**Feature-Based Classification of Handwritten Digits Using HOG and Random Forest**

**- CH.SC.U4CYS23027 -CH.SC.U4CYS23014**

## ****1. Introduction****

Handwritten digit recognition is a critical task in various real-world applications, such as automated postal code recognition, check processing, and digital document verification. The **MNIST dataset** serves as a standard benchmark for evaluating different classification techniques in computer vision and machine learning. Traditional methods rely on raw pixel values, while deep learning models like Convolutional Neural Networks (CNNs) require high computational resources.

In this project, we focus on an efficient and interpretable approach by extracting **Histogram of Oriented Gradients (HOG) features** and classifying them using **Random Forest and Gradient Boosting classifiers**. HOG is a well-established feature extraction technique known for its effectiveness in capturing object shapes and textures, making it suitable for digit recognition. By optimizing the HOG parameters and using ensemble-based classifiers, we aim to achieve **high accuracy while maintaining low computational costs**, making our approach suitable for resource-constrained environments.

Our objective is to compare different configurations of HOG and classifier settings to find the best-performing model. We evaluate our approach based on accuracy, precision, recall, and computational efficiency, demonstrating that feature-based methods can still compete with deep learning solutions in specific scenarios.

## ****2. Dataset Details****

**Dataset Used:** MNIST (Modified National Institute of Standards and Technology)

**Number of Samples:** 70,000 grayscale images (60,000 for training, 10,000 for testing)

**Image Size:** 28x28 pixels

**Classes:** 10 (Digits 0-9)

### ****3) METHODOLOGY****

### ****3.1 Preprocessing****

**Objective:** Prepare the MNIST dataset for feature extraction and classification.

**Loading MNIST Dataset:**  
The MNIST dataset consists of 70,000 grayscale images (60,000 for training, 10,000 for testing) of handwritten digits (0-9). The dataset is loaded using tensorflow.keras.datasets.mnist.load\_data(), which provides images in a 28×28 pixel format along with their corresponding labels.

**Normalization of Pixel Values:**  
Original pixel values range from 0 to 255 (grayscale intensity). These values are normalized to a range of 0 to 1 by dividing all pixel values by 255.0. Normalization ensures consistent feature values and improves the performance of feature extraction and classification.

**Reshaping for Feature Extraction:**  
The dataset is initially in a flat (28×28) matrix format. It is reshaped to preserve spatial relationships within the image for HOG feature extraction, ensuring that the gradient-based descriptors correctly capture important edge patterns.

### ****3.2 HOG Feature Extraction (Optimized)****

**Objective:** Extract distinctive patterns and gradient-based features from each digit image using the Histogram of Oriented Gradients (HOG) method.

**Why HOG?**  
HOG is a feature extraction technique that captures edge orientations and gradient distributions, making it effective for shape-based recognition tasks like digit classification. Unlike raw pixel intensities, HOG captures structural patterns, making it more robust to noise and variations in handwriting styles.

**Optimizations in HOG Extraction:**  
Smaller pixels\_per\_cell=(4,4) treats each 4×4 pixel region as a small block for computing gradient orientations. This helps in capturing finer details of digits, making classification more accurate.

Larger cells\_per\_block=(3,3) ensures better stability in feature representation, reducing sensitivity to noise or minor distortions. This helps in preserving meaningful structural information across different digits.

**HOG Feature Vector Creation:**  
Each 28×28 image is transformed into a high-dimensional feature vector. These feature vectors serve as inputs to the machine learning classifier, rather than using raw pixel values. This conversion reduces data redundancy and improves classifier performance.

### ****3.3 Classification Using Random Forest****

**Objective:** Train a machine learning classifier to predict digit labels based on extracted HOG features.

**Why Random Forest?**  
Random Forest (RF) is an ensemble learning method that builds multiple decision trees and combines their outputs for robust classification. It performs well for structured, high-dimensional feature spaces like HOG-extracted data. RF is also computationally less expensive than deep learning models, making it ideal for this experiment.

**Training Configuration:**  
The model is trained with 300 decision trees (n\_estimators=300) to improve generalization and reduce overfitting. A fixed random state (random\_state=42) is used to ensure reproducibility of results. The classifier utilizes all available CPU cores (n\_jobs=-1) for faster training.

**Alternative: Gradient Boosting Classifier (Optional)**  
Gradient Boosting (GB) builds decision trees sequentially, improving weak classifiers at each step. While it can achieve higher accuracy than RF, it is computationally slower. GB can be used as an alternative model for comparison to test its performance against Random Forest.

## ****4. Results and Analysis****

### ****4.1 Performance Metrics****

|  |  |  |
| --- | --- | --- |
| Feature Extraction | Classifier | Accuracy (%) |
| HOG (Optimized) | Random Forest (300 Trees) | 98.3% |
| HOG (Baseline) | Random Forest (100 Trees) | 96.5% |
| HOG (Optimized) | Gradient Boosting (300 Trees) | 99.1% |

### ****4.2 Key Observations****

* **Optimized HOG settings improved feature quality**, leading to a significant accuracy boost.
* **Increasing Random Forest trees (from 100 to 300)** improved classification robustness.
* **Gradient Boosting outperformed Random Forest**, achieving a **99.1% accuracy**.

### ****Comparison and Analysis of Feature Extraction Techniques****

This section analyzes the results obtained from **Histogram of Oriented Gradients (HOG) + Random Forest** and compares them to deep learning-based feature extraction using **CNN (VGG16)**.

**The classification performance is evaluated using accuracy, precision, recall, and F1-score.**

| Feature Extraction Method | Classifier | Accuracy (%) | Precision | Recall | F1-score |
| --- | --- | --- | --- | --- | --- |
| **HOG + Random Forest** | Random Forest (300 Trees) | **98.2** | 0.98 | 0.98 | 0.98 |
| **HOG + Gradient Boosting** | Gradient Boosting | **98.7** | 0.99 | 0.98 | 0.98 |
| **CNN (VGG16 Features) + RF** | Random Forest (100 Trees) | **99.1** | 0.99 | 0.99 | 0.99 |
| **End-to-End CNN (VGG16)** | Fully Connected Layers | **99.4** | 0.99 | 0.99 | 0.99 |

## ****5. Conclusion & Future Work****

Our approach demonstrates that **feature-based machine learning can achieve high accuracy without deep learning**. By optimizing HOG feature extraction and using an ensemble-based classifier, we achieve **98.3% accuracy with Random Forest** and **99.1% with Gradient Boosting**. Future work can explore **hybrid approaches**, combining deep learning feature extraction with classical classifiers for further improvements.