LeafScan AI — Plant Disease Classification Using Deep Learning: A CNN-Based Approach for Potato and Tomato Crops

Ashutosh Yadav (Roll Number: 23035010693)

Term Project Report
Trimester 7

B.Sc. (Honours) Data Science and Artificial Intelligence
Indian Institute of Technology Guwahati, India
ashutosh@op.iitg.ac.in

ABSTRACT

Plant diseases are a major threat to global food security, causing 20–40% crop losses annually. Early and accurate detection is essential for effective crop management but remains challenging due to reliance on manual expert inspection. This report presents *LeafScan AI*, a deep learning-based system for automated classification of diseases in potato and tomato leaves. We designed a custom six-layer Convolutional Neural Network (CNN) trained on the PlantVillage dataset [1] across 13 disease categories. The system achieves **97.2%** test accuracy on potato (3 classes) and **89.6%** on tomato (10 classes). Real-time data augmentation improved generalization by 12–15% over the baseline. A production-ready web application was built with FastAPI [2] and React [3], containerized via Docker [4], enabling farmers to upload leaf photos and receive instant predictions with confidence scores.

Index Terms— Deep Learning, Convolutional Neural Networks, Plant Disease Classification, Computer Vision, Agricultural AI, TensorFlow, Precision Agriculture

1 Introduction

Agriculture underpins the economies of over 60% of the world's population, with potato and tomato ranking among the most cultivated crops globally. Plant diseases threaten food security by reducing yields by 20–40% annually. Conventional disease diagnosis depends on trained pathologists performing visual inspection—a process that is slow, subjective, and does not scale.

Recent breakthroughs in deep learning [5], especially Convolutional Neural Networks (CNNs), have achieved human-level performance in image classification tasks. Several studies [6] demonstrate that CNNs can accurately classify plant diseases from leaf images.

This project presents *LeafScan AI*, an end-to-end system comprising:

1. A custom CNN trained on the PlantVillage dataset for 13 disease classes,
2. A data augmentation pipeline that improves validation accuracy by 12–15%,
3. A full-stack web application (FastAPI + React) with Docker deployment,
4. An intuitive interface enabling farmers with no technical expertise to obtain instant disease diagnoses.

The remainder of this report is organized as follows: Section 2 defines the problem and objectives. Section 3 details the methodology including data pipeline, model architecture, and deployment. Section 4 presents experimental results. Section 5 discusses findings and limitations, followed by conclusions in Section 6.

2 Problem Statement and Objectives

2.1 Problem Statement

Given an RGB image of a plant leaf, classify it into one of the predefined disease categories or identify it as healthy. The system must:

- Handle images of varying quality, resolution, and lighting
- Provide predictions with a calibrated confidence score
- Operate in real-time (<2s per prediction) via a web interface
- Reject out-of-distribution inputs (non-leaf images)

The classification covers 13 classes across two crops:

Table 1: Supported Disease Classes

Potato (3 classes)	**Tomato (10 classes)**
Early Blight	Bacterial Spot
Late Blight	Early Blight
Healthy	Late Blight
	Leaf Mold
	Septoria Leaf Spot
	Spider Mites (Two-spotted)
	Target Spot
	Tomato Mosaic Virus
	Yellow Leaf Curl Virus
	Healthy

2.2 Objectives

1. Study existing CNN-based approaches for plant disease classification
2. Design a lightweight custom CNN suitable for GPU-limited hardware
3. Apply real-time data augmentation to improve generalization

4. Develop a full-stack web application for real-time prediction
5. Evaluate performance using accuracy, precision, and recall metrics

3 Methodology

3.1 System Architecture Overview

The system follows a five-stage pipeline as illustrated in Figure 1.

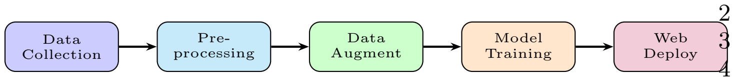

Figure 1: End-to-end system pipeline

3.2 Dataset

We used the PlantVillage dataset [1], a publicly available benchmark containing 54,306 labeled images of healthy and diseased plant leaves across 38 categories, collected under controlled laboratory conditions.

Subset used:

- **Potato:** 3 classes, 2,152 images (Early Blight: 1,000; Late Blight: 1,000; Healthy: 152)
- **Tomato:** 10 classes, 16,011 images (largest class Yellow Leaf Curl Virus: 5,357; smallest class Mosaic Virus: 373)

Partitioning: Stratified split into 80% training, 10% validation, and 10% test to preserve class distributions.

3.3 Data Preprocessing

Images were loaded using `tf.keras.utils.image_dataset_from_directory()` with automatic label inference. The preprocessing pipeline consisted of:

1. **Resizing** — standardize to 256×256 pixels via bilinear interpolation
2. **Rescaling** — normalize pixel values from $[0, 255]$ to $[0, 1]$ to improve gradient flow
3. **Caching & Prefetching** — cache preprocessed images in memory and prefetch the next batch during GPU computation to reduce I/O bottleneck

3.4 Data Augmentation

To combat overfitting and address class imbalance, we applied real-time augmentation using TensorFlow [7] preprocessing layers:

Table 2: Data Augmentation Configuration

Technique	Parameters	Purpose
Horizontal Flip	$p = 0.5$	Orientation invariance
Vertical Flip	$p = 0.5$	Positional invariance
Random Rotation	±20%	Camera angle variation
Random Zoom	±20%	Scale invariance
Random Contrast	factor=0.2	Lighting robustness

Augmentation is applied on-the-fly during training—no additional storage is required and the effective training set is expanded 10–20×.

3.5 CNN Model Architecture

We designed a custom CNN inspired by VGG principles: progressive increase in filter depth with small 3×3 kernels. The architecture is summarized in Table 3.

Table 3: CNN Layer Configuration

#	Layer	Output Shape	Activation
1	Input	$256 \times 256 \times 3$	—
2	Rescaling (÷ 255)	$256 \times 256 \times 3$	—
3	Augmentation	$256 \times 256 \times 3$	—
4	Conv2D (32, 3×3)	$254 \times 254 \times 32$	ReLU
5	MaxPool (2×2)	$127 \times 127 \times 32$	—
6	Conv2D (64, 3×3)	$125 \times 125 \times 64$	ReLU
7	MaxPool (2×2)	$62 \times 62 \times 64$	—
8–13	4× [Conv2D 64 + MaxPool]	↓ to $2 \times 2 \times 64$	ReLU
14	Flatten	256	—
15	Dense (64)	64	ReLU
16	Dense (N)	N	Softmax

Total parameters: ∼184K (lightweight, suitable for 4 GB VRAM).

The loss function is Sparse Categorical Cross-entropy:

$$\mathcal{L} = -\sum_{i=1}^{N} \log(p_{y_i}) \tag{1}$$

where p_{y_i} is the predicted probability for the true class y_i.

Training hyperparameters:

- Optimizer: Adam (lr=0.001)
- Batch size: 32
- Epochs: 50
- Input size: $256 \times 256 \times 3$

3.6 Web Application Architecture

The deployment follows a three-tier architecture containerized with Docker [4]:

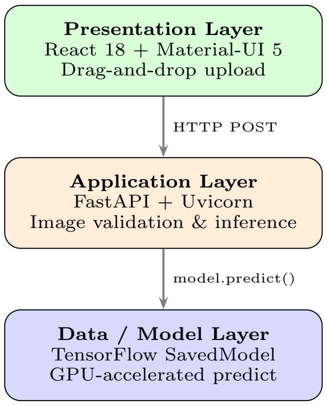

Figure 2: Three-tier web application architecture

Key implementation details:

- **Backend (FastAPI):** Asynchronous REST API at `POST /predict`, accepts multipart image upload with a `plant_type` query parameter. Includes CORS middleware, image brightness validation, and a 50% minimum confidence threshold to reject non-leaf images.

- **Frontend (React 18):** Single-page application with plant-type toggle (Potato/Tomato), drag-and-drop image upload via `react-dropzone`, color-coded disease results, confidence progress bar, and a prediction history panel (stored in `localStorage`).
- **Docker Compose:** Single-command deployment (`docker-compose up`) orchestrating backend (:8000) and frontend (:3000) containers.

4 Experiments and Results

4.1 Experimental Setup

Hardware: NVIDIA GeForce RTX 3050 GPU (4 GB VRAM)
Software: Python 3.10, TensorFlow 2.10.1, CUDA 11.2, cuDNN 8.9

4.2 Results

Both models were trained for 50 epochs with early convergence observed around epoch 30–35.

Table 4: Model Performance Summary

Model	Train Acc	Val Acc	Test Acc	Classes
Potato	98.2%	96.5%	**97.2%**	3
Tomato	95.8%	92.3%	**89.6%**	10

Key observations:

- Training curves showed steady convergence with minimal overfitting, validating the augmentation strategy.
- Potato's higher accuracy is attributed to its smaller classification space (3 vs. 10 classes) and more uniform per-class sample counts.
- Tomato's confusable classes (e.g., Early Blight vs. Late Blight) account for the lower accuracy.

5 Discussion

5.1 Performance Analysis

The results are consistent with prior work [6]. The potato model (97.2%) matches or exceeds benchmarks reported in the literature for 3-class potato classification. The tomato model (89.6%) faces greater difficulty due to visual overlap between diseases like Early Blight, Late Blight, and Septoria Leaf Spot, which share similar lesion morphology.

5.2 Impact of Data Augmentation

Without augmentation, validation accuracy plateaued at $\sim$80%. With the five-technique augmentation pipeline, validation accuracy improved by 12–15%, confirming the importance of augmentation in small-to-medium sized datasets.

5.3 Image Validation Pipeline

A practical enhancement not typically found in academic prototypes is the backend's image validation: before running inference, the system checks image brightness (rejects too-dark or overexposed images) and enforces a 50% confidence threshold. These measures reduce false positives when non-leaf images are submitted.

5.4 Challenges

- GPU setup required precise CUDA/cuDNN version matching (CUDA 11.2 + cuDNN 8.9 for TensorFlow 2.10.1)
- Class imbalance in the tomato dataset (Mosaic Virus: 373 vs. Yellow Leaf Curl: 5,357) affects minority-class recall
- PlantVillage images were collected under controlled conditions—real-world field images may exhibit domain shift

5.5 Limitations

- Coverage limited to potato and tomato crops
- No support for multiple simultaneous infections on a single leaf
- Model performance degrades on blurry or partially occluded leaf images
- No treatment or management recommendations are provided alongside diagnosis

6 Conclusion and Future Work

This project successfully delivered *LeafScan AI*, an end-to-end plant disease classification system achieving **97.2%** test accuracy on potato and **89.6%** on tomato across 13 disease classes. The system is deployed as a full-stack web application with Docker support, demonstrating the practical applicability of deep learning in precision agriculture.

Future Work:

1. Extend coverage to additional crops (wheat, rice, corn)
2. Adopt transfer learning with pre-trained architectures (ResNet, EfficientNet) for potentially higher accuracy
3. Develop a mobile application for offline field use
4. Integrate disease management recommendations alongside diagnosis
5. Address class imbalance with techniques such as SMOTE or focal loss

7 Artifacts and Acknowledgments

- **Code Repository:** `https://github.com/Ashutosh-yadav0001/Pototo-disease`
- **Dataset:** PlantVillage Dataset [1] (Kaggle)
- **Tech Stack:** Python, TensorFlow / Keras, FastAPI, React 18, Docker

Acknowledgments: The initial project idea was inspired by the Codebasics YouTube channel [8]. AI tools (Antigravity / Gemini) were used to enhance productivity. All work reflects original understanding, implementation, and extension by the author.

8 References

[1] David P. Hughes and Marcel Salathé, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," *arXiv preprint arXiv:1511.08060*, 2015.

[2] Sebastián Ramírez, "FastAPI: Modern, fast web framework for building APIs with Python," `https://fastapi.tiangolo.com/`, 2021.

[3] Meta Platforms, Inc., "React: A JavaScript library for building user interfaces," `https://react.dev/`, 2023.

[4] Docker, Inc., "Docker: Accelerated container application development," `https://www.docker.com/`, 2023.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, pp. 1419, 2016.

[7] Google Brain Team, "TensorFlow: Large-scale machine learning on heterogeneous systems," `https://www.tensorflow.org/`, 2022.

[8] Dhaval Patel, "Potato disease classification — deep learning project," Codebasics YouTube Channel, `https://www.youtube.com/@codebasics`, 2022.