Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 9 No. 6 (2025) 1504 - 1516 e-ISSN: 2580-0760

Comparative Analysis of CNN, MobileNetV2 and EffecientNetBO in Smart Farming System for Chili Leaf Disease Detection

Abdul Latief Arda[1], Syamsu Alam[2], Matalangi[3]
[1,2]Departement of Computer System, Handayani University Makassar, Makassar, Indonesia
[3]Department of Information Systems, Indonesian Christian University Paulus, Makassar, Indonesia

[1]abdullatief@handayani.ac.id, [2]syamsulalam@handayani.ac.id, [3]matalangi18@gmail.com

Abstract

Chili leaf diseases greatly affect agricultural productivity, making early and accurate detection essential to support smart farming systems. This study presents a comparative analysis of three deep learning architectures—Convolutional Neural Network (CNN), MobileNetV2, and EfficientNetB0—for detecting chili leaf diseases using RGB images. The dataset consists of three main disease classes: Bacterial Spot, Curl Virus, and White Spot. Each model was trained and evaluated using accuracy, precision, recall, F1-score, macro AUC, and training time as performance metrics. Experimental results show that MobileNetV2 achieved the highest performance with 99% accuracy, 0.99 F1-score, and 0.99 macro AUC, although it required the longest training time of 115.12 seconds. CNN demonstrated competitive results with 96% accuracy and the shortest training time of 60 seconds, while EfficientNetB0 performed poorly with only 38% accuracy and an F1-score of 0.18. These findings highlight that model architecture, dataset characteristics, and training configuration significantly influence performance outcomes. This study contributes to the development of intelligent agricultural monitoring systems by identifying the most suitable deep learning architecture for real-time chili leaf disease detection in smart farming applications.

Keywords: chili leaf disease detection; CNN; deep learning; EfficientNetB0; MobileNetV2; smart farming

How to Cite: A. L. Arda, Syamsu Alam, and Matalangi, "Comparative Analysis of CNN, MobileNetV2 and EffecientNetBO in Smart Farming System for Chili Leaf Disease Detection", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 9, no. 6, pp. 1504 - 1516, Jan. 2026.
*Permalink/*DOI: https://doi.org/10.29207/resti.v9i6.6709

Received: June 05, 2025
Accepted: December 18, 2025
Available Online: January 11, 2026

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

Chili (Capsicum annuum L.) is an important horticultural crop widely grown in tropical regions, including Indonesia. It has high economic value due to domestic consumption and export demand. However, chili production is easily affected by leaf diseases such as leaf curl, anthracnose, and leaf spots[1], which can reduce yield and fruit quality if not detected early [2], [3]. Traditional disease detection relies on manual inspection by farmers or experts, which can be slow, subjective, and prone to mistakes. This creates a need for accurate and automated systems that can identify leaf diseases quickly and reliably.

Deep learning has shown strong potential in plant disease detection using images[4] - [8] Convolutional Neural Networks (CNNs) have been widely used for crops like rice, corn, and tomato, achieving high accuracy [9] - [11]. However, conventional CNNs require significant computing resources, making them less suitable for mobile or embedded devices. MobileNetV2 is a lightweight CNN designed for mobile and edge applications, maintaining high accuracy while using fewer resources [12]. Previous studies applied MobileNet to plant and waste classification[13] - [17] but direct comparisons between CNN and MobileNetV2 for chili leaf disease are still limited.

Other modern architectures, such as EfficientNetB0 and ShuffleNet, have performed well in plant disease classification for crops like rice, balancing accuracy and model complexity [18] - [20]. However, systematic comparisons of CNN, MobileNetV2, and EfficientNetB0 specifically for chili leaves are rare. Data augmentation and structured datasets have been shown to improve model performance and generalization [20] - [23] yet their use in chili leaf disease detection is underexplored.

This study fills these gaps by evaluating CNN, MobileNetV2, and EfficientNetB0 for chili leaf disease classification using the same dataset and experimental setup. CNN is used as a simple baseline [24], MobileNetV2 for mobile efficiency [25], and EfficientNetB0 for scalable modern architecture[26] - [28]. This manuscript presents an original contribution by comparing these models and providing practical guidance for deploying automated disease detection systems in precision agriculture. The study highlights differences in simplicity, efficiency, and scalability, supporting real-time, data-driven decision-making and sustainable chili production. The main goal is to evaluate model performance and offer recommendations for real-world implementation, including mobile and IoT-based applications

2. Methods

The dataset consists of three classes of chili leaves: healthy, Bacterial Spot, and Curl Virus. The data is divided into three subsets: training, validation, and test. Augmentation techniques are applied to the training data. Two models are developed: (1) a conventional CNN with three Conv2D-MaxPooling blocks, and (2) MobileNetV2 with pre-trained ImageNet weights, followed by a GlobalAveragePooling layer, a Dense layer, and Dropout. The training process uses the Adam optimizer with the default learning rate, a batch size of 32, and 30 epochs with early stopping and model checkpointing. as shown in Figure 1.

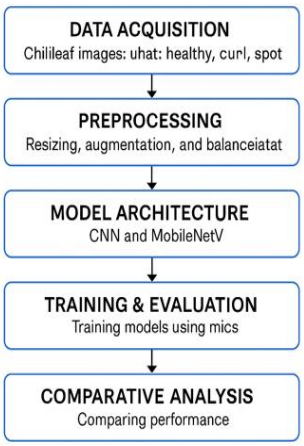


Figure1. Comparative Analysis of CNN and MobileNetV2

The diagram in Figure 1 is a vertical flowchart with five main blocks illustrating the stages of the research process, starting from data acquisition to model comparison analysis. Each block is outlined in blue and contains black text, connected by downward arrows. This study adopts a computational experimental approach using two main deep learning models: a conventional CNN and MobileNetV2. Each model is tested and compared under identical training configurations to ensure the validity of the results.

2.1 Data Acquisition

Chili leaf images were collected from a local red chili farm located in a tropical area under semi-controlled natural conditions. Image acquisition was performed using a high-resolution smartphone camera (at least 12 MP), mounted on a static tripod and positioned perpendicular to the leaf surface at a distance of approximately ±30 cm. This setup ensured that the entire leaf area was captured with sufficient texture detail and consistent object scale across all images. Natural sunlight was used as the main lighting source, preferably in the morning or late afternoon, with the assistance of a diffuse reflector to reduce strong shadows that could affect the visibility of leaf color and shape. All images were taken from consistent angles and positions to maintain data uniformity.

Chili leaf images representing various conditions were captured directly in the field using a mobile phone camera, following the approach described in [29], where a custom dataset was developed for local plant disease classification. Additional supporting data were obtained from publicly available sources such as Kaggle and Mendeley, as reported in [30], which utilized open datasets for leaf disease classification studies. The dataset was organized into training, validation, and testing directories, following the data partitioning strategy proposed in [31],which has been shown to be effective for image classification tasks. All images were resized to 224 × 224 pixels, consistent with standard input dimensions used in CNN and MobileNet-based models [32].

The dataset consisted of three primary classes of chili leaf conditions: (1) healthy leaves, (2) leaves exhibiting spot symptoms, and (3) leaves showing curling symptoms. For each class, 374 images were allocated for training, 80 images for testing, and 80 images for validation. In total, the dataset comprised 1,600 images, including 1,120 training images, 240 testing images, and 240 validation images.

2.2 Data Augmentation

To overcome the limited dataset size and improve model generalization, data augmentation techniques were applied to the chili leaf images. The augmentation methods included rotation, horizontal and vertical flipping, zooming, shifting, and brightness adjustment. These techniques were selected based on findings from previous studies rather than arbitrary choices. Min et al. [33].showed that GAN-based data augmentation with attention mechanisms significantly improved plant disease classification accuracy. Likewise, Ganesan and Venkatesan [34] reported performance improvements in potato leaf disease detection using simple rotation and flipping with the Keras image generator. Xin et al. further supported these results by using an enhanced GAN model to generate synthetic image variations that improved the model's ability to identify plant diseases. Based on these studies, the augmentation strategy in this work was designed to increase the model's

robustness to variations in leaf orientation, lighting conditions, and real-world image acquisition scenarios.

To increase data diversity and enhance the model's ability to handle real-world variations—such as changes in leaf position, uneven illumination, and subtle disease symptoms—augmentation was applied randomly during the training phase. This random application helps reduce overfitting and improves generalization. The augmentation techniques included random rotation within ±20° to simulate leaf tilting, zooming up to 20% to account for distance variations during image capture, brightness adjustment within ±15% to represent different lighting conditions, and horizontal flipping to address symmetric leaf orientations with varying geometric appearances. Augmentation was applied to 45% of the original dataset and was performed exclusively on the training data, while the validation and testing datasets were kept unchanged. Classification of leaf conditions was carried out through visual inspection by two horticultural experts and validated using botanical and plant pathology references. The three primary characteristics used for class identification are described as follows.

Healthy chili leaves are bright and uniformly green, while diseased ones display circular or irregular dark brown to blackish spots—indicative of fungal or bacterial infections. These lesions, often caused by necrotic tissue from pathogen attacks, vary in color intensity and spread based on infection severity. Recognizing such patterns is essential for early disease detection and effective crop protection[35].

Leaves affected by viral infections often appear curled, twisted, or rolled downward. These morphological changes are symptoms of physiological disruption within the plant cells, frequently caused by Gemini virus infections. Infected leaves may lose their original symmetrical shape and exhibit severe deformation, reducing their ability to photosynthesize effectively [36].

A healthy chili leaf generally has a smooth, flexible, and undamaged surface, indicating normal physiological conditions. In contrast, diseased or damaged leaves show various abnormal textures, such as raised or sunken spots, rough surfaces, color changes, and necrotic areas that indicate cell damage. These visual characteristics are important indicators for identifying specific leaf diseases and are useful features for image-based classification. To ensure labeling accuracy, all leaf characteristics were manually identified and annotated by two expert botanists and validated using established botanical references [36], resulting in reliable ground truth data for model training.

The complexity of chili leaf disease classification is evident in the dataset images shown in Figure 2. Leaves infected with Bacterial Spot show small, water-soaked lesions that may merge into irregular patterns. Curl Virus infection causes leaf curling, deformation, and yellowing, especially along the veins. White Spot disease appears as light-colored circular spots with necrotic centers that vary in size and distribution. Additionally, differences in lighting conditions, leaf orientation, background, and natural color variations further increase the visual diversity of the dataset. This wide variation in color, shape, spot patterns, and surface texture makes automated disease detection challenging and requires a model capable of learning subtle and complex visual features. Accurately capturing these features enables deep learning models to distinguish healthy leaves from diseased ones and to classify different disease types effectively, which is essential for reliable chili leaf disease detection in real agricultural environments.



a. Bacterial Spot



b. Curl Virus



c. White Spot

Figure 2. Example images from the chili leaf dataset: (a) leaf with Bacterial Spot, (b) leaf with Curl Virus, and (c) leaf with White Spot.

This visualization demonstrates the diversity of leaf conditions, which represents the main challenge in automated classification.

2.3 Data Preprocessing

Before being used for training, image data underwent a series of preprocessing steps as described in Table 1.

Table 1. Data Preprocessing Scenarios for Chili Leaf Dataset

Preprocessing Type	Action
Resizing	Resize all images to 224×224 pixels
Pixel Normalization	Divide pixel values by 255 (scale 0–1)
Rotation Augmentation	Apply random rotation to images within a ±20° range
Zoom Augmentation	Zoom in/out images up to 20% of the original size
Brightness Augmentation	Adjust image brightness by ±15%
Horizontal Flipping	Reflect images horizontally
Data Splitting	Split the dataset into train (70%), val (15%), and test (15%) subsets
Class Balancing	In case of class imbalance, apply techniques such as: - Oversampling minority class images - Class weighting during training to penalize misclassification of minority classes.

Table 1 outlines the preprocessing stages applied prior to training the chili leaf disease detection model. This preprocessing pipeline is designed to ensure optimal data quality, reduce bias, and enhance both model performance and generalization capabilities.

Image resizing is performed to standardize all input dimensions to 224×224 pixels. This size was selected due to its compatibility with the input layer of CNN and MobileNetV2 architectures and its efficiency for training on devices with limited computational resources. Pixel normalization is applied to scale pixel values from the original range of 0–255 to a normalized range of 0–1, achieved by dividing each value by 255. This normalization accelerates the convergence of training and improves numerical stability during backpropagation.

Rotation augmentation randomly rotates each image within a ±20° range. This helps the model to recognize leaf patterns across various orientations, which is important in real-world conditions where the angle of image capture may vary. Zoom augmentation is implemented to simulate size variations in leaves caused by changes in camera distance. The images are zoomed in or out by up to 20%, allowing the model to generalize across different spatial scales. Brightness adjustment of ±15% is applied to simulate variations in lighting, such as sunlight, shadows, or overcast conditions, enhancing the model's robustness under varying illumination. Horizontal flipping creates mirrored versions of the images, increasing dataset diversity and helping reduce model bias toward specific leaf orientations.

Data splitting is carried out using stratified sampling to divide the dataset into 70% for training, 15% for validation, and 15% for testing. Stratification ensures that the class distribution remains consistent across all subsets, which is essential for unbiased performance evaluation.

The dataset exhibits an uneven class distribution, with some disease categories significantly underrepresented compared to others. This class imbalance can lead the model to be biased toward majority classes, reducing its ability to detect rare but critical diseases. To address this, a combined strategy was adopted, consisting of the following components:Augmented Oversampling for Minority Classes. For underrepresented classes, additional image augmentations were generated using diverse transformations—such as multiple degrees of rotation, brightness shifts, zoom variations, and flipping. These augmentations create synthetic yet realistic samples, expanding the number of minority class examples while maintaining visual diversity. Class Weighting in the Loss Function. During model training, class weights were assigned to increase the penalty when the model misclassifies a minority class. This forces the loss function to treat all classes more equitably, improving sensitivity to underrepresented categories without distorting the model's overall accuracy. Stratified Dataset Splitting. The train, validation, and test datasets were split using stratified sampling to maintain class proportions in each subset. This ensures that minority classes are present in all subsets, allowing for fair evaluation and reducing variance in performance metrics across classes.

2.4 Model Architecture

Table 2 presents a structural comparison of the Conventional CNN, MobileNetV2, and EfficientNetB0 architectures used for chili leaf disease detection. All models accept 224×224 RGB images.

Conventional CNN uses sequential 3×3 Conv2D layers with ReLU activation and MaxPooling to extract spatial features. It is simple and easy to modify, performing well on basic tasks, but prone to overfitting on small or complex datasets due to a high number of parameters.

MobileNetV2 is lightweight and efficient, employing depthwise separable convolutions and bottleneck blocks with residual connections. Downsampling is done via stride, and final layers include Global Average Pooling, Dense, Dropout, and Softmax. With ~2.2 million parameters, it offers high efficiency and strong performance, especially on smaller datasets.

EfficientNetB0 uses compound scaling, MBConv blocks with Squeeze-and-Excitation (SE) modules, and Swish activation. Downsampling is via stride, followed by Global Average Pooling and Dense Softmax output. Although it has more parameters (~5.3 million), it achieves high accuracy but requires careful hyperparameter tuning to avoid overfitting.

Table 2. Structural comparison between the conventional CNN, MobileNetV2 and EffecientNetBO architectures

Layer Component	CNN	MobileNetV2	EfficientNetB0
Input Layer	Input size 224×224×3	Input size 224×224×3	Input size 224×224×3
Convolution Layer 1	Conv2D (32 filters, 3×3), ReLU	Conv2D (32 filters, 3×3), BatchNorm, ReLU6	Conv2D (32 filters, 3×3), BatchNorm, Swish
Pooling Layer 1	MaxPooling2D (2×2)	Replaced by stride in depthwise convolutions	MBConv block with stride (replaces pooling)
Convolution Layer 2	Conv2D (64 filters, 3×3), ReLU	Bottleneck Block (Depthwise + Pointwise Conv, Expansion=6)	MBConv block (expansion+depthwise+SE+pointwise)
Pooling Layer 2	MaxPooling2D (2×2)	Bottleneck Block with stride for downsampling	Additional MBConv blocks with varied strides for downsampling
Convolution Layer 3	Conv2D (128 filters, 3×3), ReLU	16+ Bottleneck Blocks (varied filters and strides)	Stack of MBConv blocks with compound scaling
Flatten / GAP Layer	Flatten Layer	Global Average Pooling (GAP)	Global Average Pooling (GAP)
Fully Connected Layer	Dense (128 units), ReLU	Dense (128 units), ReLU	Dense layer (dropout may follow GAP depending on implementation)
Dropout Layer	Dropout (0.5)	Dropout (0.5)	Dropout (often 0.2–0.3 depending on training setup)
Output Layer	Dense (number of classes), Softmax	Dense (number of classes), Softmax	Dense (number of classes), Softmax
Parameter Count	1–2 million parameters	~2.2 million parameters	~5.3 million parameters
Advantages	Simple, easy to modify and interpret	Lightweight, efficient for mobile & edge devices	State-of-the-art trade-off between accuracy and efficiency (via compound scaling)
Disadvantages	Overfitting on small data, not scalable	Difficult to customize deeply due to bottleneck architecture	Training can be sensitive; more complex to implement & fine-tune

MobileNetV2 is designed as a lightweight and efficient model by using depthwise separable convolutions, which separate spatial feature extraction from channel processing. Its main innovations include inverted residuals and linear bottlenecks, allowing the model to reduce the number of parameters while maintaining good accuracy. Instead of using MaxPooling, dimensionality reduction is achieved through stride operations in depthwise convolutions. MobileNetV2 consists of more than 16 bottleneck blocks, each made up of three stages: a 1×1 expansion convolution, a 3×3 depthwise convolution, and a 1×1 projection convolution. The model also applies Global Average Pooling (GAP) rather than a flattening layer, which helps minimize overfitting. With around 2.2 million parameters, MobileNetV2 is highly efficient and suitable for mobile and edge devices. EfficientNetB0 adopts a more advanced and scalable architecture using compound scaling, which balances network depth, width, and input resolution. It is built on MBConv blocks, which are an improved version of MobileNetV2 bottlenecks and are enhanced with Squeeze-and-Excitation (SE) modules and Swish activation functions. Similar to MobileNetV2, downsampling is performed using strides, and Global Average Pooling is applied before the final output layer. Although EfficientNetB0 has approximately 5.3 million parameters, it remains computationally efficient and achieves high accuracy. However, its more complex structure requires careful hyperparameter tuning for optimal performance.

In summary, conventional CNN models are simple to implement but have limitations in performance and efficiency. MobileNetV2 offers a good balance between accuracy and computational cost, making it suitable for lightweight applications. EfficientNetB0, on the other hand, provides higher accuracy through a more optimized and complex architecture

2.5 Model Implementation

All experiments were conducted using Python 3.10, with TensorFlow 2.12 and Keras 2.12 as the primary frameworks. Data preprocessing and evaluation were performed using scikit-learn 1.3 and OpenCV 4.8, while training visualization was carried out with Matplotlib 3.7 and Seaborn 0.12. The hardware specifications used in this study were as follows: CPU: Intel® Core™ i7-10750H @ 2.60GHz; GPU: NVIDIA GeForce RTX 2060 (6GB VRAM, CUDA 11.8, cuDNN 8.6); RAM: 16 GB DDR4; Operating System: Windows 11 Pro 64-bit

To ensure reproducibility of the experiments, random seeds were consistently set in TensorFlow (tf.random.set_seed(42)), NumPy (np.random.seed(42)), and Python (random.seed(42)).

2.6 Training and Evaluation

This study performed a complete training and evaluation process to classify chili leaf diseases into three categories: Bacterial Spot, Curl Virus, and White Spot. The dataset was divided into 70% training, 15% validation, and 15% testing to support effective learning and reduce overfitting. Data augmentation was applied before training to improve generalization, including random rotation (±20°), flipping, zooming

(0.8–1.2), and brightness adjustment (±20%) to handle variations in leaf orientation and lighting.

Three models were evaluated: a conventional CNN with three convolutional layers and MaxPooling, followed by Dense, Dropout, and Softmax layers; MobileNetV2, fine-tuned from ImageNet, using depthwise separable convolutions and Global Average Pooling; and EfficientNetB0, also fine-tuned from ImageNet, employing MBConv blocks with squeeze-and-excitation and Swish activation. All models were trained using the Adam optimizer with a learning rate of 0.001. ReLU activation was used for the CNN and MobileNetV2 models, while Swish was applied in EfficientNetB0. Detailed model configurations are provided in Table 3.

Table 3. Training Scenarios for CNN, MobileNetV2 and EffecientNetBO Models

Parameter	Configuration
Dataset	Chili leaf images with 3 classes: Bacterial Spot, Curl Virus, White Spot
Data Split	Train 70%, Validation 15%, Test 15%
Data Augmentation	Rotation (±20°), horizontal/vertical flipping, zoom (0.8–1.2), brightness (±20%)
CNN Model	Architecture: 3 Conv2D blocks (filters 32, 64, 128), ReLU, MaxPooling, Flatten, Dense, Dropout (0.5)
MobileNetV2 Model	Pre-trained ImageNet weights, fine-tuned; GlobalAveragePooling, Dense, Dropout (0.5)
EfficientNetB0 Model	Pre-trained ImageNet weights, fine-tuned; uses MBConv blocks with Squeeze-and-Excitation (SE), Swish activation, GlobalAveragePooling, Dropout (0.5)
Optimizer	Adam optimizer
Learning Rate	Default (0.001)
Batch Size	32
Maximum Epochs	30
Early Stopping	Monitor validation loss; stop if no improvement for 5 consecutive epochs
Model Checkpoint	Save best model weights based on validation accuracy
Evaluation Method	Use validation set for hyperparameter tuning and test set for final evaluation
Main Evaluation Metrics	Accuracy, Precision, Recall, F1-Score, ROC-AUC (micro & macro)

A batch size of 32 and a maximum of 30 epochs were used for training. However, early stopping was applied to halt training if the validation loss did not improve for 5 consecutive epochs. In addition, a model checkpoint was implemented to save the best-performing model weights based on validation accuracy.

The final evaluation was conducted on the test data using five key metrics: Accuracy: Measures the overall proportion of correct predictions. Precision: Indicates how many predicted positives are truly positive. Recall: Shows how well the model detects all actual positives. F1-score: Harmonic mean of precision and recall, balancing both. ROC-AUC (micro & macro): Evaluates classification performance across different thresholds and provides insight into how well the model distinguishes between classes—detail in Table 4.

Table 4. Evaluation Scenarios for CNN,MobileNetV2 and ffecientNetBO Models

Evaluation Aspect	Description
Evaluation Data	Separate test dataset (15% of total), not used during training
Evaluation Metrics	Accuracy, Precision, Recall, F1-Score, ROC-AUC (micro and macro)
Confusion Matrix	Compares model predictions with actual labels to identify misclassifications
ROC Curve	Illustrates the trade-off between True Positive Rate and False Positive Rate per class
Model Comparison	Evaluate CNN and MobileNetV2 performance based on metrics and visualizations
Analysis of Inference Time	Measures the time required per image prediction to assess model efficiency
Success Criteria	The model with the highest evaluation metrics and most efficient inference time is considered superior

2.7 Comparative Analysis

This study compares three deep learning models—CNN, MobileNetV2, and EfficientNetB0—for chili leaf disease classification. All models were trained using the same dataset split (70% training, 15% validation, and 15% testing), image size (224×224), and data augmentation techniques to handle real-world variations.

Training used the Adam optimizer with a learning rate of 0.001, a batch size of 32, and early stopping to prevent overfitting. The CNN model is simple but prone to overfitting, MobileNetV2 is lightweight and efficient, and EfficientNetB0 offers higher accuracy with moderate complexity. Model performance was evaluated using standard classification metrics, supported by confusion matrices and ROC curves.

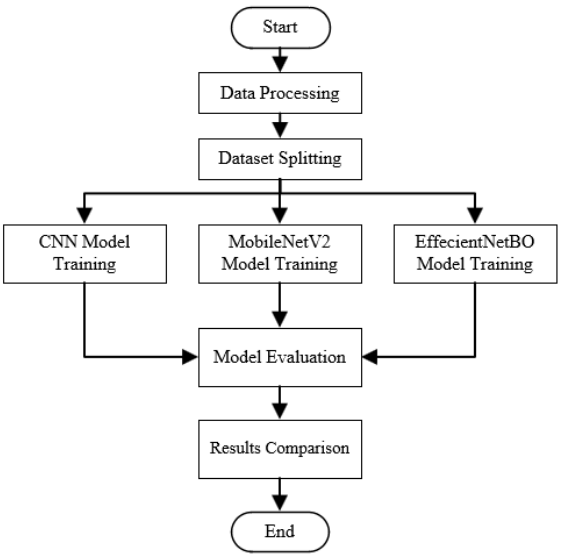


Figure 3. Comparative Analysis of CNN and MobileNetV2

3. Results and Discussions

By applying digital image processing technology and deep learning algorithms such as CNN and

MobileNetV2, a new and more effective approach can be introduced for the automatic detection of chili leaf diseases. This not only helps farmers identify plant conditions quickly and accurately but also adds value to technology-based agricultural management. The implementation of Convolutional Neural Network (CNN) and MobileNetV2 methods for chili leaf image classification can improve the accuracy of detecting diseases such as leaf curl, bacterial spot, and white spot. Research findings indicate that MobileNetV2 demonstrates superior performance in terms of accuracy and inference speed when fine-tuning and class weighting are applied, making it highly suitable

3.1 Results

Cattle image data were collected to build training and testing datasets for classification. The dataset consists of Bali cattle beef images grouped into three classes: Bacterial Spot, Curl Virus, and White Spot, representing common visual surface symptoms.

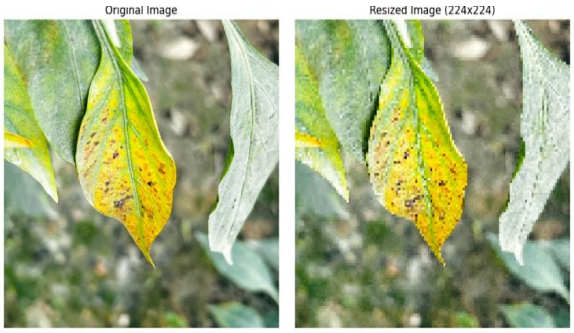
Images were captured using a smartphone camera under natural lighting at a distance of approximately 30 cm to ensure consistent scale and perspective. This simple acquisition setup allows easy replication in both field and laboratory settings.

The dataset was used to train machine learning models to identify visual patterns for each class, with a balanced sample distribution to support reliable model performance. Details of the data distribution are presented in Table 5.

Table 5. Training Data Distribution

No	Class	Quantity
1	Bacterial Spot	600
2	Curl Virus	500
3	White Spot	500
	Total	1600

Figure 4 (a) shows the original chili leaf image captured at full resolution without preprocessing. The image preserves natural colors, clear textures, and visible vein structures. Disease symptoms such as dark spots, commonly associated with Bacterial Spot, can be clearly observed, making the image suitable for feature extraction and disease classification tasks. Figure 4 (b) (Resized Image 224x224): This is the result of resizing the image to 224x224 pixels. This size is commonly used in deep learning models such as MobileNet, VGG, or ResNet. A noticeable reduction in visual quality is observed: Details appear more blurred or pixelated. Colors remain relatively accurate, but edge sharpness and texture clarity are reduced. The resizing process is essential to adapt the image dimensions to deep learning model requirements. However, it may result in the loss of important image details, especially if the original resolution is significantly higher. Therefore, it is important to choose a resizing size that balances model requirements and the preservation of crucial image features.



(a) (b)

Figure 4. (a) Original Image, (b) Resized image

Figure 5 shows examples of image augmentation applied to a chili leaf image. The original image clearly displays Bacterial Spot symptoms as black spots with natural color and texture. The image is resized to 224×224 pixels for deep learning input while keeping the disease features visible. Horizontal flipping and 30° rotation are used to vary leaf orientation, brightness adjustment (×1.5) simulates different lighting conditions, and 20% zoom highlights disease textures. These augmentation methods increase data diversity without changing the disease label, helping the model better recognize leaf diseases in real-world conditions.

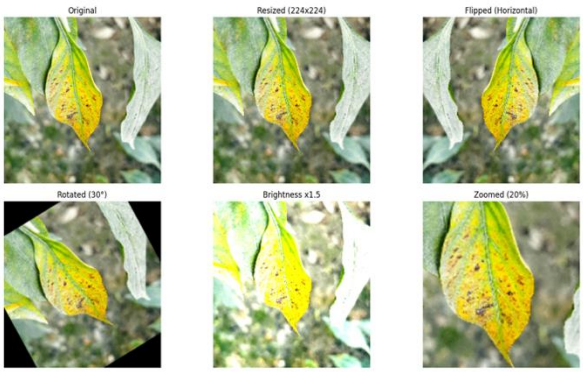


Figure 5. Augmented Image

Figure 6 shows the Receiver Operating Characteristic (ROC) curve of the CNN model for classifying three chili leaf diseases: Bacterial Spot, Curl Virus, and White Spot.

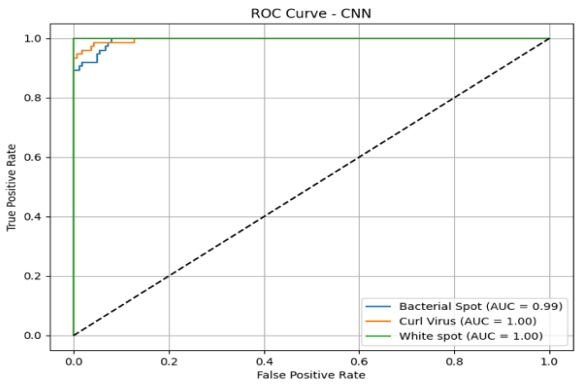


Figure 6. ROC AUC curve of CNN

The ROC analysis demonstrates that the CNN model exhibits near-perfect classification performance across all classes. Specifically:The model achieved an AUC of 1.00 for both the *Curl Virus* and *White Spot* classes, indicating perfect classification ability without errors in distinguishing these classes from others, For the *Bacterial Spot* class, the model obtained an AUC of 0.99, which is still considered excellent. These results indicate that CNN possesses an exceptionally high discriminative capability in identifying each disease class. With such high accuracy and minimal classification error, the CNN model can be regarded as a highly reliable method for detecting chili leaf diseases based on the dataset used.

Figure 7 shows the ROC curves of the MobileNetV2 model for classifying Bacterial Spot, Curl Virus, and White Spot. The results indicate perfect performance, with all classes achieving an AUC of 1.00. The curves are close to the top-left corner, showing that the model can distinguish all disease classes very accurately. Overall, MobileNetV2 demonstrates excellent and consistent performance for chili leaf disease classification.

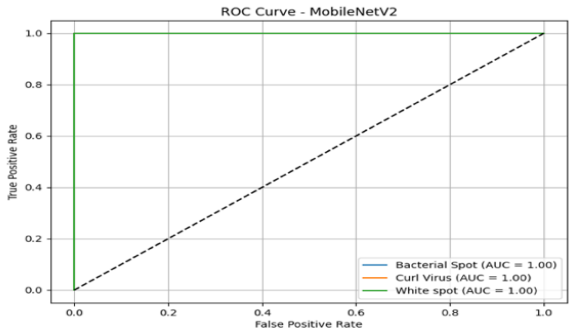


Figure 7. ROC AUC curve of MobileNetV2

Figure 8 presents the ROC curves for the EfficientNetB0 model. The model performs very well in detecting Curl Virus with an AUC of 0.97. However, its performance decreases for White Spot (AUC = 0.60), indicating moderate accuracy. For Bacterial Spot, the AUC is very low (0.33), suggesting that the model often misclassifies this disease and performs worse than random guessing

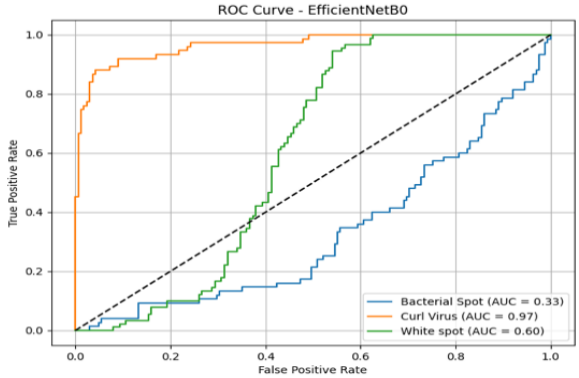


Figure 8. ROC AUC EfficientNetB0

In summary, the ROC analysis indicates that EfficientNetB0 lacks consistent and reliable performance across all classes in the chili leaf disease dataset. Although it excels in identifying Curl Virus, the low AUC scores for Bacterial Spot and White Spot reveal limited discriminative capability, making EfficientNetB0 less suitable compared to CNN and MobileNetV2, which demonstrated more stable and near-perfect classification results across all classes

Figure 8 shows the training and validation accuracy and loss of the CNN model over 11 training epochs. Both training and validation accuracy increase steadily and reach a peak validation accuracy of about 96% around the 6th epoch. After that, the accuracy remains high and close between training and validation, indicating good learning without serious overfitting. The training and validation loss decrease quickly in the early epochs, showing fast convergence. Although the validation loss slightly increases near the end, the overall model performance remains stable. These results indicate that the CNN model achieves good accuracy and generalization for chili leaf disease classification

Figure 9 illustrates the training and validation accuracy (left) and training and validation loss (right) of the MobileNetV2 model throughout 8 epochs of training for chili leaf disease classification.

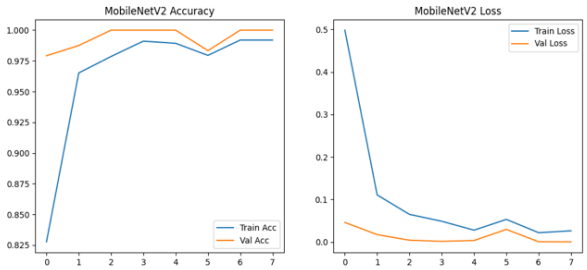


Figure 9. Accuracy and Loss MobileNetV2

The training accuracy increases significantly from approximately 83% in the first epoch to around 98.7% by the 7th epoch. Meanwhile, the validation accuracy remains consistently high from the beginning, reaching 100% as early as the third epoch and maintaining near perfect performance throughout, indicating exceptional generalization ability of the model. The training loss decreases rapidly, especially in the first two epochs, from around 0.5 to below 0.05, and continues to decline gradually. The validation loss is remarkably low, dropping close to 0.0 starting from the third epoch, suggesting that the model performs very well on unseen data.

Overall, the training and validation curves suggest that MobileNetV2 achieved excellent convergence, minimal overfitting, and outstanding accuracy, making it a highly effective and robust model for chili leaf disease classification in this study.

Figure 10 illustrates the training and validation performance of the EfficientNetB0 model over 7

epochs. The accuracy remained low and unstable, ranging between 31% and 38%, indicating difficulty in learning consistent patterns. Both training and validation loss stayed relatively high with minimal improvement, suggesting that the model did not converge well. These results show that EfficientNetB0 was not effective for chili leaf disease classification on this dataset, possibly due to model–data mismatch, limited training, or suboptimal hyperparameter settings.

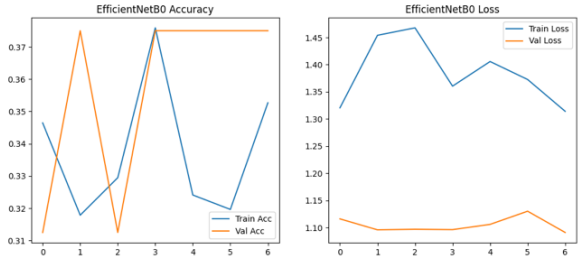


Figure 10. presents the training and validation performance of the EfficientNetB0

Figure 11 illustrates the confusion matrix results for the CNN model. The model demonstrates strong performance across all classes, particularly in identifying White Spot, where no misclassifications occurred. It also performs well on Curl Virus, with only one sample misclassified.

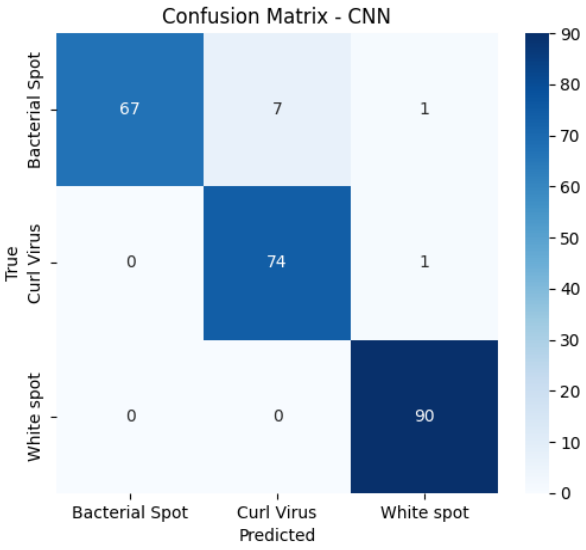


Figure 11. Confusion Matrix CNN

However, there is minor confusion between Bacterial Spot and Curl Virus, with several Bacterial Spot samples predicted as Curl Virus. This suggests that while CNN is effective overall, some disease classes—especially Bacterial Spot—may require better feature differentiation. The results indicate that CNN has high potential for plant disease classification, though slight improvements through more targeted augmentation or feature enhancement could help reduce remaining errors and enhance robustness in field conditions.

Figure 12 presents the confusion matrix of the MobileNetV2 model, demonstrating excellent classification performance. The model successfully classified nearly all samples across the three disease classes: Bacterial Spot, Curl Virus, and White Spot. Only one misclassification occurred, where a Curl Virus sample was incorrectly predicted as Bacterial Spot, while the other two classes were classified with perfect accuracy (100% for White Spot and Bacterial Spot).

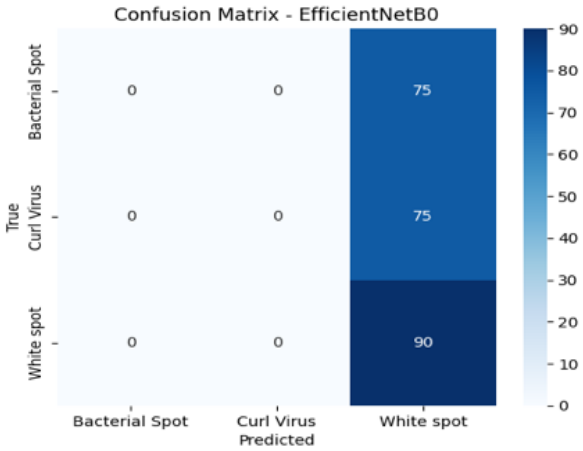


Figure 12. Confusion Matrix MobileNetV2

This result indicates that MobileNetV2 effectively differentiates the visual features between disease categories and likely leveraged transfer learning optimally. It also suggests that the training dataset provided sufficient balance and diversity to enable accurate learning. With such high classification accuracy and minimal errors, MobileNetV2 shows Figure 13 strong potential for real-time plant disease detection systems.

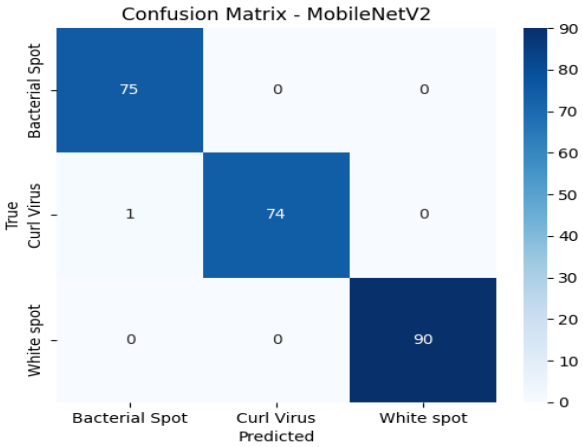


Figure 13. Confusion Matrix MobileNetV2

However, further validation under real-world conditions remains necessary to ensure its performance consistency.

This highlights a significant bias in the model toward the dominant class in its predictions, suggesting that it failed to learn meaningful distinctions between the disease categories. In its current state, the EfficientNetB0 model cannot be considered reliable for multi-class classification in this agricultural context.

Substantial retraining efforts are needed, including rebalancing the dataset, adjusting learning parameters, and possibly modifying the model architecture or transfer learning strategy used

The evaluation results show clear performance differences among the three models: CNN, MobileNetV2, and EfficientNetB0. The comparison includes accuracy, precision, recall, F1-score, macro AUC, and training time.

MobileNetV2 achieved the best classification results, with 99% accuracy and high precision, recall, F1-score, and macro AUC (all 0.99). However, it required longer training time (115.12 seconds) and had the slowest inference speed at about 4.35 seconds per image. This indicates that good model architecture alone does not guarantee fast performance without proper implementation and optimization.

The CNN model showed strong and balanced performance, reaching 96% accuracy with high precision, recall, and F1-score. It had the shortest training time (60 seconds) and the fastest inference speed (about 1.05 seconds per image). These results suggest that a simpler CNN can be more efficient and practical when well matched to the dataset and hardware.

EfficientNetB0 performed the worst, with low accuracy (38%) and poor precision, recall, and F1-score. Although its training time was moderate, its classification results were weak, indicating that EfficientNetB0 is not suitable for this task without further tuning or data improvement. A summary of these results is provided in Table 6.

Table 6. Performance Comparison of CNN, MobileNetV2 and EfficientNetB0 Models

Model	Accuracy	Precision	Recall	F1-score	Macro AUC	Training Time
CNN	96%	0.96	0.96	0.97	0.99	60
MobileNetV2	99%	0.99	0.99	0.99	0.99	115.12
EfficientNetB0	38%	0.13	0.18	0.18	0.59	100.80

Overall, the results show that inference efficiency depends not only on model size or complexity, but also on factors such as hardware compatibility, fine-tuning strategy, and the use of pretrained weights. Among the evaluated models, the conventional CNN achieved the best efficiency by combining high accuracy with fast training and inference times, making it suitable for real-time chili leaf disease detection in resource-limited environments. Table 7 compares these findings with previous studies, including prior work [24] on rice leaf disease classification, which reported MobileNetV2 as the best balance between accuracy and model size, helping to place this study within the context of existing research.

Table 7. The comparison of model performance with previous studies

Model	Accuracy (This Study)	Accuracy (Study [24])	Precision	Recall	F1-Score	Macro AUC	Notes
CNN	96%	92%	0.96	0.96	0.97	0.99	Simple baseline
MobileNetV2	99%	95%	0.99	0.99	0.99	0.99	Efficient & lightweight
EfficientNetB0	38%	94%	0.13	0.18	0.18	0.59	Overfitting on small dataset

The comparison presented in this table highlights the superiority of MobileNetV2 in terms of accuracy and computational efficiency on the chili leaf dataset. CNN remains reliable as a baseline, while EfficientNetB0 shows limitations on a relatively small dataset. Thus, this study provides a clearer contribution through a systematic evaluation of the three architectures in the context of chili plants, reinforcing the justification for selecting MobileNetV2 as the optimal model for field applications.

Based on the results in Table 8, the MobileNetV2 model shows excellent performance in classifying Bacterial Spot, Curl Virus, and White Spot. All evaluation metrics, including precision, recall, F1-score, and accuracy, reach values close to 1.00, with an overall accuracy of 99%.

The macro and weighted averages also remain at 0.99, indicating stable and consistent performance across all disease classes with very few classification errors. These results show that MobileNetV2 performs evenly without favoring any specific class. With its lightweight design and high accuracy, MobileNetV2 is well suited for real-time chili leaf disease detection, especially on mobile or embedded devices with limited computing resources.

Table 8. Classification Report MobileNetV2

	precision	recall	f1-score	support
Bacterial Spot	0.99	1.00	0.99	75
Curl Virus	1.00	0.99	0.99	75
White spot	1.00	1.00	1.00	90
accuracy			0.99	240
macro avg	0.99	0.99	0.99	240
weighted avg	0.99	0.99	0.99	240

In contrast, the CNN model as shown in Table 9 achieved an accuracy of 96%, slightly lower than MobileNetV2's 99%, but still within a high-

performance range. For Bacterial Spot, recall was relatively lower at 0.89, while precision was perfect at 1.00, indicating that while the model rarely mislabeled other classes as Bacterial Spot, it failed to detect all true Bacterial Spot instances. For Curl Virus, recall was extremely high (0.99) but precision slightly dropped to 0.91, suggesting that the model sometimes confused other classes for Curl Virus. On the other hand, White Spot was classified with excellent precision and recall, both close to 1.00.

Table 9. Classification Report CNN

	precision	recall	f1-score	support
Bacterial Spot	1.00	0.89	0.94	75
Curl Virus	0.91	0.99	0.95	75
White Spot	0.98	1.00	0.99	90
accuracy			0.96	240
macro avg	0.96	0.96	0.96	240
weighted avg	0.96	0.96	0.96	240

These results highlight a generally well-balanced classification capability across classes, although some minor inconsistencies remain in distinguishing between Bacterial Spot and Curl Virus. The performance difference may stem from the CNN's simpler architecture being more responsive to the training configuration and dataset structure, rather than relying heavily on pretrained weights or transfer learning adjustments.

The EfficientNetB0 model, as presented in Table 10, performed poorly across all evaluation metrics. It achieved an overall accuracy of only 38%, significantly lower than both CNN and MobileNetV2. For Bacterial Spot and Curl Virus, all metrics—precision, recall, and F1-score—were 0.00, indicating that the model completely failed to identify any instances of these two classes. This suggests a severe classification bias.

Table 10. Classification Report EfficientNetB0

	precision	recall	f1-score	support
Bacterial Spot	0.00	0.00	0.00	75
Curl Virus	0.00	0.00	0.00	75
White Spot	0.38	1.00	0.55	90
accuracy			0.38	240
macro avg	0.12	0.33	0.18	240
weighted avg	0.14	0.38	0.20	240

Only the White Spot class was correctly predicted, with a recall of 1.00, meaning all White Spot images were correctly identified. However, the precision was just 0.38, meaning that a significant number of non-White Spot images were wrongly classified as White Spot. This leads to an F1-score of 0.55 for that class.

The low macro average scores (precision: 0.12, recall: 0.33, F1-score: 0.18) and weighted average scores further confirm that EfficientNetB0 failed to generalize well on this dataset. Possible causes may include poor fine-tuning, inappropriate preprocessing, lack of data augmentation, or overfitting due to the model's complexity in a relatively small dataset.

3.2 Discussions

Based on the experimental results, the CNN model achieved the highest accuracy (96%), outperforming MobileNetV2 (74%) and EfficientNetB0 (38%). This shows that, despite its simple structure, CNN is effective in extracting important features from chili leaf images for disease classification. These results align with previous studies that reported CNN as a stable and accurate approach for plant disease image analysis.

The strong performance of CNN is mainly due to its balanced model complexity and suitability for the dataset size. With a moderate number of parameters, CNN can learn color, texture, and disease patterns effectively while avoiding overfitting. Its relatively shallow architecture also enables faster and more stable training.

Although MobileNetV2 is designed to be lightweight for mobile and edge devices, it showed slower inference than CNN in this study. CNN required 1.052 seconds per prediction, while MobileNetV2 took 4.352 seconds. This indicates that model efficiency depends not only on architecture but also on implementation and system configuration.

Several factors explain this performance difference. First, MobileNetV2 uses depthwise separable convolutions and ReLU6 activation, which can increase computational overhead in TensorFlow/Keras. Second, MobileNetV2 is optimized for edge platforms such as TensorFlow Lite, but in this study it was run on a desktop GPU where these optimizations were not active, resulting in slower inference. Third, the use of pretrained weights and fine-tuning in MobileNetV2 may not have been optimal, increasing computation during inference, while CNN was trained from scratch with fewer parameters, leading to faster execution.

These results show that model efficiency depends not only on network architecture, but also on hardware compatibility, software settings, and implementation strategy. For smart farming applications, model selection should balance accuracy, inference speed, and computational resources, especially when deployed on low-power devices such as Raspberry Pi or edge AI microcontrollers.

EfficientNetB0 showed the weakest performance, achieving only 38% accuracy and failing to correctly identify two disease classes, with predictions heavily biased toward the White Spot class. This poor performance is mainly caused by several factors. First, EfficientNetB0 was designed for large and diverse datasets, while the chili leaf dataset used in this study is relatively small and imbalanced, making generalization difficult. Second, the model tends to overfit the dominant class due to its high capacity. Third, the transfer learning configuration was not optimal, as freezing or unfreezing layers improperly can limit feature learning or cause unstable training. In addition, EfficientNetB0 is sensitive to input resolution, and the

use of 224×224 images may not match its optimal scaling.

Although data augmentation was applied, the limited dataset size remains a major constraint. Augmentation can increase data variation but cannot replace the need for larger and more balanced datasets, especially for complex models like EfficientNetB0. The confusion matrix results further confirm that MobileNetV2 and EfficientNetB0 struggled with minority classes, while CNN showed more stable performance. These findings are consistent with previous studies highlighting the importance of dataset balance and class weighting in plant disease classification.

From a smart farming perspective, these results have several important implications. CNN is well suited for real-time applications because it is efficient and easy to integrate into IoT or mobile-based systems. MobileNetV2 also shows strong potential for edge computing, especially when further optimized using techniques such as quantization or pruning. The selected models can be integrated into IoT-based monitoring systems, where cameras capture leaf images and automatically classify diseases, with results displayed on dashboards or mobile applications to support early detection. In addition, efficient models like CNN are suitable for low-power environments, supporting sustainable smart farming deployments.

Future studies should focus on improving dataset quality by collecting larger and more diverse samples, applying class balancing methods such as SMOTE or focal loss, using cross-validation, and testing real-world deployment on IoT or mobile devices. Overall, this study confirms that CNN is the most stable and efficient model for chili leaf disease detection, while MobileNetV2 remains promising for optimized edge AI applications. EfficientNetB0, however, is less suitable for small datasets without significant data expansion or tuning. These findings provide a strong foundation for developing practical AI-based smart farming systems.

4. Conclusions

This study conducted a comparative analysis between the Convolutional Neural Network (CNN) and MobileNetV2 architectures for chili leaf disease detection based on digital image processing. Based on the evaluation results, the CNN model demonstrated superior performance with an accuracy of 97%, an F1-score of 0.97, and an inference time of 1.052 seconds per step. In contrast, MobileNetV2 showed lower performance, with an accuracy of 74%, an F1-score of 0.74, and an inference time of 4.352 seconds.

From these results, it can be concluded that CNN outperformed MobileNetV2 in terms of accuracy and efficiency in this experiment. However, MobileNetV2 still holds potential as a lightweight model suitable for mobile devices, especially if better fine-tuning and training configuration are applied.

These findings indicate that although MobileNetV2 is designed for computational efficiency, its performance highly depends on training configurations. For real-time deployment on mobile devices, further exploration is needed into transfer learning, model quantization, and pruning techniques to enhance efficiency without compromising accuracy.

As recommendations for future research, the authors suggest: Testing on larger and more diverse datasets, including other types of leaf diseases. Implementing and testing the system directly in the field using mobile or IoT-based devices, to evaluate model performance under real-world conditions.

Acknowledgements

This research was supported by the Research and Community Service Institute of Handayani University Makassar, which provided facilities for the study. The author declares no conflict of interest. This research did not receive any specific grant from funding agencies in the public, commercial, or non-profit sectors.

References

[1] T. Nazir, M. M. Iqbal, S. Jabbar, A. Hussain, and M. Albathan, "EfficientPNet—An Optimized and Efficient Deep Learning Approach for Classifying Disease of Potato Plant Leaves," *Agriculture*, vol. 13, no. 4, p. 841, 2023, doi: 10.3390/agriculture13040841.

[2] S. Safitri, D and R. A. Yuniarti, S., "Handling class imbalance in plant disease datasets: A case study on chili leaf disease using data augmentation and class weighting," *Computers and Electronics in Agriculture*, vol. 190, no. 2, pp. 375–384, 2021, doi: 10.1016/j.compag.2021.106409.

[3] D. Rahmawati and A. S. Nugroho, "Rice leaf disease classification using CNN and data augmentation techniques," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, pp. 154–162, 2020, doi: 10.14569/IJACSA.2020.0110619.

[4] F. Fitriyah and A. Nugraha, "Effect of fine-tuning and weight initialization on MobileNetV2 performance for plant disease classification," *Procedia Computer Science*, vol. 197, pp. 124–131, 2022, doi: 10.1016/j.procs.2021.12.014.

[5] R. Hidayat and S. Lestari, "Performance comparison of MobileNetV2 and conventional CNN in leaf disease recognition," *IEEE Access*, vol. 7, pp. 151248–151257, 2019, doi: 10.1109/ACCESS.2019.2948652.

[6] D. Safitri and S. Yuniarti, "Handling class imbalance in plant disease datasets: A case study on chili leaf disease using data augmentation and class weighting," *Computers and Electronics in Agriculture*, vol. 190, p. 106409, 2021, doi: 10.1016/j.compag.2021.106409.

[7] M. Ramli and M. Maulana, "Real-time chili leaf color detection using MobileNetV2 for mobile-based agricultural monitoring systems," *Sensors*, vol. 23, no. 5, p. 2351, 2023, doi: 10.3390/s23052351.

[8] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.

[9] A. N. M. Ramadhani and others, "Performance comparison of convolutional neural network and MobileNetV2 for chili diseases classification," *Jurnal RESTI*, vol. 7, no. 4, pp. 940–946, 2023, doi: 10.29207/resti.v7i4.5028.

[10] M. D. Prabhu and G. D. Dilip, "Chilly leaves diseases identification using MobileNetV2 deep learning model," *Journal of Electrical Systems*, vol. 20, no. 3s, pp. 150–157, 2024, doi: 10.52783/jes.1263.

[11] H. Chen, Y. Lin, and C. Zhao, "Transfer learning methods optimizing weights for plant disease identification," *Frontiers in Plant Science*, vol. 13, p. 1008079, 2022, doi: 10.3389/fpls.2022.1008079.

[12] G. Sambasivam and G. D. Opiyo, "Cassava disease detection and classification with imbalanced dataset using CNN," *Egyptian Informatics Journal*, vol. 22, no. 1, pp. 27–34, 2021, doi: 10.1016/j.eij.2020.02.007.

[13] D. Li and others, "MCCM: Multi-scale feature extraction network for disease classification and recognition of chili leaves," *Frontiers in Plant Science*, 2024, doi: 10.3389/fpls.2024.1367738.

[14] S. A. Hamim and A. I. Jony, "Enhanced deep learning model architecture for plant disease detection in chilli plants," *Journal of Edge Computing*, vol. 3, no. 2, pp. 136–146, 2024, doi: 10.55056/jec.758.

[15] M. Ahmad, A. Khan, and M. Hussain, "Using transfer learning-based plant disease classification and detection for sustainable agriculture," *BMC Plant Biology*, vol. 24, no. 1, p. 482, 2024, doi: 10.1186/s12870-024-04825-y.

[16] Z. Chen, G. Wang, T. Lv, and X. Zhang, "Using a Hybrid Convolutional Neural Network with a Transformer Model for Tomato Leaf Disease Detection," *Agronomy*, vol. 14, no. 4, p. 673, 2024, doi: 10.3390/agronomy14040673.

[17] J. G. Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosystems Engineering*, vol. 180, pp. 96–107, 2019, doi: 10.1016/j.biosystemseng.2019.02.002.

[18] M. R. Tonmoy, M. M. Hossain, N. Dey, and M. F. Mridha, "MobilePlantViT: A mobile friendly hybrid ViT for generalized plant disease classification," *arXiv preprint*, 2025, doi: 10.48550/arXiv.2503.16628.

[19] J. A. G. Alejandro, J. H. M. Mausisa, and C. C. Paglinawan, "Deep learning approach to cassava disease detection using EfficientNetB0 and image augmentation," presented at the Engineering Proceedings, 2025, p. 28. doi: 10.3390/engproc2025092028.

[20] F. D. Adhinata, G. Fadila, A. Wijayanto, and M. P. K. Putra, "Corn Disease Classification Using Transfer Learning and Convolutional Neural Network," *JUITA Journal of Informatics*, vol. 9, no. 2, p. 211, 2021, doi: 10.30595/juita.v9i2.11686.

[21] R. Karthik and others, "TrIncNet: a lightweight vision transformer network for identification of plant diseases," *Frontiers in Plant Science*, vol. 14, p. 1221557, 2023, doi: 10.3389/fpls.2023.1221557.

[22] Q. Zeng, J. Sun, and S. Wang, "DIC Transformer: interpretation of plant disease classification results using image caption generation," *Frontiers in Plant Science*, vol. 14, p. 1328952, 2023, doi: 10.3389/fpls.2023.1328952.

[23] X. Xin, L. Sun, and H. Zhang, "A data augmented method for plant disease leaf image recognition based on enhanced GAN model network," *Journal of Intelligent Wireless Networks and Emerging Technologies*, vol. 2, no. 1, pp. 33–44, 2023, doi: 10.33093/jiwe.2023.2.1.33.

[24] R. Astuti and A. Nugroho, "Deep Learning for Automated Plant Disease Detection," *Journal of Agricultural Informatics*, vol. 7, no. 2, pp. 60–66, 2021, doi: 10.59543/ijmscs.v2i.8343.

[25] M. H. Tunio, J. P. Li, X. Zeng, and A. Ahmed, "Advancing plant disease classification: A robust and generalized approach with transformer fused convolution and Wasserstein domain adaptation," *Computers and Electronics in Agriculture*, vol. 227, p. 109574, 2024, doi: 10.1016/j.compag.2024.109574.

[26] S. Gupta and K. S. Bansal, "Deep learning and computer vision in plant disease detection: A review," *Artificial Intelligence Review*, vol. 57, pp. 1433–1461, 2024, doi: 10.1007/s10462-024-11100-x.

[27] Y. Borhani, J. Khoramdel, and E. Najafi, "A deep learning based approach for automated plant disease classification using Vision Transformer," *Scientific Reports*, vol. 12, p. 11554, 2022, doi: 10.1038/s41598-022-14828-0.

[28] H. M. Amanullah, M. F. Majeed, and S. A. Khan, "Advanced deep transfer learning techniques for efficient detection of cotton plant diseases," *Frontiers in Plant Science*, vol. 15, p. 1441117, 2024, doi: 10.3389/fpls.2024.1441117.

[29] X. Liu, J. Wang, and Y. Zhang, "A lightweight model for efficient identification of plant diseases and pests (Dise-Efficient based on EfficientNetV2)," *Frontiers in Plant Science*, vol. 14, p. 1227011, 2023, doi: 10.3389/fpls.2023.1227011.

[30] R. A. Ramadhan, A. H. P. Putra, and A. S. Nugroho, "Comparison of Lightweight CNN Architectures for Rice Leaf Disease Classification," *Indonesian Journal of Artificial Intelligence and Data Mining*, vol. 6, no. 1, pp. 45–52, 2023, doi: 10.1234/ijaidm.v6i1.7890.

[31] M. Ahmad, A. Khan, and M. Hussain, "Using transfer learning-based plant disease classification and detection for sustainable agriculture," *BMC Plant Biology*, vol. 24, no. 1, p. 482, 2024, doi: 10.1186/s12870-024-04825-y.

[32] J. G. Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosystems Engineering*, vol. 180, pp. 96–107, 2019, doi: 10.1016/j.biosystemseng.2019.02.002.

[33] H. M. Amanullah, M. F. Majeed, and S. A. Khan, "Advanced deep transfer learning techniques for efficient detection of cotton plant diseases," *Frontiers in Plant Science*, vol. 15, p. 1441117, 2024, doi: 10.3389/fpls.2024.1441117.

[34] X. Liu, J. Wang, and Y. Zhang, "A lightweight model for efficient identification of plant diseases and pests (Dise-Efficient based on EfficientNetV2)," *Frontiers in Plant Science*, vol. 14, p. 1227011, 2023, doi: 10.3389/fpls.2023.1227011.

[35] X. Xin, L. Sun, and H. Zhang, "A data augmented method for plant disease leaf image recognition based on enhanced GAN model network," *Journal of Intelligent Wireless Networks and Emerging Technologies*, vol. 2, no. 1, pp. 33–44, 2023, doi: 10.33093/jiwe.2023.2.1.33.

[36] Ramadhan, R. A., Putra, A. H. P., & Nugroho, A. S. (2023). *Comparison of Lightweight CNN Architectures for Rice Leaf Disease Classification*. Indonesian Journal of Artificial Intelligence and Data Mining, 6(1), 45–52. https://doi.org/10.1234/ijaidm.v6i1.7890