Plant Disease Detection Using Deep Learning

# 1. Project Title:

AI-Based Plant Disease Detection Using Convolutional Neural Networks (CNN)

# 2. Objective:

The primary objective of this project is to design and implement an AI-based system that can identify plant leaf diseases through image classification using Convolutional Neural Networks (CNNs). The motivation is to enable early detection and diagnosis of crop diseases to reduce losses in agricultural productivity, improve yield quality, and provide an accessible tool for farmers, researchers, and agri-tech developers.

# 3. Problem Statement:

Crop diseases cause significant economic losses to farmers and threaten food security globally. Traditional disease detection methods are time-consuming, require expert knowledge, and are not scalable. This project addresses these issues by creating an intelligent, automated, and scalable system to detect multiple types of plant diseases from leaf images with high accuracy using deep learning techniques.

# 4. Dataset Used:

The dataset used for training and testing the model is sourced from Kaggle and includes approximately 87,000 RGB images categorized into 38 different classes. These images represent healthy leaves and leaves affected by a variety of diseases across several crops such as apple, grape, corn, tomato, and more.

The dataset was augmented offline to increase variability and simulate real-world conditions such as lighting, rotation, and background noise. It was then split into an 80:20 ratio for training and validation, with a small test set of 33 manually selected images.

# 5. Methodology:

The system employs a deep Convolutional Neural Network (CNN) architecture built using TensorFlow and Keras. Key steps in the methodology include:

• Preprocessing: Images are resized to 128x128 pixels and normalized.  
• Augmentation: ImageDataGenerator is used for real-time augmentation during training.  
• Model Architecture: The CNN consists of multiple convolutional, max pooling, and dense layers with ReLU activation and softmax output for multi-class classification.  
• Compilation: The model is compiled using the Adam optimizer and categorical crossentropy loss.  
• Training: The model is trained on the dataset using 20 epochs and evaluated using accuracy and loss metrics.  
• Deployment: A Streamlit web application is built to make predictions from uploaded images.

# 6. System Architecture:

The system follows a pipeline consisting of:  
1. Image input through a web UI  
2. Image preprocessing (resize, normalize)  
3. Feeding into trained CNN model  
4. Prediction of the disease class  
5. Display of results to the user via the Streamlit interface

# 7. Tools and Technologies Used:

• Programming Language: Python  
• Frameworks/Libraries: TensorFlow, Keras, OpenCV, NumPy, Matplotlib, Pandas, Streamlit  
• Development Tools: Jupyter Notebook, VS Code  
• Platform: Web app built using Streamlit  
• File Format: .keras model format for loading and inference

# 8. Results and Evaluation:

The model showed strong performance on both training and validation datasets:

|  |  |
| --- | --- |
| Metric | Value |
| Training Accuracy | ~98% |
| Validation Accuracy | ~96% |
| Test Accuracy | ~94% |
| Model Format | trained\_plant\_disease\_model.keras |

The model generalizes well on unseen test data, and results indicate its robustness to noise and image variation. Model predictions were also evaluated with visual confirmation, showing high consistency.

# 9. Applications:

• Precision farming: Real-time disease detection and diagnosis in the field using mobile devices  
• Agri-tech advisory services: Integration with platforms providing expert crop management  
• Research and academia: Training tool for agricultural students and researchers  
• IoT and drone monitoring: Integration with smart farming devices and aerial surveillance systems

# 10. Limitations:

• Limited to 38 disease categories based on the dataset  
• Misclassification may occur with poor image quality or overlapping symptoms  
• Requires internet access for the web-based interface  
• Lack of localized language support for farmers

# 11. Future Scope:

• Integration with GPS to provide location-based disease trends  
• Grad-CAM visualizations to explain model predictions  
• Multilingual support for wider accessibility  
• Expansion to include more crops, pests, and fruit quality analysis  
• Integration with agricultural drones and real-time dashboards

# 12. Conclusion:

The developed plant disease detection system successfully demonstrates how deep learning can address real-world agricultural challenges. The CNN-based model provides accurate, quick, and scalable disease classification, while the Streamlit interface offers user-friendly access for non-technical users. With future improvements, this system can be a vital tool in promoting sustainable and smart agriculture.

# 13. Team Members:

• Tanya (Project Lead and Developer)

Chahat Thakur