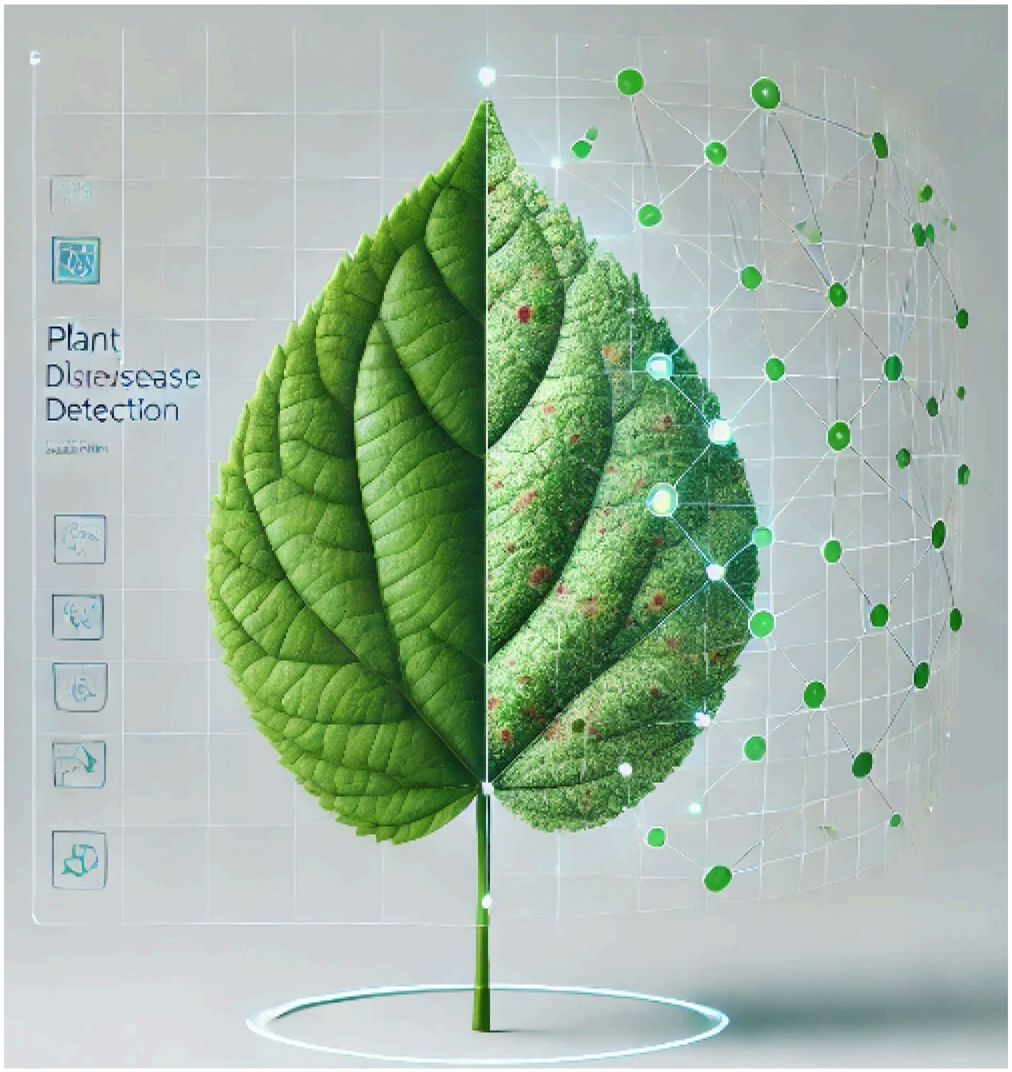
Final Project

Plant Disease Detection from Images



Bhuvaneswaran R

Batch - MA28

Contents:

Overview:

This project focuses on detecting plant diseases from leaf images using a Convolutional Neural Network (CNN) model. The goal is to develop a user-friendly web application that allows users to upload images of plant leaves to predict the type and severity of plant diseases, enabling timely and accurate intervention for better agricultural outcomes.

Objectives

The primary objectives of this project are:

1. Automatic Disease Detection: Enable farmers and gardeners to detect plant diseases early by simply uploading leaf images.
2. Accuracy and Efficiency: Develop a CNN model that achieves high accuracy, reliability, and low latency for real-time predictions.
3. User Accessibility: Create an intuitive web application interface using Streamlit, accessible to users without technical expertise.

System Architecture

The system consists of the following components:

1. User Interface (UI): A web-based interface built with Streamlit for image uploads.
2. Image Processing and Preprocessing: Processes uploaded images for optimal model input.
3. CNN Model: The core classification engine that detects diseases from leaf images.
4. Backend Processing: Processes the input images, runs the CNN model, and displays predictions.
5. Deployment: Deploy the application on a cloud platform to ensure availability and scalability.

Model Architecture

The CNN model architecture consists of:

- Convolutional Layers: Extract hierarchical features from images.
- Pooling Layers: Reduce dimensionality while retaining essential features.
- Dense Layers: Fully connected layers for final classification based on features.
- Activation Functions: ReLU in hidden layers and Softmax in the output layer to output probabilities for each disease category.
- Loss Function and Optimizer: Cross-entropy loss with Adam optimizer for efficient training and reduced error.

Dataset

- Source: New Plant Diseases Dataset from Kaggle, which includes labeled images of plant diseases and healthy leaves.
- Data Composition: Contains images for various diseases across multiple plant species, enhancing model accuracy and generalizability.
- Data Division: The dataset is divided into training, validation, and testing sets for model evaluation and performance tuning.

Preprocessing Steps

1. Resizing: Standardize image dimensions to meet model input requirements.
2. Normalization: Scale pixel values to normalize data and improve model convergence.
3. Data Augmentation: Techniques such as rotation, flipping, and scaling to expand the dataset and make the model more resilient to variations in leaf orientation and lighting.

CNN Model Implementation

- Framework: TensorFlow/Keras for building, training, and evaluating the CNN model.
- Architecture: A sequential model with alternating convolutional and pooling layers, followed by fully connected layers.
- Custom Layers: Incorporate dropout layers to reduce overfitting and increase model robustness.
- Hyperparameters: Optimized learning rate, batch size, and epoch count through experimentation for the best model performance.

Training the Model

1. Training Process: The model is trained on the preprocessed images with labeled disease categories.
2. Validation: Used to tune hyperparameters and monitor model overfitting.
3. Testing: Model performance is evaluated on an unseen test set to assess accuracy and reliability.
4. Performance Optimization: Data augmentation and transfer learning applied to enhance model accuracy.

Image Upload and Detection Workflow

1. Image Upload: Users upload an image of a plant leaf through the Streamlit interface.
2. Image Preprocessing: The uploaded image undergoes resizing and normalization.
3. Prediction: The preprocessed image is fed into the CNN model, which outputs the predicted disease category.
4. Results Display: The application displays the disease type with a confidence score, assisting users in understanding the leaf health.

Disease Labels

The model categorizes plant leaves into multiple disease labels, each representing a specific disease or healthy state. Disease categories include, but are not limited to:

- Powdery Mildew
- Leaf Spot
- Rust
- Blight
- Healthy Leaf

User Interface (UI) Design

- Platform: Built using Streamlit for ease of use and accessibility.
- Features:
    - *File Upload*: A drag-and-drop feature for leaf images.
    - *Real-Time Feedback*: Instant predictions on disease type and severity.
    - *Guidance and Documentation*: Step-by-step instructions to aid users in understanding and interpreting the results.

Performance Metrics

To evaluate the CNN model, the following metrics are used:

- Accuracy: Measures the percentage of correctly predicted cases.
- Precision: The ratio of correctly predicted positive observations to the total predicted positives.
- Recall: The ratio of correctly predicted positives to the actual positives.
- F1 Score: The harmonic mean of precision and recall, balancing the two metrics.
- Confusion Matrix: Provides insights into misclassifications, enabling further model refinement.

Future Enhancements

1. Broader Disease Classification: Expand the model to detect additional diseases.
2. Real-Time Video Processing: Enable real-time detection from video feeds for larger-scale applications.
3. Mobile Application Integration: Develop a mobile-friendly version for field use.
4. Incremental Learning: Implement a self-learning mechanism that adapts based on new data or images uploaded by users.

Dependencies

pandas

TensorFlow

Keras

Streamlit

matplotlib

NumPy

Challenges Faced

- Creating and testing a neural network is time consuming when you don't have the right tools, our model created for recognizing facial emotions required a gpu server in order to process the results faster, free gpu servers available on Google Colab had time constraints and often crashed when running for high number of epochs.
- Hyperparameter tuning for neural networks is a tedious task as it requires fast gpu's to run multiple iterations which on a normal computer/server might take days or weeks.
- Once the model is created, deploying it successfully is a bit difficult since most of the popular cloud servers that offer these kinds of services require certain prerequisites.

Conclusion

The "Plant Disease Detection from Images" project successfully applies CNN technology to automate plant disease identification. This application aids users, especially in agriculture, to diagnose plant health issues early, enabling timely interventions. With further enhancements, this tool can be invaluable in large-scale agricultural practices, contributing to higher yields and more sustainable farming.