Green Intelligent Systems and Applications

Research Article
Volume 6(1), 2026, 81−95
https://doi.org/10.53623/gisa.v6i1.1063

Potato Leaf Disease Classification Using MobileNetV3 Architecture with Adam and Stochastic Gradient Descent Optimizers

Hafizh Pebrian, Ery Hartati*

Faculty of Computer Science and Engineering, Universitas Multi Data Palembang, South Sumatra, Indonesia

*Correspondence: Ery_hartati@mdp.ac.id

SUBMITTED: 20 February 2026; REVISED: 8 March 2026; ACCEPTED: 15 March 2026

ABSTRACT: Potato leaf diseases such as Early Blight and Late Blight reduced productivity and could cause crop failure if they were not detected early. This study analyzed the comparative performance of the Adam and Stochastic Gradient Descent (SGD) optimizers using the MobileNetV3-Large architecture for potato leaf disease classification. The dataset consisted of three categories: healthy leaves, Early Blight, and Late Blight, with a total of 4,072 images. All images were processed through preprocessing stages, including resizing to 224 × 224 pixels and pixel value normalization. The data were divided into training, validation, and testing sets with a ratio of 70:20:10. Random undersampling and data augmentation techniques were applied to the training data to address class imbalance and improve the model's generalization capability. The model training process was conducted using a transfer learning approach with the MobileNetV3-Large architecture through two stages: feature extraction and fine-tuning. Model performance evaluation was based on accuracy, precision, recall, and F1-score metrics. The results showed that the Adam optimizer achieved a test accuracy of 98.75% with an F1-score of 0.9875, while the SGD optimizer achieved a test accuracy of 96.56% with an F1-score of 0.9635. The Adam optimizer also demonstrated faster and more stable convergence during the training process. This study was expected to serve as a reference for determining an appropriate optimizer for deep learning applications in image classification, particularly in plant disease detection.

KEYWORDS: Adam; CNN; image classification; MobileNetV3-Large; potato leaf disease; SGD

1. Introduction

Potato (*Solanum tuberosum* L.) was one of the most strategic food commodities in the world, including Indonesia, as it ranked among the major sources of carbohydrates after corn, wheat, and rice. Its high economic value made potato cultivation an important pillar for farmer welfare and national food security. However, potato plants were highly susceptible to various pathogen attacks on leaf tissue, which caused symptoms such as chlorosis, spots, and severe damage that reduced photosynthetic capacity and inhibited tuber formation.

The two major diseases that most severely affected potato productivity worldwide were Early Blight and Late Blight. Early Blight, caused by the fungus *Alternaria solani*, was

characterized by blackish-brown spots and was first reported in the United States around the 1880s. Meanwhile, Late Blight, caused by *Phytophthora infestans*, was far more destructive because of its ability to spread rapidly within a short period and was first reported in Europe in 1845 [1,2]. Based on research by Febrianto et al., the intensity of Late Blight and Early Blight attacks in Gowa Regency, South Sulawesi, reached up to 70%, with the highest average attack rate of 43.20% [3]. A delay in symptom identification of only a few days could lead to total crop failure, causing substantial economic losses for farmers.

This problem was exacerbated by disease identification methods that were still conducted through direct visual observation by farmers. This traditional approach proved inefficient because it depended heavily on subjective human visual perception and was prone to diagnostic errors [4]. Limited technical knowledge regarding disease differences at early stages often resulted in delayed treatment, which reduced the effectiveness of pesticide application and increased production costs without optimal results [5]. Therefore, a technology-based approach capable of providing accurate, fast, and objective diagnosis was required.

The application of digital image processing using Convolutional Neural Network (CNN) architectures provided significant advancements in automatic image object identification. CNN models were capable of performing automatic visual feature extraction directly from raw images without requiring complex manual feature engineering [6]. However, complex CNN architectures such as NASNetMobile, although achieving high accuracy, required very high computational resources [7]. This situation created a gap between technological capability and practical field requirements, as many farmers did not have access to high-performance computing infrastructure.

As a solution, the use of the MobileNetV3-Large architecture became highly relevant. This third-generation architecture developed by Google was designed to achieve a balance between high accuracy and computational efficiency on resource-constrained devices through a platform-aware Neural Architecture Search (NAS) mechanism [8]. By utilizing inverted bottleneck units and squeeze-and-excitation modules, MobileNetV3-Large produced rich feature representations while maintaining a lightweight structure, making it suitable for potential implementation on mobile or edge computing devices.

Although the model architecture was determined, the success of the classification process largely depended on the training configuration, particularly the selection of optimization algorithms. Two of the most commonly used optimizers were Adam and Stochastic Gradient Descent (SGD). Adam operated by calculating first- and second-order momentum adaptively for each parameter, which enabled rapid convergence during the early stages of training. However, its adaptive mechanism sometimes caused the model to converge to less optimal local minima and resulted in weaker generalization performance on unseen data. In contrast, SGD updated model parameters using randomly selected samples in each iteration, which theoretically provided a more dynamic convergence trajectory and often resulted in stronger generalization performance. Nevertheless, this stochastic update mechanism could produce significant gradient fluctuations, potentially leading to unstable training if learning rates were not properly configured [9].

This uncertainty was further highlighted by the phenomenon of domain shift. Research by Herwina et al. reported that the use of the Adam optimizer on MobileNetV2 for rice plant disease classification significantly improved model performance [10]. In contrast, research by Maximilliano and Rachmat found that the SGD optimizer applied to the MobileNetV3-Small

architecture for corn plant disease classification produced more stable performance [11]. However, findings from rice and corn plant studies could not be directly generalized to potato plant cases. Potato leaf images exhibited unique morphological characteristics, such as subtle concentric spot patterns in Early Blight and irregular wet lesion patterns in Late Blight. To date, no study had specifically examined how the training stability produced by Adam compared with SGD when handling the distinctive visual features of potato leaf diseases using the MobileNetV3-Large architecture.

Without a comprehensive comparative analysis, optimizer selection was often based on general assumptions, which risked producing models that were not optimal in terms of both accuracy and training stability. Therefore, this research was conducted to address this gap by comparatively evaluating the effect of Adam and SGD optimizers on the performance of the MobileNetV3-Large model for potato leaf disease classification.

This study contributed to the field of plant disease image classification in several ways. First, a comparative analysis of Adam and SGD optimizers was performed using the MobileNetV3-Large architecture specifically for potato leaf disease classification. Second, a two-stage transfer learning approach combined with random undersampling and data augmentation was implemented to address class imbalance. Finally, an empirical evaluation of each optimizer's performance was conducted using accuracy, precision, recall, and F1-score metrics to provide evidence regarding the most suitable training configuration for potato leaf disease classification.

2. Materials and Methods

2.1. Potato leaf diseases.

Potato leaf disease is a pathological disorder in potato plant leaf tissue caused by pathogen attacks such as fungi, bacteria, or viruses, which cause symptoms of spots, color changes, and complete leaf damage [12]. Leaf damage caused by pathogens reduces photosynthetic capacity, directly reducing tuber formation and decreasing potato crop yields. The most damaging potato leaf diseases are Early Blight and Late Blight. Early Blight is caused by the fungus *Alternaria solani*, characterized by blackish-brown concentric spots on old leaves, causing chlorosis and leaf drop [13]. Meanwhile, Late Blight is caused by *Phytophthora infestans* and is the most destructive disease, characterized by irregular blackish spots, white on the underside of leaves, and rapid spread until infecting the entire plant [14]. The impact of potato leaf diseases not only damages leaves as the main photosynthetic organ but also causes economic losses due to decreased crop yields and even crop failure. BMC Plant Biology reported that Late Blight was among the most destructive plant diseases globally and can cause yield losses of up to 100% if not controlled early [15].

2.2. Digital image processing.

Digital Image Processing is defined as a scientific discipline that studies techniques for processing images. The image in question can be a picture (photo) or moving images (from a webcam). Mathematically, an image can be viewed as a two-dimensional function $f(x, y)$, where x and y are spatial coordinates on a flat plane, while the value f at those coordinates represents intensity or gray level. A digital image can be represented as a two-dimensional matrix of size

$N \times M$. Each matrix element *f(x,y)* states the pixel intensity value at row coordinate *x* and column coordinate *y*. Parameter *N* indicates the number of image rows, while *M* indicates the number of image columns. This representation is written in numerical matrix form as follows [16]:

$$f\ (x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,\mathrm{M}-1) \\ f(1,0) & f(1,1) & \dots & f(1,\mathrm{M}-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(\mathrm{N}-1,0) & f(\mathrm{N}-1,1) & \cdots & f(\mathrm{N}-1,\mathrm{M}-1) \end{bmatrix} \quad (1)$$

An image *f(x, y)*, in mathematical function can be written as follows:

$0 \leq x \leq M-1$

$0 \leq y \leq N-1$

$0 \leq f(x,y) \leq G-1$

Where:
M = number of row pixels in the image array
N = number of column pixels in the image array
G = gray level scale value

The values of M, N, and G are generally powers of two: $M = 2^m; N = 2^n; G = 2^k$, where m, n and k are positive integers. The interval (0,G) is called grayscale. The size of G depends on the digitization process. Usually, gray level 0 (zero) represents black intensity and 1 (one) represents white intensity. For 8-bit images, the G value equals $2^8 = 256$ colors (gray levels) [16].

2.3. Deep learning.

Deep Learning is a subfield of machine learning developed based on the Artificial Neural Network structure with many hidden layers [17]. The fundamental advantage of this method lies in its ability to perform automatic feature extraction (automatic feature learning) directly from raw data, without requiring human intervention for manual feature engineering [17].

2.4. Convolutional Neural Network (CNN).

CNN is a type of neural network commonly used to analyze image data. CNN is very effective in detecting and recognizing image objects because it is specifically designed to process data with grid topology structure. CNN was first introduced in the 1960s and showed promising results in computer vision [18]. Technically, CNN is a trainable architecture consisting of several stages. CNN is a combination of image convolution that functions for feature extraction processes, and neural networks that function for classification. Based on the LeNet5 architecture, there are 4 main types of layers in a CNN: convolutional layer, ReLU layer, pooling layer, and fully connected layer. The CNN architecture can be seen in Figure 1 [18].

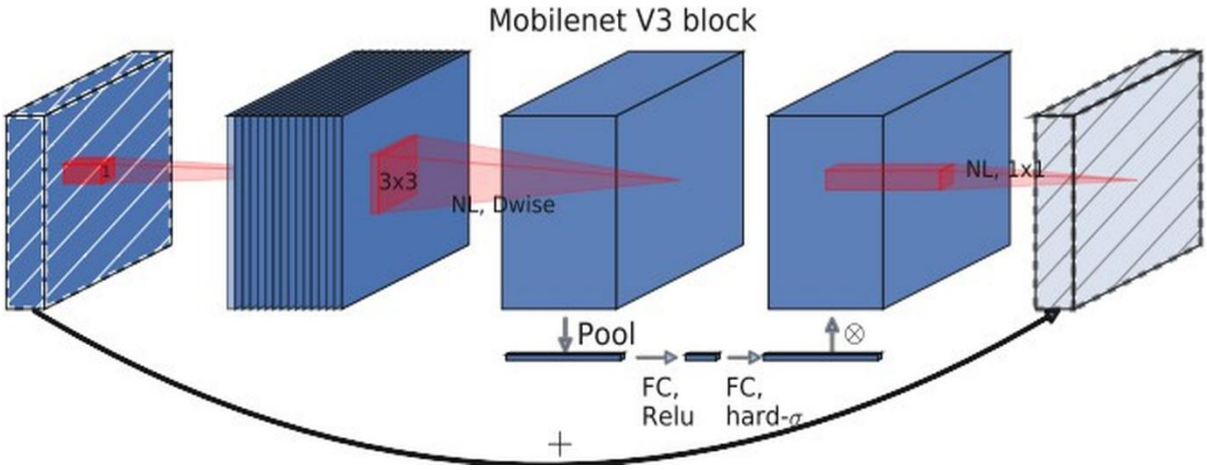

Figure 1. MobileNetV3 Architecture.

2.5. MobileNetV3 architecture.

MobileNetV3 is the third generation of CNN architecture developed by Google, with a main focus on computational efficiency on mobile devices and systems. The development of MobileNetV3 utilizes two main techniques: Platform-Aware Neural Architecture Search (NAS) to find optimal global network structures by balancing accuracy and latency, and the NetAdapt algorithm to perform fine-tuning on each layer. Additionally, supported by innovations such as the use of non-linear hard swish (h-swish) activation function and optimization on initial and final layers, it produces the best balance between performance and speed on devices with limited CPU resources [19].

2.5.1. MobileNetV3-Large.

MobileNetV3-Large is designed for devices with relatively high computing capabilities, with a target inference time of around 80 ms on mobile CPUs. This model was developed using a platform-aware Neural Architecture Search (NAS) approach, with MnasNet-A1 as the base model, then further refined through NetAdapt techniques. Its architectural structure consists of varying inverted bottleneck blocks, including different expansion ratios, use of 3×3 and 5×5 kernel sizes, and selective application of squeeze-and-excitation modules on several layers. Before entering the final classification layer, the model produces 1,280 feature channels. The activation function used is h-swish, which is applied especially in the final part of the network to suppress computational complexity without sacrificing accuracy performance [19].

Evaluation results on the ImageNet dataset showed that MobileNetV3-Large is capable of achieving Top-1 accuracy of 75.2% with 51 ms latency on Google Pixel 1 CPU. This model remains efficient with approximately 5.4 million parameters and computational requirements of 219 million MAdds operations. Compared to MobileNetV2, MobileNetV3-Large provides an accuracy increase of 3.2% while accelerating the inference process by about 20%, making it the right choice for various computer vision applications such as image classification, object detection, and semantic segmentation. Detailed specifications of the MobileNetV3-Large architecture can be seen in Table 1 [19].

Table 1. MobileNetV3-Large architecture specifications.

Input Size	Operator	Exp Size	Out	SE	NL	Stride
224×224	Conv2d 3×3	–	16	–	HS	2
112×112×16	Bottleneck 3×3	16	16	–	RE	1
112×112×16	Bottleneck 3×3	64	24	–	RE	2
56×56×24	Bottleneck 3×3	72	24	–	RE	1
56×56×24	Bottleneck 5×5	72	40	✓	RE	2
28×28×40	Bottleneck 5×5	120	40	✓	RE	1
28×28×40	Bottleneck 3×3	240	80	–	HS	2
14×14×80	Bottleneck 3×3	200	80	–	HS	1
14×14×80	Bottleneck 3×3	184	80	–	HS	1
14×14×80	Bottleneck 3×3	184	80	–	HS	1
14×14×80	Bottleneck 3×3	480	112	✓	HS	1
14×14×112	Bottleneck 3×3	672	112	✓	HS	1
14×14×112	Bottleneck 5×5	672	160	✓	HS	2
7×7×160	Bottleneck 5×5	960	160	✓	HS	1
7×7×160	Conv2d 1×1	–	960	–	HS	1
7×7×960	AvgPool 7×7	–	–	–	–	–
1×1×960	Conv2d 1×1	–	1280	–	HS	1

2.6. Transfer learning.

Transfer learning is a method of reusing models that have been thoroughly trained on large datasets. The goal is to apply the knowledge already obtained to smaller datasets, by only retraining the final layer of the model for adaptation to new classification tasks, while the initial feature layers are maintained [16].

2.7. Adam optimizer.

The Adam optimizer works by calculating first-order momentum (average gradient value) and second-order momentum (uncentered variance of gradients) at each training iteration. The model then utilizes both momentum estimates to produce adaptive learning rates for each model parameter, enabling faster and more stable convergence during the training process. The following are the Adam optimizer formulas:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1).g_t \quad (2)$$

$$v_t = \beta_2 v_t - 1 + (1-\beta_2).g_t^2 \quad (3)$$

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \quad (4)$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad (5)$$

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{\widehat{m}_t}{\sqrt{\hat{v}_t+\epsilon}} \quad (6)$$

Based on Eq. (2) – (6), the Adam optimizer moves parameters (θ) toward optimal solutions with update rate (α). Eq. (2) computes m_t as the moving average of gradients, while Eq. (3) computes v_t as the moving variance of gradients. Parameters β_1 and β_2 determine the memory weight of moment history (m_t and v_t). Eq. (4) and Eq. (5) apply bias correction to produce unbiased estimates $\widehat{m}_t$and $\hat{v}_t$, compensating for the zero-initialization of both

moments at the start of training. Finally, Eq. (6) performs the parameter update, where ε is a minimum value inserted to prevent division operations from becoming undefined [20].

2.8. SGD optimizer.

SGD is known as a fundamental optimization algorithm. Unlike Batch Gradient Descent, SGD updates model parameters by utilizing one random sample at each iteration, offering significant advantages in time efficiency and memory usage. However, this dependence on single samples has the potential to trigger high fluctuations in gradient estimation, which can disrupt stability during the training process and result in turbulent or inconsistent convergence trajectories [21]. The following is the SGD optimizer formula:

$$\theta_{t+1} = \theta_t - \eta \cdot g_t \quad (7)$$

Where:
θ_t : parameter to be optimized
η : learning rate
g_t : gradient of the loss function on the data

As shown in Eq. (7), the SGD optimizer updates parameter θ_t by subtracting the product of learning rate η and the current gradient g_t at each iteration. Unlike Adam, SGD does not maintain moment estimates, making it simpler but more sensitive to the choice of learning rate [20].

2.9. Undersampling.

Undersampling is a random undersampling (RUS) method that randomly selects the majority class and adds it to the minority class to form a new dataset. Random undersampling will discard datasets from the majority class in the training data, so that the majority class data will be reduced in number. The disadvantage of this technique is randomly deleting data in the majority class, allowing deleted data to be important or useful data in building classification models [22].

2.10. Dataset collection.

The dataset used in this study is the "Potato Disease Leaf Dataset (PLD)" obtained from Kaggle, consisting of 4,072 images collected from the Central Punjab region of Pakistan [7]. This dataset is grouped into three classes: Early Blight, Healthy, and Late Blight (Figure 2).

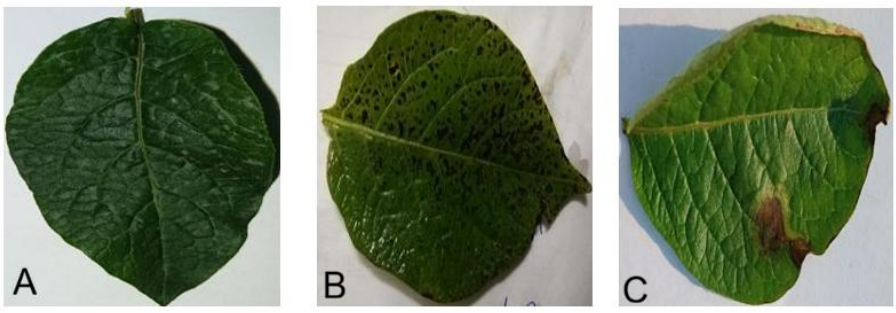

Figure 2. Potato leaf disease dataset samples: (A) Healthy leaf - healthy potato leaf without disease symptoms; (B) Early Blight - potato leaf infected with Early Blight disease showing concentric brown spots; (C) Late Blight - potato leaf infected with Late Blight disease showing irregular wet patches.

2.11. Research design.

This study designed a system to analyze the comparative performance of Adam and SGD optimizers for potato leaf disease classification using the MobileNetV3-Large architecture. The dataset consists of 4,072 images divided into three classes: Early Blight (1,628 images), Late Blight (1,424 images), and Healthy (1,020 images). The methodology was carried out through the following steps.

2.11.1. Preprocessing.

The preprocessing stage was carried out through two steps. First, each image was resized to 224 × 224 pixels to match the required input dimensions of the MobileNetV3-Large architecture. Second, pixel values were normalized to the range [0,1] by dividing each pixel value by 255, which accelerates model convergence and prevents numerical instability during the training process.

2.11.2. Data splitting.

The dataset was divided into three subsets through the following procedure. First, a splitting ratio of 70% for training, 20% for validation, and 10% for testing was applied. Second, a stratified split strategy was used to maintain class proportions across all subsets, ensuring each subset contains a representative proportion of Early Blight, Late Blight, and Healthy classes. Third, splitting was performed before undersampling and augmentation to prevent data leakage and ensure that validation and test subsets remain mutually exclusive from the training process. The data distribution is shown in Table 2.

Table 2. Data distribution, undersampling, and augmentation results.

Class	Initial Total	Training (70%)	Validation (20%)	Testing (10%)	Under sampling	Augmentation
Early Blight	1,628	1,140	325	163	714	2,856
Late Blight	1,424	997	285	142	714	2,856
Healthy	1,020	714	204	102	714	2,856
Total	**4,072**	**2,851**	**814**	**407**	**2,142**	**8,568**

2.11.3. Undersampling.

Based on the class distribution, the training data exhibited class imbalance with Early Blight having the highest number of samples (1,140) and Healthy the lowest (714). This imbalance may cause model bias toward the majority class, resulting in poor generalization on minority classes. To address this, random undersampling was applied exclusively to the training data to equalize the number of samples per class to 714 images, matching the minority class. Validation and test data were kept at their original distribution to ensure evaluation reflects real-world conditions.

2.11.4. Augmentation.

To mitigate domain bias arising from variations in lighting conditions, background, and image capture angles, image augmentation techniques were applied to the training data. The augmentation techniques applied include rotation, horizontal flip, vertical flip, and shear transformation. These techniques artificially increase the diversity of training samples, forcing

the model to recognize disease characteristics universally without being overly dependent on specific orientations or object positions. As a result, the model's generalization capability is improved and the risk of overfitting during training is reduced. Augmentation was applied only to the training data and not to the validation or test data. The results of undersampling and augmentation are shown in Table 2. The complete training parameters are shown in Table 3.

Table 3. MobileNetV3-Large training parameters.

Parameter	Configuration
Input Image Size	224 × 224
Batch Size	32
Epoch	30
Initial Stage (Freeze)	Epoch 1-10
Fine-Tuning Stage	Epoch 11-30
Unfrozen Layers	20% of the final MobileNetV3-Large layers
Dense Layer	256 neuron (*ReLU*)
Dropout	0.5
Optimizer	Adam and SGD
Learning Rate	0.001
Learning Rate Fine-Tuning	0.0001
Loss Function	Categorical Crossentropy

2.11.5. Model Development.

The MobileNetV3-Large architecture pre-trained on the ImageNet dataset was used as the primary feature extractor due to its efficiency in computational resources and strong feature extraction capability. The model structure was modified at the final layers by adding a Global Average Pooling layer to reduce feature dimensions, followed by a Dense layer with 256 neurons and ReLU activation, a Dropout layer with a rate of 0.5 as an overfitting mitigation mechanism, and a Softmax output layer with three output nodes corresponding to the three potato leaf disease classes.

2.11.6. Training.

The training process employed a two-stage transfer learning approach. In the first stage, feature extraction (epoch 1–10), all convolutional layers of MobileNetV3-Large were frozen and only the final classification layers were trained to allow the model to learn task-specific features while preserving the pre-trained weights. In the second stage, fine-tuning (epoch 11–30), approximately 20% of the final MobileNetV3-Large layers were unfrozen, specifically the last inverted bottleneck blocks, to further optimize high-level feature representations specific to the potato leaf disease dataset. To ensure an objective comparison, both Adam and SGD optimizer scenarios used identical training parameters: batch size 32, 30 epochs, learning rate 0.001, and fine-tuning learning rate 0.0001. Complete training parameters are shown in Table 3.

3. Results and Discussion

The performance of MobileNetV3-Large was evaluated using two optimization algorithms, Adam and SGD, on a potato leaf disease dataset consisting of three classes. Model performance was assessed based on confusion matrix results, including accuracy, loss, precision, recall, and F1-score metrics.

3.1. Model performance with Adam optimizer.

The test results indicated that the MobileNetV3-Large model trained using the Adam optimizer achieved a test accuracy of 98.75% with a test loss of 0.0495, demonstrating the model's ability to classify potato leaf diseases with high accuracy on data that was not seen during training. On the validation data, the model obtained a validation accuracy of 97.03% with a validation loss of 0.1140, reflecting strong and consistent performance throughout the training process. The confusion matrix for the Adam optimizer scenario is presented in Figure 3, providing a visual representation of the correct and incorrect predictions for each disease class.

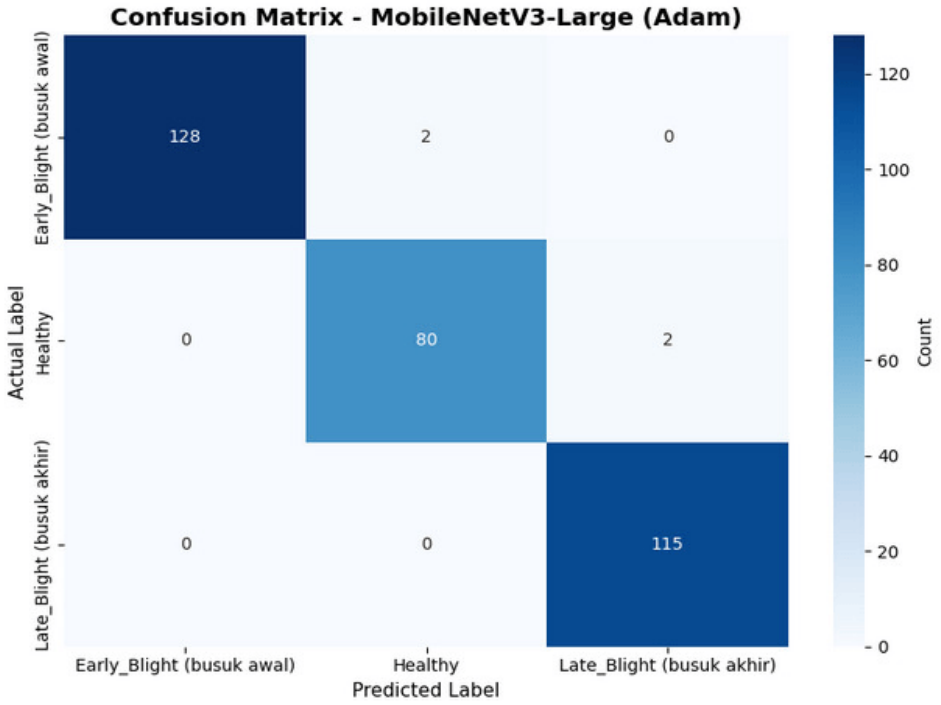

Figure 3. Confusion matrix of the model using the Adam optimizer.

Based on the classification report, the model using the Adam Optimizer showed excellent performance across all three classes: Early Blight, Healthy, and Late Blight, with high precision and recall values. A detailed comparison of precision, recall, and F1-score for both optimizers is presented in Table 4 in Section 3.3.

3.2. Model performance with SGD optimizer.

In the SGD optimizer training scenario, the MobileNetV3-Large model achieved a test accuracy of 96.56% with a test loss of 0.1409, demonstrating a satisfactory level of classification performance on unseen data. On the validation data, the model reached a validation accuracy of 94.06% with a validation loss of 0.2038, reflecting a consistent relationship between validation and test performance. The confusion matrix for the SGD optimizer scenario is presented in Figure 4.

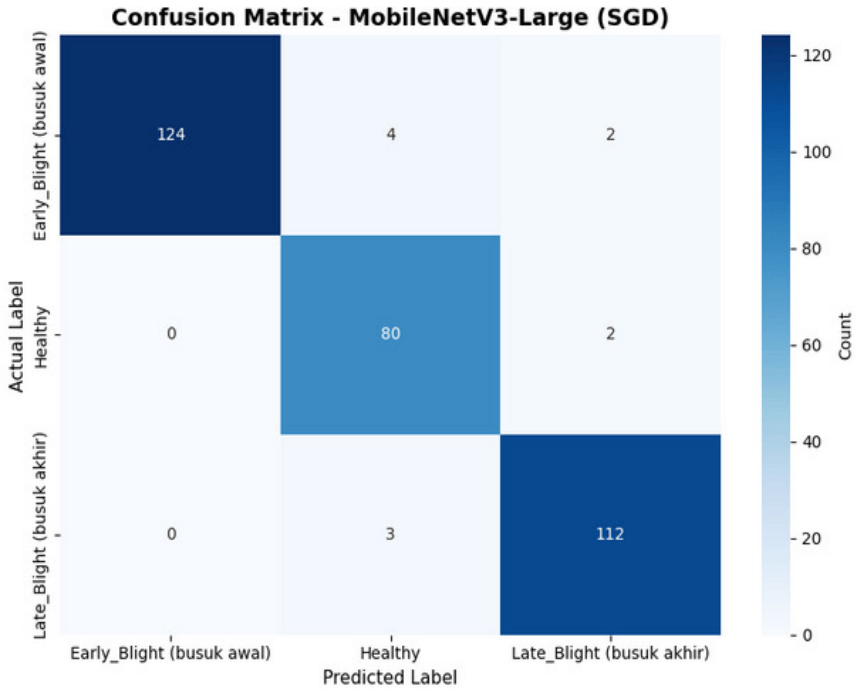

Figure 4. Confusion matrix of the model using the SGD optimizer.

The classification report showed that the MobileNetV3-Large model optimized with SGD achieved reasonably consistent performance across all classes. The Early Blight, Healthy, and Late Blight classes exhibited high precision and recall values, indicating that the model effectively captured the discriminative features of each disease. Overall, performance was relatively balanced across all classes, although the Healthy class showed a slightly lower precision compared to the others, suggesting that the model maintained good generalization capabilities despite a marginally lower overall accuracy relative to the Adam optimizer scenario. A detailed comparison of precision, recall, and F1-score for both optimizers is presented in Table 4.

Table 4. Comparison of precision, recall, and F1-Score for Adam and SGD optimizers.

Class	Adam Optimizer			SGD Optimizer		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Early Blight	1.00	0.98	0.99	1.00	0.95	0.98
Healthy	0.98	0.98	0.98	0.92	0.98	0.95
Late Blight	0.98	1.00	0.99	0.97	0.97	0.97

3.3. Comparison of Adam and SGD optimizer performance.

Based on the test results reflected in the accuracy and loss curves per epoch, the Adam optimizer demonstrated faster and more stable convergence compared to SGD, which tended to exhibit fluctuations particularly during the fine-tuning phase. This suggests that Adam is more consistent in updating model weights throughout the training process. The Adam optimizer achieved a test accuracy of 98.75% and an F1-score of 0.9875, whereas the SGD optimizer yielded a test accuracy of 96.56% and an F1-score of 0.9635. Furthermore, the gap between validation accuracy and test accuracy for Adam (~1.7%) was smaller than that of SGD (~2.5%), indicating that Adam not only outperformed SGD in terms of accuracy but also demonstrated greater consistency between validation and test performance. Therefore, the Adam optimizer proved to be more optimal than SGD for the potato leaf disease classification

task using the MobileNetV3-Large architecture. The superior performance of Adam can be attributed to its adaptive learning rate mechanism. Unlike SGD which applies a uniform learning rate to all parameters, Adam computes individual adaptive learning rates for each parameter using first and second moment estimates. This is particularly advantageous when dealing with heterogeneous visual features in potato leaf images, where Early Blight and Late Blight exhibit distinct spatial patterns. The adaptive nature of Adam allows the model to focus more precisely on relevant features during fine-tuning, resulting in faster and more stable convergence compared to SGD. A comparative overview of the final performance of both optimizers is illustrated in Figure 5 (Accuracy and Loss per Epoch) and Figure 6 (Test Accuracy and F1-Score Comparison).

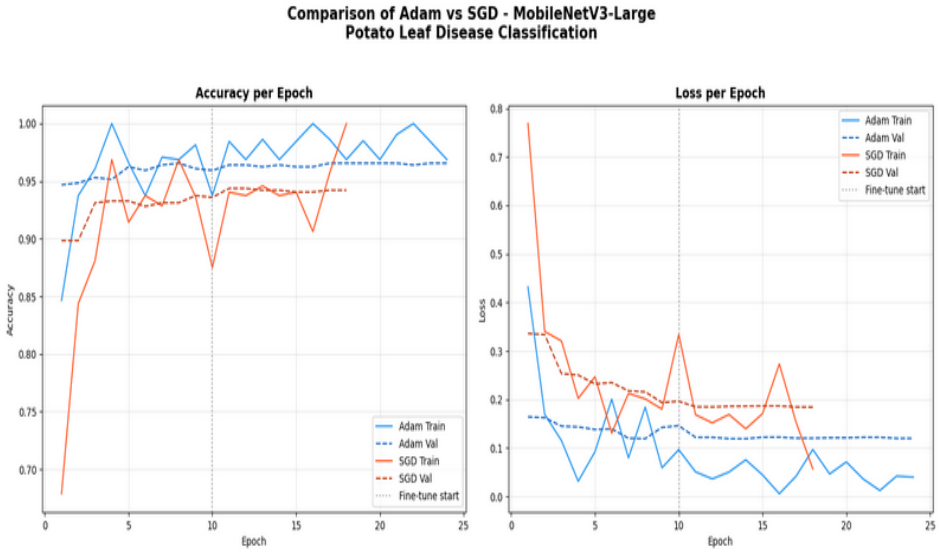

Figure 5. Accuracy and loss per Epoch graph for Adam and SGD optimizers.

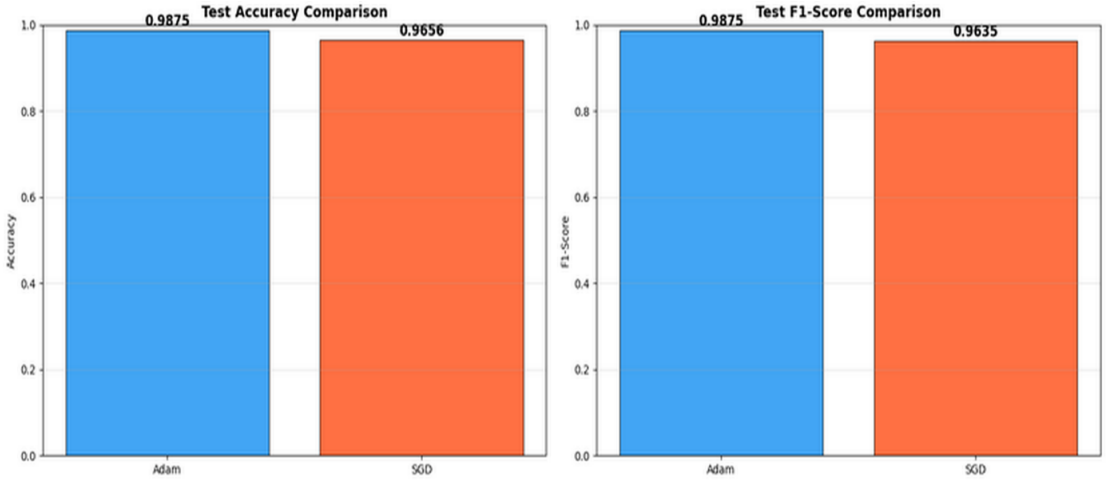

Figure 6. Comparison graph of test accuracy and F1-score for Adam and SGD optimizers.

4. Conclusions

Based on the results of the study on potato leaf disease classification using the MobileNetV3-Large architecture with a comparative analysis of the Adam and SGD optimizers, significant performance differences were identified between the two optimizers. The Adam optimizer achieved a test accuracy of 98.75%, precision of 0.9906, recall of 0.9844, and F1-score of

0.9875, with a test loss of 0.0495. Meanwhile, the SGD optimizer yielded a test accuracy of 96.56%, precision of 0.9700, recall of 0.9667, and F1-score of 0.9635, with a test loss of 0.1409. Based on all evaluation metrics, the Adam optimizer proved to be superior to SGD in the potato leaf disease classification task using the MobileNetV3-Large architecture. In terms of training stability, the Adam optimizer demonstrated faster and more stable convergence, whereas SGD tended to be more fluctuative particularly during the fine-tuning phase. The application of a two-stage transfer learning approach combined with random undersampling and data augmentation proved effective in addressing class imbalance while improving the model's generalization ability. The MobileNetV3-Large architecture, which was designed to operate optimally on resource-constrained devices, made it a suitable choice not only in terms of performance but also in terms of efficiency, opening broader opportunities for deployment without relying on large-scale computing infrastructure. Beyond a mere technical achievement, the presence of an accurate and efficient plant disease classification system had the potential to transform how farmers responded to disease threats at an early stage, which ultimately could reduce excessive reliance on pesticides, minimize crop losses, and strengthen food security and environmental sustainability as a whole. For future research, it was recommended to explore other optimizers such as RMSprop, AdamW, or SGD with Nesterov momentum, utilize larger and more diverse datasets collected directly from real field conditions, and direct further development toward implementation on mobile or edge computing devices to provide more tangible benefits for farmers.

Acknowledgments

The authors gratefully acknowledge the supervision and constructive guidance of Mrs. Ery Hartati during the course of this research.

Competing Interest

The authors declare no competing interest.

References

[1] Majeed, A.; Siyar, S.; Sami, S. (2022). Late blight of potato: From the great Irish potato famine to the genomic era—An overview. *Hellenic Plant Protection Journal*, *15*(1), 1–9. https://doi.org/10.2478/hppj-2022-0001.

[2] Wolters, P. J.; Wouters, D.; Kromhout, E. J.; Huigen, D. J.; Visser, R. G. F.; Vleeshouwers, V. G. A. A. (2021). Qualitative and quantitative resistance against early blight introgressed in potato. *Biology*, *10*(9). https://doi.org/10.3390/biology10090892.

[3] Talab, H. K.; Mohammadzamani, D. (2025). Diagnosis and classification of two common potato leaf diseases (Early Blight and Late Blight) using image processing and machine learning. *15*(1).

[4] Tiwari, S.; Srivastava, A. (2024). Farmers' knowledge and perception of late blight of potato and its management strategies in Kailali and Banke districts of Nepal. *Plant Pathology & Quarantine*, *14*(1), 1–9. https://doi.org/10.5943/ppq/14/1/1.

[5] Hasan, M. (2024). Identifikasi penyakit daun kentang menggunakan fitur GLCM dan algoritma Multi-SVM [Identification of potato leaf disease using GLCM features and Multi-SVM algorithm]. *CONTEN: Computer and Network Technology*, *4*(1), 52–57. https://doi.org/10.31294/conten.v4i1.3465.

[6] Amrulloh, I. T. A.; Sari, B. N.; Padilah, T. N. (2024). Evaluasi augmentasi data pada deteksi penyakit daun tebu dengan YOLOv8 [Evaluation of data augmentation in sugarcane leaf disease

detection using YOLOv8]. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(4), 7547–7552. https://doi.org/10.36040/jati.v8i4.10267.

[7] Fuadi, A.; Suharso, A. (2022). Perbandingan arsitektur MobileNet dan NASNetMobile untuk klasifikasi penyakit pada citra daun kentang [Comparison of MobileNet and NASNetMobile architectures for disease classification on potato leaf images]. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, *7*(3), 701–710. https://doi.org/10.29100/jipi.v7i3.3026.

[8] Chen, L.; Wei, J.; Wang, G.; Yang, X.; Qin, L. (2025). MobileNetV3–Transformer-Based Prediction of Highway Accident Severity. *Applied Sciences*. 2025; *15*(23), 12694. https://doi.org/10.3390/app152312694 .

[9] Fitriansyah, M. F.; Siregar, G. M. A. (2025). Optimasi menggunakan algoritma stochastic gradient descent (SGD): Studi bibliometrik [Optimization using stochastic gradient descent (SGD) algorithm: A bibliometric study]. *Indexia, 7(1),* 53–68. https://doi.org/10.30587/indexia.v7i1.9708.

[10] Herwina, H.; Darmatasia, D.; Shiddiq, A. K. A.; Syahputra, T. D. (2022). Deteksi penyakit pada tanaman padi menggunakan MobileNet transfer learning berbasis Android [Disease detection in rice plants using Android-based MobileNet transfer learning]. *Jagti*, *2*(2), 1–8. https://doi.org/10.24252/jagti.v2i2.41.

[11] Maximilliano, W.; Rachmat, N. (2025). Comparative analysis of MobileNetV3-Large and Small for corn leaf disease classification. *Brilliance: Research of Artificial Intelligence*, *5*(1), 325–332. https://doi.org/10.47709/brilliance.v5i1.6259.

[12] Alhammad, S. M.; Khafaga, D. S. (2025). Deep learning and explainable AI for classification of potato leaf diseases. *Frontiers in Artificial Intelligence*. https://doi.org/10.3389/frai.2024.1449329.

[13] Radwan, M.; Ali, A.; Abdelhameed, A. (2025). Potato leaf disease classification using optimized machine learning. *Potato Research*, *68*(2), 897–921. https://doi.org/10.1007/s11540-024-09763-8.

[14] Chang, C.-Y.; Lai, C.-C. (2024). Potato Leaf Disease Detection Based on a Lightweight Deep Learning Model. *Machine Learning and Knowledge Extraction*, *6*(4), 2321–2335. https://doi.org/10.3390/make6040114.

[15] Sinamenye, J. H.; Chatterjee, A.; Shrestha, R. (2025). Potato plant disease detection: Leveraging hybrid deep learning models. *BMC Plant Biology*. https://doi.org/10.1186/s12870-025-06679-4.

[16] Simangunsong, J.; Simanjuntak, N. D.; Matondang, A. A. (2025). Penerapan transfer learning untuk klasifikasi citra bunga berbasis convolutional neural network [Application of transfer learning for flower image classification based on convolutional neural network]. *Jurnal Minfo Polgan*, *14*(1), 1062–1067. https://doi.org/10.33395/jmp.v14i1.14980.

[17] Taye, M. M. (2023). Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions. *Computers*, *12*(5), 91. https://doi.org/10.3390/computers12050091.

[18] Yuliany, S.; Aradea; Rachman, A. N. (2022). Implementasi deep learning pada sistem klasifikasi hama tanaman padi menggunakan metode convolutional neural network (CNN) [Implementation of deep learning in rice plant pest classification system using convolutional neural network (CNN)]. *Jurnal Buana Informatika*, *13*(1), 54–65. https://doi.org/10.24002/jbi.v13i1.5022.

[19] Zhao, L.; Wang, L. (2022). A new lightweight network based on MobileNetV3. *Transactions on Internet and Information Systems*, *16*(1), 1–15. https://doi.org/10.3837/tiis.2022.01.001.

[20] Asy Syifa, S.; Amelia Dewi, I. (2022). Arsitektur ResNet-152 dengan perbandingan optimizer Adam dan RMSProp untuk mendeteksi penyakit paru-paru [ResNet-152 architecture with comparison of Adam and RMSProp optimizers for lung disease detection]. *Journal MIND*, *7*(2), 139–150. https://doi.org/10.26760/mindjournal.v7i2.139-150.

[21] Haqqi, M. S.; Kusumoputro, B. (2022). Komparasi metode optimasi Adam dan SGD dalam skema direct inverse control untuk sistem kendali data sikap dan ketinggian quadcopter [Comparison of Adam and SGD optimization methods in direct inverse control scheme for quadcopter attitude and

altitude control system]. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, dan Teknik Elektronika*, *10*(2), 458. https://doi.org/10.26760/elkomika.v10i2.458.

[22] Utomo, P. B.; Faruqziddan, M.; Aulia, E. H. S.; Azzahra, S. D. (2024). Perbandingan skenario balancing oversampling dan undersampling dalam klasifikasi risiko kambuh kanker tiroid menggunakan algoritma SVM linear [Comparison of oversampling and undersampling balancing scenarios in thyroid cancer recurrence risk classification using linear SVM algorithm]. *JAMI: Jurnal Ahli Muda Indonesia*, *5*(2), 172–182. https://doi.org/10.46510/jami.v5i2.320.