Rubber Tree Disease Prediction Mobile Application

Jayaprasad
Department of Computer Science Engineering
KVGCE, Sullia, Dakshina Kannada, India
jayaprasadgowda471@gmail.com

Pruthvi P. C.
Department of Computer Science Engineering
KVGCE, Sullia, Dakshina Kannada, India
pruthviofl@gmail.com

Kishor Kumar K
Department of Computer Science Engineering
KVGCE, Sullia, Dakshina Kannada, India
kishorkajjodi@gmail.com

Ashwini C. K.
Department of Computer Science Engineering
KVGCE, Sullia, Dakshina Kannada, India
ashwinichidgallu@gmail.com

Abstract*—Rubber (Hevea brasiliensis) is a commercially important crop, yet foliar diseases such as Anthracnose, Corynespora Leaf Fall, and Phytophthora-induced Dry Leaf Syndrome significantly reduce plantation yield and tree longevity. Manual diagnosis by agricultural officers is time-consuming, subjective, and often inaccessible to smallholder farmers in remote regions. This paper presents an end-to-end AI-powered mobile application for real-time detection and classification of rubber leaf diseases using deep learning. The system employs a MobileNetV2 convolutional neural network trained via a two-stage transfer learning strategy: an initial warmup phase with a frozen ImageNet backbone followed by fine-tuning of the top 30 layers using cosine-decay learning rate scheduling. Data augmentation, L2 regularization, dropout, and class-weight balancing were applied to improve generalization on the Mendeley Rubber Leaf Dataset containing 1,741 images across four classes. The model achieved a validation accuracy of 99.7% with perfect precision and recall across all classes. The trained model was deployed through a dual-inference architecture: a Python Flask REST API for cloud-based predictions and a quantized TensorFlow Lite model embedded within a cross-platform Flutter mobile application for offline inference, ensuring uninterrupted functionality in rural areas with limited connectivity. The application further incorporates a 90% confidence-threshold gating mechanism to reject ambiguous inputs, multilingual support, an analytics dashboard for disease trend monitoring, and PDF report generation for agricultural record-keeping. The proposed system demonstrates that lightweight deep learning models, combined with intelligent mobile deployment strategies, can provide scalable, accurate, and accessible diagnostic tools for precision agriculture.

***Index Terms*—Deep Learning, Transfer Learning, MobileNetV2, Rubber Leaf Disease, TensorFlow Lite, Flutter, Precision Agriculture, CNN, Image Classification**

I. INTRODUCTION

Natural rubber (Hevea brasiliensis), a perennial tropical crop originating from the Amazon basin, is one of the most economically significant plantation commodities in the world. Global demand for natural rubber has grown steadily, driven by industries ranging from automotive manufacturing to healthcare, with annual production exceeding 14 million metric tonnes. Countries such as Thailand, Indonesia, India, and Malaysia collectively account for a major share of global output, and millions of smallholder farmers depend on rubber cultivation as their primary source of livelihood.

However, rubber plantations are highly susceptible to a range of foliar diseases that directly compromise latex yield and tree longevity. Among the most destructive are Anthracnose, Corynespora Leaf Fall Disease, and Phytophthora-induced Dry Leaf Syndrome. When left undetected during early stages, these diseases escalate rapidly under tropical climatic conditions, resulting in severe economic consequences for farming communities.

Traditional disease diagnosis relies on visual inspection by trained agricultural extension officers, a process that is inherently subjective, time-consuming, and geographically constrained. In many rubber-growing regions, the ratio of extension officers to farmers is critically low, resulting in delayed interventions and preventable crop loss. Furthermore, the visual similarity between early-stage symptoms of different diseases makes accurate manual identification difficult even for experienced practitioners.

The advent of deep learning and computer vision has introduced a paradigm shift in automated plant disease detection. Convolutional neural networks, in particular, have demonstrated remarkable success in learning discriminative visual features from leaf imagery, often achieving diagnostic accuracy surpassing that of human experts. Transfer learning has further reduced the data requirements and computational cost of training high-accuracy models for domain-specific agricultural tasks.

Despite these advances, a significant gap persists between research-grade deep learning models and their practical deployment in real-world agricultural settings. Most existing studies focus exclusively on model accuracy, while neglecting critical deployment challenges such as limited internet connectivity, smartphone computational constraints, multilingual accessibility, and the need for actionable treatment guidance alongside diagnosis.

This paper addresses these limitations by presenting a complete, production-ready system for rubber leaf disease detection that bridges the gap between laboratory research and field deployment. The proposed system employs a MobileNetV2-based classification model trained using a two-stage transfer learning strategy, achieving 99.7% validation accuracy on the

Mendeley Rubber Leaf Dataset across four classes: Anthracnose, Leaf Spot, Dry Leaf, and Healthy. To ensure reliable operation in resource-constrained environments, the system implements a hybrid online-offline inference architecture comprising a Flask-based cloud API for connected scenarios and a quantized TensorFlow Lite model embedded natively within a cross-platform Flutter mobile application for offline inference.

II. Background

A. Rubber Cultivation

The rubber tree is the primary commercial source of natural rubber latex, a polymer of isoprene used extensively in automobile tyres, surgical gloves, footwear, industrial belts, and adhesives. Global natural rubber production exceeds 14 million metric tonnes annually, with the Asia Pacific region accounting for most of the supply. In India, rubber cultivation is concentrated in Kerala, Karnataka, Tripura, and the northeastern states, directly supporting the livelihoods of large numbers of smallholder farming families.

B. Foliar Diseases

Rubber plantations thrive in warm, humid tropical climates that are conducive to fungal pathogens. Anthracnose manifests as necrotic lesions with yellow halo rings and causes premature defoliation. Corynespora Leaf Fall Disease produces irregular brown lesions and can trigger aggressive defoliation. Dry Leaf Syndrome causes marginal yellowing, papery browning, and dark water-soaked bark lesions, often reducing latex output and tree viability.

C. Limitations of Manual Diagnosis

Conventional diagnosis depends on periodic visual inspection by agricultural officers or plant pathologists. This method is inherently subjective and often unreliable for early-stage infections. Limited staffing and vast plantation areas make frequent inspection impractical, and by the time an outbreak is confirmed, the window for effective intervention is often reduced.

D. Convolutional Neural Networks

Convolutional neural networks are the foundation of modern image classification systems. They use learnable convolutional filters to extract spatial features from raw pixel data without manual feature engineering. Architectures such as AlexNet, VGGNet, ResNet, and Inception have shown strong performance across image recognition tasks.

E. Transfer Learning

Training deep CNNs from scratch requires enormous labeled datasets and substantial computational resources. Transfer learning addresses this by initializing a model with weights pre-trained on ImageNet and fine-tuning it on a smaller target dataset. This strategy is especially effective when the target dataset contains fewer than 10,000 samples.

F. MobileNetV2

MobileNetV2 is a lightweight CNN architecture designed for mobile and embedded deployment. It uses inverted residual blocks with linear bottlenecks, delivering a strong trade-off between accuracy and computational cost. Its efficiency makes it suitable for real-time inference on smartphones.

G. TensorFlow Lite

TensorFlow Lite is a framework for deploying machine learning models on mobile phones and edge devices. Through post-training quantization, it reduces model size and inference latency while preserving near-original accuracy. This makes it ideal for offline diagnosis in rural areas with unstable connectivity.

H. Flutter and Flask

Flutter enables cross-platform mobile application development from a single Dart codebase. Its ecosystem supports camera access, file I/O, PDF generation, and TensorFlow Lite integration. Flask is a lightweight Python web framework well suited for REST APIs that host machine learning inference endpoints.

III. Related Work

The application of deep learning to plant disease detection has received considerable attention. Mohanty et al. achieved very high accuracy on the PlantVillage dataset using GoogleNet and AlexNet, establishing the viability of CNNs for leaf disease classification. Ferentinos evaluated multiple CNN architectures on a larger plant disease dataset and reported strong results with transfer learning. Sladojevic demonstrated the usefulness of augmentation for limited agricultural datasets.

In rubber disease detection specifically, Naik et al. used a custom CNN and obtained promising accuracy, while Vasavi et al. applied ResNet-50-based transfer learning. However, many existing studies focus only on classification performance and do not address deployment constraints such as mobile inference, offline operation, multilingual support, confidence calibration, and report generation.

IV. Methodology

A. System Architecture

The proposed system follows a two-tier client-server architecture. The client tier consists of a Flutter mobile application handling image capture, local TensorFlow Lite inference, result visualization, history management, and PDF report generation. The server tier comprises a Flask REST API hosting the full-precision Keras model for cloud-based inference, along with a disease knowledge base that returns detailed diagnostic information.

The application prioritizes online inference when connectivity is available and silently switches to the embedded TFLite model when the connection fails. This ensures uninterrupted usage in field conditions.

B. Dataset Description

The model was trained on the Mendeley Rubber Leaf Dataset, containing 1,741 labeled images distributed across four classes: Anthracnose, Dry Leaf, Healthy, and Leaf Spot. The images were captured under natural field conditions with varying backgrounds, lighting, and leaf orientations. All images were resized to 224 × 224 pixels to conform to the MobileNetV2 input specification.

C. Preprocessing

Pixel values were normalized using the MobileNetV2 preprocessing function. The dataset was partitioned into training and validation subsets using an 80:20 split with stratified sampling to preserve class proportions.

D. Data Augmentation

To reduce overfitting and improve generalization, an on-the-fly augmentation pipeline was applied during training. The pipeline included random horizontal and vertical flips, random rotation up to 20%, random zoom up to 15%, and random contrast adjustment of 10%. These transformations were implemented as preprocessing layers within the model graph.

E. Model Architecture

The classification model uses MobileNetV2 pre-trained on ImageNet as the feature extraction backbone. The original top layers were removed and replaced with a custom classification head consisting of Global Average Pooling, Batch Normalization, a Dense layer with 512 units and Swish activation, L2 regularization, another Batch Normalization layer, Dropout with rate 0.4, and a final Dense output layer with 4 units and softmax activation.

F. Two-Stage Training

Training was conducted in two stages. In Stage 1, the backbone was frozen and only the classification head was trained for 10 epochs using the Adam optimizer with learning rate 0.001. In Stage 2, the top 30 layers of the backbone were unfrozen and the model was fine-tuned for 10 additional epochs using a cosine decay learning rate schedule with a lower initial learning rate.

G. Class Imbalance Handling

The dataset shows moderate imbalance between classes. To compensate, class weights were computed using the balanced strategy and applied during training so that minority classes contributed more strongly to the loss function.

H. Callbacks

Early stopping with patience of 4 epochs was used to prevent overfitting. A model checkpoint callback saved the best-performing model based on validation loss.

I. Mobile Deployment

After training, the best Keras model was converted to TensorFlow Lite format using post-training quantization. This reduced the model size from approximately 9 MB to 2.4 MB. The quantized model was embedded in the Flutter application for offline inference.

J. Backend API

The Flask API exposes a /predict endpoint that accepts image uploads via HTTP POST. The server performs resizing, normalization, and inference, then returns the predicted class label, confidence score, and disease knowledge base information including symptoms, treatment, prevention, severity, and economic impact.

K. Mobile Application

The Flutter application uses an MVVM-style structure with modules for image upload, real-time scanning, prediction history, analytics dashboard, and PDF report generation. It supports English, Kannada, Hindi, Tamil, and Malayalam through a JSON-based translation system.

V. Results

The proposed system achieved a validation accuracy of 99.7% and a perfect F1-score of 1.00 across all four classes on the Mendeley Rubber Leaf Dataset. The two-stage training strategy was critical to this performance, helping the model adapt to the target domain without losing useful ImageNet features.

The quantized TensorFlow Lite model preserved the classification accuracy while reducing the model size significantly. Offline inference latency remained under 150 milliseconds on mid-range smartphone hardware, enabling real-time usage without network dependency.

TABLE I
Dataset Distribution

Class	Images
Anthracnose	301
Dry Leaf	450
Healthy	506
Leaf Spot	484

TABLE II
Performance Summary

Metric	Value
Validation Accuracy	99.7%
F1-score	1.00
Cloud Inference Latency	~400 ms
Offline Inference Latency	<150 ms
TFLite Model Size	2.4 MB

VI. Discussion

The results show that MobileNetV2 with two-stage transfer learning is highly effective for rubber leaf disease classification. The combination of data augmentation, dropout, L2 regularization, batch normalization, and class weighting helped the model generalize well despite the modest dataset size.

The offline capability is especially important for rural deployment, where internet access may be unreliable. The confidence-threshold gating mechanism adds an additional layer of practical reliability by rejecting ambiguous or low-quality inputs before a diagnosis is shown.

However, caution is required when interpreting the high accuracy because the evaluation was performed on a validation split from the same dataset used for training. A fully independent external test set would provide stronger evidence of true generalization across different plantation conditions and camera devices.

VII. Challenges

A. Limited Dataset

The dataset is relatively small and imbalanced, which increases the risk of overfitting. This was addressed using augmentation, regularization, class weighting, and early stopping.

B. Domain Gap

Field images captured by farmers may contain glare, blur, occlusion, and cluttered backgrounds. This domain gap can affect classification accuracy in real-world use.

C. Offline-Online Synchronization

Maintaining consistency between cloud inference and on-device inference is non-trivial because the two model formats behave slightly differently due to quantization.

D. Multilingual Support

Supporting multiple languages requires careful manual translation of disease terminology and system messages to avoid ambiguity or incorrect agricultural advice.

E. Deployment Constraints

Cloud deployment may face issues related to Python compatibility, TensorFlow installation, memory limits, and cold-start latency.

F. Confidence Calibration

Softmax confidence values are not always well calibrated. Choosing a 90% threshold is a practical compromise, but adaptive thresholding may improve user experience in future versions.

VIII. Future Scope

Future work can expand the system to include more rubber diseases such as Powdery Mildew, Abnormal Leaf Fall, Bird's Eye Spot, and Pink Disease. Severity grading can be added to estimate mild, moderate, and severe infection levels. Progression tracking across repeated scans may help monitor disease spread and recovery after treatment.

A geospatial disease mapping feature using GPS coordinates can support plantation-level outbreak monitoring. Drone-based aerial scanning could further extend the system from single-leaf diagnosis to large-scale plantation surveillance. Future versions may also integrate adaptive confidence calibration and dynamic multilingual content management.

IX. Conclusion

This paper presented a complete AI-powered mobile application for rubber leaf disease detection and diagnosis. By combining MobileNetV2 transfer learning, TensorFlow Lite deployment, Flask-based cloud inference, and Flutter mobile development, the proposed system achieves both high accuracy and practical usability in field conditions.

The model achieved 99.7% validation accuracy on the Mendeley Rubber Leaf Dataset and was successfully deployed in a lightweight offline-capable format. The combination of intelligent inference switching, multilingual support, confidence gating, analytics, and report generation makes the solution suitable for real-world precision agriculture.

References

[1] S. P. Mohanty, D. P. Hughes, and M. Salathe´, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, 2016.
[2] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.
[3] S. Sladojevic et al., "Deep neural networks based recognition of plant diseases by leaf image classification," *Computational Intelligence and Neuroscience*, 2016.
[4] A. Naik et al., "Rubber leaf disease classification using convolutional neural networks," 2021.
[5] Vasavi et al., "Transfer learning based rubber leaf disease detection using ResNet-50," 2022.
[6] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. CVPR*, 2018.
[7] Google, "TensorFlow Lite documentation," 2024. [Online]. Available: https://www.tensorflow.org/lite
[8] Google, "Flutter documentation," 2024. [Online]. Available: https://flutter.dev/docs
[9] Pallets Projects, "Flask documentation," 2024. [Online]. Available: https://flask.palletsprojects.com