Summer Project Report

DeepLeaf

Plant Disease Recognition Using Deep Learning and Mobile Application Deployment

By
Suyash Gupta
B.Tech Artificial Intelligence and Data Science
IIIT Bangalore

Date: June - August 2025

Abstract

- Developed a deep learning-based system for recognizing plant diseases from leaf images, inspired by the work done on Cattle Monitoring Model in the Summer School at Aivancity University, Cachan.
- Used the PlantVillage dataset to train a convolutional neural network (CNN) model for 15 disease/healthy classes across multiple crops.
- Converted the trained PyTorch model to ONNX format for mobile deployment.
- Built an Android application in Kotlin that can work offline to predict plant diseases from images captured via camera or selected from the gallery.
- Provided farmers and agricultural users with instant, offline, and accessible plant disease detection.

Project Details

- **Objective:** To create a portable and offline mobile app that can accurately detect plant diseases using deep learning.
- **Dataset:** PlantVillage dataset from Kaggle.
- **Number of Classes:** 15 classes:
  - Pepper__bell___Bacterial_spot
  - Pepper__bell___healthy
  - Potato___Early_blight
  - Potato___Late_blight
  - Potato___healthy
  - Tomato_Bacterial_spot
  - Tomato_Early_blight
  - Tomato_Late_blight
  - Tomato_Leaf_Mold
  - Tomato_Septoria_leaf_spot
  - Tomato_Spider_mites_Two-spotted_spider_mite
  - Tomato__Target_Spot
  - Tomato__Tomato_YellowLeaf__Curl_Virus

- Tomato__Tomato_mosaic_virus
- Tomato_healthy

- **Technologies Used:**
  - Python (for model training)
  - PyTorch (deep learning framework)
  - Torchvision (data preprocessing and augmentation)
  - ONNX & ONNX Runtime (for mobile inference)
  - Kotlin, XML & Android Studio (for Android app development)

- **Model Development:**
  - Image preprocessing: resized to 128×128, normalized, and augmented (rotation, flipping).
  - Architecture: Pretrained ResNet Model fine-tuned for 15-class classification with appropriate hyoperparameters.
  - Training: 15 epochs, Adam Optimizer, CrossEntropyLoss Function.
  - Validation Accuracy: 98.76
  - Export: Converted to ONNX (Open Neural Network Exchange) format for integration into the Android app.

- **Android Application Features:**
  - Capture an image using the device camera.
  - Select an existing image from the gallery.
  - Display predicted disease and confidence score.
  - Works fully offline for faster accessibility.

Challenges and Solutions

- **Large Model Size:** Initial trained model was too large for smooth mobile performance.

  Solution: Reduced image resolution to 128×128, used a smaller pretrained model variant, and fine-tuned only key layers to shrink file size.

- **Incorrect Predictions:** Early versions of the model frequently confused similar disease classes such as Tomato Early Blight and Tomato Late Blight.

  Solution: Added more diverse training samples and applied stronger data augmentation (rotation, flip, brightness change) to improve generalization.

- **Custom CNN Model Giving Verbose and Poorly Generalized Results:** The initial self-built CNN model often overfit to training data and gave inconsistent predictions in real-world tests.

  Solution: Switched to a pretrained MobileNetV3 model for its balance of accuracy, speed, and mobile efficiency, and fine-tuned it for the 15 target classes.

- **Mobile Runtime Errors:** ONNX Runtime on Android threw input-output shape mismatch errors.

  Solution: Verified model input shape during export and aligned mobile preprocessing pipeline with training preprocessing steps.

- **Limited Dataset Variability:** PlantVillage dataset images were captured under controlled lighting and backgrounds, which could hurt real-world performance.

  Solution: Augmented training set with random color jitter, contrast variation, and background noise to simulate real-world conditions.

- **Performance on Low-End Devices:** The app initially lagged on devices with lower RAM and slower processors.

  Solution: Reduced batch size for inference to 1, quantized the model during ONNX export, and optimized the Kotlin code to avoid unnecessary object creation.

- **Incorrect Image Orientation in Predictions:** Some images from the gallery appeared rotated or mirrored, affecting predictions.

  Solution: Implemented an image orientation correction step before preprocessing.

- **Debugging on Physical Devices:** Emulator predictions worked fine but failed on actual smartphones due to file path handling differences.

  Solution: Used Android's ContentResolver API to correctly read images from camera and gallery.

- **Balancing Accuracy and Speed:** Higher accuracy models slowed down predictions on mobile, while faster models reduced accuracy.

  Solution: Performed comparative benchmarking of architectures and selected a balanced configuration that achieved high accuracy with acceptable speed.

- **Color Variations in Leaf Images:** Lighting differences in field images changed the leaf color tone, confusing the model.

  Solution: Normalized color channels during preprocessing to minimize lighting effects.

- **Maintaining Offline Capability:** Ensuring model inference without internet while keeping app size reasonable.

  Solution: Packaged the ONNX model within the app bundle.

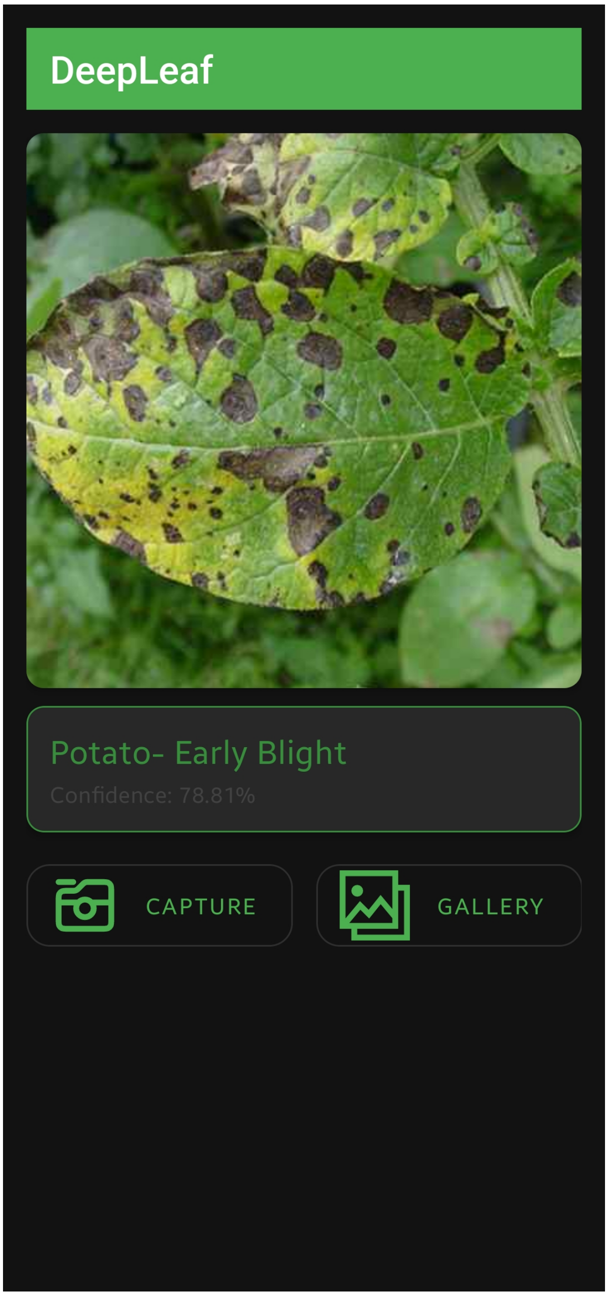
Results

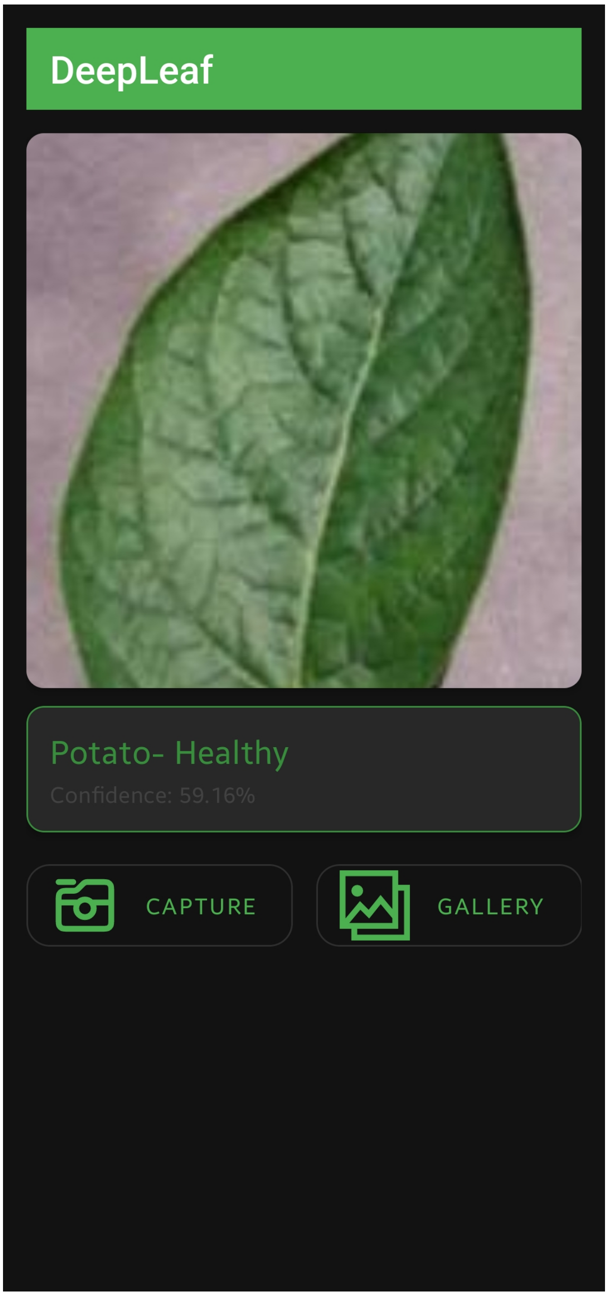
- Model achieved strong accuracy on validation and test datasets.
- Mobile application predicts the disease in under 2 seconds per image.
- Offline prediction capability ensures usability in remote farming areas.

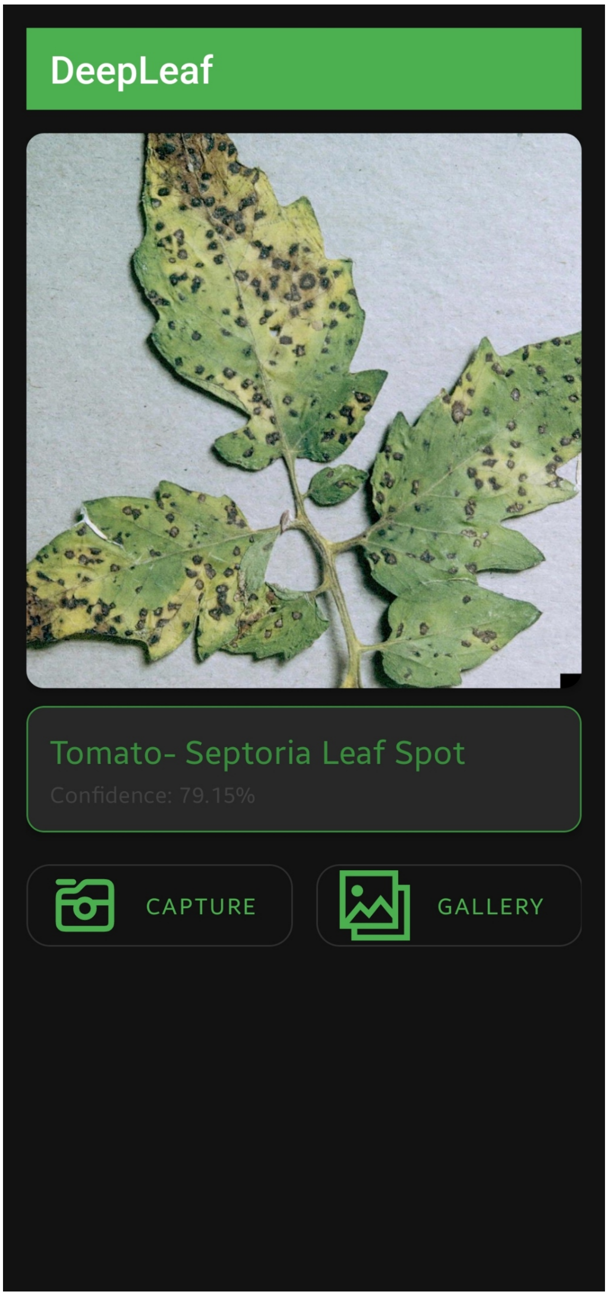
Future Possibilities

- Expand dataset to cover over 100 plant diseases.
- Add real-time video-based disease detection.
- Include multi-language support for farmers across different regions.
- Add a discussion forum within the app for farmers to share their experiences, disease photos, and solutions.
- Allow optional cloud storage of results for agricultural experts to remotely analyze farmer-submitted cases.
- Connect the system with IoT-based agricultural sensors to combine environmental data (humidity, temperature, soil moisture) with visual disease detection for better parametric analysis.

Screenshot Report


DeepLeaf
Potato- Early Blight
Confidence: 78.81%
CAPTURE
GALLERY


DeepLeaf
Potato- Healthy
Confidence: 59.16%
CAPTURE
GALLERY


DeepLeaf
Tomato- Septoria Leaf Spot
Confidence: 79.15%
CAPTURE
GALLERY