A Comparative Review of Pretrained Deep Learning Models for Plant Disease Detection

Pratyaksh Mathur
University of Texas at Arlington
pxm8261@mavs.uta.edu

Vandit Goel
University of Texas at Arlington
vxg5699@mavs.uta.edu

Abstract

Plant diseases pose a significant threat to global food security, necessitating automated detection systems that are both accurate and deployable in real-world agricultural settings. This paper presents a comprehensive comparative evaluation of twelve pretrained deep learning architectures for binary plant disease classification (healthy vs. diseased), including classical models (AlexNet, VGG16), lightweight architectures (MobileNet_v2, ShuffleNet_v2, SqueezeNet, MnasNet, EfficientNet-Lite4), and modern designs (ResNet50, DenseNet121, Xception, InceptionV4, ConvNeXt). Using transfer learning on the PlantVillage dataset with targeted data augmentation to address class imbalance, we evaluate model performance through accuracy, precision, recall, F1-score, and AUC-ROC metrics. Critically, we assess generalization capability through cross-dataset evaluation on PlantDoc, which contains real-world images with natural backgrounds. Our results reveal that architectural complexity does not guarantee superior performance: VGG16 achieved the highest AUC-ROC (0.9990) on PlantVillage, while EfficientNet-Lite4 attained the best accuracy (98.20%). However, cross-dataset evaluation exposed significant generalization gaps, with ConvNeXt demonstrating the best transfer performance (83.90% on PlantDoc) despite ranking lower on PlantVillage. Lightweight models exhibited severe overfitting, with MobileNet_v2 and ShuffleNet_v2 dropping to 44.49% and 40.25% accuracy respectively on PlantDoc. For practical deployment, AlexNet emerges as a recommended choice, offering balanced performance (96.53% PlantVillage, 81.36% PlantDoc) with the fastest inference speed (651 samples/second). Our findings underscore the importance of cross-dataset evaluation for robust plant disease detection systems.

1. Introduction

Plant diseases pose a significant threat to global food security, leading to considerable yield losses and reduced crop quality. Traditional methods of disease diagnosis rely on visual inspection by experts, which is both time-consuming and error-prone, especially in resource-limited agricultural regions. Recent advances in computer vision and deep learning have provided promising alternatives by enabling automated, accurate, and scalable disease detection systems. Among these, convolutional neural networks (CNNs) and their variants have emerged as the dominant approach for extracting discriminative features from plant leaf images.

The **PlantVillage** dataset [12], a publicly available benchmark comprising a wide range of plant species and disease classes, has become the standard testbed for evaluating deep learning models in this domain. Leveraging pretrained models through transfer learning has further accelerated progress, allowing researchers to adapt architectures originally developed for large-scale image recognition tasks (e.g., ImageNet) to agricultural applications with limited labeled data.

This paper presents a comprehensive review of state-of-the-art pretrained models that have been applied to plant disease detection using the PlantVillage dataset. The models examined include **Mobilenet, ResNet50, SqueezeNet, MnasNet1, VGG16, AlexNet, DenseNet-121, Inception V4, Xception, ShuffleNet, EfficientNet-Lite4 and ConvNext**. These architectures represent diverse design philosophies, ranging from lightweight networks optimized for mobile deployment (e.g., MobileNet, ShuffleNet) to deeper and more complex models targeting higher accuracy (e.g., DenseNet, EfficientNet).

In this study we do not perform multi-class classification across the original PlantVillage labels. Instead, we recast the task as a binary classification problem (”healthy” vs ”unhealthy”) by mapping all disease-specific labels to a single ”unhealthy” class by changing the labels if each and every images in the dataset. This relabeling simplifies the

prediction target but increases class imbalance (many crops have far more diseased or healthy images), which we address through targeted data-augmentation and careful batch sampling during training. We discuss these choices and their impact on evaluation in the Dataset and Implementation sections below.

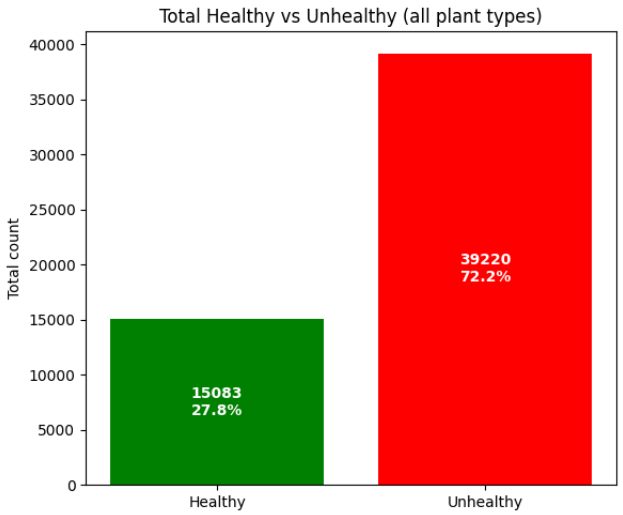

Figure 1. Distribution of healthy vs. diseased

To address the challenge of accurate and efficient plant disease detection, this paper adopts a comparative evaluation of multiple deep learning models on the PlantVillage dataset. Instead of developing a new architecture from scratch, **transfer learning** is employed to leverage the feature extraction capabilities of the aforementioned models. Each model is fine-tuned and tested under the same experimental conditions, allowing for a fair assessment of accuracy, computational efficiency, and deployment feasibility. By systematically analyzing their performance, the paper identifies which architectures are most suitable for practical agricultural applications, thereby providing a clear pathway for selecting models that balance precision and resource constraints in real-world plant disease detection systems.

2. Related Work

Comparative analyses of CNN models are crucial for selecting appropriate architectures for specific applications. Existing surveys often provide comprehensive overviews of architecture, applications, and performance on standard datasets like ImageNet, MNIST, and CIFAR-10/100. These studies systematically evaluate metrics like accuracy, parameter count, and FLOPs to provide insights into model suitability.

CNNs have found wide application across various domains, including healthcare, remote sensing, security, and agriculture. In agriculture, convolutional neural networks are specifically pointed out as being useful for tasks like monitoring crops, spotting pests, predicting harvest amounts, and detecting/treating diseases. Given the complexity of visual data in agricultural settings, deep learning is essential for automated feature extraction, moving beyond traditional manual feature engineering methods.

The general strategy employed in this domain is **transfer learning**. Because acquiring large, richly annotated agricultural datasets is often resource-intensive and time-consuming, researchers leverage models pretrained on massive general visual datasets (like ImageNet) and fine-tune them for specific PDD tasks. Reviews specifically addressing CNN use in agriculture emphasize the utility of these techniques for maximizing resource use and accuracy. [3, 8, 16, 17]

2.1. Differentiation of the Current Research

This research, differs significantly from existing literature in both scope and rigorous validation methodology, specifically addressing the practical challenges of deployment in agricultural informatics:

1. **Comprehensive Architectural Scope:** We systematically evaluate and compare a broad spectrum of pretrained CNN models, ranging from established deep architectures (e.g., VGG16, ResNet50, DenseNet-121, Inception V4, Xception) to contemporary efficient and lightweight designs (e.g., MobileNet, ShuffleNet, MnasNet1, EfficientNet-Lite4, ConvNext). This breadth allows for identifying not only the most accurate models but also those offering the best balance between accuracy and computational efficiency (measured via inference speed), which is paramount for practical edge deployment.
2. **Reframing the Classification Task:** Instead of focusing purely on multi-class disease differentiation, this study recasts the problem into a crucial **binary classification** task: distinguishing ”healthy” from ”unhealthy” leaves. This shift, though seemingly simpler, demands sophisticated techniques to mitigate the aggravated class imbalance and the risk of overfitting to spurious cues. We employ targeted data augmentation (including replacing majority class images with augmented variants to encourage learning disease-relevant features rather than background artifacts) and batch-wise balanced sampling to ensure robust training.
3. **Cross-Dataset Generalization Validation:** Crucially, we move beyond reporting performance solely on the clean, in-distribution PlantVillage test set. We deploy a **cross-dataset evaluation** strategy using the independent PlantDoc dataset. The PlantDoc dataset, derived from internet-scraped images, inherently possesses the real-world domain variations (natural backgrounds, diverse lighting) that models trained on PlantVillage typi-

cally fail to generalize to. This robust evaluation, measured using comprehensive metrics (including precision, recall, F1-score, AUC-ROC, sensitivity, and specificity), provides a realistic assessment of model deployability and robustness to environmental shifts—a critical but often overlooked metric in prior PDD studies.

3. Dataset

For this study, we make use of **PlantVillage** [7] [9] dataset, which is one of the most widely used benchmark datasets for plant disease detection and classification tasks. The dataset consists of 54,303 labeled images of healthy and diseased plant leaves belonging to 14 distinct classes. These classes cover a variety of important crops such as **Apple, Blueberry,Cherry, Corn, Grape, Orange, Peach, Bell Pepper, Potato, Raspberry, Soybean, Squash, Strawberry, Tomato**, each with multiple disease categories in addition to healthy leaves (Figure 2). It containes images of 17 fungal diseases, 4 bacterial diseases, 2 mold(oomycete) diseases, 2 viral disease, and 1 disease caused by a mite. 12 crop species also have images of healthy leaves that are not visibly affected by a disease. See Table 5.

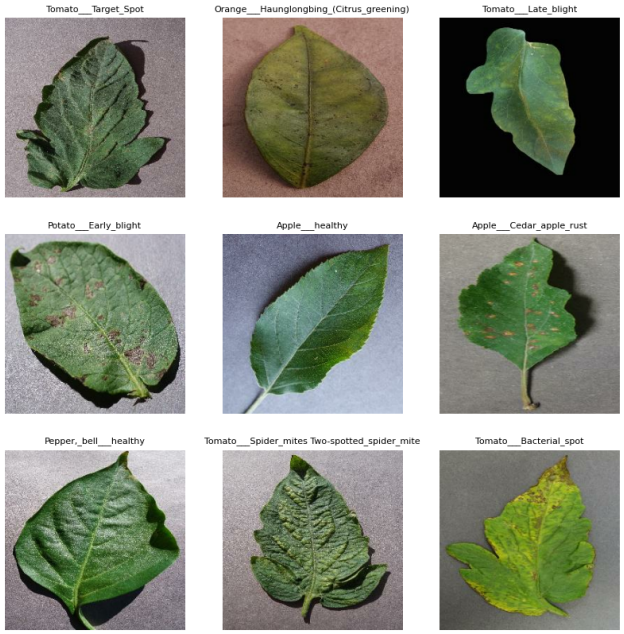

Figure 2. Sample images from dataset.

The images in PlantVillage were originally collected under controlled conditions with uniform backgrounds, making them suitable for initial model training. However, this also introduces a limitation in terms of generalization to real-world conditions, where background noise, varying illumination, and complex environmental factors are present [10].

The PlantVillage dataset exhibits a significant degree of class imbalance, with certain crop-disease categories being heavily overrepresented while others contain relatively few samples. Such imbalance can lead to biased model training, where the network becomes disproportionately optimized for the majority classes, resulting in poor generalization for minority categories. To mitigate this issue, several strategies are commonly employed, including oversampling the minority class, undersampling the majority class, applying class-weighted sampling, or using advanced data augmentation techniques. Additionally, relying solely on accuracy as a performance metric can be misleading under such circumstances; therefore, metrics like precision, recall, and F1-score are used to provide a more balanced evaluation of model performance.

In this study, data augmentation was adopted as the primary approach to alleviate class imbalance. Various augmentation techniques—such as rotation, flipping, illumination adjustment, and scaling—were applied to artificially increase the number of samples in minority classes. For the majority classes, augmentation was performed selectively, with the augmented images replacing existing samples to prevent further expansion of the dominant categories. Despite these measures, residual imbalance may still persist. To further address this, batch-wise balanced training was implemented using TensorFlow's data pipeline, ensuring that each batch presented to the model contains an approximately equal representation of all classes. This strategy promotes fairer learning and enhances the robustness of the trained models across all plant disease categories.

3.1. PlantDoc Dataset for Cross-Dataset Evaluation

To evaluate the generalization capability of models trained on PlantVillage, we employ the **PlantDoc** dataset [14] as an independent test set. PlantDoc is a publicly available dataset specifically designed for visual plant disease detection, containing 2,598 images across 13 plant species and up to 17 disease classes. Unlike PlantVillage, which was collected under controlled laboratory conditions with uniform backgrounds, PlantDoc consists of internet-scraped images that exhibit significant variation in lighting conditions, backgrounds, image quality, and leaf orientations. This makes PlantDoc an excellent benchmark for assessing how well models trained on controlled-environment data generalize to real-world conditions [14].

Table 1. PlantDoc test set class distribution (binary labels)

Class	Samples
Healthy	90
Diseased	146
Total	**236**

The PlantDoc dataset includes 27 distinct folder categories representing different plant-disease combinations such as "Apple Scab Leaf", "Tomato leaf bacterial spot", "Corn Gray leaf spot", and healthy leaves like "Apple leaf" or "Tomato leaf". For our binary classification task, we mapped these multi-class labels to binary targets: folders containing only the plant name followed by "leaf" (e.g., "Apple leaf", "Tomato leaf") were labeled as healthy (0), while folders containing disease indicators such as "rust", "scab", "spot", "blight", "rot", "mildew", "bacterial", "virus", or "mosaic" were labeled as diseased (1). This mapping ensures consistency with our PlantVillage-trained models while testing their ability to generalize to visually diverse, real-world plant disease images.

4. Methodology for Comparative Evaluation

4.1. Data Augmentation

Data augmentation increases the effective size and diversity of the training set by applying label-preserving transformations. For minority classes this helps the model see more varied examples of disease symptoms, reducing the risk that the classifier will ignore rare categories. For majority classes we perform selective augmentation and sometimes replace original images with augmented variants for two reasons:

First, prevent overfitting to trivial background cues: PlantVillage images were collected in controlled conditions with uniform backgrounds. If the model memorizes background or camera-specific artifacts present in the majority class, it may not generalize to field images. Replacing some majority-class originals with augmented variants (cropped, color-jittered, or slightly rotated) encourages the model to learn disease-relevant features instead of spurious patterns.

Second, Maintain class balance in batches while preserving dataset size: Instead of only increasing the dataset by duplicating/augmenting majority classes (which worsens imbalance), we use augmentation to create alternative views and then sample batches that are balanced across the two binary labels (healthy vs unhealthy). This preserves the overall dataset scale while providing more diverse training signals.

We used the PlantVillage dataset (loaded via TensorFlow Datasets) and converted the original multi-class labels of the form "PlantType___DiseaseName" into a binary target by assigning $0 =$ healthy and $1 =$ diseased (determined by testing whether the suffix after "___" equals "healthy").

To mitigate class imbalance we applied a class-specific augmentation and replication scheme implemented in a tf.data pipeline: labels are mapped to binary via a static lookup array and all image augmentations are performed on float32 images in the $\{0, 1\}$ range and converted back to uint8 for downstream use. For the healthy class we applied aggressive stochastic augmentations: random horizontal and vertical flips, a random $90°$ rotation ($k \in 0, 1, 2, 3$), random saturation in $[0.8, 1.25]$, random hue offset up to ± 0.05, random brightness delta up to ± 0.12, and random contrast in $[0.8, 1.25]$. For the diseased class we applied milder perturbations and a replacement policy: for each diseased sample, with probability 0.5 we replace the original with an augmented variant produced by random horizontal flip, random $90°$ rotation, contrast in $[0.9, 1.1]$, brightness ± 0.08 and hue ± 0.03; otherwise the original image is retained. (See Figure 3 for example augmentations.)

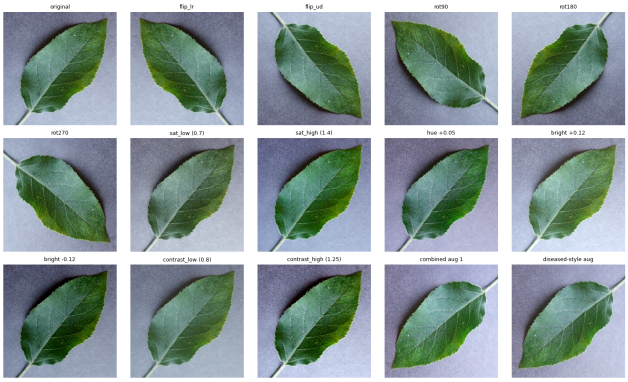

Figure 3. Example of Data-Augmentation

To balance classes we computed H and D as the healthy and diseased training counts and set the healthy replication multiplier $m = \max(1, \lceil (D/H) \rceil - 1) \quad (\text{if } H > 0)$, then concatenated the original healthy set with m independently-augmented passes of the healthy set to produce $\approx H \times (m + 1)$ healthy examples; augmented diseased samples were concatenated thereafter. See Table 2 for the final class distribution after augmentation.

Table 2. Distribution after augmentation

Class	Samples
Healthy	45249 (53%)
Diseased	39220 (47%)
Total	**84469**

4.2. Model Training

All models were trained using transfer learning with pre-trained ImageNet weights, leveraging PyTorch's torchvision library for standard architectures and the timm library for advanced architectures. This approach provides strong feature initialization while reducing training time and computational requirements.

4.2.1. Transfer Learning Strategy

For each architecture, we employed a layer freezing strategy to preserve learned low-level features while adapting the network to our binary classification task. All backbone/feature extraction layers were frozen, and only the final classifier layers were made trainable.

4.2.2. Training Configuration

Adam optimizer was used with a lower learning rate (10^{-4}) for stable convergence across diverse architectures. Employed Binary Cross-Entropy with Logits (BCEWithLogitsLoss) with sigmoid activation, outputting a single logit per sample. A ReduceLROnPlateau scheduler monitored validation loss and reduced the learning rate when no improvement was observed. This adaptive approach allows aggressive initial learning while ensuring fine-grained convergence in later stages. To prevent overfitting, training was halted if validation loss did not improve for the specified patience period. The model checkpoint with the lowest validation loss was retained as the final model. (See Table 3 for full hyperparameter details.)

Table 3. Training Hyperparameters

Hyperparameter	Value
Optimizer	Adam
Learning Rate	1×10^{-4}
Loss Function	Binary Cross-Entropy with Logits
Max Epochs	5
Early Stopping Patience	2 epochs
LR Scheduler	ReduceLROnPlateau (mode=min, patience=1)
Random Seed	42

4.2.3. Memory Optimization Techniques

Training large pretrained models requires significant GPU memory. We implemented several optimization strategies:

- **Mixed Precision Training (AMP):** Automatic Mixed Precision was enabled using PyTorch's `torch.amp` module, performing forward passes in FP16 while maintaining FP32 master weights. This reduces memory consumption by approximately 50% and accelerates training on modern GPUs.
- **Gradient Accumulation:** For memory-constrained scenarios, gradients were accumulated over 2 steps before performing weight updates, effectively simulating larger batch sizes without proportional memory increase.
- **Model-Specific Batch Sizes:** Larger architectures (InceptionV4, Xception) used reduced batch sizes (16) compared to smaller models (AlexNet: 64, VGG16/ResNet50: 32) to fit within GPU memory constraints.
- **Gradient Checkpointing:** Where supported, gradient checkpointing was enabled to trade computation for memory by recomputing intermediate activations during backpropagation.

4.2.4. Data Pipeline

The training pipeline was implemented as a hybrid TensorFlow-PyTorch system for optimal efficiency:

1. **Data Loading:** PlantVillage dataset was loaded via TensorFlow Datasets (tfds) with streaming access.
2. **Augmentation:** Class-specific augmentations were applied using TensorFlow's `tf.data` pipeline with parallel processing (`num_parallel_calls=AUTOTUNE`).
3. **Conversion:** Augmented images were converted on-the-fly to PyTorch tensors via a custom `IterableDataset` wrapper, avoiding disk caching and minimizing memory footprint.
4. **Normalization:** Images were resized to 224×224 pixels and normalized using ImageNet statistics (mean $= [0.485, 0.456, 0.406]$, std $= [0.229, 0.224, 0.225]$).
5. **Prefetching:** TensorFlow prefetching (buffer size = 2) ensured GPU was never idle waiting for data.

4.2.5. Model Summary

Table 4 provides an overview of all evaluated architectures, categorized by their design philosophy and computational requirements.

Table 4. Summary of Evaluated Model Architectures

Model	Params (M)	ImageNet Top-1 (%)
AlexNet	61.1	56.5
VGG16	138.4	71.6
ResNet50	25.6	76.1
DenseNet121	8.0	74.4
InceptionV4	42.7	80.0
Xception	22.9	79.0
ConvNeXt Base	88.6	84.1
MobileNetV2	3.5	71.9
ShuffleNetV2	2.3	69.4
SqueezeNet 1.0	1.2	58.1
MNASNet 1.0	4.4	73.5
EfficientNet-Lite4	13.0	80.4

4.3. Evaluation Metrics

To ensure a fair and comprehensive comparison across different pretrained architectures, several quantitative metrics were employed to evaluate model performance including accuracy, precision, recall and F1-score.

Furthermore, model performance was also analyzed using the **Receiver Operating Characteristic (ROC)** curve, which plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various classification thresholds. The **Area Under the Curve (AUC)** serves as a single scalar value summarizing the overall discriminative capability of the model—where a higher AUC indicates stronger classification performance.

Confusion matrices were generated for each model to provide a detailed breakdown of classification outcomes, including True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

Additionally, **inference speed** (samples per second) was measured to assess the practical deployability of each model, as real-world plant disease detection systems often require rapid classification for timely intervention. Such a multi-metric evaluation ensures that both overall accuracy, per-class discrimination, and computational efficiency are considered in determining the most effective pretrained model for plant disease detection.

4.4. Cross-Dataset Evaluation on PlantDoc

To assess the generalization capability of the trained models beyond the PlantVillage dataset, we conducted cross-dataset evaluation using the PlantDoc dataset [14]. This evaluation is critical because models trained on PlantVillage's controlled laboratory images may overfit to dataset-specific artifacts such as uniform backgrounds and consistent lighting conditions [10]. Testing on PlantDoc, which contains internet-scraped images with natural backgrounds and varying conditions, provides a realistic assessment of model robustness.

The cross-dataset evaluation pipeline was implemented using PyTorch with the following methodology:

Preprocessing: All test images from PlantDoc were resized to 224×224 pixels to match the input dimensions used during training. Standard ImageNet normalization (mean $= [0.485, 0.456, 0.406]$, std $= [0.229, 0.224, 0.225]$) was applied to ensure consistency with the pretrained model expectations.

Label Mapping: The PlantDoc dataset's multi-class folder structure was converted to binary labels. A rule-based classifier examined folder names for disease-indicating keywords (e.g., "rust", "blight", "spot", "scab", "mildew", "bacterial", "virus"). Folders without these keywords and containing only "<PlantType> leaf" patterns were classified as healthy.

Inference: Models were loaded from their trained checkpoints and evaluated in inference mode with mixed-precision (FP16) for computational efficiency. For each test image, the model produced a class probability via sigmoid activation, and the predicted class was determined by thresholding at 0.5. The sigmoid output (probability of the diseased class) was retained for ROC analysis.

Metrics Computation: The same comprehensive metrics used for PlantVillage evaluation were computed on the PlantDoc test set: accuracy, precision, recall, F1-score, and AUC-ROC. Confusion matrices were generated to analyze per-class performance, with particular attention to sensitivity (true positive rate for diseased samples) and specificity (true negative rate for healthy samples). This cross-dataset evaluation reveals how well each architecture captures generalizable disease features versus dataset-specific patterns.

5. Results

5.1. PlantVillage Dataset Evaluation

Table 6 summarizes the performance metrics for all evaluated models on the PlantVillage test set. The results reveal several important findings regarding the effectiveness of different architectural approaches for plant disease detection.

5.1.1. Classical Models Analysis

The classical architectures demonstrated surprisingly strong performance despite their relatively simpler design. VGG16 achieved the highest AUC-ROC score of 0.9990 among all models, with an accuracy of 97.05% and F1-score of 0.9801. The model exhibited excellent recall (99.85%), indicating near-perfect sensitivity in detecting diseased samples. AlexNet, despite being the earliest architecture evaluated, achieved 96.53% accuracy with remarkable training efficiency (538.35 seconds), making it highly suitable for resource-constrained deployment scenarios.

5.1.2. Lightweight Models Analysis

Among the lightweight architectures, EfficientNet-Lite4 emerged as the top performer overall, achieving the highest accuracy (98.20%) and F1-score (0.9875) across all evaluated models. This result validates the efficacy of compound scaling strategies employed in the EfficientNet family. MobileNet_v2 also demonstrated competitive performance with 96.17% accuracy while maintaining computational efficiency suitable for mobile deployment.

However, lightweight models exhibited greater variance in performance. SqueezeNet1_0 and MnasNet1_0 showed lower accuracy (82.01% and 84.22% respectively), though both maintained high AUC-ROC scores (>0.97), indicating good ranking capability despite suboptimal classification thresholds. The confusion matrix analysis reveals that these models tend toward high recall at the expense of precision, suggesting a bias toward predicting the diseased class.

5.1.3. Modern Architectures Analysis

Modern architectures presented mixed results. ResNet50 and DenseNet121 achieved strong performance (97.09% and 96.76% accuracy respectively), benefiting from their sophisticated connectivity patterns that facilitate gradient

flow and feature reuse. Both models demonstrated excellent precision (>0.99), indicating low false positive rates critical for practical deployment.

Interestingly, ConvNeXt_Base, despite being the most recently developed architecture, achieved only 91.03% accuracy. Similarly, Xception and InceptionV4 showed lower accuracy (87.66% and 88.09%) compared to classical alternatives, with notably lower precision values.

5.2. Training Dynamics Analysis

Figure 4 presents the training and validation loss curves across all model clusters. Several observations emerge from this analysis:

Classical Models: Both AlexNet and VGG16 demonstrated smooth convergence with training and validation losses decreasing consistently across epochs. The gap between training and validation loss remained minimal, suggesting good generalization without significant overfitting.

Lightweight Models: These architectures showed more varied convergence patterns. EfficientNet-Lite4 exhibited rapid initial convergence, while MnasNet1_0 showed higher validation losses that plateau at suboptimal values, explaining their lower test accuracy. MobileNet_v2 and ShuffleNet_v2 converge reasonably well.

Modern Architectures: ResNet50 and DenseNet121 displayed stable learning dynamics with low final loss values. However, InceptionV4 showed elevated validation loss despite low training loss, indicating potential overfitting on the PlantVillage dataset.

5.3. ROC Curve Analysis

The ROC curves in Figure 5 provide insight into the classification capability across different operating thresholds. All models achieved AUC-ROC values exceeding 0.93, indicating strong discriminative ability for the binary classification task.

VGG16 and AlexNet demonstrated near-perfect ROC curves that closely hug the upper-left corner, reflecting their exceptional ability to distinguish healthy from diseased samples across all threshold values. Among lightweight models, EfficientNet-Lite4 showed the steepest initial rise, achieving high true positive rates at very low false positive rates.

5.4. Cross-Dataset Generalization: PlantDoc Evaluation

Table 7 presents the performance metrics when models trained on PlantVillage were evaluated on the PlantDoc dataset without fine-tuning.

5.4.1. Generalization Performance Analysis

ConvNeXt emerged as the best generalizing model with 83.90% accuracy and the highest F1-score (0.8782) on PlantDoc. Notably, this represents a ranking reversal from PlantVillage results, where ConvNeXt ranked lower. This suggests that ConvNeXt's modern design captures more generalizable disease features that transfer across datasets.

Classical models demonstrated robust generalization. AlexNet achieved 81.36% accuracy on PlantDoc (compared to 96.53% on PlantVillage), representing only a 15.2 percentage point drop. VGG16 showed similar robustness with 75.42% accuracy. The simpler feature hierarchies learned by classical networks appear to transfer better to out-of-distribution samples.

Severe degradation in lightweight models. MobileNet_v2 dropped from 96.17% to 44.49% accuracy, while ShuffleNet_v2 fell to 40.25%. The results reveal extreme prediction bias: ShuffleNet_v2 achieved perfect precision (1.0) but only 3.42% recall, indicating it classified nearly all samples as healthy. This behavior suggests these models overfit to PlantVillage-specific background and lighting patterns.

ResNet50's catastrophic failure (32.63% accuracy, below random baseline for some classes) is particularly notable given its strong PlantVillage performance. The AUC-ROC of 0.2811 (below 0.5) indicates inverted predictions, where the model's probability rankings are negatively correlated with true labels.

5.4.2. ROC Analysis on PlantDoc

ConvNeXt, SqueezeNet1_0, and VGG16 maintained reasonable ROC curves with AUC values above 0.90, indicating preserved ranking ability despite lower classification accuracy. However, ResNet50's ROC curve falls below the diagonal random classifier line, confirming systematic prediction errors. Among lightweight models, only SqueezeNet1_0 (AUC=0.909) maintained strong discriminative capability on PlantDoc. See Figure 6.

5.5. Computational Efficiency

Table 8 presents inference speed measurements.

AlexNet processes 182.0 samples/second, balancing speed with strong generalization performance (81.36% accuracy on PlantDoc). VGG16, despite achieving 75.42% accuracy on PlantDoc, maintains practical throughput at 152.6 samples/second.

ConvNeXt_Base, while demonstrating the best cross-dataset generalization (83.90% accuracy), requires significantly more computational resources with only 96.9 samples/second—nearly half the speed of the fastest models. This trade-off between accuracy and inference speed is critical for deployment.

6. Conclusion

This study presented a comprehensive evaluation of twelve pretrained deep learning architectures for binary plant

disease classification, comparing classical, lightweight, and modern network designs across both in-distribution (PlantVillage) and cross-dataset (PlantDoc) evaluation scenarios.

Our experimental results reveal several key insights for deploying plant disease detection systems in real-world agricultural settings:

Architectural Complexity Does Not Guarantee Superior Performance. Classical architectures, particularly VGG16 and AlexNet, consistently ranked among the top performers across both datasets. VGG16 achieved the highest AUC-ROC (0.9990) on PlantVillage, while AlexNet demonstrated the best efficiency-accuracy trade-off with 96.53% accuracy on PlantVillage, 81.36% on PlantDoc, and the fastest inference speed (182 samples/second). These findings challenge the assumption that deeper, more complex networks necessarily provide improved plant disease detection.

Generalization Remains a Critical Challenge. Cross-dataset evaluation exposed significant generalization gaps across all model categories. While EfficientNet-Lite4 achieved the highest in-distribution accuracy (98.20%), ConvNeXt demonstrated superior cross-dataset transfer (83.90% on PlantDoc), representing a notable ranking reversal from PlantVillage results. This suggests that optimizing for benchmark performance may not translate to robust real-world deployment.

Lightweight Models Exhibit Severe Overfitting. Despite their computational efficiency, lightweight architectures including MobileNet_v2 and ShuffleNet_v2 showed catastrophic performance degradation on PlantDoc (44.49% and 40.25% respectively), with extreme prediction biases indicating overfitting to dataset-specific visual patterns. ResNet50 similarly failed with 32.63% accuracy and inverted probability rankings (AUC-ROC of 0.2811).

Practical Recommendations. For resource-constrained agricultural applications requiring both accuracy and efficiency, AlexNet emerges as the recommended choice due to its balanced performance across datasets and minimal computational requirements. For applications prioritizing maximum generalization capability, ConvNeXt offers superior cross-dataset transfer despite higher computational costs.

Limitations and Future Work. This study focused on binary disease classification; extending the evaluation to multi-class disease identification would provide additional insights. Furthermore, investigating domain adaptation techniques and data augmentation strategies specifically designed to improve cross-dataset generalization represents a promising direction for future research. The development of specialized architectures that prioritize generalization over benchmark optimization may ultimately prove more valuable for practical agricultural deployment.

Our findings underscore the importance of evaluating plant disease detection models beyond single-dataset benchmarks, as model robustness to varying image conditions is paramount for real-world agricultural applications.

References

[1] Ali Hussein Ali, Ayman Youssef, Mahmoud Abdelal, and Muhammad Adil Raja. An ensemble of deep learning architectures for accurate plant disease classification. *Ecological Informatics*, 81:102618, 2024.

[2] Asadulla Y Ashurov, Mehdhar SAM Al-Gaashani, Nagwan A Samee, Reem Alkanhel, Ghada Atteia, Hanaa A Abdallah, and Mohammed Saleh Ali Muthanna. Enhancing plant disease detection through deep learning: a depthwise cnn with squeeze and excitation integration and residual skip connections. *Frontiers in Plant Science*, 15:1505857, 2025.

[3] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. Review of image classification algorithms based on convolutional neural networks. *Remote Sensing*, 13(22):4712, 2021. 2

[4] Shuang Cong and Yang Zhou. A review of convolutional neural network architectures and their optimizations. *Artificial Intelligence Review*, 56(3):1905–1969, 2023.

[5] Jaya Gupta, Sunil Pathak, and Gireesh Kumar. Deep learning (cnn) and transfer learning: a review. In *Journal of Physics: Conference Series*, page 012029. IOP Publishing, 2022.

[6] Noredine Hajraoui, Mourade Azrour, Yousef Farhaoui, and Ahmad El Allaoui. Artificial intelligence and deep transfer learning for plant disease detection and classification. 2025.

[7] David P. Hughes and Marcel Salathé. An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060*, 2015. 3

[8] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 33(12):6999–7019, 2021. 2

[9] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7, 2016. 3

[10] Mehmet Alican Noyan. Uncovering bias in the plantvillage dataset. *arXiv preprint arXiv:2206.04374*, 2022. 3, 6

[11] Omkar Oak, Rukmini Nazre, Soham Naigaonkar, Suraj Sawant, and Himadri Vaidya. A comparative analysis of cnn-based deep learning models for landslide detection. In *2024 Asian Conference on Intelligent Technologies (ACOIT)*, pages 1–6. IEEE, 2024.

[12] G Prathibha Priyadarshini and S Zahoor-Ul-Huq. A systematic literature review on cnn-based deep learning models for plant disease detection and classification to enhance agricultural productivity. In *International Conference on Sustainability Innovation in Computing and Engineering (ICSICE 2024)*, pages 1207–1218. Atlantis Press, 2025. 1

[13] Shagun Sharma and Kalpna Guleria. Deep learning models for image classification: comparison and applications. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 1733–1738. IEEE, 2022.

[14] Davinder Singh, Naman Jain, Pranjali Jain, Pratik Kayal, Sudhakar Kumawat, and Nipun Batra. Plantdoc: A dataset for visual plant disease detection. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, page 249–253, New York, NY, USA, 2020. Association for Computing Machinery. 3, 6

[15] Neelam Sulaiya and Shyamol Banerjee. A review on plant leaf disease detection using cnn deep learning models. *International Journal of Advanced Research and Multidisciplinary Trends (IJARMT)*, 2(1):642–653, 2025.

[16] Maria Trigka and Elias Dritsas. A comprehensive survey of deep learning approaches in image processing. *Sensors*, 25(2):531, 2025. 2

[17] Hewa Majeed Zangana, Corresponding Author, Ayaz khalid Mohammed, and Firas Mahmood Mustafa. Advancements and applications of convolutional neural networks in image analysis: A comprehensive review. *Jurnal Ilmiah Computer Science*, 3:16–29, 2024. 2

[18] Xia Zhao, Limin Wang, Yufei Zhang, Xuming Han, Muhammet Deveci, and Milan Parmar. A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, 57(4):99, 2024.

A. Tables

Table 5. Crop disease status

Crop Species	Diseased	Health
Apple	1527	1645
Blueberry	0	1502
Cherry	1052	854
Corn	2690	1162
Grape	3640	423
Orange	5507	0
Peach	360	2291
Bell Pepper	997	1478
Potato	2000	152
Raspberry	0	371
Soybean	0	5090
Squash	1835	0
Strawberry	1109	456
Tomato	16570	1592

Table 6. Performance Metrics on PlantVillage Test Set

Model	**Category**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**	**Train Time (s)**
EfficientNet-Lite4	Lightweight	**0.9820**	0.9913	0.9838	**0.9875**	0.9982	1156.42
VGG16	Classical	0.9705	0.9625	0.9985	0.9801	**0.9990**	1688.80
ResNet50	Modern	0.9709	**0.9912**	0.9686	0.9798	0.9958	987.65
DenseNet121	Modern	0.9676	0.9945	0.9607	0.9773	0.9970	1245.32
AlexNet	Classical	0.9653	0.9564	0.9978	0.9766	0.9979	538.35
MobileNet_v2	Lightweight	0.9617	0.9958	0.9514	0.9731	0.9962	678.54
ShuffleNet_v2	Lightweight	0.9332	0.9859	0.9212	0.9525	0.9883	589.21
ConvNeXt_Base	Modern	0.9103	0.8907	**0.9993**	0.9419	0.9962	3341.80
InceptionV4	Modern	0.8809	0.8626	0.9949	0.9240	0.9403	1139.24
Xception	Modern	0.8766	0.8569	0.9970	0.9217	0.9739	1002.68
MnasNet1_0	Lightweight	0.8422	0.9981	0.7845	0.8785	0.9908	712.36
SqueezeNet1_0	Lightweight	0.8201	0.8025	0.9982	0.8897	0.9741	498.76

Table 7. Cross-Dataset Performance on PlantDoc Test Set

Model	**Category**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**
ConvNeXt	Modern	**0.8390**	0.8253	0.9384	**0.8782**	**0.9173**
AlexNet	Classical	0.8136	0.7802	0.9726	0.8659	0.8932
VGG16	Classical	0.7542	0.7245	0.9726	0.8304	0.9137
DenseNet121	Modern	0.7161	0.7724	0.7671	0.7698	0.7426
EfficientNet-Lite4	Lightweight	0.6949	0.8426	0.6233	0.7165	0.7760
Xception	Modern	0.6398	0.6356	0.9795	0.7709	0.7594
InceptionV4	Modern	0.6229	0.6223	0.9932	0.7652	0.7412
SqueezeNet1_0	Lightweight	0.6229	0.6213	**1.0000**	0.7664	0.9091
MnasNet1_0	Lightweight	0.5085	**0.9688**	0.2123	0.3483	0.7309
MobileNet_v2	Lightweight	0.4449	0.7778	0.1438	0.2428	0.6237
ShuffleNet_v2	Lightweight	0.4025	1.0000	0.0342	0.0662	0.7981
ResNet50	Modern	0.3263	0.3860	0.1507	0.2167	0.2811

Table 8. Computational Efficiency Metrics

Model	**Inference Time (s)**	**Samples/Second**
AlexNet	1.30	182.0
VGG16	1.55	152.6
Xception	1.44	163.8
InceptionV4	1.43	164.7
ConvNeXt_Base	2.44	96.9
DenseNet121	1.43	165.3
MnasNet1_0	1.39	169.6
MobileNetV2	1.48	159.1
ResNet50	1.46	161.8
ShuffleNetV2	1.46	161.8
SqueezeNet1_0	1.47	160.1
tf_efficientnet_lite4	1.58	149.4

B. Figures

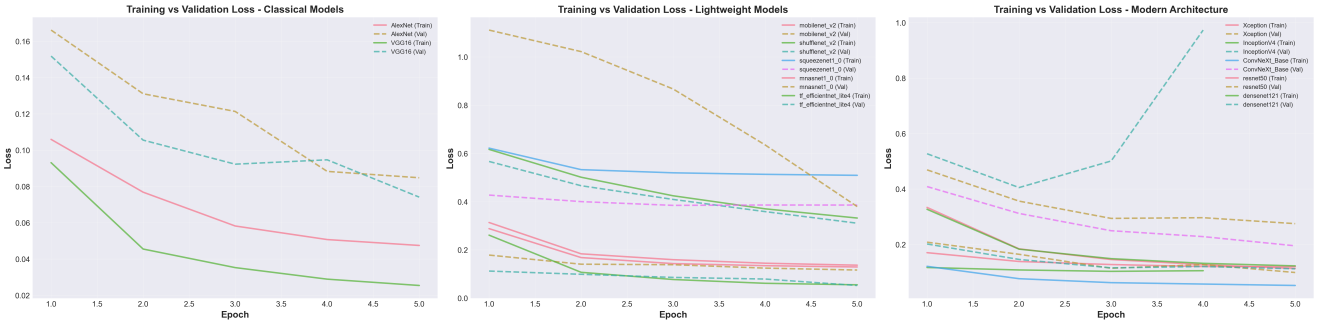

Figure 4. Training vs. Validation Loss Curves by Model Cluster

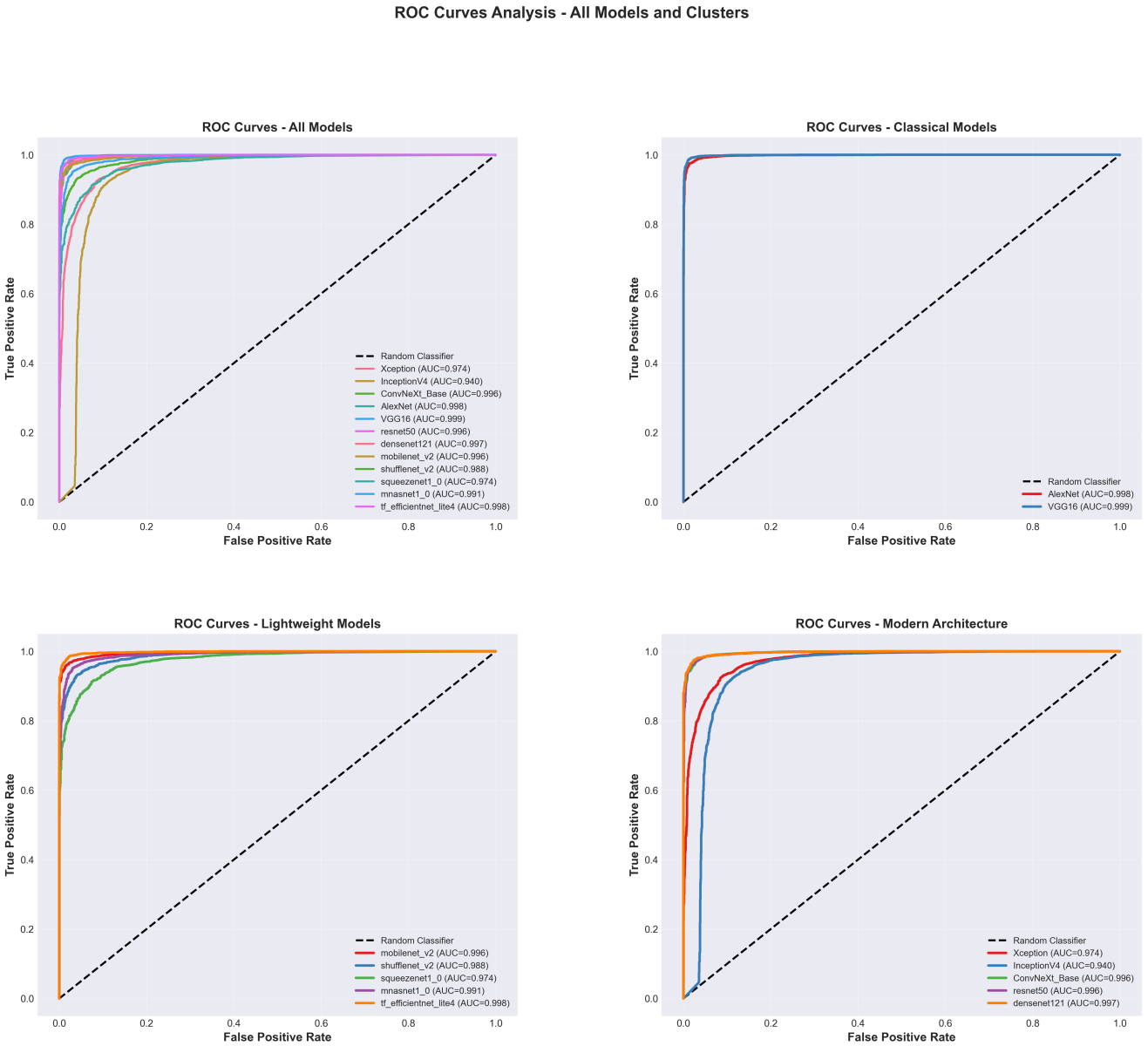

Figure 5. ROC Curves Analysis - PlantVillage Dataset

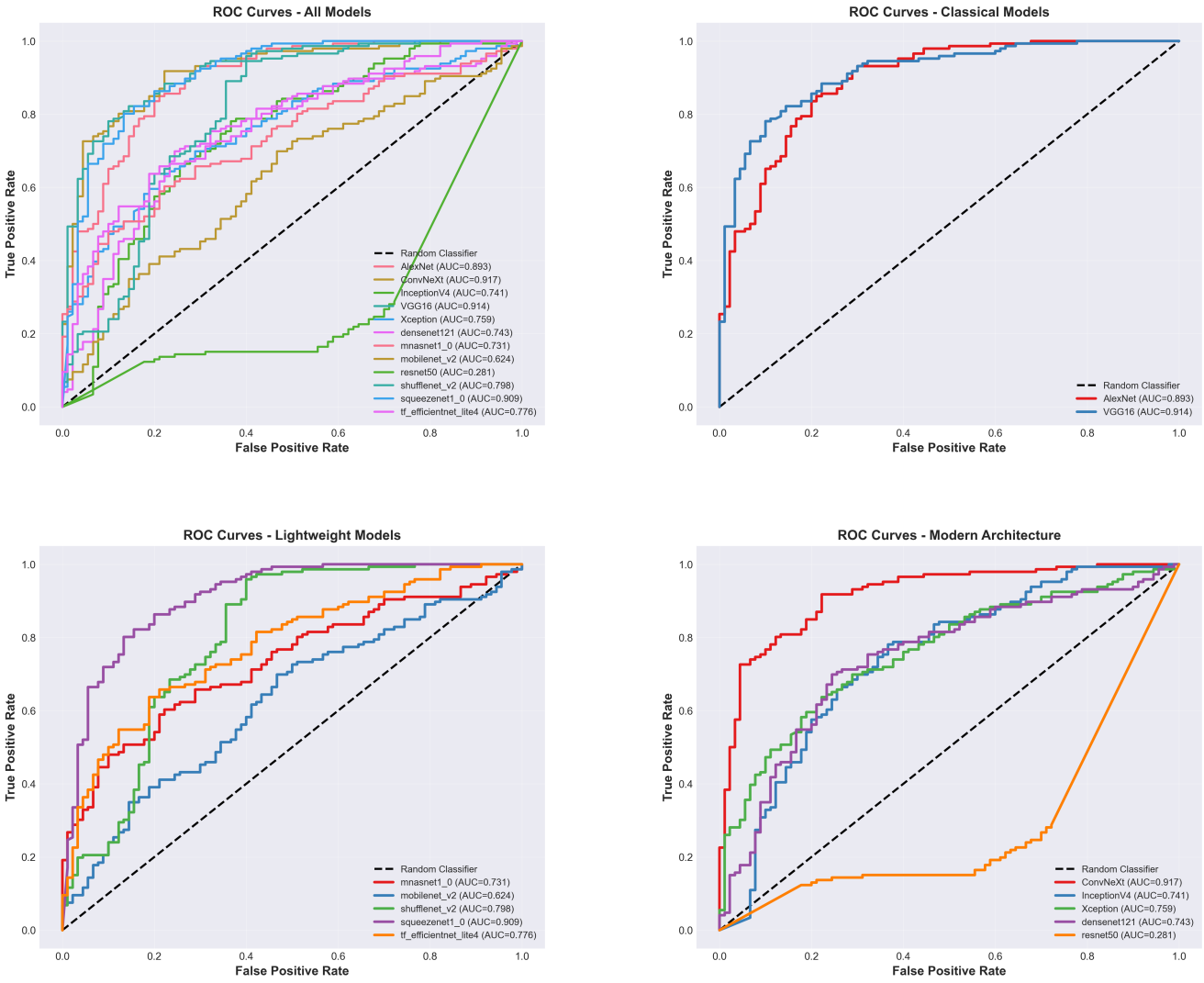

Figure 6. ROC Curves Analysis - PlantDoc Cross-Dataset Evaluation