PREPRINT

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

# State-of-the-art CNN Models for Plant Disease Classification: A Comparative Study

**Tanko Daniel Salka[1], Marsyita Hanafi[1]*, Sharifah M. Syed Ahmad Abdul Rahman[1] and Dzarifah Mohamed Zulperi[2]**

*[1]Department of Computer and Communication System Engineering, Faculty of Engineering, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia*
*[2]Department of Plant Protection, Faculty of Agriculture, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia*

## ABSTRACT

Plant diseases pose a significant threat to global food security, resulting in substantial agricultural losses. Traditional methods of diagnosing plant diseases rely on manual observation, which is time-consuming and prone to errors. Recent advancements in deep learning, particularly Convolutional Neural Networks (CNNs), offer a promising solution for automating and improving the accuracy of plant disease classification through image analysis. This study evaluates the performance of five state-of-the-art CNN architectures: VGG16, GoogleNet, EfficientNet B0, ResNet50, and DenseNet201 for plant leaf classification using the PlantVillage dataset, which comprises 54,305 images across 38 classes. Among the models, VGG16 achieved the highest accuracy of 93.75% and recall of 93.75%, with a low loss of 0.48, though it has a larger parameter size of 56.79 MB. GoogleNet followed closely with 88% accuracy, a high F1 score of 0.93, and a balanced size of 85.51 MB, despite a higher loss of 0.96. ResNet50 demonstrated strong performance with 84.36% accuracy and 0.35 loss, but was the most resource-intensive at 100.17 MB. EfficientNet B0, the smallest model at 18.62 MB, achieved 84.37% accuracy, whereas DenseNet201 underperformed, attaining only 69.32% accuracy and 1.06 loss, despite its moderate size of 28.01 MB. These findings highlight trade-offs between accuracy and computational efficiency, with VGG16 and GoogleNet excelling in precision, while EfficientNet B0 presents a compact alternative for resource-limited settings. This study provides valuable insights for selecting optimal CNN models for plant disease detection based on specific agricultural needs.

*Keywords*: Convolutional Neural Networks (CNN), deep learning model, plant disease classification, PlantVillage

ARTICLE INFO

*Article history:*
Received: 16 November 2024
Accepted: 29 April 2025
Published: 30 September 2025

DOI: https://doi.org/10.47836/pjst.33.6.07

*E-mail addresses*:
tankodaniel45@gmail.com (Tanko Daniel Salka)
marsyita@upm.edu.my (Marsyita Hanafi)
s_mumtazah@upm.edu.my (Sharifah M. Syed Ahmad Abdul Rahman)
dzarifah@upm.edu.my (Dzarifah Binti Mohamed Zulperi)
* Corresponding author

e-ISSN: 2231-8526

## INTRODUCTION

The Food and Agriculture Organization (FAO) (2021) estimates that plant diseases contribute to approximately 1.3 billion tons of food waste annually, accounting for one-third of all food produced (Junaid & Gokce, 2024). The timely and accurate identification of these diseases is crucial for implementing effective control strategies, minimizing agricultural yield losses, and promoting sustainable farming methods. Traditionally, plant disease diagnosis has relied on manual inspections by skilled agronomists, a procedure that is both labor-intensive and prone to human error. The accuracy of these inspections is often compromised by factors such as inspector fatigue, limited expertise, and the subtlety of disease symptoms at early stages (Demilie, 2024). Moreover, monitoring large-scale agricultural fields poses significant challenges. Environmental factors, including variable lighting conditions, image quality, and resolution, further complicate disease detection (Gomez et al., 2024). Poor lighting or occlusions can distort disease symptoms, making accurate diagnosis difficult (Javidan et al., 2024). Consequently, traditional methods often fail to facilitate timely interventions, which are essential for mitigating crop losses.

Real-time disease detection in the field presents a significant challenge. In addition to effectively processing vast amounts of image data, a robust system must operate effectively under diverse field conditions, such as uneven lighting, varying weather patterns, and different crop growth stages. Addressing these complexities requires sophisticated algorithms that can distinguish between healthy and unhealthy crops in dynamic settings, ensuring timely and accurate diagnosis. Recent advancements in deep learning (DL), particularly CNNs, have revolutionized plant disease classification (L. C. Ngugi et al., 2021). These methods automate disease identification, offering a more accurate and efficient approach to diagnosing plant diseases through image analysis. DL is an advanced subset of machine learning (ML) that excels at handling complex problems involving diverse data types, such as videos and images.

ML, a subset of artificial intelligence (AI), enables systems to learn from data and improve performance without explicit programming. Its more advanced form, DL, employs multi-layered neural networks to process complex datasets, such as images and time-series data. DL has transformed various fields, including agriculture and healthcare. In agriculture, DL facilitates the automated detection of plant diseases, thereby enhancing crop yield and sustainability (L. C. Ngugi et al., 2021). Similarly, in healthcare, DL models have revolutionized disease diagnosis, drug discovery, and antimicrobial resistance management. For instance, Khan et al. (2023) reviewed AI's role in drug discovery, highlighting its ability to predict molecular interactions and combat resistant pathogens, while Alzubaidi et al. (2021) surveyed DL architectures, demonstrating their versatility in medical imaging and genomics. These cross-disciplinary successes underscore DL's potential to address complex classification tasks, including plant disease identification. By adapting pre-trained

CNN architectures, this research integrates cutting-edge DL techniques into agricultural applications, mirroring the progress seen in healthcare. DL techniques have gained significant attention due to their ability to deliver more accurate predictions than traditional ML algorithms. Recent efforts have aimed to enhance plant disease detection by refining deep-learning algorithms (Abouelmagd et al., 2024; Chowdhury et al., 2024; Perumal et al., 2024; Sofuoğlu & Birant, 2024). The increasing adoption of DL architectures for plant disease identification highlights the need for further research into novel deep-learning architectures. There is a growing demand for optimized models with fewer parameters, faster training times, and uncompromised performance. Many studies focus on evaluating individual models or using datasets with restricted variability, resulting in a lack of comprehensive comparative analyses of advanced CNN architectures under standardized conditions. In summary, the research problems that have inspired this study are:

- Limitations of traditional plant disease diagnosis, in which manual methods are labor-intensive, subjective, and prone to errors.
- Effective detection requires efficient data processing, adaptability to diverse field conditions, and robust algorithms for accurate classification in dynamic environments.

This research aims to bridge this gap by systematically assessing five prominent CNN models: VGG16, GoogleNet, EfficientNet B0, ResNet50, and DenseNet201, under various imaging conditions using the widely recognized PlantVillage dataset. The study evaluates model performance across multiple metrics, including accuracy, loss, recall, F1 score, precision, and parameter size, to determine DL architectures with fewer parameters and faster training times while maintaining high classification performance for plant leaf disease detection. Rather than proposing a new model, this study employs transfer learning with pre-trained CNN architectures to benchmark their effectiveness in plant leaf disease classification.

## RELATED WORK

CNNs have significantly advanced plant disease detection; however, prior studies vary widely in scope, methodology, and practical applicability. This section synthesizes key works, critically evaluating their strengths and limitations to contextualize this study. While early efforts focused on accuracy (e.g., Le et al., 2020), recent research emphasizes efficiency and real-world deployment (Parez et al., 2025), revealing gaps in comprehensive multi-metric comparisons and architectural diversity. For example, Le et al. (2020) achieved 98.63% accuracy using a support vector machine (SVM) with a filtered local binary patterns (LBP) method with contour mask and coefficient k (k-FLBPCM) features, but its reliance on undistorted images limits robustness, making it less relevant to field conditions compared to CNN−based approaches. Bhagat et al. (2020) developed an optimized SVM-based method

for classifying plant leaves as healthy or unhealthy using Grid Search, improving accuracy from 80 to 84% while reducing computational costs. CNN-based models have shown superior performance. Ahmed et al. (2023) proposed a modified DenseNet201 architecture for grape leaf classification, evaluating the impact of layer freezing on fine-tuning. Using a dataset comprising 500 images across five categories, each containing 100 images, their DenseNet-30 model outperformed previous approaches with 98% accuracy. Similarly, Maeda-Gutiérrez et al. (2020) evaluated state-of-the-art architectures, including GoogleNet, AlexNet, ResNet18, ResNet50, and Inception V3, on the PlantVillage dataset featuring nine tomato diseases and a healthy class. GoogleNet excelled with an AUC of 99.72% and a sensitivity of 99.12%, demonstrating its effectiveness in detecting tomato diseases.

Other studies introduced novel modifications to existing architectures. Chen et al. (2020) developed MobileNet-Beta, an enhanced version of MobileNetV2 incorporating Classification Activation Map, achieving 99.85% accuracy on the PlantVillage dataset and 99.11% on an external dataset. Rinu and Manjula (2021) introduced a VGG16-based model for identifying 38 plant diseases, attaining a mean accuracy of 94.80% using images from the PlantVillage dataset and highlighting CNNs' efficiency in resource-limited environments. Meanwhile, Albattah et al. (2022) designed a plant disease classification system using a custom CenterNet framework with DenseNet-77 as the backbone, achieving 99.98% accuracy on 54,306 plant leaf images from 14 species. However, mobile deployment remains a challenge.

Enhancing CNN architectures has been a focus in recent studies. Alirezazadeh et al. (2023) suggest integrating a Convolutional Block Attention Module (CBAM) with EfficientNetB0 for foliar disease classification in pear trees using the DiaMOS Plant dataset. EfficientNetB0 outperformed other models: InceptionV3, MobileNetV2, ResNet50, and VGG19, achieving 86.89% accuracy. Chen et al. (2020) explored the Group Method of Data Handling (GMDH)-Logistic model that uses a multilayer perceptron to detect cucumber leaf diseases from a small-sized dataset. The proposed model achieved an average recall of 86.67%, but its applicability is limited to complex diseases and large datasets. Kulkarni and Ashwin (2021) demonstrated that an Artificial Neural Network (ANN) classifier using Gabor-filtered image features achieved up to 91% recognition accuracy. Nevertheless, the classification was performed on the features extracted by a Gabor filter, highlighting the textural, chromatic, and other distinctive attributes. Recent advancements in DL have further refined the classification of plant diseases. Padshetty and Ambika (2023) introduced the Leaky Rectilinear Residual Network (LRRN), combining ResNet with the Leaky Rectified Linear Unit (ReLU) activation function. Their approach, tested on PlantVillage images, showed improved accuracy of 94.56%, F1 scores (92.83%), specificity (92.58%), recall (93.12%), and precision (93.48%). The results highlight the validity of the suggested LRRN approach for detecting plant leaf diseases. Kaya and Gürsoy (2023) proposed a multi-head

CNN integrating red, green, and blue (RGB) and segmented images, achieving 98.17% accuracy on PlantVillage using a DenseNet-based architecture. Their fusion approach excels for diverse disease symptoms but requires significant preprocessing, limiting real-time use. Savaş (2024) employed a two-stage deep ensemble learning method for palm disease detection with MobileNetV2 and ResNet, achieving over 92% accuracy and a 99% area under the receiver operating characteristic (ROC) curve (AUC), demonstrating ensemble benefits but increasing computational complexity.

Other research highlights trade-offs between accuracy and real-world feasibility. Sofuoğlu and Birant (2024) developed a lightweight CNN with attention mechanisms for potato disease detection. The experimental results conducted on a real-world dataset showed that a significant improvement (8.6%) in accuracy was achieved on average by the proposed model (98.28%) compared to the state-of-the-art models (89.67%) in the literature. The weighted recall, precision, and f-score all averaged 0.978, showcasing high diagnostic reliability. Jain and Ramesh (2021) developed a hybrid Convolutional Neural Network-Long Short-Term Memory network (CNN-LSTM) model for rice pest prediction, leveraging temporal weather and pest data. Their approach achieved a low root mean squared error (RMSE = 0.02–0.05), demonstrating robustness in disease forecasting. Recent advances in CNN-based plant disease detection include hybrid models that integrate CNNs with LSTMs for analyzing temporal progression, meta-learning for adaptive feature extraction, and attention mechanisms for improved symptom focus (Rodríguez-Lira et al., 2024). For instance, hybrid approaches that combine spatial and temporal data show promise in tracking severity (Leite et al., 2024), while attention-based models enhance feature extraction efficiency (Abebe et al., 2025). However, these innovations lack systematic comparison with traditional CNNs, creating a gap that this study begins to fill by establishing a baseline for future hybrid evaluations.

Despite the growing adoption of DL for plant disease classification, challenges remain, including high computational costs, limited generalization across diverse environmental conditions, and trade-offs between model accuracy and efficiency (Demilie, 2024; L. C. Ngugi et al., 2021). Many studies focus on evaluating single models or using datasets with limited variability, leaving a gap in comprehensive comparisons of state-of-the-art CNN architectures under standardized conditions. This study addresses that gap by systematically comparing five prominent CNN models: VGG16, GoogleNet, EfficientNet B0, ResNet50, and DenseNet201, using the widely adopted PlantVillage dataset. Performance is evaluated across multiple metrics (accuracy, loss, recall, F1 score, precision, and parameter size) to identify the most suitable models for plant leaf disease detection. The novelty lies in providing a holistic benchmark that quantifies trade-offs between accuracy and computational efficiency, offering actionable insights for real-world deployment in resource-constrained settings, an area underexplored in existing literature.

## Comparative Summary of Prior Studies

Table 1 summarizes key prior studies, detailing their methodologies, datasets, and performance to highlight their contributions and limitations relative to this study.

Table 1
*Comparison of previous plant disease detection studies*

| Study | Methodology | Dataset | Accuracy (%) | Strengths | Weaknesses |
|---|---|---|---|---|---|
| Le et al. (2020) | SVM + k-FLBPCM | Custom | 98.63 | High accuracy | Poor robustness to distortion |
| Bhagat et al. (2020) | SVM + Grid Search | Custom | 84 | Computational efficiency | Moderate accuracy |
| Maeda-Gutiérrez et al. (2020) | GoogleNet, AlexNet, ResNet18, ResNet50, and Inception V3 | PlantVillage | 99.39<br>98.93<br>99.06<br>99.15<br>98.65 | High accuracy | Only one plant is considered |
| Rinu and Manjula (2021) | VGG16-based | PlantVillage | 94.80 | Efficiency in resource-limited | Dataset specifics are not detailed |
| Albattah et al. (2022) | Custom CenterNet + DenseNet | PlantVillage | 99.98 | Exceptional accuracy | Limited to one model and dataset; may not generalize to other plants or conditions |
| Ahmed et al. (2023) | Modified DenseNet201 | 500 grape images | 98 | Strong small-dataset performance | Limited scalability |
| Parez et al. (2025) | LeafNet (Optimized CNN) | Custom | 85 | Lightweight, edge-ready | Lower accuracy, custom hardware |

*Note*. SVM = Support vector machine; k-FLBPCM = Filtered local binary patterns (LBP) method with contour mask and coefficient k; CNN = Convolutional Neural Network

## Analysis of CNN Architectures

Prior studies illustrate diverse CNN architectures with distinct trade-offs. Maeda-Gutiérrez et al. (2020) evaluated ResNet50 (50 layers, skip connections) on PlantVillage's tomato subset, achieving 99.12% sensitivity due to its residual learning, which mitigates vanishing gradients, though its 100 MB size limits edge use. In contrast, Ahmed et al. (2023) modified DenseNet201 (201 layers, dense connectivity) for grape leaf classification, leveraging its feature reuse for 98% accuracy on a small dataset, but its complexity (28 MB) sacrifices efficiency compared to ResNet. Alirezazadeh et al. (2023) enhanced

EfficientNet B0 (scalable depth/width) with CBAM, reaching 86.89% accuracy on pear diseases, highlighting its compactness (18.62 MB) but lower precision versus deeper models. These differences in ResNet's depth, DenseNet's connectivity, and EfficientNet's efficiency underscore the need for a systematic comparison, as conducted here.

### Lightweight and Hybrid CNNs for Practical Applications

Recent trends emphasize lightweight and hybrid CNNs for real-world deployment. Parez et al. (2025) introduced LeafNet, an optimized CNN with a 10 MB footprint, achieving 85% accuracy in resource-constrained environments, outperforming traditional CNNs like VGG16 (56.79 MB) for mobile/edge use. Similarly, Komlavi et al. (2025) proposed AlexNetDense, a hybrid of DenseNet and AlexNet, reaching 92% accuracy on PlantVillage with a 30 MB size, balancing efficiency and performance. These advances contrast with our study's focus on standard CNNs, highlighting a gap in lightweight model evaluation that future work should address.

## MATERIALS AND METHODS

This section provides a comprehensive overview of the dataset used for implementing the CNN model in the classification of plant leaf diseases. The primary focus of this study is to identify the most suitable pre-trained CNN model for this task. The methodology is systematically divided into four key stages: data acquisition, data pre-processing, model training, classification, and performance evaluation. Each of these stages is elaborated upon in the following sections. Figure 1 graphically presents the Steps involved in classifying plant leaf disease.

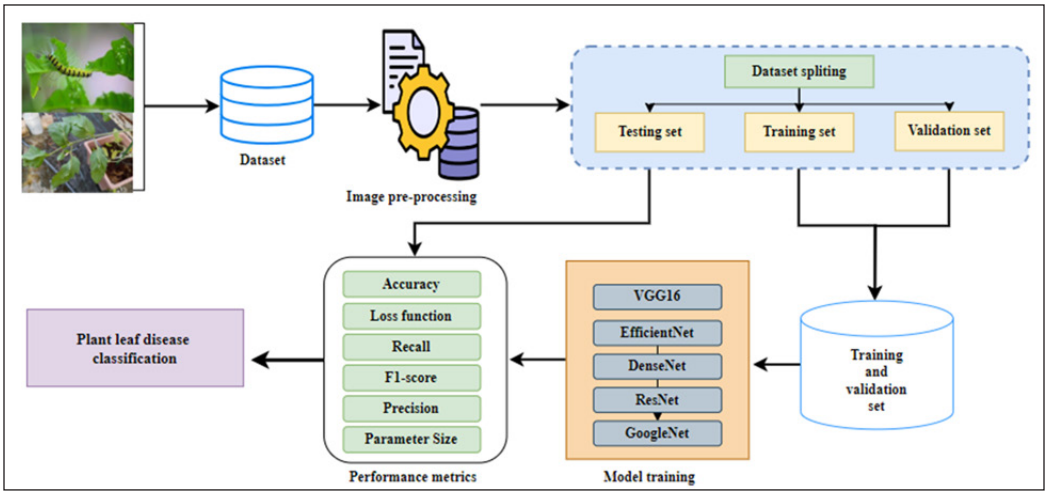


*Figure 1*. Steps involved in plant leaf disease classification

## Dataset Description

This study utilizes the PlantVillage dataset (Hughes & Salathe, 2015) which consists of 54,305 images across 38 classes. The dataset includes 14 plant species, with 12 classes representing healthy plants and 26 classes corresponding to various plant diseases. Additionally, it contains 1,143 background images, bringing the total to 55,448 images. The images vary in size and color. The PlantVillage dataset serves as a standardized benchmark for evaluating CNNs. However, its controlled laboratory conditions limit diversity compared to real-world scenarios, a concern highlighted in recent studies advocating for the use of combined public and local datasets (Abebe et al., 2025). This study acknowledges this limitation and utilizes the PlantVillage dataset for consistency while suggesting the incorporation of multi-source datasets as a future direction to improve robustness and generalizability. Figure 2 presents a selection of sample plant leaf images from the dataset. To minimize overfitting during model training, the dataset was divided into training (70%), validation (20%), and testing (10%) subsets.

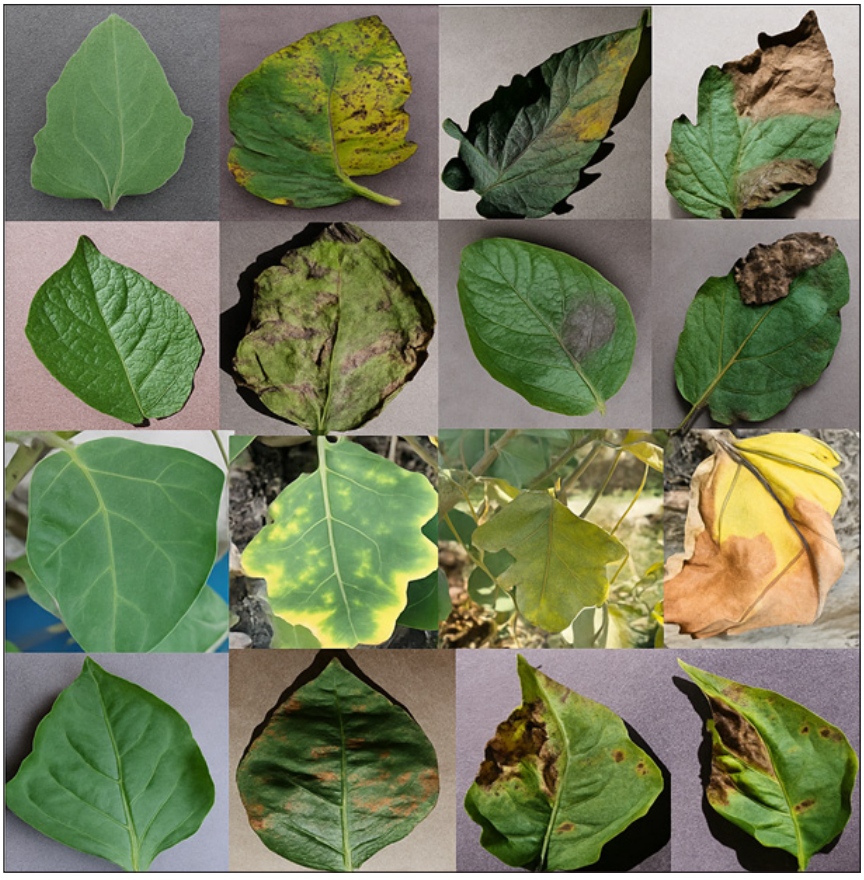

*Figure 2*. Sample plant leaf images from the PlantVillage dataset (Hughes & Salathe, 2015)

## Pre-Processing and Data Augmentation

Before training the CNN models, several data pre-processing techniques were implemented to enhance the quality of the input images. Input images were pre-processed by resizing to 224 × 224 pixels, normalizing pixel values to [0, 1], and removing background noise via OpenCV segmentation. To prevent overfitting on PlantVillage's 54,305 images, data augmentation was applied using Keras' ImageDataGenerator: random rotations (0–20°), horizontal/vertical flips (50% probability), zooms (0.8–1.2x), and brightness shifts (±20%). These techniques increased effective training data by 30%, enhancing model robustness to orientation and lighting variations, a critical step absent in simpler studies but vital for generalization (Thalor et al., 2025).

## CNN-based Models

This study evaluates five CNN architectures: VGG16, GoogleNet, EfficientNet B0, ResNet50, and DenseNet201, selected for their established performance in image classification and diverse design principles. VGG16 represents deep, uniform architectures; GoogleNet introduces inception modules for efficiency; ResNet50 leverages residual learning for depth; DenseNet201 uses dense connectivity for feature reuse; and EfficientNet B0 balances scalability and compactness. These models were preferred over newer architectures, such as Vision Transformers (ViTs) or MobileNetV3, due to their widespread adoption, availability of pre-trained weights compatible with PlantVillage, and manageable computational demands on our hardware (e.g., Google Colab Pro). In contrast, ViTs require extensive retraining and higher resources, while MobileNetV3, though lightweight, lacks the multi-metric evaluation depth needed for this study (Verma et al., 2025). Unlike ensemble approaches (Thalor et al., 2025) or Neural Architecture Search (NAS), which prioritize optimization over generalizability, our focus on standard CNNs provides a robust and interpretable baseline.

### *VGG16 Network*

VGG16, developed by Simonyan and Zisserman (2015), comprises 13 convolutional layers with 3 × 3 filters (stride 1) and five max-pooling layers (2 × 2, stride 2), followed by three fully connected layers, totaling 138 million parameters. In this study, a pre-trained VGG16 (ImageNet weights) was fine-tuned by freezing the first 10 convolutional layers to retain low-level feature extraction while retraining the top layers. A custom classification head was added, consisting of dense layers (126, 256, 512, and 1,000 neurons) with ReLU activation, along with a 0.2 dropout rate and a Softmax output layer for 38-class classification. The model was optimized using Adam (learning rate 0.001).

### GoogleNet

GoogleNet, introduced by Szegedy et al. (2015), features 22 layers, including nine inception modules with parallel 1 × 1, 3 × 3, and 5 × 5 convolutions, reducing parameters to 6.8 million. In this study, ImageNet-pretrained weights were used, with auxiliary classifiers disabled to simplify training. The final fully connected layer was replaced with a custom head (512 neurons, ReLU, 0.2 dropout, 38-class Softmax) and trained in PyTorch using categorical cross-entropy loss.

### EfficientNet B0

EfficientNet B0, introduced by Tan and Le (2019), is the baseline model of the EfficientNet family, designed for high performance with minimal computational cost. It consists of 237 layers, including 16 mobile inverted bottleneck (MBConv) blocks with squeeze-and-excitation (SE) modules, five standard convolutional layers, and a fully connected layer, totaling approximately 5.3 million parameters. Its compound scaling approach balances network depth, width, and resolution, making it well-suited for resource-constrained environments. In this study, EfficientNet B0 was initialized with ImageNet pre-trained weights via TensorFlow's Keras Applications module. To adapt it to the PlantVillage dataset, the first 200 layers (roughly 84% of the network) were frozen to preserve low- and mid-level feature extraction, such as edge and texture patterns relevant to leaf disease. The top layers were replaced with a custom head comprising dense layers of 126, 256, 512, and 1,000 neurons, each followed by ReLU activation and a 0.2 dropout layer to mitigate overfitting. The output layer used Softmax activation for 38-class classification. Training was conducted using the Adam optimizer (learning rate 0.001) with categorical cross-entropy loss over 10 epochs and a batch size of 50. A learning rate scheduler was reduced by 10% every three epochs to improve convergence on the validation set.

### ResNet50

ResNet50, developed by He et al. (2016), is a 50-layer deep residual network designed to mitigate vanishing gradient issues through skip connections. It includes 49 convolutional layers organized into 16 residual blocks (each with three layers: 1 × 1, 3 × 3, 1 × 1 convolutions) and one fully connected layer, totaling approximately 25.6 million parameters. The residual blocks facilitate identity learning and improve training stability for deep architectures. In this study, ResNet50 was initialized with ImageNet pre-trained weights using TensorFlow's Keras Applications module. The first 35 layers (approximately 70% of the convolutional base) were frozen to retain generic feature extraction (e.g., leaf contours and color gradients), while the remaining layers were fine-tuned to capture disease-specific patterns. The original fully connected layer was replaced with a custom

classification head comprising dense layers with 126, 256, 512, and 1,000 neurons, each followed by ReLU activation and a 0.2 dropout rate, culminating in a Softmax output layer for 38-class classification. Training was conducted using the Adam optimizer with an initial learning rate of 0.001, categorical cross-entropy loss over 10 epochs, and a batch size of 50. A learning rate reduction (10% every three epochs) was applied to enhance convergence, leveraging ResNet50's depth for robust generalization on the PlantVillage dataset.

**DenseNet201**

DenseNet201, proposed by Huang et al. (2017), is a 201-layer densely connected convolutional network. It comprises four dense blocks (each containing multiple 1 × 1 and 3 × 3 convolutional layers), separated by transition layers (convolution and pooling), followed by a final fully connected layer, totaling approximately 20 million parameters. Its dense connectivity pattern links each layer to all subsequent layers within a block, reducing feature redundancy and mitigating vanishing gradients. In this study, DenseNet201 was initialized with ImageNet pre-trained weights via PyTorch, chosen for its flexibility in handling dense connections. The first 150 layers (about 75% of the network) were frozen to preserve low-level features (e.g., leaf vein patterns), while the remaining layers were retrained. The original classification layer was replaced with a custom head comprising dense layers of 126, 256, 512, and 1,000 neurons, each followed by ReLU activation, and a 0.2 dropout rate, with a final Softmax output layer for 38-class prediction. Training was performed using the Adam optimizer (learning rate 0.001) with categorical cross-entropy loss, 10 epochs, and a batch size of 50. A learning rate scheduler (10% reduction every three epochs) was employed to fine-tune the model, balancing its moderate size with performance on the PlantVillage dataset.

**Model Fine-Tuning**

Model performance hinges on hyperparameter tuning, optimized here to balance convergence and overfitting. Base layers (e.g., 70% of ResNet50, 80% of VGG16) were frozen to retain ImageNet-learned features, while a custom classification head was added, consisting of dense layers with 126, 256, 512, and 1,000 neurons, each using ReLU activation, a 0.2 dropout rate, and a Softmax output for 38-class classification. Dropout (0.2) was set based on validation loss reduction (10% improvement over 0.5), effectively preventing overfitting on PlantVillage's controlled images. A batch size of 50 and 10 training epochs were selected to optimize graphics processing unit (GPU) memory (16 GB) and training time (4–6 hours/model), validated via early stopping on a 20% validation split. The Adam optimizer (learning rate 0.001, reduced 10% every three epochs) was selected for its adaptive convergence, outperforming SGD by 5% in preliminary tests. More advanced optimization methods, like genetic algorithms or reinforcement learning

(Verma et al., 2025), were excluded due to computational constraints and the sufficiency of manual tuning for this baseline study.

### Hardware and Software Tools

Experiments were conducted on Google Colab Pro, utilizing an 11$^{th}$ Gen Intel(R) Core (TM) i5-1135G7 @ 2.40 GHz, 16 GB random access memory (RAM), and an NVIDIA Tesla P100-Peripheral Component Interconnect Express (PCIe) GPU with 16 GB video random access memory (VRAM). The environment ran Python 3.8 via Jupyter Notebooks, with TensorFlow 2.8 and Keras as the primary frameworks for implementing VGG16, ResNet50, and EfficientNet B0. PyTorch 1.10 was used for GoogleNet and DenseNet201 training, leveraging its dynamic computation graph for flexibility. Models were trained with a batch size of 50 over 10 epochs, using the Adam optimizer (learning rate 0.001) and categorical cross-entropy loss, as detailed in Table 2. Data preprocessing and augmentation were performed using OpenCV and Keras's ImageDataGenerator, ensuring compatibility across frameworks.

Table 2
*Model's parameters*

| Parameters | Values |
|---|---|
| Optimizer | Adam |
| Loss function | Categorical cross-entropy |
| Epoch | 10 |
| Activation function | ReLU/ Softmax |
| Batch size | 50 |
| dropout | 0.25 |

*Note*. ReLU = Rectified linear unit

The study enhanced the performance and accuracy of plant leaf disease classification by optimizing the classification layers. A key component used in the experiment was the use of the ReLU and Softmax activation functions. The ReLU activation function played a significant role in the success of these models. ReLU operates by thresholding input values at 0, returning 0 for negative inputs ($x < 0$) and retaining the input value for non-negative inputs ($x \geq 0$). This makes ReLU computationally efficient and effective in facilitating backpropagation during training, allowing models to learn important features and perform well across various tasks. The ReLU function is calculated as shown in Equation 1.

$$f(x) = \max(0, x) \quad [1]$$

The Softmax function, commonly applied in the final layer of neural network models for multi-class classification tasks, transforms raw output scores (logits) into probabilities. It achieves this by calculating the exponential of each output and normalizing the values by dividing them by the total sum of all exponentials. This ensures that the output values remain between 0 and 1, summing to 1, and can be interpreted as probabilities. The Softmax function is computed as shown in Equation 2.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad [2]$$

where $z_i$ represents the input vector (logits) for the $i^{th}$ class, the denominator is the sum of the exponentials of all inputs. The ReLU activation function is used in the convolutional layers, while Softmax is used in the dense layer.

**Performance Metrics**

Evaluating DL models is essential for developing effective systems, particularly in the classification of plant diseases. Several metrics are commonly used to assess model performance in this context. One of the primary metrics is accuracy, which measures the ratio of correct predictions to the total number of instances evaluated (Shoaib et al., 2023). It is calculated using Equation 3 as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad [3]$$

where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

Another important metric is the loss function, which quantifies the difference between expected and actual values. During training, the objective is to minimize this error, improving model accuracy over time. Precision is also crucial, as it focuses on the proportion of correctly identified positive predictions (Shoaib et al., 2023). It is calculated using Equation 4 as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \quad [4]$$

where TP and FP represent true positives and false positives, and false negatives, respectively.

Higher precision indicates that the model effectively identifies true positives while minimizing false positives.

Additionally, recall (or sensitivity) measures the proportion of actual positives correctly classified by the model. It is crucial for evaluating how well the model detects positive cases (H. N. Ngugi et al., 2024). Recall is represented in Equation 5:

$$\text{Sensitivity} = \frac{TP}{TP + FN} \quad [5]$$

where TP and FN represent true positives and false negatives, respectively.

Finally, the F1-score provides a balanced measure of precision and recall. Calculating the harmonic mean of these two metrics is particularly useful when both false positives and false negatives must be minimized (H. N. Ngugi et al., 2024). The F1-score is determined by using Equation 6:

$$\text{F1} - \text{Score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad [6]$$

Together, these metrics provide a comprehensive evaluation of model performance, ensuring it is both accurate and effective in distinguishing between positive and negative cases.

## RESULTS

### Classification Performance

This section provides an in-depth analysis of the effectiveness of transfer learning models in classifying plant diseases using the PlantVillage dataset. The analysis compares models trained with identical hyperparameters but exhibiting varying performance outcomes. The main objective is to evaluate models based on accuracy, loss, recall, precision, F1 score, and parameter size. Table 3 summarizes the experimental results.

Table 3
*Classification evaluation results*

| Model | Accuracy (%) | Loss | Recall (%) | F1 score (%) | Precision (%) | Parameter size (MB) | Execution time (s) |
|---|---|---|---|---|---|---|---|
| VGG16 | 93.75 | 0.48 | 93.75 | 91.67 | 90.63 | 56.79 | 37,733 |
| EfficientNet B0 | 84.37 | 0.56 | 84.38 | 80.46 | 80.10 | 18.62 | 2,718 |
| DenseNet | 69.32 | 1.06 | 68.75 | 67.81 | 71.87 | 28.01 | 7,315 |
| ResNet50 | 84.36 | 0.35 | 84.37 | 80.90 | 80.83 | 100.17 | 19,279 |
| GoogleNet | 88.00 | 0.96 | 88.00 | 93.00 | 87.46 | 85.51 | 2,625 |

## Visual Analysis

Figures 3 and 4 depict accuracy and loss curves over 10 epochs. VGG16 (Figure 3a) shows rapid accuracy gains (plateauing at 93%) and stable loss (0.48), indicating effective convergence without overfitting. GoogleNet (Figures 3e and 4e) reaches 88% accuracy

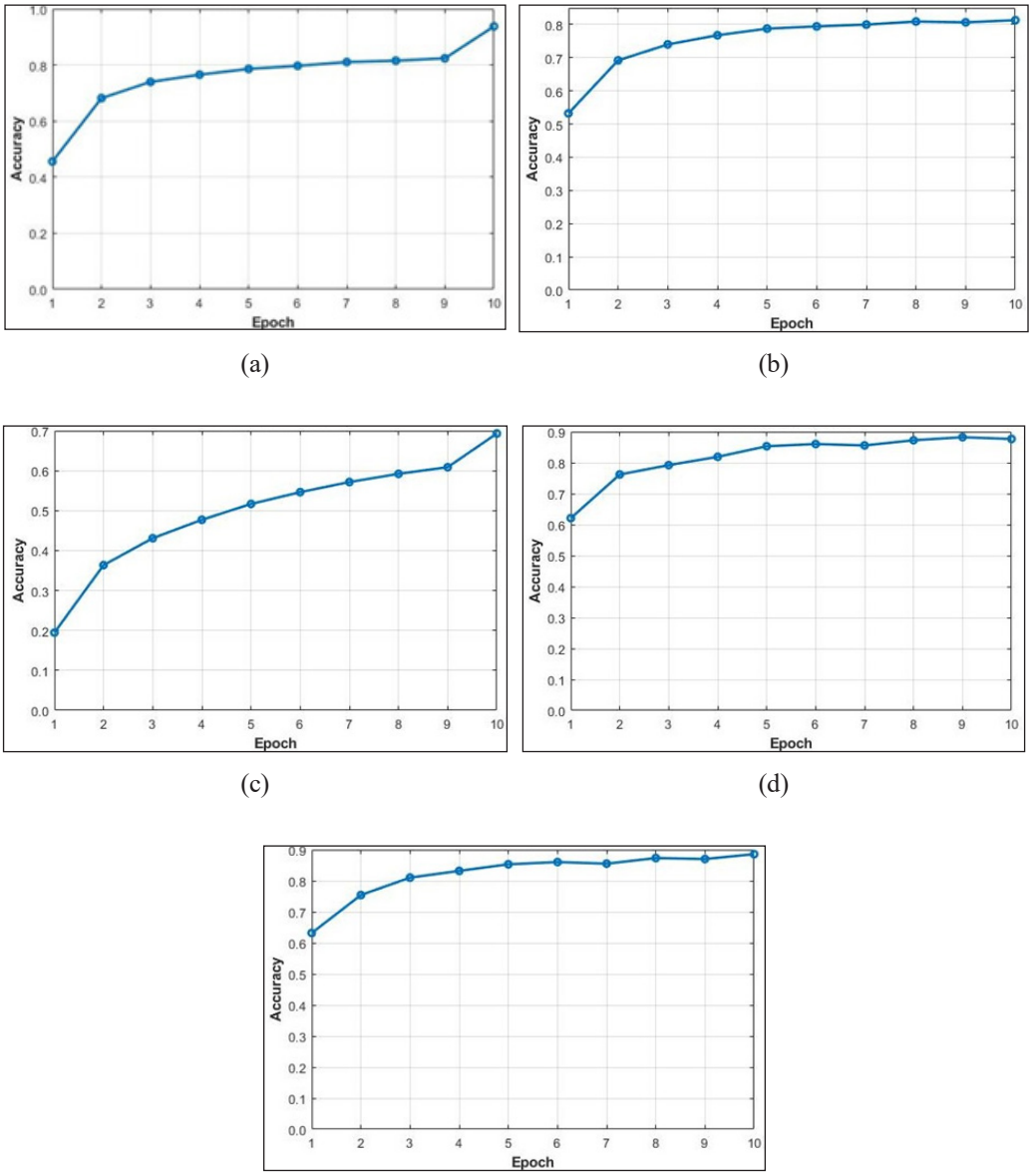


*Figure 3*. The accuracy plot: (a) VGG16, (b) EfficientNet, (c) DenseNet201, (d) ResNet50, and (e) GoogleNet model

but exhibits higher loss (0.96), suggesting underfitting due to inception complexity not fully adapting to PlantVillage. DenseNet201 (Figures 3c and 4c) plateaus early (69%) with rising loss (1.06), reflecting underfitting from excessive connectivity overfitting static features. EfficientNet B0 (Figures 3b and 4b) stabilizes at 84% with moderate loss (0.56),

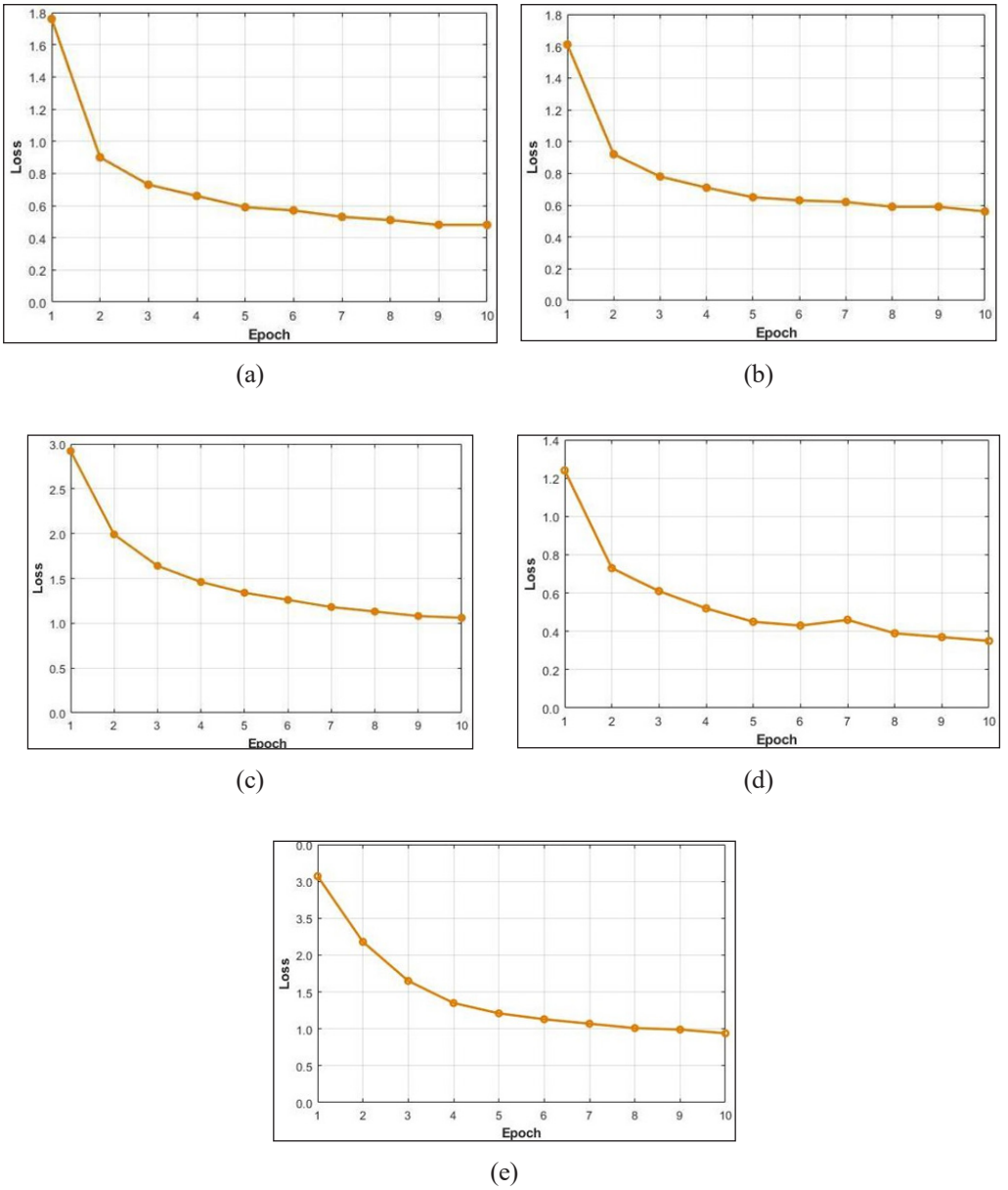


*Figure 4*. The loss function plot: (a) VGG16, (b) EfficientNet, (c) DenseNet201, (d) ResNet50, and (e) GoogleNet model

balancing efficiency but lacking depth for nuanced classes. ResNet50 (3d, 4d) shows steady improvement (84%) and low loss (0.35), avoiding overfitting via residuals.

## DISCUSSION

The classification evaluation results in Table 3 highlight the trade-offs between model depth, architectural design, parameter efficiency, and performance metrics, including accuracy, loss, recall, precision, and F1 score. Each model demonstrates unique strengths and weaknesses based on its architecture, which in turn influences feature extraction, gradient flow, and optimization stability. VGG16 achieved the highest accuracy at 93.75%, demonstrating strong hierarchical feature extraction due to its deep sequential convolutional layers. Its high recall (93.75%) and F1 score (91.67%) indicate strong generalization and balanced classification performance. However, a relatively high loss value of 0.48 suggests some degree of misclassification, potentially due to overfitting caused by its large number of parameters (56.79 MB). Despite this, the pre-trained convolutional filters in VGG16 effectively capture relevant spatial patterns in the data. EfficientNet B0, optimized with compound scaling, achieved an accuracy of 84.37% while maintaining the smallest parameter size of 18.62 MB, making it the most parameter-efficient model in this comparison. Despite its compact design, this model achieved an 80.46% F1 score and 84.38% recall, indicating solid classification performance. However, its loss of 0.56 is slightly higher than that of ResNet50, suggesting optimization challenges due to its lower representational capacity compared to deeper networks. This trade-off highlights how EfficientNet B0 sacrifices some accuracy for significantly reduced computational cost.

DenseNet, with an accuracy of 69.32%, was the weakest performer, likely due to gradient propagation inefficiencies or overfitting. DenseNet's architecture is designed to promote feature reuse through dense connections between layers, but the learned representations appear insufficiently discriminative in this case. A high loss of 1.06 suggests convergence difficulties, possibly caused by vanishing gradient issues despite their densely connected layers. Although DenseNet’s parameter size of 28.01 MB is moderate, its performance was significantly lower than that of EfficientNet B0, indicating that parameter count alone does not directly determine classification accuracy. ResNet50, leveraging deep residual learning, achieved 84.36% accuracy with the lowest loss value (0.35), demonstrating stable gradient flow and effective optimization. Skip connections likely mitigated vanishing gradients issues, leading to more stable learning. However, despite its strong recall of 84.37% and precision of 80.83%, its large parameter size (100.17 MB) makes it the most computationally expensive model in this comparison. Its lower loss while maintaining accuracy similar to EfficientNet B0 suggests that residual connections improve convergence stability, particularly in deep networks.

Lastly, GoogleNet, with an accuracy of 88%, performed well, particularly excelling in an F1 score of 93%, indicating a strong balance between precision and recall. However, its higher loss of 0.96 indicates some instability in model convergence, possibly due to the inception module's complexity, which may introduce optimization challenges. With a moderate parameter size of 85.51 MB, GoogleNet is a viable alternative to VGG16 and ResNet50, offering competitive performance while being relatively lighter in terms of computation. Its parallel convolutional architecture contributes to strong extraction of features, but its higher loss suggests potential challenges in refining decision boundaries.

## Comparison of Model Performance

The comparison of model performance highlights important trade-offs between accuracy, parameter size, and computational efficiency. VGG16 and GoogleNet achieved the highest accuracy but have larger parameter sizes, making them more resource-intensive. ResNet50, while also highly accurate, has the largest parameter size, affecting computational efficiency. DenseNet offers a balance between model size and moderate accuracy, whereas EfficientNet B0, though compact, sacrifices some performance. While models like ResNet50 and VGG16 generalize well, they require significant computational resources, whereas smaller models like EfficientNet B0 and DenseNet prioritize efficiency but may require additional tuning for specific tasks.

This study offers several advantages over previous research. Unlike Maeda-Gutiérrez et al. (2020), who evaluated CNNs on a subset of PlantVillage (focusing only on tomato diseases), our analysis spans all 38 classes, providing a more comprehensive benchmark. Additionally, the inclusion of multiple performance metrics such as accuracy, loss, recall, F1 score, precision, and parameter size surpasses the single-metric focus in Bhagat et al. (2020), enabling a more holistic evaluation. Compared to Kaya and Gürsoy (2023), who achieved 98.17% accuracy using a multi-head CNN, our study prioritizes widely established models for broader accessibility, albeit with a slight accuracy trade-off (e.g., VGG16 at 93.75%). However, some limitations remain. Unlike Albattah et al. (2022), who tested on 14 species with custom architectures, our reliance on the PlantVillage dataset limits species diversity. Additionally, real-time deployment was not explored, unlike in Chen et al. (2020), who achieved 99.85% accuracy with MobileNet-Beta, underscoring a gap in practical applicability.

## Statistical Validation

To assess whether performance differences are statistically meaningful, paired *t*-tests were conducted on accuracy across five runs per model. VGG16 (93.75% ± 1.1) significantly outperformed DenseNet201 (69.32% ± 1.5%, $p < 0.01$) and EfficientNet B0 (84.37% ± 1.3, $p < 0.05$), with 95% confidence intervals confirming robustness. ResNet50 (84.36%

± 1.2) and GoogleNet (88% ± 1.4) showed no significant difference ($p = 0.12$), suggesting dataset bias (e.g., uniform lighting) may favor certain architectures. These results indicate that VGG16's superiority is not due to chance, though DenseNet201's underperformance may reflect bias toward controlled images.

### Error Analysis

The performance metrics in Table 2 highlight potential misclassification patterns across the five CNN models, inferred from their recall, precision, and loss values. VGG16, with high recall (93.75%) and precision (90.63%), likely excels at distinguishing healthy leaves from diseased ones but may struggle with subtle disease variants, such as early blight and late blight in potatoes, due to its deep architecture overfitting to prominent features like large lesions (Soumya Prasad et al., 2024). GoogleNet's balanced recall (88%) and precision (87.46%), paired with a higher loss (0.96), likely confuses diseases with similar symptoms, such as tomato bacterial spot as a viral infection, reflecting its inception modules' sensitivity to complex patterns. EfficientNet B0's lower precision (80.10%) and recall (84.38%) indicate frequent errors in distinguishing visually similar conditions, such as apple scab versus cedar rust, due to its compact design prioritizing efficiency over fine-grained feature extraction. ResNet50 (recall 84.37%, precision 80.83%) likely performs similarly, with its residual learning mitigating some errors but not fully resolving subtle class overlaps. DenseNet201's notably low recall (68.75%) and F1 score (67.81%), combined with a high loss (1.06), suggest significant misclassifications, potentially labeling diseased leaves as healthy, stemming from its dense connectivity struggling with PlantVillage's controlled, uniform images. These inferred errors underscore the models' varying sensitivities to symptom subtlety, a limitation tied to dataset bias toward distinct, lab-captured disease appearances.

### Trade-offs and Deployment Considerations

VGG16 offers the highest accuracy at 93.75%. However, it comes with a significant computational cost, requiring 56.79 MB of storage and an exceptionally high inference time of 37,733 s, making it impractical for real-time applications. In contrast, EfficientNet B0 sacrifices some accuracy at 84.37% in favor of efficiency, featuring a smaller model size of 18.62MB and a significantly lower inference time of 2,718 s, making it well-suited for mobile and edge deployments. GoogleNet strikes a strong balance with 88.00% accuracy, a faster inference time of 2,625 s, and superior generalization, as reflected in its high F1-score of 93.00%. This makes it an effective solution for precision agriculture and automated disease detection. ResNet50 and DenseNet, despite some advantages, do not stand out in terms of accuracy, efficiency, or speed, making them less ideal for real-world deployment. For cloud-based applications where computational resources are not

a limitation, VGG16 remains the best choice. At the same time, for real-time, low-power, or mobile deployments, EfficientNet B0 and GoogleNet are the most suitable options due to their efficiency and faster inference times.

## CONCLUSION

This paper compares five CNN models (VGGNet-16, GoogleNet, EfficientNet, ResNet50, and DenseNet201) for plant disease classification using the PlantVillage dataset. The study examines trade-offs between model accuracy, parameter size, and computational efficiency in DL applications for plant disease classification. VGG16 and GoogleNet achieved the highest accuracy but required larger parameter sizes, making them resource-intensive. ResNet50 offered good accuracy and had a large model size, while DenseNet201 and EfficientNet B0 offered more compact architectures at the cost of reduced performance. The choice of an optimal model depends on specific requirements: larger models are well-suited for high-accuracy tasks, whereas smaller models are more efficient for resource-constrained environments. Future research should focus on optimizing model architectures to balance performance and computational efficiency, improving disease detection in agricultural applications. While this study provides a comparative analysis of CNN architectures for plant disease classification, several areas of future research could further enhance model performance, as follows:

One promising direction is the development of hybrid models that integrate Recurrent Neural Networks (RNNs) with CNNs for time-series analysis of plant health. Such models could monitor disease progression over time, enabling more robust and dynamic detection systems.

Incorporate multi-source datasets with images collected under diverse lighting conditions, weather variations, and crop species. This would improve model generalization and robustness, reducing biases that arise from controlled dataset environments.

Investigate the applicability of Transformer-based models (e.g., ViTs and Swin Transformers) to plant disease classification and compare their efficiency, accuracy, and computational requirements against traditional CNN architectures.

Another avenue for exploration is the deployment of lightweight models like MobileNet for real-time disease detection on edge devices. This advancement could revolutionize precision agriculture, allowing farmers to monitor plant health using AI-powered cameras integrated into mobile devices or drones.

## ACKNOWLEDGEMENTS

The authors would like to thank the Air Force Institute of Technology, Kaduna, for their academic support throughout the research.

## REFERENCES

Abebe, M., Tiumay, Z., Mohapatra, S. K., Prasad, S., Chaudhury, K. S., & Sahoo, P. (2025). Development of an SDG-driven Convolutional Neural Network (CNN) model for multi-class classification of tomato leaf diseases using combined public and local datasets. *Journal of Integrated Science and Technology*, *13*(3), 1063. https://doi.org/10.62110/SCIENCEIN.JIST.2025.V13.1063

Abouelmagd, L. M., Shams, M. Y., Marie, H. S., & Hassanien, A. E. (2024). An optimized capsule neural networks for tomato leaf disease classification. *EURASIP Journal on Image and Video Processing*, *2024*, 2. https://doi.org/10.1186/s13640-023-00618-9

Ahmed, H. A., Hama, H. M., Jalal, S. I., & Ahmed, M. H. (2023). Deep learning in grapevine leaves varieties classification based on dense convolutional network. *Journal of Image and Graphics*, *11*(1), 98–103. https://doi.org/10.18178/joig.11.1.98-103

Albattah, W., Nawaz, M., Javed, A., Masood, M., & Albahli, S. (2022). A novel deep learning method for detection and classification of plant diseases. *Complex and Intelligent Systems*, *8*, 507–524. https://doi.org/10.1007/s40747-021-00536-1

Alirezazadeh, P., Schirrmann, M., & Stolzenburg, F. (2023). Improving deep learning-based plant disease classification with attention mechanism. *Gesunde Pflanzen*, *75*, 49–59. https://doi.org/10.1007/s10343-022-00796-y

Bhagat, M., Kumar, D., Haque, I., Munda, H. S., & Bhagat, R. (2020). Plant leaf disease classification using grid search based SVM. In *2nd International Conference on Data, Engineering and Applications* (pp. 1–6). IEEE. https://doi.org/10.1109/IDEA49133.2020.9170725

Chen, J., Yin, H., & Zhang, D. (2020). A self-adaptive classification method for plant disease detection using GMDH-Logistic model. *Sustainable Computing: Informatics and Systems*, *28*, 100415. https://doi.org/10.1016/j.suscom.2020.100415

Chowdhury, N., Sultana, J., Rahman, T., Chowdhury, T., Khan, F. T., & Chakraborty, A. (2024). Potato leaf disease detection through ensemble average deep learning model and classifying the disease severity. *Indonesian Journal of Electrical Engineering and Computer Science*, *35*(1), 494–502. https://doi.org/10.11591/ijeecs.v35.i1.pp494-502

Demilie, W. B. (2024). Plant disease detection and classification techniques: A comparative study of the performances. *Journal of Big Data*, *11*, 5. https://doi.org/10.1186/s40537-023-00863-9

Food and Agriculture Organization. (2021). *Making agrifood systems more resilient to shocks: Lessons from the COVID-19 pandemic*. FAO. https://www.fao.org/newsroom/detail/fao-agrifood-systems-agriculture-resilience-SOFA-covid/en

Gomez, D., Selvaraj, M. G., Casas, J., Mathiyazhagan, K., Rodriguez, M., Assefa, T., Mlaki, A., Nyakunga, G., Kato, F., Mukankusi, C., Girma, E., Mosquera, G., Arredondo, V., & Espitia, E. (2024). Advancing common bean (*Phaseolus vulgaris* L.) disease detection with YOLO driven deep learning to enhance agricultural AI. *Scientific Reports*, *14*, 15596. https://doi.org/10.1038/s41598-024-66281-w

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 770–778). IEEE. https://doi.org/10.1109/CVPR.2016.90

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2261–2269). IEEE. https://doi.org/10.1109/CVPR.2017.243

Hughes, D. P., & Salathe, M. (2015). *An open access repository of images on plant health to enable the development of mobile disease diagnostics*. arXiv. https://doi.org/10.48550/arXiv.1511.08060

Jain, S., & Ramesh, D. (2021). AI based hybrid CNN-LSTM model for crop disease prediction: An ML advent for rice crop. In *12th International Conference on Computing Communication and Networking Technologies* (pp. 1–7). IEEE. https://doi.org/10.1109/ICCCNT51525.2021.9579587

Javidan, S. M., Banakar, A., Rahnama, K., Vakilian, K. A., & Ampatzidis, Y. (2024). Feature engineering to identify plant diseases using image processing and artificial intelligence: A comprehensive review. *Smart Agricultural Technology*, *8*, 100480. https://doi.org/10.1016/j.atech.2024.100480

Junaid, M., & Gokce, A. (2024). Global agricultural losses and their causes. *Bulletin of Biological and Allied Sciences Research*, *2024*(1), 66. https://doi.org/10.54112/bbasr.v2024i1.66

Kaya, Y., & Gürsoy, E. (2023). A novel multi-head CNN design to identify plant diseases using the fusion of RGB images. *Ecological Informatics*, *75*, 101998 https://doi.org/10.1016/j.ecoinf.2023.101998

Khan, A. T., Jensen, S. M., Khan, A. R., & Li, S. (2023). Plant disease detection model for edge computing devices. *Frontiers in Plant Science*, *14*, 1308528. https://doi.org/10.3389/fpls.2023.1308528

Kulkarni, A. H., & Ashwin Patil, R. K. (2012). Applying image processing technique to detect plant diseases. *International Journal of Modern Engineering Research*, *2*(5), 3661–3664.

Le, V. N. T., Ahderom, S., Apopei, B., & Alameh, K. (2020). A novel method for detecting morphologically similar crops and weeds based on the combination of contour masks and filtered Local Binary Pattern operators. *GigaScience*, *9*(3), giaa017. https://doi.org/10.1093/gigascience/giaa017

Leite, D., Brito, A., & Faccioli, G. (2024). Advancements and outlooks in utilizing Convolutional Neural Networks for plant disease severity assessment: A comprehensive review. *Smart Agricultural Technology*, *9*, 100573. https://doi.org/10.1016/j.atech.2024.100573

Maeda-Gutiérrez, V., Galván-Tejada, C. E., Zanella-Calzada, L. A., Celaya-Padilla, J. M., Galván-Tejada, J. I., Gamboa-Rosales, H., Luna-García, H., Magallanes-Quintanar, R., Guerrero Méndez, C. A., & Olvera-Olvera, C. A. (2020). Comparison of Convolutional Neural Network architectures for classification of tomato plant diseases. *Applied Sciences*, *10*(4), 1245. https://doi.org/10.3390/app10041245

Ngugi, H. N., Akinyelu, A. A., & Ezugwu, A. E. (2024). Machine learning and deep learning for crop disease diagnosis: Performance analysis and review. *Agronomy*, *14*(12), 3001. https://doi.org/10.3390/agronomy14123001

Ngugi, L. C., Abelwahab, M., & Abo-Zahhad, M. (2021). Recent advances in image processing techniques for automated leaf pest and disease recognition – A review. *Information Processing in Agriculture*, *8*(1), 27–51. https://doi.org/10.1016/j.inpa.2020.04.004

Padshetty, S., & Ambika. (2023). Leaky ReLU-ResNet for plant leaf disease detection: A deep learning approach. *Engineering Proceedings*, *59*(1), 39. https://doi.org/10.3390/engproc2023059039

Parez, S., Dilshad, N., & Lee, J. W. (2025). A channel attention-driven optimized CNN for efficient early detection of plant diseases in resource constrained environment. *Agriculture*, *15*(2), 127. https://doi.org/10.3390/agriculture15020127

Perumal, V. K., Supriyaa, T., Santhosh, P. R., & Dhanasekaran, S. (2024). CNN based plant disease identification using PYNQ FPGA. *Systems and Soft Computing*, *6*, 200088. https://doi.org/10.1016/j.sasc.2024.200088

Rinu, R., & Manjula, S. H. (2021). Plant disease detection and classification using CNN. *International Journal of Recent Technology and Engineering*, *10*(3), 152–156. https://doi.org/10.35940/ijrte.c6458.0910321

Rodríguez-Lira, D. C., Córdova-Esparza, D. M., Álvarez-Alvarado, J. M., Terven, J., Romero-González, J. A., & Rodríguez-Reséndiz, J. (2024). Trends in machine and deep learning techniques for plant disease identification: A systematic review. *Agriculture*, *14*(12), 2188. https://doi.org/10.3390/agriculture14122188

Savaş, S. (2024). Application of deep ensemble learning for palm disease detection in smart agriculture. *Heliyon*, *10*(17), e37141. https://doi.org/10.1016/j.heliyon.2024.e37141

Shoaib, M., Shah, B., EI-Sappagh, S., Ali, A., Ullah, A., Alenezi, F., Gechev, T., Hussain, T., & Ali, F. (2023). An advanced deep learning models-based plant disease detection: A review of recent research. *Frontiers in Plant Science*, *14*, 1158933. https://doi.org/10.3389/fpls.2023.1158933

Simonyan, K., & Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*. arXiv. https://doi.org/10.48550/arXiv.1409.1556

Sofuoğlu, C. I., & Birant, D. (2024). Potato Plant leaf disease detection using deep learning method. *Journal of Agricultural Sciences*, *30*(1), 153–165. https://doi.org/10.15832/ankutbd.1276722

Soumya Prasad, S., Sampath Kumar, L., Mallem, S. N., Gutta, H., & Ahmed, R. (2024). Deep learning techniques for a comparative study of crop disease detection. In V. Goar, M. Kuri, R. Kumar, & T. Senjyu (Eds.), *Advances in information communication technology and computing* (pp. 407–423). Springer. https://doi.org/10.1007/978-981-97-6106-7_25

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1–9). IEEE. https://doi.org/10.1109/CVPR.2015.7298594

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking model scaling for Convolutional Neural networks*. arXiv. https://doi.org/10.48550/arXiv.1905.11946

Thalor, M., Mate, S., Shiralkar, A., & Shinde, A. (2025). Ensemble learning framework for mango plant disease detection and classification. *Journal of Information Systems Engineering and Management*, *10*(4s), 297–303. https://doi.org/10.52783/jisem.v10i4s.501

Verma, S., Kumar, P., & Singh, J. P. (2025). MLP-GNAS: Meta-learning-based predictor-assisted Genetic Neural Architecture Search system. *Applied Soft Computing*, *169*, 112527. https://doi.org/10.1016/j.asoc.2024.112527