PLANT DISEASE DETECTION USING DEEP LEARNING

Introduction

Objective:

The goal of this project is to develop a deep learning-based model for plant disease detection using images of plant leaves. The trained model will be deployed in a Streamlit-based web application, where users can upload leaf images to receive predictions about potential diseases.

This study leverages CNNs (Convolutional Neural Networks) and transfer learning with pre-trained models such as EfficientNetB0, Xception, and DenseNet121, optimizing performance using data augmentation, mixed precision training, and learning rate adjustments.

Dataset Description

Dataset Source:

The dataset used is the New Plant Diseases Dataset (Augmented) from Kaggle, containing labeled images of plant leaves affected by different diseases.

Preprocessing Steps:

- **Resizing:** All images resized to 96×96 pixels for compatibility with the models.
- **Normalization:** Pixel values scaled to the 0-1 range to improve convergence.
- **Augmentation:** Random transformations applied to enhance generalization, including:
    - Rotation
    - Width & Height Shift
    - Zoom & Shear
    - Horizontal Flip

Model Architecture and Training

Pre-trained Models Used (Transfer Learning)

Three deep learning models were fine-tuned for plant disease classification:

1. **EfficientNetB0**
2. **Xception**
3. **DenseNet121**

Each model was initialized with ImageNet weights and modified to:

- Include Global Average Pooling, Batch Normalization, and Dropout layers.
- Replace the classifier with a fully connected layer matching the number of plant disease classes.

Training Configuration

Hyperparameters:

Parameter	**Value**
Batch Size	16
Epochs	10
Optimizer	Adam (learning rate = 0.0001)
Loss Function	Sparse Categorical Crossentropy

Training Strategy:

- **Early Stopping:** Prevents overfitting by monitoring validation loss.
- **Learning Rate Reduction:** Reduces learning rate when validation loss stagnates.
- **Model Checkpoints:** Saves the best model based on validation accuracy.

Performance Evaluation

Each model was evaluated using the validation dataset based on accuracy, precision, recall, and F1-score.

Results Summary

Model	**Accuracy**
EfficientNetB0	0.96
Xception	0.71
DenseNet121	0.92

Best Model: EfficientNetB0 (Highest Accuracy = 96%)

Deployment & Application Features

A Streamlit-based web app was developed to allow users to upload images and receive real-time predictions.

Features:

- Image Upload: Accepts JPEG, PNG, and BMP files.
- Preprocessing Pipeline: Converts images to 96×96 pixels and normalizes them.
- Model Selection: Users can choose between EfficientNetB0, Xception, and DenseNet121.
- Real-Time Prediction: Model inference runs instantly on uploaded images.
- Disease Interpretation: Displays predicted class and a confidence score.

Conclusion & Future Work

Key Findings:

- EfficientNetB0 achieved the highest accuracy (96%) and demonstrated superior generalization.
- DenseNet121 (92%) outperformed Xception (71%), indicating that deeper architectures benefit from transfer learning in plant disease detection.
- Transfer learning improved performance significantly, reducing the training time while maintaining high accuracy.

Future Enhancements:

- Train models on higher resolution images (e.g., 224×224) for better feature extraction.
- Experiment with self-supervised learning techniques for label-efficient training.
- Optimize the web application UI/UX for better user experience.
- Expand the dataset to include more plant species and diseases for broader applicability.

Deliverables

- **Trained Models:** EfficientNetB0_best.keras, Xception_best.keras, DenseNet121_best.keras.
- **Streamlit Application:** Plant Disease Detection Web App.
- **Codebase:** Python scripts for model training, preprocessing, and deployment.
- **Project Report:** This document summarizing results, methodology, and future scope.
