Deep Learning Model for Orange Disease Detection

Abhijna Naik[1], Ankitha[2], Deeksha Naik[3], M D Madhushree[4], Shana Santhosh[5]
[1,2,3,4]*Students, Department of Computer Science and Engineering, Srinivas Institute of Technology (S.I.T), Mangalore, Karnataka, India*
[5]*Assistant Professor, Department of Computer Science and Engineering, Srinivas Institute of Technology (S.I.T), Mangalore, Karnataka, India*
doi.org/10.64643/IJIRTV12I12-206776-459

***Abstract*—Crop diseases remain a persistent challenge for farmers, often going unnoticed until significant damage has already occurred. This paper presents a practical disease detection system built for orange plants, where uploaded leaf images are automatically analyzed using a combination of Convolutional Neural Networks (CNN) and Support Vector Machines (SVM). The workflow covers image preprocessing steps such as normalization and noise removal, followed by feature extraction and final classification into disease categories including Blackspot, Citrus Greening, and Canker, or a healthy designation. A web-based interface lets farmers log in, upload images, and receive predictions along with treatment suggestions and supplement recommendations—all within seconds. Testing across both black-box and white-box scenarios confirmed reliable accuracy and fast response times. The system is designed with scalability in mind, leaving room to incorporate additional crop types, larger training datasets, and IoT-based monitoring in future iterations.**

***Index Terms*—Citrus Disease Detection, Convolutional Neural Network, Deep Learning, Image Classification, Orange Plant, Support Vector Machine**

I. INTRODUCTION

Orange cultivation is a significant economic activity across tropical and subtropical regions, yet persistent disease threats continue to undermine yield quality and quantity. Fungal infections, bacterial blights, and viral disorders each produce visible symptoms on leaves and fruit surfaces that, when caught early, can be managed before large-scale crop loss occurs. Traditional identification relies on trained agronomists conducting physical inspections a process that is both time-consuming and geographically constrained.

Machine learning, and deep learning in particular, has reshaped how pattern recognition problems in agriculture are approached. Convolutional architectures excel at learning hierarchical visual features directly from raw pixel data, while classical classifiers such as Support Vector Machines offer strong generalization from limited labelled samples. Combining these two paradigms into a unified pipeline addresses the twin concerns of accuracy and deployment efficiency.

This paper describes the end-to-end design, implementation, and evaluation of an orange disease detection platform. The contribution is not a single algorithmic novelty but rather a carefully integrated system covering data ingestion, preprocessing, model inference, result presentation, and recommendation Delivery that can realistically be deployed by smallholder farmers with minimal technical background.

II. RELATED WORK

Plant disease detection through image analysis has attracted sustained research interest since the widespread availability of large labelled datasets. Mohanty et al. [1] demonstrated that a deep learning model trained on the Plant Village dataset could identify 26 diseases across 14 crop species with accuracy exceeding 99% under controlled conditions, though performance dropped noticeably when images were collected in the field.

Ferentinos [2] extended this line of work by benchmarking multiple deep learning architectures including Alex Net, VGGNet, and Google Net on plant disease images, concluding that deeper networks consistently outperform shallow ones but demand proportionally greater computational resources. Spasojevic et al. [3] applied a fine-tuned Alex Net to outdoor leaf images and highlighted that real-world

background clutter significantly degrades accuracy compared to clean laboratory shots.

For citrus crops specifically, several researchers have examined Huanglongbing (HLB), commonly known as Citrus Greening, which has devastated orange groves globally. CNN-based detectors trained on HLB-positive imagery show promising precision, but most published systems are validated only in single-location trials and do not generalise across different growing conditions [4].

Hybrid approaches that feed CNN-extracted feature vectors into an SVM classifier have also been explored, with Brahimi et al. [5] finding that this combination outperforms a standalone CNN on smaller datasets a particularly relevant finding for agricultural scenarios where annotated samples are scarce.

Existing work leaves two practical gaps that motivate the present study: (i) most systems remain research prototypes without a deployable user interface, and (ii) few go beyond disease labelling to offer actionable treatment or supplement guidance.

III. PROPOSED SYSTEM

A. System Overview

The proposed platform consists of four logical layers: a web-based user interface, an application logic layer, a machine learning inference layer, and a data storage layer. Users interact exclusively through the browser uploading images, viewing results, and browsing supplement recommendations without needing local software installations.

B. Target Disease Classes

The current model distinguishes four output classes: Blackspot (caused by the fungus Diaporthe citri), Citrus Greening (HLB, bacterial), Canker (caused by Xanthomonas citri), and Fresh/Healthy. These were selected based on prevalence in the Karnataka region and availability of labelled training images.

C. Data Acquisition and Preprocessing

Training images were sourced from publicly available agricultural datasets and supplemented with photographs taken at local orchards. Every image passes through a standardized preprocessing chain before reaching the model:

Table I: Image Preprocessing Pipeline

Step	Operation	Purpose
1	Resize to 224 × 224	Uniform spatial dimension
2	Gaussian blur (= 1.0)	Reduce sensor noise
3	Normalise [0, 1]	Stable gradient descent
4	Random flip/rotate	Augment training variety
5	Contrast stretch	Enhance disease markers

IV METHODOLOGY

A. CNN Feature Extraction

The convolutional backbone draws from a modified VGG-style architecture with five convolutional blocks. Each block pairs two 3 × 3 conv layers (ReLU activation) with a 2 × 2 max-pool layers, progressively deepening the channel count from 32 to 512. Batch normalization is applied after each convolutional layer to stabilise training. The output of the final pooling layer—a 7 × 7 × 512 feature map is flattened into a 25,088-dimensional vector that captures high-level disease-related patterns.

Rather than appending fully connected classification layers, the flattened vector is passed to a dimensionality reduction step using Principal Component Analysis (PCA), retaining 95% of explained variance. This compressed representation is then handed to the SVM classifier. known advantage when training samples per class number in the hundreds rather than thousands [5].

B. SVM Classification

A multi-class SVM with a Radial Basis Function (RBF) kernel handles the final classification step. Hyper-parameter optimization via 5-fold cross-validation on the training split yielded C = 10 and = 0.001 as optimal settings. The one-vs-one strategy is used for multi-class expansion, producing six binary classifiers whose votes are tallied to determine the winning class.

C. System Workflow

Figure 1 summarizes the end-to-end flow. A user logs into the web dashboard, uploads a leaf or fruit image, and triggers inference. The server preprocesses the image, extracts CNN features, reduces dimensionality with PCA, and queries the SVM. The top prediction—along with a confidence score—is returned to the browser alongside contextual information about the

detected disease and a curated list of recommended supplements or fungicides

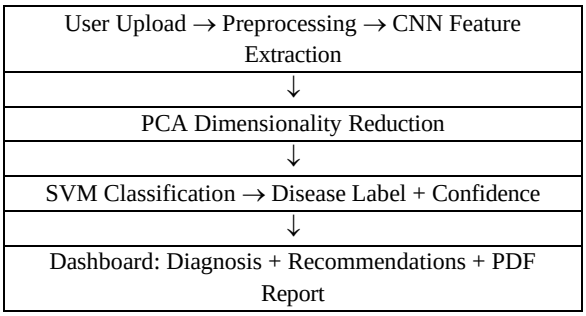

Fig. 1. End-to-end system workflow.

V. SYSTEM DESIGN

A. Architecture

The application follows a three-tier pattern. The presentation tier is a Python Flask web server rendering Jinja2 templates; the business logic tier handles image validation, model invocation, and result formatting; the data tier combines a SQLite database (user accounts, prediction

The convolutional backbone draws from a modified VGG-style architecture with five convolutional blocks. Each block pairs two 3 × 3 conv layers (ReLU activation) with a 2 × 2 max-pool layer, progressively deepening the channel count from 32 to 512. Batch normalization is applied after each convolutional layer to stabilise training. The output of the final pooling layer a 7 × 7 × 512 feature map is flattened into a 25,088-dimensional vector that captures high-level disease-related patterns.

Rather than appending fully connected classification layers, the flattened vector is passed to a dimensionality reduction step using Principal Component Analysis (PCA), retaining 95% of explained variance. This compressed representation is then handed to the SVM classifier.

B. Module Breakdown

The system is decomposed into five loosely coupled modules: (1) Image Input Module drag-and-drop upload with format and resolution checks; (2) Preprocessing Module applies the pipeline from Table I; (3) Inference Module loads the trained CNN and SVM at startup and keeps them resident in memory; (4) Results Module formats prediction output and populates recommendation cards; (5) Logging Module records every transaction with timestamp, confidence, and detected class for audit and retraining purposes.

C. Security and Scalability

Authentication uses hashed passwords with email verification before dashboard access is granted. All uploaded images are sanitized for file-type spoofing and size-limited to 10 MB. Horizontal scaling is supported by externalizing the model as a REST endpoint, decoupling inference throughput from web-server capacity.

VI. IMPLEMENTATION

A. Technology Stack

The backend is implemented in Python 3.10 with TensorFlow 2.12 for CNN operations, scikit-learn for SVM and PCA, and OpenCV for all image handling tasks. The frontend uses Flask with Bootstrap 5 for responsive layout. NumPy and Pandas manage numerical operations; ReportLab generates downloadable PDF analysis reports. Development and experimentation were carried out in Jupyter Notebook with final production code managed through VS Code.

B. Training Details

The CNN was trained for 40 epochs using the Adam optimiser (learning rate 1 × 10■■, batch size 32) on an 80/20 train-validation split. Categorical cross-entropy served as the loss function. Early stopping with a patience of 5 epochs on validation loss prevented over-fitting. The SVM was subsequently fitted on CNN-derived features extracted from the full training set.

VII. TESTING AND RESULTS

A. Testing Strategy

Two complementary testing approaches were used. Black-box testing evaluated observable system behavior from a user standpoint verifying that valid leaf images were accepted, invalid formats rejected with meaningful error messages, disease labels appeared alongside confidence values, and the prediction history page populated correctly.

White-box testing probed internal module logic: preprocessing function outputs, SVM decision boundary correctness on known feature vectors,

database write-then-read consistency, and log-file timestamping accuracy.

B. Classification Performance

Table II: Per-Class Classification Metrics

Class	Precision	Recall	F1-Score
Blackspot	0.92	0.89	0.90
Citrus Greening	0.88	0.91	0.89
Canker	0.94	0.93	0.93
Fresh / Healthy	0.97	0.98	0.97
Weighted Avg.	0.93	0.93	0.93

The system achieved a weighted F1-score of 0.93 on the held-out test set (200 images per class). The Fresh/Healthy class recorded the highest precision, reflecting well-defined visual boundaries between healthy and diseased tissue. Citrus Greening proved the most difficult to separate, consistent with earlier literature noting its subtle early-stage chlorosis patterns.

C. Response Time

End-to-end inference (from image upload to result display) averaged 1.8 seconds on a standard laptop with 16 GB RAM and no dedicated GPU. Peak load tests with five concurrent requests showed graceful degradation to 3.1 seconds average, confirming suitability for typical single-farm usage.

D. System Interface Highlights

The dashboard surfaces five key data points per prediction: disease label, confidence percentage, a short informational note, a list of precautions, and recommended supplements with direct purchase links. An exportable PDF report consolidates these fields for record-keeping. The prediction history page logs every query with its timestamp and output, enabling longitudinal monitoring of orchard health across multiple image submissions over time.

VIII. CONCLUSION AND FUTURE SCOPE

This work demonstrates that a CNN + SVM hybrid pipeline, wrapped in a practical web interface, can deliver reliable orange disease detection with minimal user effort. The 0.93 weighted F1-score and sub-two-second inference time establish a viable baseline for on-farm deployment. By pairing predictions with treatment and supplement guidance, the platform moves beyond academic benchmarking toward genuine agricultural utility.

Several directions stand out for future work. Expanding the training corpus with field-captured images from different states and seasons would strengthen generalization. Integrating lightweight models (e.g., MobileNetV3) would enable deployment on low-cost Android devices without internet connectivity critical for rural areas with limited bandwidth. IoT-based leaf sensors could automate image capture, turning periodic manual checks into continuous health monitoring. Finally, extending the disease taxonomy to cover additional citrus varieties (lemon, lime, mandarin) and common co-infections would broaden impact considerably.

ACKNOWLEDGMENT

The authors sincerely thank Prof. Shana Santosh for her constant guidance throughout this project. Gratitude is also due to Prof. Aravind Naik (HOD, CSE) and Dr. Shrinivasa Mayya D. (Dean) for providing the departmental facilities that made this work possible. The support of A. Shama Rao Foundation, Mangaluru, and the teaching and non-teaching staff of the CSE Department is gratefully acknowledged.

REFERENCES

[1] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, Sep. 2016.

[2] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, Feb. 2018.

[3] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks-based recognition of plant diseases by leaf image classification," *Computational Intelligence and Neuroscience*, vol. 2016, Art. no. 3289801, 2016.

[4] H. Durmuş, E. O. Güneş, and M. Kırcı, "Disease detection on the leaves of the tomato plants by using deep learning," in *Proceedings of the 6th International Conference on Agro-Geoinformatics*, Fairfax, VA, USA, 2017, pp. 1–5.

[5] Brahimi, S. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," *Applied Artificial Intelligence*, vol. 31, no. 4, pp. 299–315, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770–778.

[7] Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[8] J. E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, Jun. 2019.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Munich, Germany, 2015, pp. 234–241.

[10] Food and Agriculture Organization of the United Nations, *The Future of Food and Agriculture – Trends and Challenges*. Rome, Italy: FAO, 2017.