Mobile Application for Automated Plant Disease Detection

V. VISHAL[1], T. DURGESH[2], A. ABISHEK[3], Mrs. N. KANAGADURGA[4]
[123] Department of Computer Science and Engineering, E.G.S. Pillay Engineering College (Autonomous), Nagapattinam, Tamil Nadu, India
Corresponding Author: V. Vishal | Email: vishal@egspec.org

How to Cite this Article:
VISHAL, V., DURGESH, T., ABISHEK, A. & KANAGADURGA, N. (2026). Mobile Application for Automated Plant Disease Detection. International Journal of Creative and Open Research in Engineering and Management, <i>02</i>(6).
https://doi.org/10.55041/ijcope.v2i6.239

https://doi.org/10.55041/ijcope.v2i6.239

Abstract—The widespread impact of plant diseases on agricultural yield demands an intelligent, accessible, and real-time detection solution. This paper presents the Mobile Application for Automated Plant Disease Detection, a lightweight smartphone-based system that leverages a fine-tuned MobileNetV2 Convolutional Neural Network (CNN) to classify plant leaf diseases in real time. The system accepts smartphone camera images, applies preprocessing including resizing to 224×224 pixels and normalisation, and classifies the plant health state as Healthy, Early Blight, Late Blight, Leaf Curl, or Powdery Mildew with associated confidence scores. A Flask API backend hosts the trained model and communicates with a React Native mobile frontend to return classification results within 1.2 seconds on a 4G network connection. Firebase Cloud Messaging delivers real-time push notifications and treatment recommendations directly to the farmer's device. The system is deployed entirely on standard Android and iOS smartphones without any specialised hardware, sensors, or wearable devices. Experimental evaluation on the PlantVillage dataset with over 54,000 annotated leaf images demonstrated classification accuracy exceeding 96%, API response latency below 800 milliseconds, and zero dependency on dedicated agricultural equipment. Usability testing with agricultural practitioners confirmed intuitive operation without prior technical training. These results confirm that the proposed application offers an efficient, portable, and institutionally deployable solution for modern precision agriculture.

Keywords—Plant disease detection; MobileNetV2; convolutional neural network; deep learning; precision agriculture; smartphone application; transfer learning; PlantVillage dataset; real-time classification; push notification

I. INTRODUCTION

The global agricultural sector sustains the livelihood of over 500 million smallholder farming households, producing the majority of the world's food supply. Plant diseases pose one of the most significant threats to crop productivity and food security, causing estimated annual losses of up to 40% of global food production. Timely and accurate identification of plant diseases is essential for effective disease management, pesticide application, and crop yield optimisation.

- (1) A fine-tuned MobileNetV2 CNN achieving over 96% accuracy on five plant disease categories using the PlantVillage dataset;
- (2) A lightweight REST API backend supporting real-time mobile image inference with sub-800ms server response time;
- (3) A complete cross-platform mobile application with real-time push notifications, treatment recommendations, and a disease monitoring dashboard;

Conventional disease identification relies on manual inspection by trained agronomists and laboratory pathogen testing. These methods are time-consuming, expensive, and not scalable to the vast agricultural areas cultivated by smallholder farmers in developing nations, including rural India. In most farming communities, expert agronomists are not available on-demand, and by the time disease diagnosis is completed, significant crop damage has already occurred.

The rapid proliferation of smartphone devices among rural farming communities in India and other developing regions has opened a transformative opportunity for deploying AI-powered agricultural tools at the farm level. Convolutional Neural Networks (CNNs) have demonstrated exceptional capability for image-based classification tasks including plant disease identification, achieving accuracy levels that match or exceed expert human diagnosis under controlled conditions.

This paper presents the Mobile Application for Automated Plant Disease Detection, a real-time, smartphone-based disease identification system using a fine-tuned MobileNetV2 CNN. The system captures plant leaf images via smartphone camera, processes them through a cloud-hosted Flask API, and returns classified disease predictions with confidence scores and treatment recommendations within 1.2 seconds. The key contributions of this work are:

- (4) Empirical performance evaluation demonstrating practical viability for field deployment by farmers without technical expertise.

The remainder of this paper is structured as follows: Section II reviews related work in CNN-based plant disease detection. Section III describes the proposed system methodology and algorithmic design. Section IV details the system architecture and implementation. Section V presents experimental results and a comparative discussion. Section VI concludes the paper and outlines directions for future research.

II. LITERATURE REVIEW

Mohanty, Hughes, and Salathé [1] demonstrated the use of deep CNNs for plant disease classification using the PlantVillage dataset. The study trained GoogLeNet and AlexNet models on 54,306 images across 26 plant species and 38 disease categories, achieving 99.35% accuracy under controlled conditions. The paper established the foundational benchmark for AI-based plant disease detection and highlighted challenges of real-world deployment under varying lighting conditions.

Ramcharan et al. [2] proposed a MobileNet-based transfer learning framework for cassava disease detection on mobile devices. The lightweight CNN architecture achieved 93% classification accuracy on field-collected images

Ferentinos [4] presented a smartphone-based deep learning system for real-time plant disease monitoring in field conditions, integrating a custom CNN with a REST API backend for mobile image upload and classification. The model achieved 95.8% accuracy on the PlantVillage test set with robust performance under natural sunlight and varying leaf orientations, demonstrating suitability for field deployment.

Kamilaris and Prenafeta-Boldú [5] presented an IoT-integrated crop health monitoring framework combining smartphone image capture with soil moisture sensors and weather data APIs. The hybrid CNN-LSTM model and demonstrated real-time inference on Android smartphones without cloud dependency, validating on-device deep learning for agricultural applications.

Chen, Liu, and Gao [3] examined transfer learning using VGG16, ResNet50, and InceptionV3 for tomato leaf disease identification. InceptionV3 achieved the highest accuracy of 96.4% for multi-class classification. The research highlighted the importance of data augmentation techniques including random cropping, flipping, and colour jitter for improving generalisation on small agricultural datasets.

achieved superior prediction accuracy compared to image-only approaches, validating the framework on rice and wheat crops in South Asian settings.

A review of the literature reveals a consistent gap: while CNN-based classification achieves high accuracy, existing mobile solutions lack real-time push notification delivery, treatment recommendation generation, and integrated monitoring dashboards accessible to non-technical farming communities. The proposed system addresses these gaps by delivering a complete end-to-end mobile agricultural intelligence platform.

III. PROPOSED METHODOLOGY

A. Problem Formulation

Let $I = \{i_1, i_2, \ldots, i_n\}$ denote the set of n leaf images captured by a smartphone camera, where each image i_k is characterised by pixel matrix $P_k \in \mathbb{R}$^(224×224×3). The disease classification problem is to produce a mapping function $f: I \to C$ such that: (i) $f(i_k) = c_1$ if the leaf is Healthy; (ii) $f(i_k) = c_2$ if Early Blight is detected; (iii) $f(i_k) = c_3$ for Late Blight; (iv) $f(i_k) = c_4$ for Leaf Curl; and (v) $f(i_k) = c_5$ for Powdery Mildew. The Softmax output layer produces confidence scores $\sigma(z)_k$ for each class, where the argmax determines final classification.

B. MobileNetV2 Architecture

MobileNetV2 employs depthwise separable convolutions with inverted residual blocks to reduce the parameter count to 3.4 million, making it suitable for mobile and edge deployment while maintaining high classification accuracy. The architecture uses linear bottlenecks to prevent information loss and shortcut connections to improve gradient flow during training.

Phase 1 – Feature Extraction: The MobileNetV2 backbone pretrained on ImageNet is used as a frozen feature extractor. Input images resized to 224×224×3 pixels are passed through the depthwise separable convolution layers to produce a 7×7×1280 feature map.

Phase 2 – Fine-tuning: A Global Average Pooling layer reduces the feature map to a 1280-dimensional vector. Two fully connected layers (Dense 256 with ReLU, Dense 5 with Softmax) are trained on the PlantVillage dataset subsets for

Phase 3 – Confidence Thresholding: The Softmax output vector is evaluated against a confidence threshold $\theta = 0.75$. If $\max(\sigma(z)) < \theta$, the image is flagged as uncertain and the application prompts the user to re-capture the leaf under better lighting conditions. This threshold was empirically determined through validation set analysis to minimise false positive disease detections.

Phase 4 – Treatment Mapping: The classified disease category is mapped to a treatment recommendation database containing fungicide names, dosage instructions, and cultural management practices specific to each disease category.

C. Complexity Analysis

The MobileNetV2 inference on a single 224×224 image requires approximately 300 million multiply-accumulate (MAC) operations, enabling sub-100ms on-device inference on modern mid-range smartphones. The Flask API server adds approximately 600ms network and preprocessing overhead, resulting in total end-to-end response times of under 1.2 seconds on a 4G mobile connection. Total system complexity is therefore O(1) per image with respect to dataset size, enabling sub-second execution for real-time field use.

the five target disease categories. The top 20 layers of MobileNetV2 are unfrozen in the second training phase to enable task-specific fine-tuning.

IV. SYSTEM ARCHITECTURE AND IMPLEMENTATION

A. Architectural Design

The proposed system adopts a client-server architecture combining a React Native mobile frontend with a Python Flask API backend. The mobile application is developed using React Native 0.72 targeting both Android 10+ and iOS 13+ platforms from a single JavaScript codebase. The backend API is hosted on a Flask 2.3 server running on an Intel Core i5 machine with 16 GB RAM, serving the TensorFlow/Keras-loaded MobileNetV2 model for inference.

The application is structured into five functional modules: (1) Image Capture Module, (2) CNN Disease Detection Module, (3) Disease Analysis Engine, (4) Alert and Recommendation System, and (5) Dashboard and Report Generation Module. Module communication uses RESTful JSON APIs with Base64-encoded image transmission.

B. Image Capture and Preprocessing Module

The Image Capture Module uses the React Native Camera library (RNCamera) to access the device rear camera at a minimum resolution of 640×480 pixels. Captured images are converted to JPEG format and transmitted to the Flask API as a multipart form-data POST request. Server-side preprocessing applies OpenCV-based BGR-to-RGB colour space conversion, bilinear interpolation resizing to 224×224 pixels, and pixel normalisation to the [0, 1] range as required by the MobileNetV2 input pipeline.

C. CNN Disease Detection Module

The MobileNetV2 model is loaded at Flask server startup from a serialised .h5 file and cached in memory for subsequent inference requests. Each preprocessed image is expanded to a batch dimension of 1 and passed through model.predict() to obtain the Softmax probability vector over the five disease classes. The argmax and maximum probability values are extracted and returned to the mobile client as a JSON response containing the classified disease label and confidence score.

D. Alert and Recommendation System

The Alert and Recommendation System delivers real-time disease notifications using Firebase Cloud Messaging (FCM). When the disease confidence score exceeds the threshold $\theta = 0.75$ and the classification is not Healthy, an FCM push notification is dispatched to the farmer's device containing the disease name, confidence percentage, and a recommended treatment action. Visual alert banners are colour-coded: green for Healthy, amber for disease confidence between 0.75 and 0.90, and red for confidence exceeding 0.90, corresponding to potential severity levels.

The treatment recommendation database contains disease-specific entries for all five categories including recommended fungicides (e.g., Mancozeb for Early Blight), application dosages, spray intervals, and cultural management practices. Recommendations are retrieved by disease key lookup and displayed on the mobile application within the same API response cycle.

E. Dashboard and Report Module

The Dashboard and Report Generation Module provides a graphical React Native interface displaying disease detection history using time-series charts via the Victory Native charting library. The dashboard presents colour-coded health state indicators, session statistics including total healthy scans, disease events, and mean confidence scores, and a PDF session summary export feature using React Native Print. This module enables farmers and agricultural supervisors to review crop health patterns following field inspection sessions and generate records for advisory consultations.

V. EXPERIMENTAL RESULTS AND EVALUATION

A. Classification Performance

The MobileNetV2 model was trained on a balanced subset of 25,000 images from the PlantVillage dataset, with 5,000 images per disease class distributed as 70% training, 15% validation, and 15% test splits. Data augmentation was applied during training including random horizontal flipping, brightness variation (±30%), and rotation up to 15 degrees. Training was conducted over 20 epochs using the Adam optimiser at a learning rate of 0.001 with categorical cross-entropy loss.

The trained model achieved 96.7% overall classification accuracy on the held-out test set, with per-class F1-scores ranging from 94.2% for Leaf Curl to 98.1% for Powdery Mildew. The confusion matrix analysis revealed that the predominant misclassification source was between Early Blight and healthy leaf samples with minor spot formations, an expected ambiguity under varying lighting conditions.

B. System Performance

Performance testing was conducted across 500 consecutive API requests on a standard Intel Core i5 server with 8 GB RAM. The mean API response time was 743 milliseconds with a standard deviation of 87 milliseconds. End-to-end latency from image capture to push notification delivery averaged 1.18 seconds on a 4G mobile connection. Server CPU utilisation remained below 55% during continuous inference sessions, confirming practical deployability on low-cost server hardware.

C. Comparative Analysis

Table I presents a feature-level comparison of the proposed system against conventional manual inspection methods and existing software-based plant disease tools.

Feature	Manual Method	Proposed System
Disease Accuracy	Variable	96.7%
Hardware needed	Specialised	Smartphone only
Real-time alerts	None	FCM Push
Treatment advice	Manual	Automated DB
Dashboard	None	In-app charts
Response time	>2 days	<1.2 seconds
Expert required	Yes	No

Table I: Feature Comparison

D. Usability Evaluation

Usability testing was conducted with ten agricultural practitioners, including farmers and agricultural extension officers from Nagapattinam district, Tamil Nadu. Participants were asked to complete a full disease detection workflow (camera capture → API classification → notification receipt) without prior training. All ten participants completed the task successfully within an average of three minutes and forty seconds. Post-task feedback identified the colour-coded dashboard and the treatment recommendation detail view as the most valued usability features.

VI. CONCLUSION AND FUTURE SCOPE

This paper presented the Mobile Application for Automated Plant Disease Detection, a real-time CNN-based plant health classification system delivered through a standard smartphone interface without specialised hardware. The proposed two-module architecture combining a fine-tuned MobileNetV2 Flask API backend with a React Native mobile frontend achieves 96.7% classification accuracy across five disease

Future enhancements identified for subsequent development include: (i) on-device model deployment using TensorFlow Lite for offline operation in areas without internet connectivity; (ii) integration with government agricultural advisory APIs including ICAR and AgriStack India for expanded treatment databases; (iii) multi-disease detection per leaf image to identify categories with sub-1.2 second end-to-end response times. The complete Firebase push notification and treatment recommendation pipeline eliminates the delay between disease occurrence and farmer action, providing practical value in precision agriculture and smallholder farming contexts.

Experimental evaluation confirmed high classification accuracy, sub-second API response times, and positive usability outcomes with real-world farming practitioners. The integration of multi-disease classification, real-time notifications, treatment recommendations, and a monitoring dashboard within a standard mobile application represents a comprehensive advance over existing fragmented or server-dependent agricultural intelligence solutions.

co-occurring infections simultaneously using multi-label classification; (iv) drone-captured aerial image analysis support for large-scale field monitoring; and (v) multilingual interface support for Tamil, Hindi, and regional Indian languages to improve rural farmer accessibility.

REFERENCES

| Mohanty, S. P., Hughes, D. P., and Salathé, M. (2023). Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science, 7, 1419.

| Ramcharan, A., Baranowski, K., McCloskey, P., Ahmed, B., Legg, J., and Hughes, D. P. (2024). Deep Learning for Image-Based Cassava Disease Detection. Frontiers in Plant Science, 8, 1852.

| Chen, J., Liu, Q., and Gao, L. (2024). Visual Tea Leaf Disease Recognition Using a Convolutional Neural Network Model. Symmetry, 11(3), 343.

4] Ferentinos, K. P. (2023). Deep Learning Models for Plant Disease Detection and Diagnosis. Computers and Electronics in Agriculture, 145, 311–318.

5] Kamilaris, A. and Prenafeta-Boldú, F. X. (2025). Deep Learning in Agriculture: A Survey. Computers and Electronics in Agriculture, 147, 70–90.

6] Howard, A. G. et al. (2022). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint, Google Research.

7] Hughes, D. and Salathé, M. (2021). An Open Access Repository of Images on Plant Health to Enable the Development of Mobile Disease Diagnostics. arXiv preprint, EPFL.