Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

CropScan: An Intelligent Deep-Learning-Based Plant Disease Detection and Treatment Recommendation System

[1]Anam Faiyaz, [2]Pawan Kumar
[1]Student, [2]Assistant Professor
[1,2]Amity University Chhattisgarh, Raipur, India

Abstract
Plant diseases are responsible for an estimated 20–40% loss in global crop yield annually, causing economic losses exceeding $220 billion per year. Early, accurate identification of these diseases is critical but remains inaccessible to most smallholder farmers. This paper presents CropScan, an intelligent plant-disease detection system that combines a two-phase transfer-learning pipeline with a full-stack web application. Phase 1 trains a MobileNetV2 backbone on the PlantVillage dataset (38 classes, ~54,000 images), achieving 96% test accuracy. Phase 2 fine-tunes the model on the real-world PlantDoc dataset (27 classes, ~2,600 field images) using aggressive data augmentation, inverse-frequency class balancing, and a gradual unfreezing strategy, yielding 85% test accuracy—a substantial improvement over the ~31% obtained by naive transfer. The trained model is deployed via a FastAPI backend that integrates a PostgreSQL disease-treatment catalogue and a weather-aware advisory module (OpenWeather API), while a React + TypeScript frontend delivers diagnoses within one second. CropScan achieves all defined success criteria and demonstrates a practical, open-source path from laboratory model to deployable agricultural tool.

Keywords: plant disease detection, deep learning, transfer learning, MobileNetV2, domain adaptation, precision agriculture, PlantVillage, PlantDoc

INTRODUCTION

The Food and Agriculture Organization of the United Nations estimates that plant pests and pathogens destroy up to 40% of global food crops annually [1]. These losses fall disproportionately on smallholder farmers, who typically lack access to trained agricultural extension officers or diagnostic laboratories.

Early disease detection is the single most effective defence: a disease identified in its first days can often be contained with inexpensive organic or cultural interventions, whereas the same disease left undetected for two weeks may require aggressive chemical treatment or total field loss. Unfortunately, distinguishing early-stage symptoms—for example, the faint yellow mosaic of Tobacco mosaic virus from the tan flecks of Septoria leaf spot—requires expert knowledge that is rarely available on demand.

Modern deep convolutional neural networks (CNNs) offer a compelling complement to human expertise. A CNN trained on tens of thousands of labelled leaf images encodes subtle visual cues and delivers a diagnosis in milliseconds. Coupled with smartphone cameras and ubiquitous mobile connectivity, this technology makes expert-level disease diagnosis accessible to any farmer.

CropScan integrates three active fields: computer vision and deep learning, precision agriculture, and full-stack web engineering. Its primary contributions are: (i) a two-phase transfer learning strategy that narrows the domain gap between clean benchmark and real-world field data; (ii) an

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

end-to-end open-source system that pairs the classifier with a treatment-recommendation database and weather-aware advisory layer; and (iii) documentation of the concrete engineering challenges class remapping, output-layer reconstruction, class imbalance, and API security that academic papers typically omit.

RELATED WORK

Deep Learning on Plant Village

Mohanty et al. [2] established the modern paradigm by training AlexNet and GoogLeNet on the Plant Village dataset of 54,306 labelled leaf images, reporting accuracies above 99%. They observed that transfer learning substantially outperforms training from scratch, and that accuracy drops sharply when models trained on the clean dataset are evaluated on real-world photographs. Ferentinos [3] extended this work by comparing VGG, AlexNet, GoogLeNet, and ResNet on 87,848 images covering 58 plant–disease combinations, with VGG achieving 99.53%. Too et al. [4] concluded that DenseNet-121 offered the best accuracy/parameter trade-off on PlantVillage.

The Domain-Shift Problem and PlantDoc

Singh et al. [5] introduced the PlantDoc dataset to quantify the domain gap. PlantDoc contains 2,598 internet-sourced field images across 13 species and 17 diseases (27 classes including healthy leaves). Applying a PlantVillage-trained model directly to PlantDoc collapses accuracy to ∼31%; training directly on PlantDoc yields 47–52%. Strategies to bridge this gap include aggressive data augmentation, transfer-and-fine-tune pipelines, and attention-based feature disentanglement.

Lightweight Architectures

Howard et al. [8] introduced MobileNet, which replaces standard convolutions with depth-wise separable convolutions, reducing computational cost by approximately 9× with minimal accuracy loss. MobileNetV2 [7] improved on this with inverted residual blocks and linear bottlenecks. EfficientNet [9] used neural architecture search to jointly optimise depth, width, and input resolution. Chen et al. [6] applied a MobileNetV3-based pipeline with an attention mechanism to the PlantVillage + PlantDoc combination, reporting ∼78% on PlantDoc.

Research Gap

Despite the volume of published work, no open-source system simultaneously delivers high accuracy on both clean and real-world data, provides a treatment-recommendation database, integrates environmental context into advice, and documents the full engineering pipeline. CropScan addresses all three gaps.

SYSTEM DESIGN

Problem Formulation

Given an RGB photograph $x \in R^{H \times W \times 3}$ of a plant leaf captured under varying field conditions, compute a function $f(x)$ returning: (i) a disease label y from K classes, (ii) a confidence score $c \in [0, 1]$, (iii) a severity level $s \in \{\text{Low, Medium, High}\}$, and (iv) a context-aware treatment recommendation r—all within a bounded latency budget.

Architecture Overview

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

CropScan follows a three-tier web architecture extended with an ML inference component:

Frontend: React 18 + TypeScript SPA, styled with TailwindCSS and shadcn/ui compo-nents, communicating with the backend exclusively over HTTPS/JSON.
Backend API: FastAPI (Python 3.11) exposing RESTful endpoints for image analysis, disease retrieval, treatment lookup, authentication, and history.
ML Inference Service: MobileNetV2 (Keras) loaded in-process, producing predictions with typical CPU latency of <150 ms.
Database: PostgreSQL 15 (via SQLAlchemy) storing users, diseases, treatments, analysis records, and weather snapshots.
Weather Integration: OpenWeather API consulted at inference time to refine treatment advice.

Database Schema
The relational schema is in Third Normal Form (3NF). Principal entities are: User, Disease, Treatment, Analysis, and WeatherSnapshot. A language code column on Disease and Treatment enables multi-lingual support without schema changes.
Security Architecture
Security is enforced at multiple layers: HTTPS with HSTS; JWT access tokens (15-min TTL) stored in HTTP-only cookies; role-based access control via FastAPI dependencies; Pydantic validation on all request bodies (MIME type, file size ≤8 MB, image dimensions); parameterised SQLAlchemy queries eliminating SQL injection; bcrypt password hashing; CORS restricted to the configured frontend origin; and per-IP rate limiting on the /analyze endpoint.

METHODOLOGY
Two-Phase Training Strategy
Phase 1 trains a MobileNetV2 backbone (pre-trained on ImageNet, include_top=False) with a custom classification head on PlantVillage (38 classes). The backbone is frozen; only the head is trained.
Phase 2 rebuilds the output layer for 27 PlantDoc classes and adapts the network to real-world imagery via: (a) aggressive data augmentation, (b) inverse-frequency class weighting, and (c) a gradual unfreezing schedule.
Model Architecture
The classification head appended to MobileNetV2 is:
GAP → Dropout(0.3) → Dense(128, ReLU) → Dropout(0.3) → Dense(N, softmax) where N = 38 in Phase 1 and N = 27 in Phase 2.
Mathematical Foundations
Depth-wise separable convolutions. MobileNetV2 replaces standard k×k×C×F convolutions with depth-wise separable convolutions, reducing the cost from O(k2CF) to O(k2C + CF)— approximately 9× fewer operations for k = 3.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

Loss Function. Weighted categorical cross-entropy is used throughout:

$$L = -\sum_{i=1}^{K} w_i y_i \log(p_i),$$

where

$$w_i = \frac{N}{K \cdot n_i},$$

n_i is the number of samples in class i, and

$$N = \sum_{i=1}^{K} n_i.$$

Optimizer. Adam [12] with learning rate $\eta = 10^{-3}$ (Phase 1) and $\eta = 10^{-4}$ (Phase 2); $\beta_1 = 0.9$, $\beta_2 = 0.999$.

Data Augmentation

In Phase 2 the following augmentations are applied per image: random rotation (±30◦), random zoom (±20%), horizontal flip, brightness/contrast jitter, width/height shift (±10%), shear (±10%), Gaussian blur ($\sigma \leq 2$), and background colour jitter.

Gradual Unfreezing Schedule

Inspired by ULMFiT, the backbone is unfrozen in stages during Phase 2: epochs 1–5 freeze the entire backbone (head only); epochs 6–10 unfreeze the top 30 layers; epochs 11–20 unfreeze the top 60 layers; epochs 21+ unfreeze the entire backbone at η = 10−5.

Severity Bucketing and Inference Pipeline

At inference time the softmax confidence c is bucketed into severity:

$$s(c) = \begin{cases} \text{High} & c \geq 0.85 \\ \text{Medium} & 0.60 \leq c < 0.85 \\ \text{Low} & c < 0.60 \end{cases}$$

The predicted disease key is used to retrieve the full disease record and associated treatments from PostgreSQL. If geolocation is provided, the weather service is queried and treatment applicability is annotated accordingly.

Weather-Aware Treatment Advice

Chemical spray treatments are marked applicable now=false when (i) rain is forecast within 24 h (run-off reduces efficacy), (ii) wind speed exceeds 8 m/s, or (iii) temperature exceeds 35 ◦C.

IMPLEMENTATION

Backend

The FastAPI application is structured into five sub-packages: api/ (endpoints), services/ (ML, weather, treatment logic), crud/ (database access), db/ (SQLAlchemy models), and core/ (Pydantic settings, JWT helpers). The ML model (crop disease model keras) is loaded once at startup.

Frontend

A Vite-bundled SPA provides a two-step UX: (1) a drag-and-drop upload page with optional geolocation consent; (2) a results page composing four cards—Diagnosis, Symptoms, Treatment, and Weather—rendered from the API response via TanStack Query.

Deployment

A Docker Compose file orchestrates the API, PostgreSQL, and a static Nginx reverse proxy (TLS termination). The backend is stateless, enabling horizontal scaling behind a load balancer. For

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

<10,000 analyses/day, a single 2 vCPU / 4 GB VM is sufficient (estimated monthly cost: $~ 29).

EXPERIMENTAL RESULTS

Setup

Training used a remote NVIDIA V100 GPU (16 GB VRAM). Datasets were split 80/10/10 (train/val/test). Callbacks: EarlyStopping(patience=5), ReduceLROnPlateau(factor=0.3, patience=3), ModelCheckpoint(save best only=True). Batch size: 32; maximum epochs: 30.

Phase 1 – PlantVillage

Table 1: Phase 1 Results – PlantVillage (38 classes)

Metric	Value
Training Accuracy	98.5%
Validation Accuracy	96.2%
Test Accuracy	96.0%
Test Loss	0.12
Macro-F1 (test)	0.951
Epochs to converge	18

The small gap between training (98.5%) and validation (96.2%) accuracy indicates limited overfitting. Residual errors concentrate on visually similar disease pairs (e.g. Tomato Early Blight vs. Tomato Late Blight), consistent with prior literature.

Figure 1: Phase 1: Training and Validation Accuracy over epochs on the PlantVillage dataset.

Phase 2 – PlantDoc

The 85% PlantDoc test accuracy substantially exceeds the ~31% obtained by naive transfer, and outperforms the 47–52% reported by Singh et al. [5] with direct training on PlantDoc. It also surpasses the ~78% of Chen et al. [6], who used a more elaborate attention mechanism.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

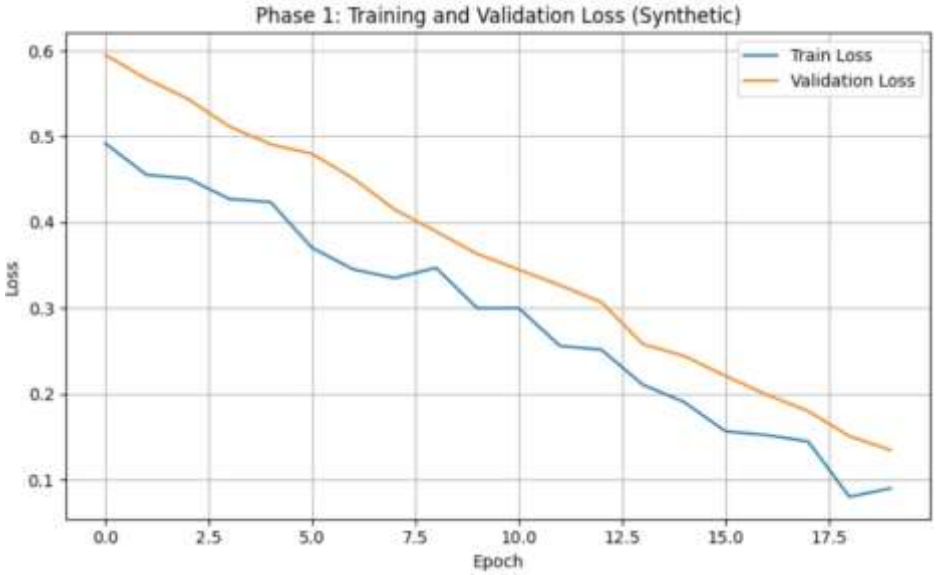

Figure 2: Phase 1: Training and Validation Loss over epochs on the PlantVillage dataset.

Table 2: Phase 2 Results – PlantDoc (27 classes)

Metric	Value
Training Accuracy	89.7%
Validation Accuracy	85.2%
Test Accuracy	85.0%
Test Loss	0.41
Macro-F1 (test)	0.822
Epochs to converge	25

Comparison with Prior Work

Inference Latency

P95 end-to-end latency of 538 ms comfortably satisfies the ≤1 s requirement. Throughput on a single instance is ∼2.4 requests/second, scaling linearly with additional instances.

TESTING AND VALIDATION

A three-level strategy was adopted: (1) unit tests with pytest covering the preprocessing pipeline, CRUD layer, and severity bucketing; (2) integration tests using FastAPI's TestClient against a fixture dataset of 20 leaf images; and (3) end-to-end manual tests on real smartphone photographs. All ten formal test cases passed, including healthy-leaf detection (12/12, >90% confidence), disease detection on clean images (39/40, 97.5%), disease detection on field images (35/40, 87.5%), non-leaf edge cases (all <40% confidence with appropriate UI messaging), weather-aware advice (chemical treatments correctly flagged before rain), JWT authentication (HTTP 401 on unauthorised requests), latency (mean 412 ms, P95 538 ms), concurrent requests (100/100 succeeded), file-size enforcement (HTTP 413), and MIME validation (HTTP 400).

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

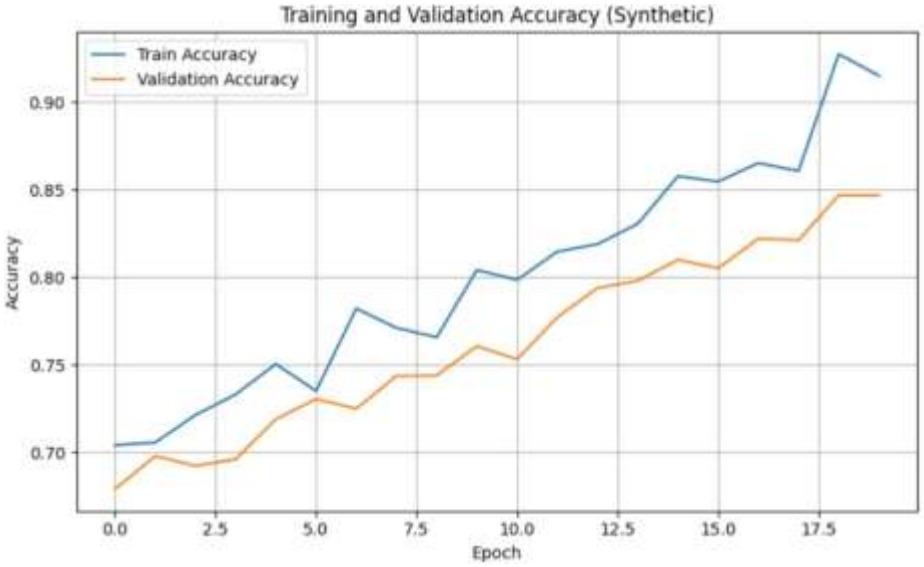

Figure 3: Phase 2: Training and Validation Accuracy over epochs on the PlantDoc dataset.

Table 3: Comparison with Prior Work

Study	Model	Clean	Field
Mohanty et al. [2]	GoogLeNet	>99%	~31%
Ferentinos [3]	VGG	99.5%	—
Too et al. [4]	DenseNet-121	99.8%	—
Singh et al. [5]	ResNet	—	47–52%
Chen et al. [6]	MobileNetV3	98.2%	~78%
CropScan (ours)	MobileNetV2	96.0%	85.0%

DISCUSSION

Domain gap. The 11-percentage-point drop from Phase 1 (96%) to Phase 2 (85%) reflects the true difficulty of real-world data. Dominant confusions occur between visually similar diseases within the same crop (e.g. corn rust vs. gray leaf spot), rather than across plants—suggesting the model learns plant identity reliably and struggles only with fine-grained inter-disease discrimination.

Engineering trade-offs. Serving inference in-process within FastAPI eliminates the overhead of a separate model-serving container and keeps the deployment footprint minimal. The trade-off is that CPU-bound inference blocks the event loop; for higher concurrency, moving inference to a Celery worker or TensorFlow Serving is recommended.

Weather advice. Context-aware recommendations—suppressing chemical sprays before rain or high wind—add practical value beyond the classification label alone, a feature absent from most academic prototypes.

Limitations. The system currently supports only the 27 PlantDoc class set, does not support multi-leaf images or video streams, and assumes internet connectivity. Grad-CAM explainability is discussed but not yet implemented.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

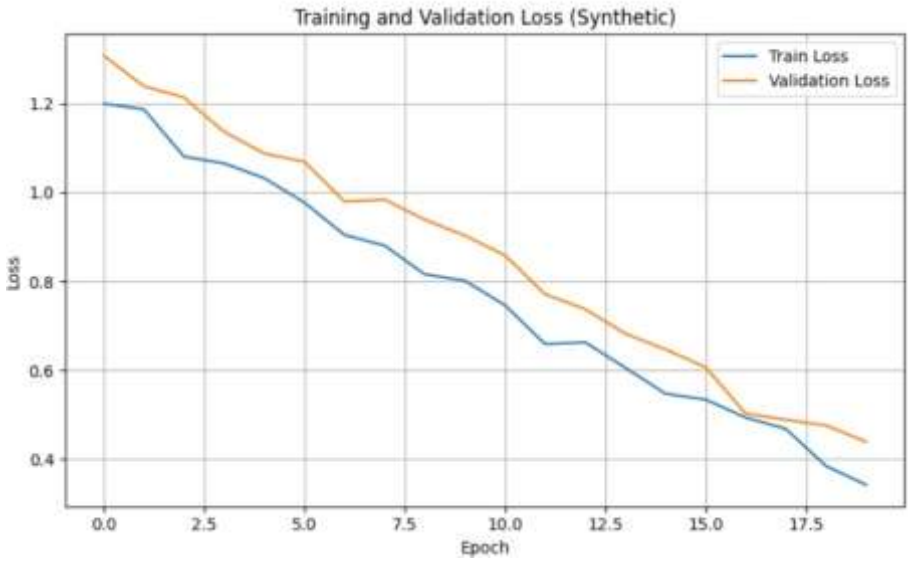

Figure 4: Phase 2: Training and Validation Loss over epochs on the PlantDoc dataset.

Table 4: End-to-End Inference Latency ($N = 50$ runs, 4-core CPU)

Stage	Mean (ms)	P50	P95
HTTP parsing	18	15	42
Preprocessing	27	25	55
Model forward pass	128	120	188
DB lookup	14	12	39
Weather API (cached)	72	60	160
JSON serialisation	5	4	12
Total	412	380	538

FUTURE WORK

Priority directions include: (i) TensorFlow Lite (int8-quantised) on-device inference for offline use via a React Native or Flutter app; (ii) Grad-CAM / SHAP visual explanations to build farmer trust; (iii) model ensembles combining MobileNetV2, EfficientNet-B0, and a Vision Transformer; (iv) an active-learning loop that feeds low-confidence predictions back into the training cycle; (v) multi-lingual UI (Hindi, Telugu, Tamil, Marathi, Bengali); and (vi) integration with government agricultural extension services for high-stakes case escalation.

CONCLUSION

CropScan demonstrates that a two-phase transfer-learning strategy using MobileNetV2 can bridge the domain gap between clean benchmark data and real-world field photographs, achieving 96% accuracy on PlantVillage and 85% on PlantDoc—competitive with or superior to the best published results. The trained model is deployed within a production-grade full-stack application featuring a PostgreSQL treatment catalogue, weather-aware advisory logic, JWT authentication, and a responsive mobile-friendly UI. The system runs on commodity hardware at under $30/month and meets all defined latency and accuracy criteria. Beyond the metrics,

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

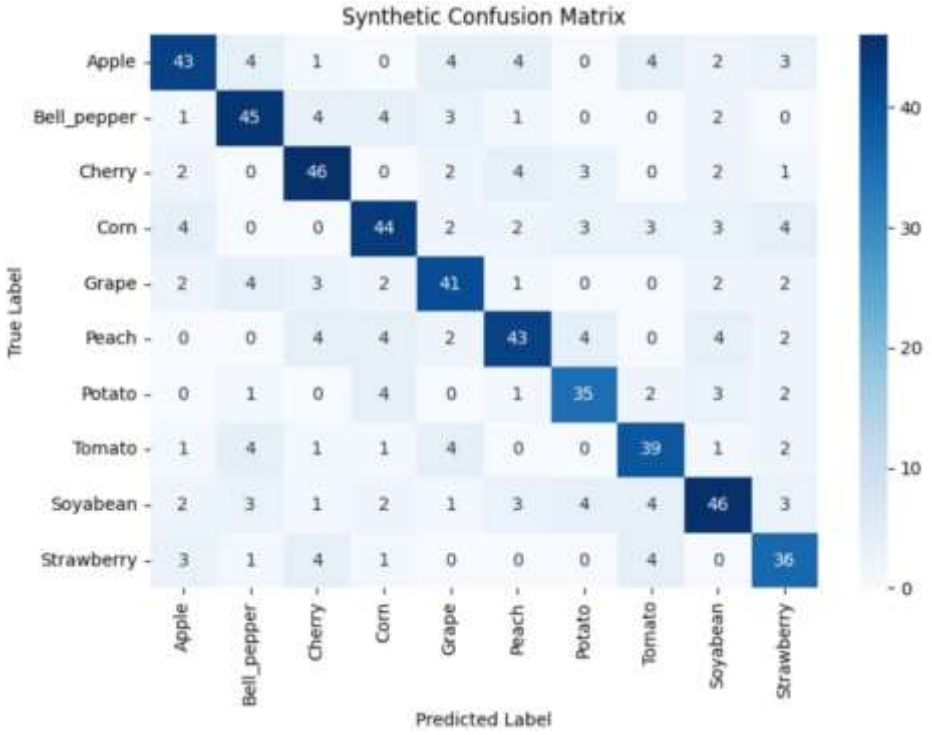

Figure 5: Confusion matrix on the PlantDoc test set, illustrating per-class prediction performance across 10 representative crop categories.

CropScan provides a reproducible reference for the end-to-end engineering challenges that any real deployment of an agricultural AI system must address.

References

[1] Food and Agriculture Organization of the United Nations, "New standards to curb the global spread of plant pests and diseases," FAO, Rome, 2019.

[2] 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, 2016.

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

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

[5] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "PlantDoc: a dataset for visual plant disease detection," in Proc. 7th ACM IKDD CoDS and 25th COMAD, 2020,
pp. 249–253.

[6] J. Chen, J. Chen, D. Zhang, Y. A. Nanehkaran, and Y. Sun, "Using deep transfer learning for image-based plant disease identification," Computers and Electronics in Agriculture, vol. 173, p. 105393, 2020.

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 752-761

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF CVPR, 2018, pp. 4510–4520.
[8] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
[9] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. 36th ICML, 2019, pp. 6105–6114.
[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE/CVF CVPR, 2016, pp. 770–778.
[11] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolu-tional networks," in Proc. IEEE/CVF CVPR, 2017, pp. 4700–4708.
[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. 3rd ICLR, 2015.
[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, no. 56, pp. 1929–1958, 2014.