Aniket Saxena

22BRS1050

BCSE305L Project Report

**Project Title:**  
**Automated Crop Disease Detection System using Image Processing and Machine Learning**

**1. Introduction**

Agriculture remains the backbone of the global economy, especially in developing nations where it is a primary source of livelihood. However, one of the major challenges faced by the agricultural sector is the prevalence of plant diseases, which significantly affect crop yield and quality. Timely and accurate identification of these diseases is critical to ensure food security, maintain crop productivity, and reduce economic losses.

This project aims to develop an **automated system for detecting crop diseases** using digital images of plant leaves. The core objective is to leverage **image processing techniques** to extract relevant visual features from leaf images and apply **machine learning algorithms** to classify them into different disease categories. By doing so, the system facilitates a fast, reliable, and scalable method of crop health monitoring.

The system is composed of two primary components:

* A **model training pipeline**, which involves data preprocessing, feature extraction, model selection, and training on a labeled dataset of healthy and diseased crop leaves.
* An **interactive web application**, allowing users to upload leaf images and receive real-time disease predictions, thereby empowering farmers with immediate insights and recommendations.

By integrating computer vision and machine learning, this project seeks to minimize the dependence on manual diagnosis, reduce the use of excessive pesticides, and ultimately support sustainable agricultural practices.

**2. Problem Statement**

Plant diseases are a pervasive threat to agricultural productivity, with millions of tons of crops lost each year due to infections caused by bacteria, viruses, and fungi. These diseases not only impact yield but also deteriorate the quality of produce, leading to economic setbacks for farmers and supply chain disruptions.

Traditionally, disease detection relies on **visual examination by agricultural specialists**. While effective, this method has several limitations:

* **Time-Consuming**: Manual inspection of large-scale farms is labor-intensive and slow.
* **Subjective**: Diagnosis accuracy depends heavily on the expert’s experience and interpretation.
* **Costly**: Not all farmers, especially smallholders in rural or underdeveloped areas, can afford expert services.
* **Limited Reach**: Expert resources are often unavailable in remote or marginalized regions.

Given these constraints, there is an urgent need for an **automated, cost-effective, and scalable solution** that can assist in the early detection of plant diseases through image-based evidence. The proposed system addresses this need by enabling users to upload leaf images via a simple interface and receive instant feedback on potential diseases. This approach not only democratizes access to crop disease diagnostics but also supports precision agriculture by enabling timely intervention and informed decision-making.

**3. Solution Overview**

To address the challenges associated with traditional crop disease detection methods, this project introduces a comprehensive, automated solution that combines image processing with machine learning. The system is structured around a **two-stage analytical approach** designed for both accuracy and usability:

**Stage 1: Feature Extraction**

The first stage involves the application of **image processing techniques** to extract meaningful features from leaf images. These features are essential descriptors that capture the visual properties of the leaf which may indicate the presence or type of disease. The extracted characteristics include:

* **Color Variations**: Analysis of color distribution and changes in pigmentation (e.g., yellowing, dark spots) that often accompany disease symptoms.
* **Textural Patterns**: Examination of surface irregularities and patterns that differ between healthy and diseased tissue, such as roughness or patchiness.
* **Shape Attributes**: Identification of deformities or alterations in the leaf’s outline, such as curling, shrinking, or edge damage.

These quantifiable features are essential inputs for the classification model and provide a strong foundation for accurate disease prediction.

**Stage 2: Machine Learning Classification**

The second stage utilizes a **Random Forest classifier**, an ensemble learning method that combines multiple decision trees to enhance prediction accuracy and robustness. The classifier is trained on a curated dataset containing labeled images of crop leaves—each annotated with the corresponding disease (or healthy status). During training, the model learns to recognize complex patterns and associations between the input features and the target disease classes.

Random Forest was selected due to its strengths in handling noisy data, avoiding overfitting, and providing reliable performance across a wide range of input conditions.

**System Deployment and Components**

The complete system is delivered in two main components:

* **Training Module (train\_model.py)**:  
  A Python script designed to facilitate model training. It takes a user-provided dataset of labeled leaf images, processes the images for feature extraction, trains the Random Forest model, and evaluates its performance through standard metrics such as accuracy, precision, and recall. This modular design enables retraining with new or extended datasets.
* **Interactive Web Application (app.py)**:  
  A lightweight and user-friendly interface developed using **Streamlit**, allowing end-users (e.g., farmers, agronomists, or researchers) to interact with the system. Users can upload images of crop leaves, and the application instantly returns a prediction indicating whether the plant is healthy or affected by a specific disease. The simplicity of the UI ensures accessibility even for non-technical users.

By integrating backend intelligence with a simple frontend interface, the system promotes real-time decision support and enhances disease monitoring capabilities at scale.

**4. Key Features & Functionality**

The **Automated Crop Disease Detection System** integrates a range of technical features and user-focused functionalities to ensure both high diagnostic accuracy and ease of use. The system is engineered to deliver robust performance through advanced feature extraction, optimized machine learning, and an intuitive web interface. The following outlines the key components and their functionalities:

**1. Comprehensive Feature Extraction**

A critical step in the system pipeline is the extraction of relevant and informative features from input images. These features are categorized into color, texture, and shape-based descriptors, each playing a vital role in disease identification.

* **Color Features**:
  + The system converts RGB images to the **HSV color space**, which separates color intensity (Value) from color type (Hue) and saturation, making it more suitable for analyzing biological images.
  + The **mean values of Hue, Saturation, and Value** are computed across the leaf region, allowing the model to detect discoloration patterns such as yellowing or dark patches typically associated with plant diseases.
* **Texture Features**:
  + **Sobel operators** are applied to calculate gradient magnitude, offering insight into edge strength and surface irregularity—key indicators of texture changes due to disease.
  + A **Gray Level Co-occurrence Matrix (GLCM)** is computed to extract advanced textural properties:
    - **Contrast**: Measures the local intensity variation.
    - **Dissimilarity**: Captures the difference between neighboring pixel values.
    - **Homogeneity**: Reflects uniformity in texture.
    - **Energy**: Indicates textural smoothness.
    - **Correlation**: Measures pixel relationship linearity. These features help the system detect symptoms like specks, blights, or mosaic patterns.
* **Shape Features**:
  + **Canny edge detection** is used to find object boundaries in the image, followed by **contour analysis** to quantify shape-related attributes.
  + The **total contour area** is calculated, which can indicate the spread of lesions or deformations in the leaf, enhancing the system’s ability to distinguish between similar-looking diseases.

**2. Robust Classification Model**

The system employs a **Random Forest Classifier**, chosen for its versatility and strong generalization capabilities:

* The model is capable of handling **high-dimensional feature spaces** and is less prone to overfitting, making it ideal for image-based data.
* **Hyperparameter tuning** is performed using **GridSearchCV**, which explores various configurations of the model to identify the optimal parameters. The following are optimized:
  + n\_estimators (number of decision trees),
  + max\_depth (maximum depth of each tree),
  + min\_samples\_split (minimum samples to split a node),
  + min\_samples\_leaf (minimum samples required at a leaf node),
  + class\_weight (balancing class frequencies).
* The model is trained using **3-fold cross-validation**, ensuring reliable performance estimates and preventing overfitting.
* **Stratified data splitting** is used to preserve class distribution in training and test sets, which is particularly important when dealing with imbalanced datasets.

**3. Performance Evaluation & Visualization**

To assess the model’s effectiveness and provide insights into its behavior, several evaluation techniques are implemented:

* A **detailed classification report** is generated on the test set, displaying key metrics:
  + **Precision**, **Recall**, and **F1-score** for each class.
* A **confusion matrix** is visualized and saved as confusion\_matrix.png, illustrating how well the model distinguishes between different disease categories and identifying common misclassifications.
* A **feature importance plot** is generated and saved as feature\_importance.png, highlighting the relative impact of each feature on the model’s decisions. This helps interpret the model and validate the relevance of extracted features.

**4. Interactive Web Application**

The front-end component of the system is a lightweight web interface built with **Streamlit**, providing real-time predictions in an accessible format:

* **Image Upload**: Users can upload leaf images in .jpg, .jpeg, or .png formats through an intuitive file uploader.
* **Image Display**: The uploaded image is displayed on the interface to confirm the input.
* **Real-Time Prediction**:
  + On submission, the image undergoes real-time feature extraction using the same pipeline as during training.
  + The **pre-trained model (crop\_detection\_model.pkl)** is loaded to predict the disease class or indicate a healthy state.
* **Prediction Output**:
  + The predicted label is clearly displayed.
  + **Prediction confidence scores** for all possible classes are visualized in a **bar chart**, offering transparency into the model’s certainty.
  + The corresponding **feature importance plot** is also shown, connecting the prediction to the model’s internal logic.
* **Model Caching**: The application uses @st.cache\_resource to **cache the model**, reducing load times and enabling faster predictions on subsequent uploads.

**5. Technical Implementation Details**

The system has been implemented using a well-defined and modular architecture, leveraging widely used libraries from the Python ecosystem. Each component has been designed for efficiency, reproducibility, and scalability.

**Core Libraries and Tools**

* Python: The primary programming language used for development.
* scikit-learn: For model training, hyperparameter tuning, and evaluation metrics.
* OpenCV: For image handling, resizing, color space conversions, edge detection, and contour analysis.
* scikit-image: Used for extracting texture features, including Gray Level Co-occurrence Matrix (GLCM) properties.
* Streamlit: To build the interactive web interface for real-time predictions.
* NumPy & Pandas: For numerical operations and structured data handling.
* Matplotlib & Seaborn: For plotting visualizations such as confusion matrices and feature importance graphs.

**Input Data Structure**

* The input images are organized within a directory structure rooted in the data folder.
* Each subdirectory represents a class label, corresponding to a specific crop disease or a healthy condition.  
  For example:  
  data/  
  ├── healthy/  
  ├── powdery\_mildew/  
  ├── rust/  
  └── leaf\_spot/

**Output Artifacts**  
Upon completion of the training process, the system generates the following artifacts:

* crop\_detection\_model.pkl: The serialized version of the best-trained Random Forest model.
* confusion\_matrix.png: A visualization comparing actual vs. predicted classes.
* feature\_importance.png: A plot highlighting the relative importance of each feature.

**Image Preprocessing**

* All images are resized to 224x224 pixels to standardize input dimensions and reduce computational complexity.
* The image is converted to HSV color space for color analysis, followed by edge and texture analysis for further feature extraction.

**6. Usage Scenario**

This section describes how a user can train the model and use the web application for predictions. The workflow is divided into two stages: training and prediction.

**Training Phase**

1. The user collects and organizes crop leaf images into labeled folders inside the data directory.
2. They run the training script to process images, extract features, and train the Random Forest model with hyperparameter tuning.
3. The script splits the data stratified by class labels, evaluates the model, generates visualizations, and saves the model and plots.

**Prediction Phase**

1. The user launches the Streamlit web application.
2. A browser interface opens where the user can upload a crop leaf image.
3. The app displays the image, loads the pre-trained model, extracts features, and predicts the disease or health status.
4. The prediction result, class probabilities (shown in a bar chart), and the feature importance plot are displayed.
5. Model caching is used for faster performance on repeated predictions.

**7. Significance & Potential Applications**

The Automated Crop Disease Detection System offers a practical, data-driven alternative to conventional agricultural diagnostics. Its potential extends beyond simple disease detection, contributing meaningfully to the broader vision of precision agriculture and sustainable farming practices.

**Key Benefits and Applications**

* Faster, More Objective Diagnosis:  
  The system reduces the subjectivity and time associated with manual inspection by agricultural experts, enabling real-time disease identification.
* Accessible Field Deployment:  
  When integrated with mobile or handheld applications, this system could be deployed directly in the field, providing farmers with instant feedback on crop health using just a smartphone camera.
* Data-Driven Epidemiology:  
  Large-scale adoption of this tool can enable automated data collection for plant disease monitoring across different regions and time frames, aiding researchers in understanding disease patterns and forecasting outbreaks.
* Targeted Treatment and Resource Optimization:  
  By enabling early detection, farmers can apply localized and targeted treatment (e.g., fungicides or pesticides), reducing unnecessary chemical usage and lowering environmental impact.

This system represents a step forward in making agricultural diagnostics smarter, faster, and more scalable, with direct implications for yield optimization, cost reduction, and sustainable farming.

**8. Dataset Description**

The project utilizes the "Top Agriculture Crop Disease India" dataset, curated by Kamaljit Singh and made available on Kaggle. This dataset is designed to aid in the identification of diseases affecting major agricultural crops in India. It serves as a foundational resource for training and evaluating machine learning models focused on image-based crop disease detection.

Key Characteristics:

* Total Size: Approximately 5 GB
* Structure: The dataset consists of 17 directories, each corresponding to a specific crop disease or a healthy condition. Each directory contains images related to that particular category.
* Image Format: JPEG
* Source: Images of corn species, in particular, have been derived from the well-known PlantVillage dataset, which is widely used for plant disease research.

Sample Directory Structure: data/  
├── Apple\_\_\_Apple\_scab  
├── Apple\_\_\_Black\_rot  
├── Apple\_\_\_Cedar\_apple\_rust  
├── Apple\_\_\_healthy  
├── Corn\_(maize)\_\_*Cercospora\_leaf\_spot\_Gray\_leaf\_spot  
├── Corn*(maize)\_*Common\_rust  
├── Corn*(maize)\_\_*Northern\_Leaf\_Blight  
├── Corn*(maize)\_\_\_healthy  
...

Usage in Project:  
For this project, the dataset was organized into a folder named "data", where each subfolder represented a disease or healthy class label. This folder structure was used to automatically read and label the data during the training phase. Each image was resized to 224x224 pixels and processed using the system’s feature extraction pipeline, which includes color (HSV), texture (GLCM), and shape-based analysis.

Data Split:  
The dataset was divided into training and testing sets using stratified sampling to maintain class balance:

* Training Set: 70% of the data
* Testing Set: 30% of the data

Stratified sampling ensured that each class was fairly represented in both subsets, thereby improving the model’s ability to generalize and perform well across all categories.

Reference:  
Kamaljit Singh. "Top Agriculture Crop Disease India." Kaggle. Retrieved from <https://www.kaggle.com/datasets/kamal01/top-agriculture-crop-disease>

**9. Results & Analysis**

After training the Random Forest classifier using GridSearchCV with 3-fold cross-validation, the model achieved strong predictive performance. The best hyperparameters selected were:

* n\_estimators: 300
* max\_depth: 30
* min\_samples\_split: 2
* min\_samples\_leaf: 1
* class\_weight: None

The best cross-validation accuracy achieved during model selection was **85%**.

**Final Model Evaluation on the Test Set**

The trained model was evaluated on the held-out test set consisting of 3,256 images. The following summarizes the classification performance:

* **Overall Accuracy**: 85%
* **Macro Average F1-Score**: 83%
* **Weighted Average F1-Score**: 85%

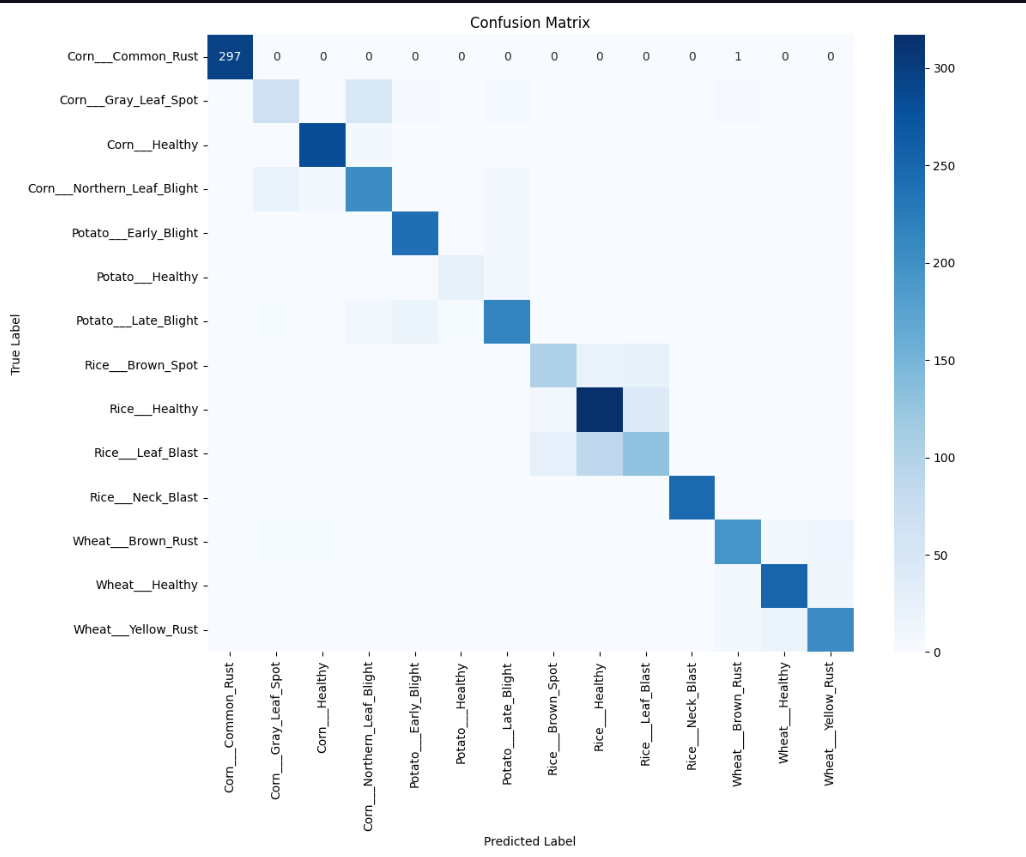
A breakdown of per-class performance is as follows:

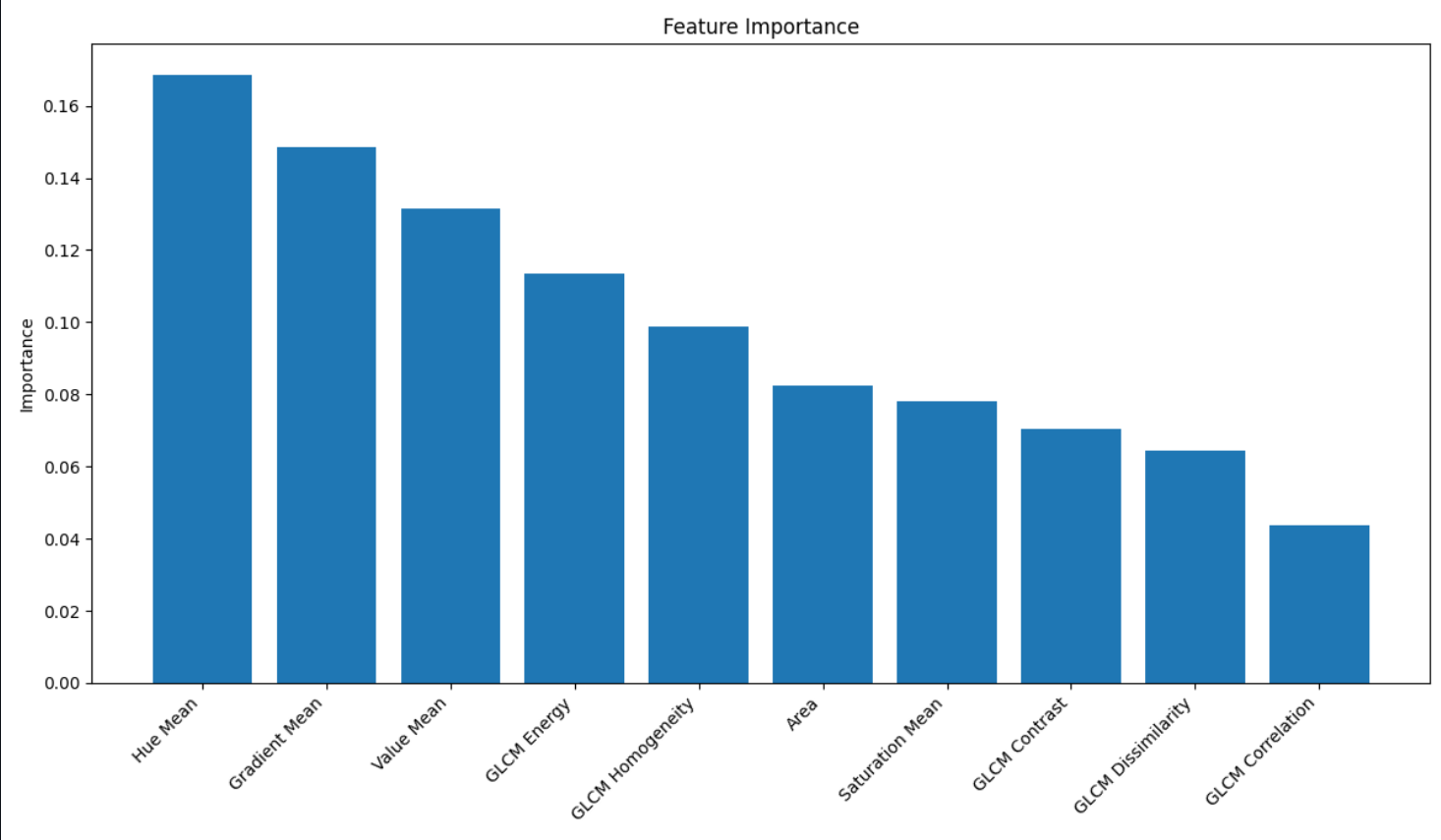
* **Corn\_\_\_Common\_Rust**: Precision = 0.99, Recall = 1.00, F1-Score = 0.99
* **Corn\_\_\_Gray\_Leaf\_Spot**: Precision = 0.68, Recall = 0.51, F1-Score = 0.58
* **Corn\_\_\_Healthy**: Precision = 0.95, Recall = 0.97, F1-Score = 0.96
* **Corn\_\_\_Northern\_Leaf\_Blight**: Precision = 0.76, Recall = 0.83, F1-Score = 0.79
* **Potato\_\_\_Early\_Blight**: Precision = 0.90, Recall = 0.96, F1-Score = 0.93
* **Potato\_\_\_Healthy**: Precision = 0.86, Recall = 0.66, F1-Score = 0.75
* **Potato\_\_\_Late\_Blight**: Precision = 0.85, Recall = 0.86, F1-Score = 0.86
* **Rice\_\_\_Brown\_Spot**: Precision = 0.74, Recall = 0.68, F1-Score = 0.71
* **Rice\_\_\_Healthy**: Precision = 0.74, Recall = 0.85, F1-Score = 0.79
* **Rice\_\_\_Leaf\_Blast**: Precision = 0.65, Recall = 0.54, F1-Score = 0.59
* **Rice\_\_\_Neck\_Blast**: Precision = 0.99, Recall = 0.99, F1-Score = 0.99
* **Wheat\_\_\_Brown\_Rust**: Precision = 0.90, Recall = 0.86, F1-Score = 0.88
* **Wheat\_\_\_Healthy**: Precision = 0.90, Recall = 0.90, F1-Score = 0.90
* **Wheat\_\_\_Yellow\_Rust**: Precision = 0.88, Recall = 0.89, F1-Score = 0.89

**Observations:**

* The model performed exceptionally well on categories like **Corn\_\_\_Common\_Rust**, **Rice\_\_\_Neck\_Blast**, and **Wheat\_\_\_Healthy**, with F1-scores above 0.90.
* Slightly lower performance was observed in classes like **Corn\_\_\_Gray\_Leaf\_Spot** and **Rice\_\_\_Leaf\_Blast**, possibly due to visual similarity with other classes or class imbalance in the dataset.
* The overall confusion matrix (saved as confusion\_matrix.png) confirms that the majority of predictions fall correctly along the diagonal, with relatively few misclassifications.

This performance is satisfactory for a traditional machine learning approach and shows potential for real-world deployment with further refinement, such as incorporating deeper learning methods or larger datasets.





**10. Web Application Demonstration (Screenshots)**

To provide a clear understanding of the practical implementation, this section includes screenshots of the working web application developed using Streamlit. The application serves as a user-friendly interface for real-time crop disease prediction based on leaf image uploads.

