Reference:

[1] M. Z. Hasan et al., “A Leaf Disease Classification Model in Betel Vine Using Machine Learning Techniques,” in Proc. 2nd Int. Conf. on Robotics, Electrical and Signal Processing Techniques (ICREST), IEEE, Dhaka, Bangladesh, 2021, pp. 362–366, doi: 10.1109/ICREST51555.2021.9331142. ↩ back

[2] M. M. Hasan et al., “An Approach to Identify Diseases in Betel Leaf Using Deep Learning Techniques,” in Proc. 4th Int. Conf. on Sustainable Technologies for Industry 4.0 (STI), IEEE, Dhaka, 2022, doi: 10.1109/STI56238.2022.10103348. ↩ back

[3] R. H. Hridoy et al., “Early Recognition of Betel Leaf Disease Using Deep Learning with Depth-wise Separable Convolutions,” in Proc. IEEE Region 10 Symposium (TENSYMP), 2021, pp. 1–7, doi: 10.1109/TENSYMP52854.2021.9551009. ↩ back

[4] E. A. Taufik et al., “Efficient Leaf Disease Classification and Segmentation Using Midpoint Normalization Technique and Attention Mechanism,” in Proc. IEEE Int. Conf. on Image Processing (ICIP), 2025, pp. 2091–2096, doi: 10.1109/ICIP55913.2025.11084601. ↩ back

[5] R. H. Hridoy, “Betel Leaf Disease Recognition Using Deep Learning,” B.Sc. Thesis, Daffodil Int. Univ., Dhaka, Bangladesh, 2021. ↩ back

[6] M. F. Ahmed et al., “Leveraging Pre-trained Models within a Semi-Supervised and Explainable AI Real-Time Framework: A Pioneering Paradigm for Betel Leaf Disease Detection,” preprint, 2024. ↩ back

[7] K. K. Biswas et al., “Deep Learning based Betelvine Leaf Disease Detection (Piper Betle L.),” in Proc. Int. Conf. on Computer Communication and Control (ICCCA), IEEE, 2020. ↩ back

[8] C. N. Maitipe et al., “Betel Plant Tech: Betel Disease Forecasting System and Finding Marketplace,” in Proc. 13th Int. Conf. on Computing Communication and Networking Technologies (ICCCNT), IEEE, 2022, doi: 10.1109/ICCCNT54827.2022.9984376. ↩ back

[9] F. Lubis et al., “Classification of Types of Betel Leaves (Piper Betel Linn) Using an Android-Based Neural Network Backpropagation,” in Proc. 7th Int. Conf. on Electrical, Telecommunication and Computer Engineering (ELTICOM), IEEE, 2023, doi: 10.1109/ELTICOM61905.2023.10443107. ↩ back

[10] R. K. Veeresha et al., “Deep Learning-Based Classification of Areca Nut Yellow Leaf Disease with ResNet-50 CNN,” in Proc. Int. Conf. on Recent Advances in Science & Engineering Technology (ICRASET), IEEE, 2024, doi: 10.1109/ICRASET63057.2024.10895610. ↩ back

[11] B. Dey et al., “Assessing Deep Convolutional Neural Network Models and Their Comparative Performance for Automated Medicinal Plant Identification from Leaf Images,” Heliyon, vol. 10, 2024, doi: 10.1016/j.heliyon.2023.e23655. ↩ back

[12] E. do Rosário and S. M. Saide, “Segmentation of Leaf Diseases in Cotton Plants Using U-Net and a MobileNetV2 as Encoder,” in Proc. Int. Conf. on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD), IEEE, 2024, doi: 10.1109/ICABCD62167.2024.10645269. ↩ back

[13] J. Hu et al., “Enhancing Betel Nut Pest and Disease Identification in Hainan, China with Swin Transformer–YOLOv5: A Machine Learning-Based Approach for Improved Precision,” in Proc. IEEE 6th Int. Conf. on Pattern Recognition and Artificial Intelligence (PRAI), 2023, pp. 539–544, doi: 10.1109/PRAI59366.2023.10332045. ↩ back

[14] S. Amritraj et al., “An Automated and Fine-Tuned Image Detection and Classification System for Plant Leaf Diseases,” in Proc. Int. Conf. on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI), IEEE, 2023, doi: 10.1109/RAEEUCCI57140.2023.10134461. ↩ back

[15] A. Baurai et al., “Leaf Disease Detection Using KNN,” in Proc. 4th Int. Conf. on Technological Advancements in Computational Sciences (ICTACS), IEEE, 2024, doi: 10.1109/ICTACS62700.2024.10840461. ↩ back

[16] S. Abinaya and G. Sivakamasundari, “Potato Leaf Disease Detection Using Deep Learning,” in Proc. Int. Conf. on Intelligent Computing and Control Systems (ICICCS), IEEE, 2025, doi: 10.1109/ICICCS65191.2025.10984928. ↩ back

Table:

Title	Dataset and URL	Description	Method	Accuracy/metric	Research question	pros	cons	citation
A Leaf Disease Classification Model in Betel Vine Using Machine Learning Techniques	Authors' betel-vine dataset (Google Drive link: **https://drive.google.com/drive/folders/1KnNB7hiMB16Waonh6Mrazlgwkssv6M5M**	1275 images; 2 classes (bacterial leaf spot, stem leaf); 80/20 split	SVM, Logistic Regression, KNN, Random Forest after feature extraction (GLCM + statistical features).	SVM (RBF kernel) — 83.69 %, Precision 82.4 %, Recall 85.6 %, F1 83.9 %. LR—81.7 %, RF—75.2 %, KNN—66.7 % (Table I, p. 364).	Can classical ML methods (SVM, RF, KNN, LR) detect and classify betel-vine leaf diseases using low-cost imaging and handcrafted features?	Uses real-field images; evaluates four ML models; detailed metrics (confusion matrices Table II, ROC Fig. 5); public dataset link.	Only two classes; moderate accuracy; no deep-learning comparison; limited generalization study..	[1]
An Approach to Identify Diseases in Betel Leaf Using Deep Learning Techniques	Authors' betel-leaf dataset	5,792 images, 2 classes: Bacterial Leaf Spot, Stem Leaf; images captured in Rajshahi (Bangladesh), resized 299×299; CLAHE for contrast + GMM segmentation; split: Train 3,851 / Val 1,283 / New Test 658.	Transfer-learned CNNs (VGG16, VGG19, ResNet50, AlexNet, InceptionV3); classic ML baselines (SVM, Logistic Regression).	InceptionV3—94.83% (test, best); VGG16—91.20%; VGG19—90.77%; ResNet50—60.02%; AlexNet—78.24%; SVM—63%; LR—58%.	(1) Which pre-trained CNN best classifies two common betel-leaf diseases? (2) Do CLAHE + GMM preprocessing steps improve performance? (3) Can a lightweight, mobile-supportive pipeline deliver high test accuracy on a separate set?	Clear pipeline (Fig. 6); separate new test set (658); full metric table (Acc/Spec/Prec/Rec/F1); ROC AUC ≈ 0.948 for InceptionV3; practical capture protocol	Only 2 classes; single-region data; no public dataset URL; limited deployment analysis.	[2]
Early Recognition of Betel Leaf Disease using Deep Learning with Depth-wise Separable Convolutions	Authors' dataset	10,662 images; 4 classes; train 8,600 / val 1,031 / test 1,031. Images captured via smartphone, resized to 300 × 300 px, mixed backgrounds.	BLCNN (Betel Leaf CNN) — 3 × Depth-wise Separable Conv layers + 2 × Fully Connected layers; activation: Swish; optimizer: Adam; loss: Categorical Cross-Entropy; compared with standard CNN (ReLU).	BLCNN – 96.02 % (test); baseline CNN – 89.53 %. Per-class: Healthy Acc 98.64 %, Foot Rot 97.87 %, Leaf Rot 97.77 %, Misc 97.77 %. (Tables II–III, p. 5).	(1) Can depth-wise separable convolutions improve early detection of betel-leaf diseases while reducing parameters? (2) Which activation (ReLU, SELU, Swish) yields optimal performance? (3) Can early disease signs (≤ 4 days) be recognized for real-time use?	Custom 10 K image dataset; clear per-class metrics; Swish + depth-wise convs ↑ 6.5 % accuracy vs standard CNN; reduced parameters/training time; class-wise confusion matrix	Dataset not public; limited to 2 disease types + 2 aux classes; no deployment or XAI analysis; regional data bias.	[3]
EFFICIENT LEAF DISEASE CLASSIFICATION AND SEGMENTATION USING MIDPOINT NORMALIZATION TECHNIQUE AND ATTENTION MECHANISM	Betel Leaf Image Dataset from Bangladesh — *Mendeley Data* DOI: **10.17632/g7fpgj57wc.2**	1,000 total images, 4 classes — *Healthy, Dried, Bacterial Leaf Disease, Fungal Brown Spot Disease*; 250 images/class (Fig. 1 p. 2); 225×225 px resized; 80 / 20 split for training/testing; preprocessed with MPN (Midpoint Normalization) vs CLAHE.	SE-ConvNet (Squeeze-and-Excitation CNN with MPN) for classification; U-Net + SE Attention for segmentation. Compared against MobileNetV2, VGG16, CNN, and vanilla U-Net	SE-ConvNet – 93 %, U-Net + SE Attention – Dice 72.44 %, IoU 58.54 %	(1) Can MPN preprocessing enhance feature quality compared to CLAHE and improve classification? (2) Do SE attention blocks improve lightweight CNN performance for both classification and segmentation? (3) How can attention help small models remain interpretable and efficient for agricultural vision tasks?	High accuracy with low computational cost (28 MB model vs 64 MB VGG16); visual GradCAM++ heatmaps confirm disease-region focus ; interpretable attention; works on small datasets; uses public data.	Only 1 K images; single attention head; limited crop diversity; real-time performance not tested.	[4]
BETEL LEAF DISEASE RECOGNITION USING DEEP LEARNING	**Daffodil International University Dataset**	9,983 total images, divided into 4 classes: Foot Rot (2,101), Leaf Rot (2,787), Miscellaneous (683), Healthy (4,412); color images, resized to 224×224 and 299×299 px; used 80/20 train-test split	BLCNN (custom CNN) and three pre-trained models — VGG16, InceptionV3, EfficientNet-B0	EfficientNet-B0—96.77%, InceptionV3—94.55%, VGG16—92.78%, BLCNN—89.44%	Can CNN architectures effectively classify betel-leaf diseases using a large, field-collected dataset? Which pretrained model yields the best trade-off between performance and computational cost?	Large, balanced dataset; compares four CNN models; includes class-wise metrics (Precision, Recall, F1, Accuracy); EfficientNet-B0 gives best result; discusses environmental & societal impact.	Only two disease types; no deployment/app; dataset not publicly available.	[5]
Leveraging pre-trained models within a semi-supervised and explainable AI RealTime framework: A pioneering paradigm for betel leaf disease detection	Betel leaf image dataset from Bangladesh — **Mendeley Data (DOI: 10.17632/g7fpgj57wc.2)**	Original: 1,000 images (4 classes: Healthy, Bacterial Leaf, Dried Leaf, Fungal Brown Spot). Augmented: 2,589 images. Split (per Table 2, augmented): Train/Test counts ≈ Healthy 578/65, Bacterial 590/66, Dried 583/65, Fungal 577/65. Input 224×224; 80% train, 20% test/val.	Supervised TL: DenseNet-201, InceptionV3, EfficientNetV2-B0; SSL: FixMatch, MixMatch, MeanTeacher (all integrated with DenseNet-201); XAI: SmoothGrad, Vanilla Saliency, Grad-CAM++, Faster Score-CAM; simple Flask/React real-time app.	DenseNet-201—99.23% (supervised); FixMatch—98.00% (30% labels); MixMatch—91.27%; MeanTeacher—81.00%.	(1) Which pre-trained CNN performs best for supervised betel-leaf classification? (2) Can SSL with limited labels approach supervised accuracy? (3) Do XAI maps make predictions interpretable for end-users? (4) Can the pipeline run real-time in a simple web app?	High supervised accuracy; SSL reaches 98% with only 30% labels; multiple XAI visualizations; working web app; threshold/label-ratio study; shows generalization on external mango-leaf set.	Region-specific dataset; modest scale; DenseNet-201 is relatively heavy; data "on request"; limited field robustness reporting.	[6]

Deep Learning based Betelvine leaf Disease Detection (Piper BetleL.)	Custom betelvine leaf dataset	1,014 images; 3 classes — Anthracnose 358, Phytophthora 456, Healthy 200; split 810/150/54 for train/val/test; images resized to 256×256; manual VIA annotation for diseased regions	Mask-RCNN with ResNet-50 + FPN, modified with Swish activation and RAdam optimizer; compared to Faster-RCNN and vanilla Mask-RCNN	Mask-RCNN(ResNet50+FPN)—mAP@IoU=0.7 = 0.8407 (Prec 0.7901; Rec 0.7814; F1 0.7857). Baselines: Faster-RCNN—mAP 0.7432; vanilla Mask-RCNN—mAP 0.8311.	Can an instance-segmentation pipeline accurately detect and localize betelvine leaf diseases in real images, and do Swish + RAdam tweaks improve performance vs. standard detectors?	Localizes lesions; improved mAP vs. baselines; small-GPU friendly; clear evaluation setup.	Modest dataset size; 3 classes only; results at a single IoU threshold; no public dataset link.	**[7]**
Betel Plant Tech: Betel Disease Forecasting System and Finding Marketplace	Custom Sri Lankan betel leaf dataset (no public URL; images collected by authors + Kaggle supplement)	1,000+ leaf images, 4 disease types (*Bacterial Leaf Blight, Brown Spots, Leaf Rot, Pest Damage*). Separate datasets for yield and disease spread: yield dataset = 144 records (2010–2021; 3 districts), weather dataset = 11 years (wind, humidity, temperature). Split ≈ 70 / 30 train-test.	ResNet-34 (disease ID), Sequential Model (propagation level classification), Random Forest Regression (yield forecast), Decision Tree Regression (disease spread prediction).	ResNet-34 — 92 %, Sequential Model — 92 %, Random Forest Regression — 98 %, Decision Tree Regression — 97 % (see Figs 8–11 pp. 5–6).	(1) Can a mobile ML system detect, classify, and predict betel-leaf diseases and their spread? (2) Can environmental and yield data be integrated into an app that supports disease mitigation and market access for farmers?	End-to-end smartphone app; multi-module design (image classification + forecasting + marketplace); integrates weather and yield data; high accuracy across tasks.	Dataset not public; small samples per task; lacks real-time latency analysis; no cross-region testing.	**[8]**
Classification of Types of Betel Leaves (Piper Betel Linn) Using An Android-Based Neural Network Backpropagation	Authors' custom dataset (not publicly released)	625 total images; 5 classes of betel leaves — *Green (Water Rider), Black, Irian, Kerakap, Serimaja*; 100 images per class; split 75% (train) / 25% (test); captured with 14 MP Realmi smartphone camera under outdoor lighting	Backpropagation on Neural Network (BPNN); features: HSV color, GLCM texture, and shape metrics (metric, eccentricity); trained 30 epochs, batch 20, learning rate 0.001 (Adam optimizer).	BPNN – 91.4 % overall (Precision ≈ 93–100 %, Recall ≈ 84–100 % per class); confusion matrix in Table III (p. 175).	Can a low-cost, mobile-ready Backpropagation NN classify *five morphologically similar betel types* using HSV, texture, and shape features?	imple, low-compute model; 5-class dataset; Android-ready; clear preprocessing pipeline; interpretable features.	Small dataset; sensitivity to lighting; no disease detection; lacks cross-device testing.	**[9]**
Deep Learning-Based Classification of Areca Nut Yellow Leaf Disease with ResNet-50 CNN	Custom *Arecanut Leaf Dataset* (authors' collection; no public URL)	3 classes — *Healthy Arecanut Leaf, Yellow Leaf Disease, Other Leaves*; split 70% train / 15% validation / 15% test; resized to 224×224 px; balanced number of samples (≈60 healthy, 54 diseased, 42 others in test set); color RGB images preprocessed using Keras	Transfer Learning (ResNet-50) pretrained on ImageNet; removed top layers, added Global Average Pooling + Dense(1024 ReLU) + Dense(3 Softmax); Adam optimizer, categorical cross-entropy loss; trained for 6 epochs, batch size 32.	ResNet-50—Training Acc 98.72%, Validation Acc 99.35%, Test Acc ≈ 99% (Precision 0.97–1.00, Recall 0.96–1.00, F1 0.98–1.00) (Fig. 7, p. 6).	(1) Can a pre-trained CNN (ResNet-50) reliably classify *Areca nut yellow leaf disease* against healthy and other leaves? (2) How effective is transfer learning with limited data for precision agriculture?	Excellent accuracy (≈99%); balanced dataset; robust evaluation (precision, recall, F1, confusion matrix Fig. 8 p. 6); distinguishes diseased vs. non-diseased clearly; suitable for early diagnosis.	Small dataset (hundreds of images); limited to one region; minor misclassification (2 YLD → Healthy); no mobile deployment tested.	**[10]**
Assessing deep convolutional neural network models and their comparative performance for automated medicinal plant identification from leaf images	Public + Field image dataset (Kaggle + field photos); data available on request	5878 total images, 30 species, 20 families. Two versions: PI (public, plain background) and PFI (public + field, complex backgrounds). Split: 85% train / 5% validation / 10% test, resized 224×224 px.	Seven DCNN models: VGG16, VGG19, ResNet50V2, Xception, InceptionV3, InceptionResNetV2, DenseNet201 (best). Each trained with Adam optimizer, categorical cross-entropy, batch size 32, 20 epochs	DenseNet201—99.64% (PI); DenseNet201—97% (PFI). (2nd best: ResNet50V2—98.6% PI, 95.1% PFI)	1) Which CNN architecture best identifies medicinal plant species from leaf images across plain vs. field settings? (2) How does dataset complexity (background variation) affect accuracy? (3) Can DCNNs generalize across inter-/intra-family variation?	Evaluates 7 state-of-the-art DCNNs under same pipeline; strong metrics (Acc, Prec, Recall, F1); includes inter- vs. intra-family analysis (Figs. 6–10); tested both public and field data; open code repo (GitHub link).	Dataset not fully public; modest species coverage (30 spp.); relies solely on 2D leaves (no stems/flowers); slight misclassification between *Ficus auriculata* and *Moringa oleifera* (Fig. 6).	**[11]**
Segmentation of Leaf Diseases in Cotton Plants Using U-Net and a MobileNetV2 as Encoder	Authors' custom cotton-leaf dataset	2,320 images total (original 387 + 6× data-augmented copies); 3 segmentation classes — *Damage, Leaf, Background*; masks labeled via Labelbox; images resized 256×256 px	U-Net with MobileNetV2 encoder (α = 0.35, ImageNet weights); Dice-loss objective; Nadam optimizer (lr = 1e-4); Softmax output; metrics: Precision, Recall, IoU, Dice.	MobileNetV2 U-Net — Dice 98.61 %, IoU 97.01 %, Precision 99.56 %, Recall 94.91 %	(1) Can a lightweight U-Net + MobileNetV2 achieve high-accuracy segmentation of cotton leaf diseases on limited data? (2) How does it compare with classical or heavier encoder–decoder models?	Excellent Dice / IoU; very low loss (0.014); 5-fold validation; detailed per-epoch metrics (Tables IV–VI p. 4); qualitative results show strong mask overlap; mobile-deployable encoder.	No public dataset; region-specific (Mozambique); assumes ICT literacy for field users; limited disease-type annotation.	**[12]**

Enhancing Betel Nut Pest and Disease Identification in Hainan, China with Swin Transformer-YOLOv5: A Machine Learning-Based Approach for Improved Precision	Hainan Betel Nut Pest and Disease Dataset (authors' collection; no public link)	472 original field images, augmented to 839 total via nearest-neighbor interpolation + random transformations (rotation, noise, scaling). 7 pest/disease classes — *Big Spot Disease (97)*, *Spiny Whitefly (28)*, *Sooty Mould (15)*, *Bacterial Leaf Spot (95)*, *Small Spot (57)*, *Algal Spot (145)*, *Middle Spot (35)*. All resized to 800×800 px	YOLOv5s + Swin Transformer (hybrid detection model); trained with transfer learning on COCO-pretrained weights using PyTorch 1.7; optimizer = Adam; loss = CIoU-based. Compared to baseline YOLOv5s.	YOLOv5s+SwinTransformer—Precision 0.684 (+1.6 %), Recall 0.512 (−1.7 %), mAP 0.569 (+6.1 %); baseline YOLOv5s—mAP 0.508 (Fig. 8, p. 5).	(1) Can integrating Swin Transformer attention improve YOLOv5 accuracy on limited, field-collected betel-nut datasets? (2) How does the hybrid model's precision, recall, and mAP compare to vanilla YOLOv5? (3) Is the approach suitable for future on-device (microcomputer) deployment for pest management?	+6.1 % mAP improvement; faster convergence (Fig. 7 p. 5); handles 7-class detection; visually demonstrates bounding-box predictions (Fig. 9 p. 5); scalable to real-time apps.	Small dataset (839 imgs); recall slightly dropped (−1.7 %); no public dataset; deployment prototype not implemented.	**[13]**
An Automated and Fine-Tuned Image Detection and Classification System for Plant Leaf Diseases	Authors' custom dataset + PlantVillage (Healthy class)	2,000 images, 8 classes — *Anthracnose, Aphids, Chilli Leaf Curl, Cucumber Mosaic Virus, Flea Beetles, Healthy, Spot, Whiteflies*; each class has 200 images (Train 1,600 / Val 400); balanced dataset ; all resized to 512×512 px	YOLOv5 (fine-tuned); baseline comparison with YOLOv3 and YOLOv4; backbone = CSPDarknet53 ; loss = CIoU; trained for 200 epochs, batch size 32, input 512×512 px; hardware: Google Colab GPU.	YOLOv5—mAP@0.5 = 0.614, mAP@0.5–0.95 = 0.484, 130 FPS; YOLOv4—mAP@0.5 = 0.241; YOLOv3—mAP@0.5 = 0.236 (Table II, p. 5)	(1) Can YOLOv5 detect and classify multiple plant diseases with high accuracy in real time? (2) Does dataset balancing and fine-tuning improve precision and speed vs. YOLOv3/v4? (3) Can this pipeline generalize to multiple species' leaf diseases under a unified model?	Fast inference (130 FPS); higher mAP (+37.3%) over YOLOv4; balanced dataset; detailed preprocessing pipeline; bounding-box visualizations	Only 2k images; no public dataset link; limited crop variety; tested only on validation set	**[14]**
Leaf Disease Detection Using KNN	Plant Pathology 2020 FGVC7 (Apple Leaf Dataset) — Kaggle	Dataset contains 4 classes — *Healthy, Rust, Scab, Multiple Diseased*; apple leaf images captured under controlled illumination; train/val/test split ≈ 80/20; preprocessed via ROI extraction, Canny edge detection, Histogram equalization, SMOTE for class balancing	KNN (K-Nearest Neighbors) classifier with Exploratory Data Analysis (EDA) preprocessing pipeline; includes Sobel filters, ROI segmentation, K-means clustering; used 5×5 de-noising mask; Euclidean distance metric.	KNN—75.9% (Val Acc), Precision 77%, Recall 76%, F1 76%	(1) Can EDA-driven preprocessing enhance KNN performance for multi-class leaf disease classification? (2) How effective is classical ML vs. CNNs on a small balanced dataset? (3) What preprocessing pipeline (edge detection + ROI) best improves traditional classifier results?	Simple and interpretable; effective pipeline (EDA + K-means + ROI); visualizes edges and regions	Moderate accuracy (≤76%); lacks deep learning; dataset domain-limited (apple leaves only); small test set.	**[15]**
Potato Leaf Disease Detection using Deep Learning	PlantVillage Potato Leaf Dataset (Kaggle)	3,000 images, 3 classes — *Healthy*, *Early Blight*, *Late Blight*; resized to 150×150 px; preprocessed via normalization and augmentation (flipping, rotation). Split: 80% train, 10% val, 10% test	MobileNetV3-Small (best), compared with MobileNet, ResNet50, and VGG19; implemented with TensorFlow/Keras, backend API via FastAPI, frontend via ReactJS; deployed on cloud (Heroku/AWS) (Figs. 3–7).	MobileNetV3-Small—98.75 %, ResNet50—98.14%, VGG19—96.48%, MobileNet—91.68%; AUC = 96.71%, Recall = 88.77%, Precision = 97.77%, F1 = 93.8%	1) Can lightweight CNN architectures like MobileNetV3 provide real-time, web-based potato leaf disease detection with high accuracy? (2) How do model depth and parameter count affect inference speed vs. accuracy? (3) Can TensorFlow-based architectures generalize across field images under varied lighting?	High accuracy; fast inference (~130 FPS via optimized FastAPI backend); real-time web app using ReactJS; small model size → mobile deployable; ablation vs. 3 baselines.	Dataset limited to *PlantVillage* (lab images); recall slightly lower (88.7%); lacks explainability or real-field validation.	**[16]**