Received: 10 September 2025 | Revised: 22 December 2025 | Accepted: 6 February 2026 | Published online: 6 March 2026

RESEARCH ARTICLE

Journal of Computational and Cognitive Engineering
2026, Vol. 5(2) 292–303
DOI: 10.47852/bonviewJCCE62027632

BON VIEW PUBLISHING

# A Weighted Ensemble of EfficientNetV2 Variants with Optimizer Tuning for Tomato Leaf Disease Classification

**Aliyu Tetengi Ibrahim[1], Ibrahim Hayatu Hassan[1], Mohammed Abdullahi[1], Abeer Rashad Mirdad[2], Muhammad I. Khan[2], Saeed Ali Bahaj[3,*] and Fatima Khan Nayer[2]**

[1] *Department of Computer Science, Ahmadu Bello University, Nigeria*

[2] *AIDA Lab, Prince Sultan University, Saudi Arabia*

[3] *MIS Department College of Business Administration, Prince Sattam bin Abdulaziz University, Saudi Arabia*

**Abstract:** Tomatoes are one of the most widely grown horticultural crops but are highly susceptible to leaf diseases that could significantly affect yield and quality. Early and accurate disease detection is necessary to enhance crop productivity and food security. The present study presents a deep learning system for automatic tomato leaf disease classification using a weighted ensemble of EfficientNetV2 models. Seven variants of EfficientNetV2 (B0–B3, S, M, and L) were fine-tuned on an augmented version of the filtered PlantVillage dataset. To improve feature visibility under varying lighting conditions, images were preprocessed using contrast-limited adaptive histogram equalization. Five optimization algorithms, which include Adam, Adamax, AdamW, Nadam, and RMSProp, were tested to assess their impact on model convergence and generalization. Among individual models, Adam-optimized EfficientNetV2L performed the best in accuracy with a measure of 99.50%. For classifiability resilience improvement, 16 ensemble configurations (eight unweighted and eight weighted ensembles using different combinations of EfficientNetV2-S, M, and L variants and five optimizers) were explored, with the best weights discovered using brute-force grid search. The best-performing weighted ensemble achieved 99.89% in accuracy, precision, recall, and F1-score, which was evaluated on a held-out test set not used during training or validation, demonstrating its strong generalizability. The proposed framework offers a scalable and reliable solution for early tomato disease detection with significant potential for real-time plant health monitoring in precision agriculture.

**Keywords:** EfficientNetV2, weighted ensemble learning, tomato leaf disease detection, transfer learning, technological development

## 1. Introduction

Agriculture continues to be a backbone for national development, with most emerging economies depending on it for a stride toward food safety, employment, and GDP increase [1]. Tomatoes are an important economic and nutritional commodity among the most grown horticultural crops in the world. Tomatoes are a crucial component in human diets and one of the major products in international agricultural trade, cultivated in open fields and greenhouses alike. Global production of tomato reached over 180 million tons in 2018, according to the Food and Agriculture Organization Statistical Database (FAOSTAT) of the United Nations, and at the same time, Asia emerged as the leading producer of this crop [2].

However, tomato cultivation is increasingly threatened by a wide spectrum of diseases that include late blight, bacterial wilt, leaf curl, and Pepino mosaic virus. These diseases not only diminish crop quality and yield but also impose substantial economic losses. Many of these pathogens manifest through visual symptoms on the foliage of a plant. Therefore, correct identification should be timely and accurate for the effective management of disease. Conventional diagnostic approaches normally rely on the heavy, subjective, and time-consuming visual expertise of farmers or pathologists. This is as identified by Liu and Wang [3]. This method is fraught with subjectivity, inefficiency, and high error rates, especially given the morphological similarities of diseases and the complexity introduced by environmental variability, as noted by Atila et al. [4].

These limitations highlight the urgent need for robust and automated plant disease diagnosis systems that are capable of accuracy, along with scalability and real-time deployment. Recent progress in artificial intelligence, particularly in the field of deep learning (DL), has given high promise for image analysis in agriculture. Although early machine learning models such as support vector machines (SVM) and artificial neural networks (ANN) required handcrafted features [5–7], deep convolutional neural networks (CNN) have the potential to enable end-to-end feature learning with superior performance and generalization capability [8, 9].

***Corresponding author:** Saeed Ali Bahaj, MIS Department College of Business Administration, Prince Sattam bin Abdulaziz University, Saudi Arabia. Email: s.bahaj@psau.edu.sa

DL has recently been widely used in image segmentation and classification. A study by Raman et al. [10] presented a panoptic-aware object-wise depth estimation model employing Feature Pyramid Networks for instance-level depth and occlusion reasoning. Transformer-based methods have also pushed state-of-the-art forward: Liu et al. [11] proposed F-DETR, a multi-branch Transformer detector with multi-scale feature fusion, which elevates both efficiency and performance. For classification, Yang et al. [12] have suggested the Modality Fusion Vision Transformer that fuses Hyperspectral Imaging (HSI) and Light Detection and Ranging (LiDAR) data via cross-attention and spectral self-attention, achieving up to 99.91% accuracy and surpassing previous methods.

It is for this reason that the integration of DL with agricultural diagnostics has resulted in innovative applications, starting from intraoperative surgery to smart farming [13]. In tomato disease classification, CNN-based models have demonstrated considerable potential due to their ability for automatic processing and extracting important image features such as shape, color, and texture. However, key challenges significantly affect these CNN-based models, particularly data imbalance, few available labeled datasets, and model overfitting.

Transfer learning has become a valuable strategy in dealing with such challenges by allowing the models to leverage pretrained weights on large-scale datasets for domain-specific tasks. Additionally, the EfficientNetV2 family of architectures introduces compound scaling strategies that properly balance model size, accuracy, and training efficiency.

On the other hand, our proposed architecture synergistically integrates optimizer adjustment (among Adam, Adamax, AdamW, Nadam, and RMSProp) with a weighted ensemble of seven EfficientNetV2 variants (B0–B3, S, M, L), where ensemble weights are set based on the performance of individual models. This integration encourages generalization, stability, and classification accuracy, mitigating the aforementioned deficiencies and offering a compact, stable solution for real-world tomato leaf disease identification. The main contributions are as follows:

1) Comprehensive model evaluation: This work leverages the EfficientNetV2 family of model architectures, B0–B3, S, M, L, for the classification of tomato leaf diseases and ensures thorough exploration of model capacities in standardized training conditions to allow for a fair and consistent model performance evaluation.
2) Optimizer impact analysis: Five optimization algorithms (Adam, Adamax, AdamW, Nadam, and RMSProp) were systematically applied to each EfficientNetV2 variant, enabling an in-depth assessment of optimizer architecture interactions to identify the most effective training configuration.
3) Sixteen ensemble configurations were constructed: Five unweighted and five weighted ensembles combining EfficientNetV2-S, M, and L trained with the same optimizer, plus three unweighted and three weighted ensembles combining the same architecture trained with four different optimizers (Adam, AdamW, Nadam, RMSProp). Optimal weights for the weighted ensembles were determined via grid search, allowing for the selection of the most effective ensemble configuration.
4) Comparative performance benchmarking: A rigorous comparative analysis was conducted among the proposed approaches and against state-of-the-art methods from the literature, using accuracy, F1-score, recall, and precision as evaluation metrics. This benchmarking highlights the competitiveness and practical relevance of the proposed methodology.

## 2. Literature Review

The study of Wu et al. [14] proposed data augmentation with DCGAN for tomato leaf disease detection. An augmentation of the original dataset with DCGAN-generated images and classification with GoogLeNet resulted in an average top 1 accuracy of 94.33%. Likewise, Basavaiah and Arlene Anthony [15] employed a set of feature extraction methods for the classification of disease in tomato leaves. Both decision tree and random forest classifiers were employed, where the highest accuracy of 94% was reported by the random forest.

Similarly, Hong et al. [16] proposed a DL-based approach for the classification of tomato disease from five architectures: ResNet50, Xception, MobileNet, ShuffleNet, and Densenet121_Xception. The authors found that the highest classification accuracy by Densenet121_Xception was 97.10%.

In another study, Gadekallu et al. [17] proposed a new tomato disease classification method using a Deep Neural Network (DNN) optimized by the hybrid method of principal component analysis and the whale optimization algorithm for feature extraction. Grid search has been used for hyperparameter tuning of DNN. The approach gave 99% training accuracy and 94% testing accuracy.

To further solve classification problems, Thangaraj et al. [18] proposed an Xception model for the classification of tomato leaf diseases through transfer learning. Among different optimizers tried, Adam performed best, giving a top 1 accuracy of 99.55%.

In this regard, Abbas et al. [19] came up with a DL model for tomato disease detection using a Conditional Generative Adversarial Network (GAN) to generate synthetic images. A DenseNet121 model, trained via transfer learning on both real and synthetic images from the PlantVillage dataset, achieved accuracies of 99.51%, 98.65%, and 97.11% on classifying tomato leaf images into 5, 7, and 10 disease classes, respectively. In another attempt, Paymode and Malode [20] present a DL model based on a the Visual Geometry Group (VGG) for multi-crop leaf disease classification. Images of grape leaves achieved 98.4% accuracy, while images of tomato leaves achieved an accuracy of 95.71%.

A previous work by Ahmed et al. [21] proposed a fast and efficient tomato leaf disease classifier using a MobileNetV2-based feature extractor coupled with a custom classifier. Runtime augmentation was used to prevent class imbalance and leakage. Compact, fast, and accurate, MobileNetV2 was selected for achieving a classification accuracy of 99.30%.

A study by Rahman et al. [22] proposed an image processing framework to detect diseases on the leaves of tomatoes; it segmented images using Otsu's method and extracted 1560 texture features through the Gray-Level Co-occurrence Matrix (GLCM). Its SVM classifier resulted in an accuracy of 100% for healthy leaves, 95% for early blight, 90% for Septoria leaf spot, and 85% for late blight.

Sun et al. [23] developed a hybrid Eff-Swin model (EfficientNetV2+Swin Transformer) to classify tomato diseases with 99.70% accuracy. This outperforms standalone CNN or Transformer baselines. To further boost the performance, Nazir et al. [24] also in their paper combine EfficientNetV2 with spatial–channel attention to focus on areas of disease and with 98.12% accuracy using transfer learning on PlantVillage image. A study by Sinamenye et al. [25] proposed a hybrid model combining EfficientNetV2B3 and Vision Transformer (ViT) for improved potato plant disease detection, leveraging the high performance of both CNN and Transformer architectures. Similarly, Tiwari et al. [26] in their work employ ensembles of pretrained CNNs such as DenseNet121, ResNet50, MobileNet, and Xception to achieve 97% accuracy in leaf diseases of different plant species.

Although many techniques have been explored for the detection of tomato leaf disease, existing approaches still manifest significant weakness. Several rely on either older architecture exploration, such as GoogLeNet + DCGAN, or shallow hybrid models, such as CNN–SVM, lacking strong generalization. Others are dependent on handcrafted features or single-optimizer transfer learning models, such as Xception and MobileNetV2, and are not very robust across the disease types. More advanced approaches such as DenseNet-CGAN, YOLOv7, and CNN–Swin hybrids are computationally heavy and seldom investigate optimizer tuning or weighted ensembling. Overall, previous research has focused much on architecture design and augmentation but little on optimization strategies by utilizing different optimizers. To our knowledge, only a few studies couple optimizer tuning with weighted ensemble learning on state-of-the-art architectures such as EfficientNetV2; thus, a clear gap exists in optimizing both training dynamics and ensemble composition for high-performance disease classification.

In order to bridge this gap, our study provides a new framework that seeks to examine the impact of multiple optimizers and weighted ensemble techniques on seven implementations of EfficientNetV2. This combined solution is aimed at improving classification accuracy and model generalizability.

## 3. Research Methodology

This section covers the materials and methodologies employed in the present study.

### 3.1. EfficientNetV2

EfficientNetV2 was introduced based on the need to improve model training speed and reduce memory size while keeping performance superior. This convolutional network architecture stands for a new family that surpasses its predecessors both in training speed and parameter efficiency. The introduction of EfficientNetV2 is toward more resource-efficient models that demonstrate improved performance across diverse tasks. Experimental results are indicative that EfficientNetV2 models achieve a notably faster time in training compared to state-of-the-art models. It has also managed to keep a more compact model size that can go up to 6.8 times smaller [27, 28].

### 3.2. Dataset

Table 1 shows the distribution of images used in this study. These were derived from the publicly available PlantVillage dataset [29]. The dataset was filtered to include 14,529 images across nine tomato disease categories and as one healthy class. A 70:18:12 ratio was used for training, validation, and testing, respectively. Images were also checked for quality and duplicates prior to training. Sample images illustrating various classes within the tomato dataset are depicted in Figure 1.

#### *3.2.1. Data preprocessing*

This study did not employ the use of sophisticated preprocessing, such as noise reduction, segmentation, or GAN-based augmentation. Contrast-limited adaptive histogram equalization (CLAHE) was used to enhance local contrast without amplifying noise and preserving disease-relevant patterns. Images were resized to 224 × 224 pixels for compatibility with the pretrained models. The dataset was divided into a 70:30 ratio for training and validation/testing. The validation set was further divided 60:40 for validation and testing.

### 3.3. Proposed model

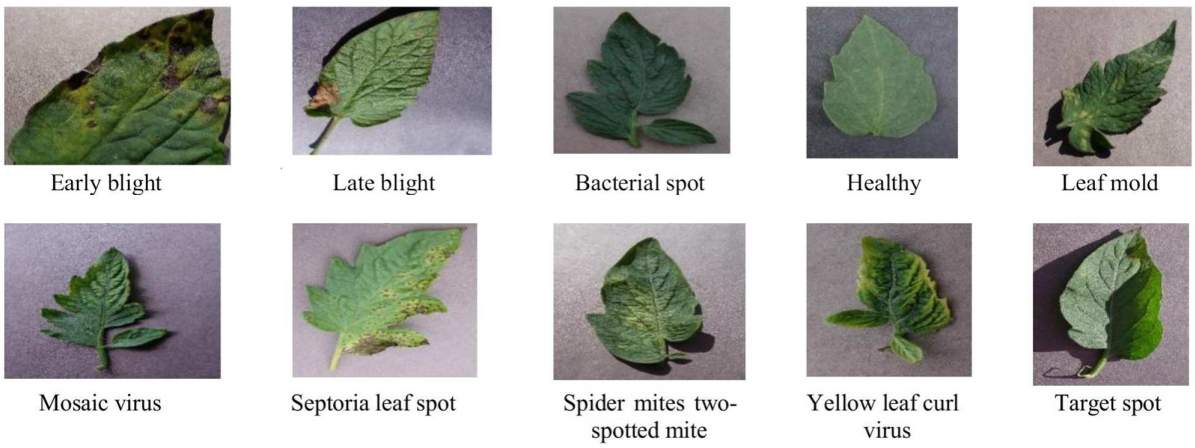
The proposed model pipeline comprises five major stages. First, all input images are enhanced using CLAHE. Second, enhanced images are then expanded through extensive data augmentation, including rotation (20°), width/height shifting (0.1), shearing (0.1), zooming (0.1), brightness variation (0.2–1.0), and horizontal/vertical flipping, to address data insufficiency. Third, both original and augmented images undergo preprocessing involving resizing to 224 × 224 and pixel normalization to improve optimization, generalization, and learning efficiency. Fourth, each pretrained model is fine-tuned using a uniform strategy. The final classification layer is replaced with a global average pooling layer, batch normalization, and two dense layers (128 neurons each) with Leaky Rectified Linear Unit (Leaky ReLU) activation, dropout (0.25), and additional batch normalization. Fifth, extracted features are then passed to a final 10-neuron SoftMax classification layer, trained with categorical cross-entropy loss. Dropout is intentionally excluded between the last dense layer and the classifier due to observed performance degradation.

To evaluate optimization behavior, the models are trained and tested using five optimizers: Adam, Adamax, AdamW, Nadam, and RMSProp. A total of 16 ensemble models is constructed. The first 10 ensembles (with and without weighting) each combine three base models: EfficientNetV2S, EfficientNetV2M, and EfficientNetV2L, paired with one of the five optimizers (Adam, AdamW, Adamax, Nadam, RMSProp). Ensembles 11–13 each incorporate four identical base models

**Table 1**
**Summary of sample images of tomato leaves extracted from each class within the tomato dataset**

| Image classes | Number of training images | Number of validation images | Number of test images |
|---|---|---|---|
| Early blight | 560 | 144 | 96 |
| Late blight | 1069 | 275 | 183 |
| Bacterial spot | 1191 | 307 | 204 |
| Healthy | 891 | 229 | 153 |
| Leaf mold | 533 | 137 | 91 |
| Mosaic virus | 209 | 54 | 36 |
| Septoria leaf spot | 992 | 255 | 170 |
| Spider mites | 939 | 241 | 161 |
| Yellow leaf curl virus | 3000 | 771 | 515 |
| Target spot | 786 | 202 | 135 |
| Total | 10,170 | 2615 | 1744 |

**Figure 1**
**Sample images showcasing tomato leaves from various classes within the tomato dataset**




(EfficientNetV2S only, EfficientNetV2M only, or EfficientNetV2L only), trained separately with Adam, AdamW, Nadam, and RMSProp; these are also repeated with weighted versions. Weighted ensembles use brute-force grid search to determine optimal model weights. For ensembles with three base models, 1000 weight combinations are evaluated; for those with four models, 10,000 combinations are tested. The weight search range is 0–10, which is implemented using nested Python loops. Although computationally expensive, this method ensures exhaustive and transparent evaluation. Optimal weights are assigned to reduce the influence of weaker models and then enhance the contribution of stronger ones. Table 2 summarizes the parameters used across all

**Table 2**
**The parameters of the ensemble models**

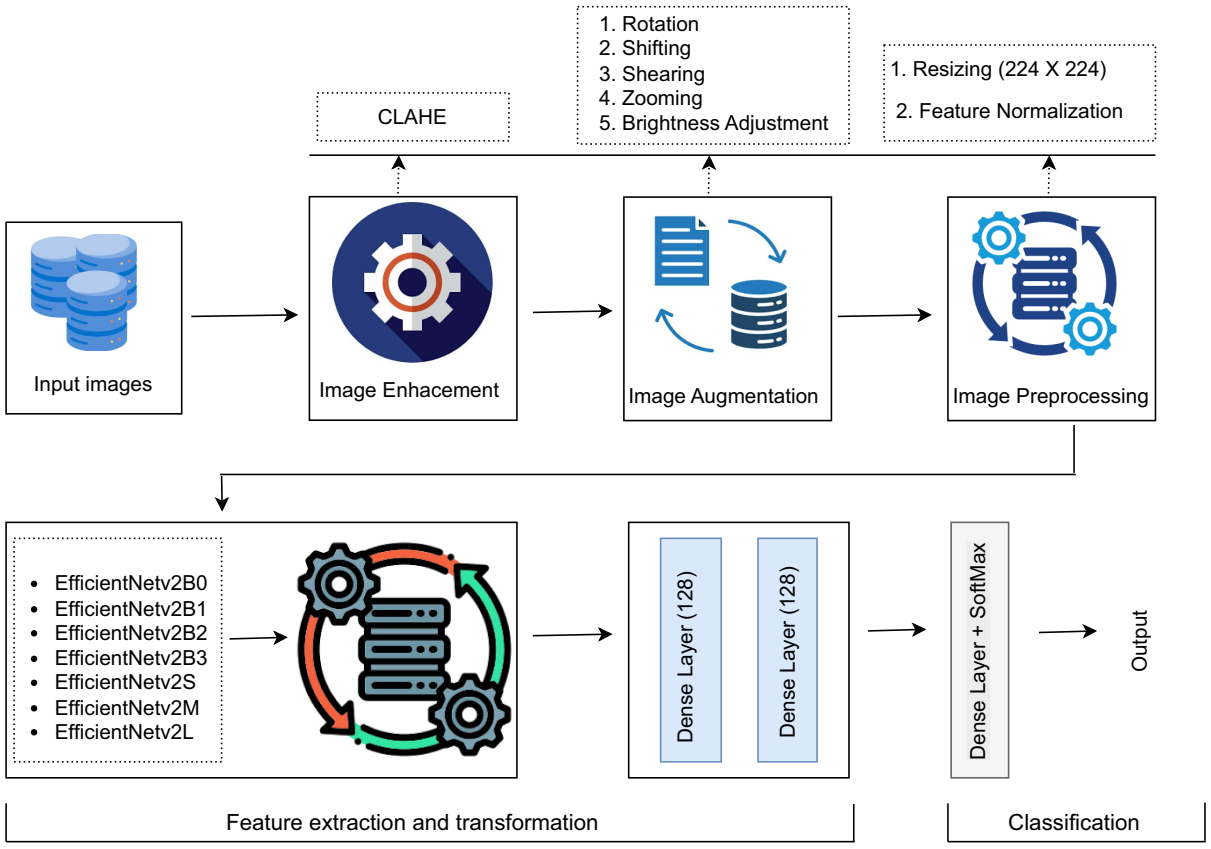
| Ensemble model | Optimizer(s) | Pretrained models | Method |
|---|---|---|---|
| Ensemble_1 | Adamax | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Unweighted |
| Ensemble_2 | Adam | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Unweighted |
| Ensemble_3 | AdamW | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Unweighted |
| Ensemble_4 | Nadam | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Unweighted |
| Ensemble_5 | RMSProp | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Unweighted |
| Ensemble_6 | Adamax | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Weighted [0.6, 0.4, 0.9] |
| Ensemble_7 | Adam | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Weighted [0.4, 0.1, 0.4] |
| Ensemble_8 | AdamW | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Weighted [0.2, 0.1, 0.2] |
| Ensemble_9 | Nadam | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Weighted [0.2, 0.2, 0.1] |
| Ensemble_10 | RMSProp | EfficientNetV2S, EfficientNetV2M, EfficientNetV2L | Weighted [0.2, 0.4, 0.3] |
| Ensemble_11 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2S, EfficientNetV2S, EfficientNetV2S | Unweighted |

(*Continued*)

**Table 2**
(*Continued*)

| Ensemble model | Optimizer(s) | Pretrained models | Method |
|---|---|---|---|
| Ensemble_12 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2M, EfficientNetV2M, EfficientNetV2M | Unweighted |
| Ensemble_13 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2L, EfficientNetV2L, EfficientNetV2L | Unweighted |
| Ensemble_14 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2S, EfficientNetV2S, EfficientNetV2S | Weighted [0.2, 0.2, 0.0, 0.3] |
| Ensemble_15 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2M, EfficientNetV2M, EfficientNetV2M | Weighted [0.2, 0.0, 0.1, 0.1] |
| Ensemble_16 | Adam, AdamW, Nadam, RMSProp | EfficientNetV2L, EfficientNetV2L, EfficientNetV2L | Weighted [0.2, 0.7, 0.3, 0.6] |

**Figure 2**
**Architecture of the proposed model**




ensembles, and Figure 2 depicts the architecture of the proposed model.

## 3.4. Optimizers

DL performance is heavily dependent on the chosen optimizer, which prescribes the method of updating model weights and how efficiently the network converges during training. This work adopts five widespread variants: Adamax, Adam, Nadam, AdamW, and RMSProp. These were chosen for their adaptive learning rate mechanism and efficiency in deep convolutional networks. Recent benchmarking studies have shown that adaptive-moment optimizers, such as Adam and Nadam, as well as AdamW, usually converge well and realize high accuracy across many CNN models [30, 31].

### *3.4.1. Adam*

Adam is an optimization algorithm popularly used in many DL studies that adjusts the learning rates of each parameter

adaptively, which incorporates elements of both momentum and scaling [32]. It merges the advantages of RMSProp and Stochastic Gradient Descent (SGD) with Momentum, making it suitable for dynamic objectives and scenarios with noisy or sparse gradients. Adam maintains two additional variables $m_t$ and $v_t$ for each variable to be trained, which are mathematically represented in Equations (1) and (2).

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)_{g_t} \tag{1}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)_{g_t^2} \tag{2}$$

The variables $m_t$ and $v_t$ represent approximations of the first moment (mean) and the second moment (variance) of the gradients, respectively.

*3.4.2. Adamax*

Adamax is a derivative of Adam that relies on the infinity norm. It is a first-order gradient-based optimization technique [33]. Its adaptability to adjust the learning rate based on data characteristics makes it particularly well-suited for learning time-variant processes. In Adam, the update rule for individual weights involves scaling their gradients inversely proportional to a (scaled) $L_2$ norm of their current and past gradients. Adamax extends Adam from the $L_2$ norm to the l∞ norm, and its formulation is presented in Equations (3)–(5).

$$u_t = \beta_2^{\infty} v_{t-1} + \left(1-\beta_2^{\infty}\right)|g_t|^{\infty} \tag{3}$$

$$= \max(\beta_2 \cdot v_{t-1}, |g_t|) \tag{4}$$

The Adamax update rule can be derived by substituting $\sqrt{\hat{v}_t + \epsilon}$ with $u^t$ to obtain the Adamax update rule, as specified in Equation (5).

$$\theta_{t+1} = \theta_t - \frac{\eta}{u_t}\hat{m}_t \tag{5}$$

*3.4.3. RMSProp*

RMSProp is an adaptive learning method designed to address the issue of diminishing learning rates. It utilizes an exponentially decaying average to discard past gradients, facilitating quicker convergence when a convex bowl is encountered [34]. This approach bears resemblance to Adadelta and is expressed as in Equations (6) and (7).

$$E|g^2|_t = \gamma E|g^2|_{t-1} + (1-\gamma)_{g_t^2} \tag{6}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E|g^2|_t + \epsilon}} g_t \tag{7}$$

where $E|g^2|_t$ is the running average and $g_t = \nabla_\theta J(\theta)$.

*3.4.4. AdamW*

AdamW is a stochastic optimization technique that adapts the conventional application of weight decay in Adam by separating weight decay from the gradient update process [35]. This separation is illustrated through a modification commonly employed for $L_2$ regularization in Adam, where $w_t$ denotes the weight decay rate at time $t$ as shown in Equations (8) and (9).

$$g_t = \nabla f(\theta_t) + w_t \theta_t \tag{8}$$

While utilizing AdamW, the adjustment involves incorporating the weight decay term directly into the gradient update step.

$$\theta_{t+1,i} = \theta_{t,i} - \eta\left(\frac{1}{\sqrt{\hat{v}_t + \epsilon}} \cdot \hat{m}_t + w_{t,i}\theta_{t,i}\right), \forall t \tag{9}$$

In the given representation, $\theta_t$ represents the parameters at time step $t$, $\eta$ denotes the learning rate, $\beta_1$ and $\beta_2$ are the exponential decay rates for the moment estimates, $m_t$ and $v_t$ represent the first and second moment estimates at time step $t$, $\epsilon$ is a small constant used for numerical stability, and $g_t = \nabla f(\theta_t)$ signifies the gradient of the loss function $f$ with respect to parameters $\theta_t$ at time step $t$.

*3.4.5. Nadam*

Nadam, which refers to Nesterov accelerated Adaptive Moment Estimation, combines the Adam and Nesterov Momentum methods. Nesterov Accelerated Gradient is a method that is designed to expedite convergence by enhancing gradient vectors in favorable directions [32]. Nadam is particularly well-suited for scenarios involving extensive data and parameters, as well as when dealing with volatile objective functions. This concept can be expressed mathematically as depicted in Equation (10).

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}\left(\beta_1 \hat{m}_t + \frac{(1-\beta_t)\, g_t}{1-\beta_1^t}\right) \tag{10}$$

## 3.5. Leaky ReLU

Given an input tensor X, the rectified linear unit (ReLu) activation layer applies the elementwise ReLu function to the input tensor [36] as shown in Equation (11). The Leaky ReLU is a modified version of the activation function derived from ReLU [33]. Unlike ReLU's flat slope for negative values, Leaky ReLU introduces a small slope. The mathematical representation of Leaky ReLU is expressed in Equation (12).

$$Y = \max(0, X) \tag{11}$$

The output tensor $Y$ has the same shape as the input tensor $X$.

$$y_i = \begin{cases} x_i & if\ x_i \geq 0 \\ \frac{x_i}{a_i} & if\ x_i < 0 \end{cases} \tag{12}$$

Here, $a_i$ represents a constant parameter within the range of $(1, +\infty)$.

## 3.6. SoftMax classifier

The SoftMax classifier is employed in the output layer for tomato disease classification. This classifier operates by assigning a probability distribution to each class. The class with the highest probability has its probability normalized to 1, and all other probabilities are adjusted proportionally. The SoftMax function computes a vector of real numbers and normalizes these values to produce a vector of probabilities, each ranging between 0 and 1 [36]. For an input vector u with a length of n $(u_1, u_2, u_3, \ldots\ldots\ldots, u_n)$, the SoftMax function $S(u)$ calculates the probability $p(v_i)$ for each element $v_i$ within the range of 1 *to n* as depicted in Equation (13).

$$p(v_i) = \frac{e^{u_i}}{\sum_{j=1}^{n} e^{u_j}} \tag{13}$$

Here, $p(v_i)$ represents the probability that the input belongs to class $i$, $u_i$ is the $i^{th}$ element of the vector $u$, and $e$ denotes the base of the natural logarithm.

### 3.7. Experimental setup

The implementation of the model was carried out in the Python programming language, utilizing DL frameworks, specifically TensorFlow and Keras. The experiment was conducted on the Kaggle cloud platform, which is equipped with 2 CPU cores, 13 gigabytes of RAM, and 1 Nvidia Tesla P100 GPU.

### 3.8. Evaluation metrics

To assess the effectiveness of the proposed model, we employed a range of performance metrics, encompassing accuracy, precision, recall, and F1-score computed based on Equations (14)–(17), respectively, as presented in Reference [37].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1 - Score = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{17}$$

## 4. Results and Discussion

The study evaluates various pretrained EfficientNetV2 architectures for classifying tomato leaf diseases and compares their performance with other CNN models from the literature. The dataset was divided into training, validation, and test sets across 10 disease classes. These were used to train the models. Results show that EfficientNetV2 variants from V2L to V2B0 achieved notably higher accuracy than the other architectures tested.

### 4.1. Performance comparison of the proposed model

Five different approaches were employed in this study to evaluate and compare the performance of the proposed model. This aims to identify the most effective one among them. Each of these techniques involved training and evaluating the proposed model using the same dataset, parameters, and configurations. These techniques are used to evaluate the (i) performance of individual models with a single optimizer, (ii) performance of ensemble models with a single optimizer, (iii) performance of weighted ensemble models with a single optimizer, (iv) performance of ensemble models with multiple optimizers, and (v) performance of weighted ensemble models with multiple optimizers.

#### *4.1.1. Performance of individual models with different optimizers*

A comparative analysis is conducted based on the performance of the proposed model utilizing all seven pretrained model architectures that are entailed in EfficientNetV2 based on five different optimizers: Adamax, Adam, AdamW, Nadam, and RMSProp. The proposed model demonstrated strong performance across all optimizers. EfficientNetV2L achieved the highest accuracy of 99.50% when paired with the Adam optimizer, as indicated in Table 3. The lowest accuracy of 98.68%, on the other hand, was achieved when EfficientNetV2L was paired with the Adamax optimizer, as indicated in Table 4. EfficientNetV2L, paired with the other three optimizers, also yielded impressive results, with slight variations compared to the highest accuracy model. Specifically, EfficientNetV2L achieved an accuracy of 99.46% with AdamW, 99.37% with Nadam, and 99.29% with RMSProp, as indicated in Tables 5, 6, and 7, respectively.

**Table 3**
**Performances of the pretrained models using Adam optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| EfficientNetV2B0 | 98.70 | 98.69 | 98.70 | 98.71 |
| EfficientNetV2B1 | 98.15 | 98.13 | 98.15 | 98.19 |
| EfficientNetV2B2 | 98.41 | 98.41 | 98.41 | 98.44 |
| EfficientNetV2B3 | 98.81 | 98.81 | 98.81 | 98.82 |
| EfficientNetV2S | 99.08 | 99.08 | 99.08 | 99.09 |
| EfficientNetV2M | 99.10 | 99.10 | 99.10 | 99.12 |
| EfficientNetV2L | 99.50 | 99.50 | 99.50 | 99.51 |

**Table 4**
**Performances of the pretrained models using Adamax optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| EfficientNetV2B0 | 96.54 | 96.49 | 96.54 | 96.66 |
| EfficientNetV2B1 | 96.46 | 96.42 | 96.46 | 96.54 |
| EfficientNetV2B2 | 96.16 | 96.14 | 96.16 | 96.28 |
| EfficientNetV2B3 | 97.11 | 97.11 | 97.11 | 97.16 |
| EfficientNetV2S | 97.69 | 97.67% | 97.69 | 97.71 |
| EfficientNetV2M | 98.47 | 98.47 | 98.47 | 98.48 |
| EfficientNetV2L | 98.68 | 98.68 | 98.68 | 98.71 |

**Table 5**
**Performances of the pretrained models using AdamW optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| EfficientNetV2B0 | 98.76 | 98.74 | 98.76 | 98.77 |
| EfficientNetV2B1 | 98.47 | 98.46 | 98.47 | 98.50 |
| EfficientNetV2B2 | 98.36 | 98.35 | 98.36 | 98.39 |
| EfficientNetV2B3 | 98.87 | 98.87 | 98.87 | 98.88 |
| EfficientNetV2S | 98.87 | 98.87 | 98.87 | 98.88 |
| EfficientNetV2M | 99.33 | 99.33 | 99.33 | 99.34 |
| EfficientNetV2L | 99.46 | 99.47 | 99.46 | 99.47 |

**Table 6**
**Performances of the pretrained models using Nadam optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| EfficientNetV2B0 | 98.68 | 98.67 | 98.68 | 98.69 |
| EfficientNetV2B1 | 98.51 | 98.50 | 98.51 | 98.52 |
| EfficientNetV2B2 | 98.11 | 98.10 | 98.11 | 98.14 |
| EfficientNetV2B3 | 99.03 | 99.02 | 99.03 | 99.03 |
| EfficientNetV2S | 98.95 | 98.95 | 98.95 | 98.95 |
| EfficientNetV2M | 99.48 | 99.48 | 99.48 | 99.49 |
| EfficientNetV2L | 99.37 | 99.37 | 99.37 | 99.37 |

**Table 7**
**Performances of the pretrained models using RMSProp optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| EfficientNetV2B0 | 98.47 | 98.46 | 98.47 | 98.49 |
| EfficientNetV2B1 | 98.22 | 98.20 | 98.22 | 98.27 |
| EfficientNetV2B2 | 97.95 | 97.95 | 97.95 | 98.00 |
| EfficientNetV2B3 | 98.81 | 98.81 | 98.81 | 98.82 |
| EfficientNetV2S | 98.85 | 98.85 | 98.85 | 98.87 |
| EfficientNetV2M | 99.20 | 99.20 | 99.20 | 99.21 |
| EfficientNetV2L | 99.29 | 99.29 | 99.29 | 99.30 |

#### *4.1.2. Performance of ensemble models with single optimizers*

Ensemble learning combines the predictions of several models, taking advantage of their diversity and generally leading to better performance. For this study, three pretrained EfficientNetV2S, V2M, and V2L models were used as base models; their predictions were combined by averaging and selecting the class with the highest score. The use of ensemble techniques improved the overall results compared to the individual predictions. Finally, the highest outcomes came up to 99.85% in the case of Adam and Nadam, the lowest being 99.35% for the Adamax in Table 8. During this phase, the performance from the optimization techniques Adam and Nadam gave the same results.

#### *4.1.3. Performance of the weighted ensemble models with single optimizers*

Weighted ensemble models give more or less importance to individual base models depending on their performance. The better the model, the stronger its say in the final prediction. All the same ensemble methods from the earlier section (4.1.2) are used here, but this time, the assignment of weights has been optimized through a grid search. Three-model ensembles used 1000 different weight combinations, while four-model ensembles used 10,000, calculated as $c^n$, where c represents the maximum range (in this case, 10) and n denotes the number of models, choosing the combination that granted the highest accuracy. Weighting increased the performance and generalization of all models; Adam reached the highest accuracy of 99.89%, while Adamax stayed the lowest at 99.48%, as shown in Table 9. Weighted ensembles reduced differences in performance among optimizers, although AdamW and Nadam continued to produce identical performance, which evinced the advantage of weighting based on performance to increase the robustness of models.

#### *4.1.4. Performance of ensemble models with multiple optimizers*

Creating an ensemble DL model with different optimizers involves training multiple or different CNN models, with each model using a different optimizer, and then combining their predictions to make a final prediction. Each optimizer converges to different local minima during training. The ensemble model technique can explore a broader range of solutions from many that can potentially improve generalization to unseen data by combining multiple models that were trained with different optimizers. Ensemble models trained with different optimizers introduce diversity in the ensemble, which can help prevent overfitting. If one model overfits to the training data because of the characteristics of its optimizer, then the impact of its predictions can be reduced by combining it with predictions from other models with a different optimizer. Three ensemble model techniques are explored in this section, utilizing four optimizers (Adam, AdamW, Nadam, and RMSProp). Adamax is excluded from the base models participating in this ensemble due to its lower performance in the previous section. Specifically, Ensemble_11, Ensemble_12, and Ensemble_13 utilize four EfficientNetV2S, four EfficientNetV2M, and four EfficientNetV2L models, respectively, as their base models. Each of these ensembles employs Adam, AdamW, Nadam, and RMSProp as optimizers for their base models. It is observed that Ensemble_13 outperforms the other ensembles by achieving an accuracy of 99.64%, while Ensemble_11 achieves lower results with an accuracy of 99.25%, as presented in Table 10.

**Table 10**
**Performances of ensemble pretrained models using multiple optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| Ensemble_11 | 99.25 | 99.25 | 99.25 | 99.26 |
| Ensemble_12 | 99.41 | 99.41 | 99.41 | 99.41 |
| Ensemble_13 | 99.64 | 99.64 | 99.64 | 99.64 |

**Table 8**
**Performances of ensemble pretrained models using single optimizer**

| Model | Optimizer | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|---|
| Ensemble_1 | Adamax | 99.35 | 99.35 | 99.35 | 99.35 |
| Ensemble_2 | Adam | 99.85 | 99.85 | 99.85 | 99.85 |
| Ensemble_3 | AdamW | 99.80 | 99.79 | 99.79 | 99.79 |
| Ensemble_4 | Nadam | 99.85 | 99.85 | 99.85 | 99.85 |
| Ensemble_5 | RMSProp | 99.69 | 99.69 | 99.69 | 99.70 |

**Table 9**
**Performances of weighted ensemble pretrained models using single optimizer**

| Model | Optimizer | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|---|
| Ensemble_6 | Adamax | 99.48 | 99.48 | 99.48 | 99.49 |
| Ensemble_7 | Adam | 99.89 | 99.89 | 99.89 | 99.89 |
| Ensemble_8 | AdamW | 99.87 | 99.87 | 99.87 | 99.87 |
| Ensemble_9 | Nadam | 99.87 | 99.87 | 99.87 | 99.87 |
| Ensemble_10 | RMSProp | 99.75 | 99.75 | 99.75 | 99.75 |

*4.1.5. Performance of weighted ensemble models with multiple optimizers*

The techniques utilized in Section 4.1.4 are replicated in this section with the additional step of assigning weights to the individual base models. A grid search algorithm is employed to search for the optimum weight combinations, just as used in Section 4.1.3. This further enhances the performance of the ensemble model. The algorithm generates 10,000 ($10\hat{4}$) weight combinations, and the combination yielding the highest performance is selected and assigned to the base models. It is observed that the performances of the three ensemble models have increased, with Ensemble_16 achieving the highest accuracy of 99.81% and Ensemble_14 achieving the lowest accuracy of 99.45%, as shown in Table 11.

**Table 11**
**Performances of weighted ensemble pretrained models using multiple optimizer**

| Model | Acc. | F1-sc. | Rec. | Prec. |
|---|---|---|---|---|
| Ensemble_14 | 99.45 | 99.44 | 99.45 | 99.45 |
| Ensemble_15 | 99.52 | 99.52 | 99.52 | 99.53 |
| Ensemble_16 | 99.81 | 99.81 | 99.81 | 99.81 |

*4.1.6. Performance comparison of the proposed model with other existing works in the literature*

Previously, multiple studies have introduced methods for detecting tomato plant diseases based on images of tomato leaves. Our proposed approach is compared to some of these studies that utilized the tomato PlantVillage dataset. The evaluation of these existing methods focused solely on accuracy, which is the metric commonly reported by many studies, as detailed in Table 12.

The proposed approach overcomes major limitations in previous tomato disease classification studies with the combination of seven EfficientNetV2 variants with optimizer tuning and weighted ensembling, which yields a peak accuracy of 99.89%.

**Table 12**
**Comparison of results with existing models from the literature**

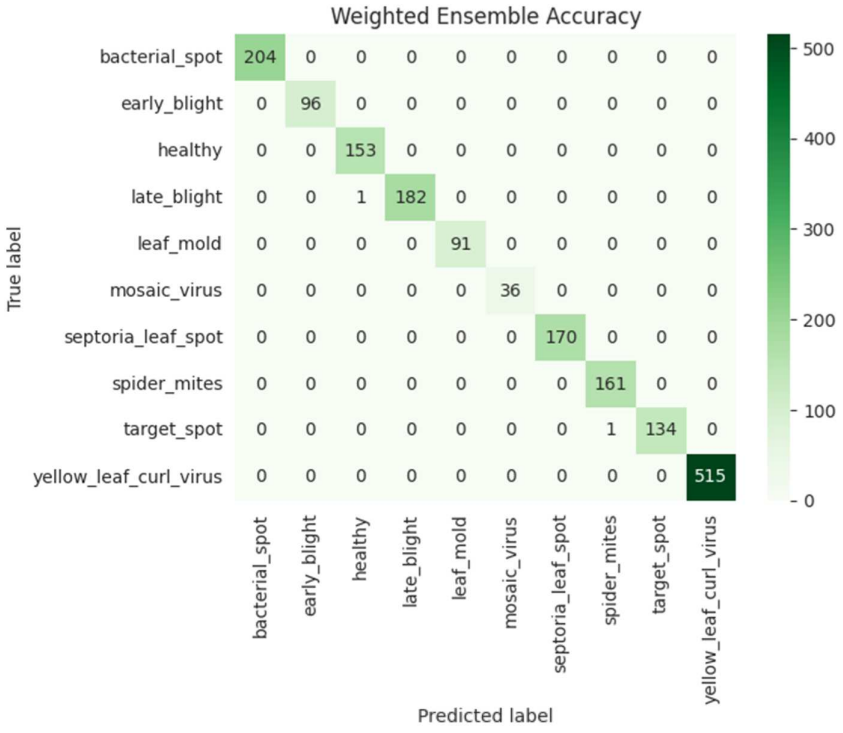
| Reference | Methods | Acc. |
|---|---|---|
| [14] | DCGAN, GoogLeNet | 94.33 |
| [15] | Random forest, decision tree | 94.00 |
| [16] | Resnet50, Xception, MobileNet, etc. | 97.10 |
| [17] | PCA, WO, DNN | 94.00 |
| [20] | VGG16 | 95.71 |
| [19] | C-GAN, DenseNet121 | 99.51, |
| [18] | Xception | 99.55 |
| [21] | MobileNetV2 | 99.30 |
| Proposed | Proposed model based on Ensemble_7 | 99.89 |

Multi-optimizer usage, extensive augmentation, and holding out the test set strictly improve the generalization under challenging conditions. Efficient scaling by EfficientNetV2 and careful balancing of the ensemble also give strong performance without heavy computational demands imposed by GAN-based or CNN–Transformer hybrids. Overall, the framework delivers superior accuracy, robustness, and scalability compared to prior approaches.

## 4.2. Confusion matrix

Confusion matrices were used to evaluate the proposed models by showing correct and incorrect predictions across all classes. They help reveal which classes are frequently confused and report outcomes such as TP, TN, FP, and FN. The Ensemble_7 model misclassified only two samples, demonstrating the strong performance of the weighted ensemble approach. Figure 3 presents the confusion matrices for Ensemble_7 across the 10 tomato disease classes.

**Figure 3**
**Confusion matrix of the proposed model**



### 4.3. Discussion

The study trains seven EfficientNetV2 variants (B0–L) for 100 epochs on Kaggle using five optimizers, where the highest single-model accuracy achieved is 99.50% by EfficientNetV2L-Adam (Table 6). Sixteen ensemble models were developed. First of all, five ensembles built from EfficientNetV2-S, M, and L reached up to 99.85% accuracy when using Adam and Nadam. Weighted ensembles (Models 6–10) further improved performance, with the highest accuracy of 99.89% achieved by the weighted ensemble based on Adam (Table 2). Ensembles using four identical backbones (Models 11–13) showed that the EfficientNetV2L-based ensemble is the best at 99.64%. Applying optimal weights again in Models 14–16 enables Ensemble-16 to reach an accuracy of 99.81%. Performance differences across models reflect both optimizer behavior and architectural complexity. Also, larger variants, especially M and L, consistently surpassed their smaller counterparts due to their richer feature-learning abilities. These tendencies justify the proposed performance-weighted ensemble strategy that can grasp strong points across diverse model sizes. Overall, the final weighted ensemble had the best accuracy of 99.89%, indicating its robustness against challenging real-world scenarios such as light variations, background noise, and similar symptoms due to other diseases. This further supports the results as shown in the confusion matrix in Figure 3.

## 5. Conclusion

The proposed weighted EfficientNetV2 ensemble greatly improves tomato leaf disease classification, offering more stable and reliable predictions under challenging conditions. It provides a practical, scalable solution for automated plant disease monitoring and supports early detection in precision agriculture.

## 6. Future Work

Future research will extend the weighted ensemble framework to more diverse plant disease datasets, further evaluating its generalization across crops and environments. This work will also investigate the integration of lightweight attention modules together with ViT-based backbones that grant better robustness under challenging visual conditions. Another effort will be in the study of meta-heuristic optimization methods for the automation and scaling of the optimal ensemble weights more efficiently compared with grid search.

## Acknowledgment

The authors would like to acknowledge the support of Prince Sultan University, Riyadh Saudi Arabia for paying the Article Processing Charges (APC) of this publication.

## Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

## Data Availability Statement

Data are available from the corresponding author upon reasonable request.

## Author Contribution Statement

**Aliyu Tetengi Ibrahim:** Conceptualization, Methodology, Formal analysis, Writing – original draft. **Ibrahim Hayatu Hassan:** Conceptualization, Methodology, Formal analysis, Resources, Data curation, Writing – original draft. **Mohammed Abdullahi:** Validation, Formal analysis, Resources, Data curation, Writing – review & editing, Visualization. **Abeer Rashad Mirdad:** Conceptualization, Software, Validation, Investigation, Writing – original draft, Writing – review & editing, Supervision, Project administration. **Muhammad I. Khan:** Methodology, Validation, Investigation, Resources, Data curation, Writing – review & editing, Visualization, Project administration. **Saeed Ali Bahaj:** Conceptualization, Software, Writing – original draft. **Fatima Khan Nayer:** Investigation, Visualization, Supervision, Project administration.

## References

[1] Yusuf, H. M., Yusuf, S. A., Abubakar, A. H., Abdullahi, M., & Hassan, I. H. (2024). A systematic review of deep learning techniques for rice disease recognition: Current trends and future directions. *Franklin Open*, *8*, 100154. https://doi.org/10.1016/j.fraope.2024.100154

[2] Anandhi, D. R. F. R., & Sathiamoorthy, S. (2023). Enhanced sea horse optimization with deep learning-based multimodal fusion technique for rice plant disease segmentation and classification. *Engineering, Technology & Applied Science Research*, *13*(5), 11959–11964. https://doi.org/10.48084/etasr.6324

[3] Liu, J., & Wang, X. (2020). Tomato diseases and pests detection based on improved Yolo V3 convolutional neural network. *Frontiers in Plant Science*, *11*, 898. https://doi.org/10.3389/fpls.2020.00898

[4] Atila, Ü., Uçar, M., Akyol, K., & Uçar, E. (2021). Plant leaf disease classification using EfficientNet deep learning model. *Ecological Informatics*, *61*, 101182. https://doi.org/10.1016/j.ecoinf.2020.101182

[5] Salih, H. S., Ali, M. H., & Khan, M. I. (2025). IoT-enabled cloud storage data access control model based on blockchain technology. *International Journal of Theoretical & Applied Computational Intelligence*, *2025*, 125–144. https://doi.org/10.65278/IJTACI.2025.14

[6] Khan, M. A., Akram, T., Sharif, M., & Saba, T. (2020). Fruits diseases classification: Exploiting a hierarchical framework for deep features fusion and selection. *Multimedia Tools and Applications*, *79*(35), 25763–25783. https://doi.org/10.1007/s11042-020-09244-3

[7] Mohammed, H. F., Alkakjea, H. A. M., Al-mafrachi, B. N. A. R., Al-Khaffaf, M. S., & Saad, A. (2025). A novel deep learning approach for classification of abnormal teeth in panoramic X-rays. *International Journal of Theoretical & Applied Computational Intelligence*, *2025*, 22–34. https://doi.org/10.65278/IJTACI.2025.7

[8] Karimi, M., Karimi, Z., Khosravi, M., Delaram, Z., Dehsheikhim, M. H., Najafabadi, S. A., ..., & Tavakoli, N. (2025). Feature selection methods in big medical databases: A comprehensive survey. *International Journal of Theoretical & Applied Computational Intelligence*, *2025*, 181–209. https://doi.org/10.65278/IJTACI.2025.21

[9] Khan, M. A., Sharif, M. I., Raza, M., Anjum, A., Saba, T., & Shad, S. A. (2022). Skin lesion segmentation and classification: A unified framework of deep neural network features fusion

and selection. *Expert Systems*, *39*(7), e12497. https://doi.org/10.1111/exsy.12497

[10] Raman, R., Jiang, W., & Bakshi, S. (2025). PODE: Panoptic-aware object-wise depth estimation for occlusion prediction. *IEEE Transactions on Consumer Electronics*, *71*(4), 10906–10916. https://doi.org/10.1109/TCE.2025.3610083

[11] Liu, F., Zheng, Q., Tian, X., Shu, F., Jiang, W., Wang, M., . . . , & Saponara, S. (2025). Rethinking the multi-scale feature hierarchy in object detection transformer (DETR). *Applied Soft Computing*, *175*, 113081. https://doi.org/10.1016/j.asoc.2025.113081

[12] Yang, B., Wang, X., Xing, Y., Cheng, C., Jiang, W., & Feng, Q. (2024). Modality fusion vision transformer for hyperspectral and LiDAR data collaborative classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *17*, 17052–17065. https://doi.org/10.1109/JSTARS.2024.3415729

[13] Ayesha, N., Hassan, I. H., Mirdad, A. R., & Khan, A. R. (2025). EfficientNet deep learning model for lung cancer early diagnosis from computed tomography scan images with transfer learning. *Journal of Advances in Information Technology*, *16*(7), 999–1008. https://doi.org/10.12720/jait.16.7.999-1008

[14] Wu, Q., Chen, Y., & Meng, J. (2020). DCGAN-based data augmentation for tomato leaf disease identification. *IEEE Access*, *8*, 98716–98728. https://doi.org/10.1109/ACCESS.2020.2997001

[15] Basavaiah, J., & Arlene Anthony, A. (2020). Tomato leaf disease classification using multiple feature extraction techniques. *Wireless Personal Communications*, *115*(1), 633–651. https://doi.org/10.1007/s11277-020-07590-x

[16] Hong, H., Lin, J., & Huang, F. (2020). Tomato disease detection and classification by deep learning. In *2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering*, 25–29. https://doi.org/10.1109/ICBAIE49996.2020.00012

[17] Gadekallu, T. R., Rajput, D. S., Reddy, M. P. K., Lakshmanna, K., Bhattacharya, S., Singh, S., . . . , & Alazab, M. (2021). A novel PCA–whale optimization-based deep neural network model for classification of tomato plant diseases using GPU. *Journal of Real-Time Image Processing*, *18*(4), 1383–1396. https://doi.org/10.1007/s11554-020-00987-8

[18] Thangaraj, R., Anandamurugan, S., & Kaliappan, V. K. (2021). Automated tomato leaf disease classification using transfer learning-based deep convolution neural network. *Journal of Plant Diseases and Protection*, *128*(1), 73–86. https://doi.org/10.1007/s41348-020-00403-0

[19] Abbas, A., Jain, S., Gour, M., & Vankudothu, S. (2021). Tomato plant disease detection using transfer learning with C-GAN synthetic images. *Computers and Electronics in Agriculture*, *187*, 106279. https://doi.org/10.1016/j.compag.2021.106279

[20] Paymode, A. S., & Malode, V. B. (2022). Transfer learning for multi-crop leaf disease image classification using convolutional neural network VGG. *Artificial Intelligence in Agriculture*, *6*, 23–33. https://doi.org/10.1016/j.aiia.2021.12.002

[21] Ahmed, S., Hasan, M. B., Ahmed, T., Sony, M. R. K., & Kabir, M. H. (2022). Less is more: Lighter and faster deep neural architecture for tomato leaf disease classification. *IEEE Access*, *10*, 68868–68884. https://doi.org/10.1109/ACCESS.2022.3187203

[22] Rahman, S. U., Alam, F., Ahmad, N., & Arshad, S. (2023). Image processing based system for the detection, identification and treatment of tomato leaf diseases. *Multimedia Tools and Applications*, *82*(6), 9431–9445. https://doi.org/10.1007/s11042-022-13715-0

[23] Sun, Y., Ning, L., Zhao, B., & Yan, J. (2024). Tomato leaf disease classification by combining EfficientNetv2 and a swin transformer. *Applied Sciences*, *14*(17), 7472. https://doi.org/10.3390/app14177472

[24] Nazir, T., Iqbal, M. M., Jabbar, S., Hussain, A., & Albathan, M. (2023). EfficientPNet—An optimized and efficient deep learning approach for classifying disease of potato plant leaves. *Agriculture*, *13*(4), 841. https://doi.org/10.3390/agriculture13040841

[25] Sinamenye, J. H., Chatterjee, A., & Shrestha, R. (2025). Potato plant disease detection: Leveraging hybrid deep learning models. *BMC Plant Biology*, *25*(1), 647. https://doi.org/10.1186/s12870-025-06679-4

[26] Tiwari, S., Choudhury, T., & Kotecha, K. (2024). Ensembling of transfer learning for enhanced precision agriculture in plant disease classification. In *Emerging Trends in Expert Applications and Security: Proceedings of ICE-TEAS 2024*, *2*, 135–144. https://doi.org/10.1007/978-981-97-3991-2_12

[27] Hayat, M., Ahmad, N., Nasir, A., & Tariq, Z. A. (2024). Hybrid deep learning EfficientNetV2 and vision transformer (EffNetV2-ViT) model for breast cancer histopathological image classification. *IEEE Access*, *12*, 184119–184131. https://doi.org/10.1109/ACCESS.2024.3503413

[28] Tan, M., & Le, Q. (2021). EfficientNetV2: Smaller models and faster training. In *Proceedings of the 38th International Conference on Machine Learning, 139*, 10096–10106.

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

[30] Dogo, E. M., Afolabi, O. J., & Twala, B. (2022). On the relative impact of optimizers on convolutional neural networks with varying depth and width for image classification. *Applied Sciences*, *12*(23), 11976. https://doi.org/10.3390/app122311976

[31] Sayın, K. A., Gürsoy, N. K., Yolcu, T., & Gürsoy, A. (2025). On the synergy of optimizers and activation functions: A CNN benchmarking study. *Mathematics*, *13*(13), 2088. https://doi.org/10.3390/math13132088

[32] Barati, B., Erfaninejad, M., & Khanbabaei, H. (2025). Evaluation of effect of optimizers and loss functions on prediction accuracy of brain tumor type using a light neural network. *Biomedical Signal Processing and Control*, *103*, 107409. https://doi.org/10.1016/j.bspc.2024.107409

[33] Panchbhai, K. G., Lanjewar, M. G., & Naik, A. V. (2025). Modified MobileNet with leaky ReLU and LSTM with balancing technique to classify the soil types. *Earth Science Informatics*, *18*(1), 77. https://doi.org/10.1007/s12145-024-01521-1

[34] Pinjarkar, L., Nittala, A., Mattada, M. P., Pinjarkar, V., Neole, B., & Math, M. K. (2025). Optimizing neural radiance field: A comprehensive review of the impact of different optimizers on neural radiance fields. *Bulletin of Electrical Engineering and Informatics*, *14*(1), 476–484. https://doi.org/10.11591/eei.v14i1.8315

[35] Shen, Y., Yang, Z., Khan, Z., Liu, H., Chen, W., & Duan, S. (2025). Optimization of improved YOLOv8 for precision tomato leaf disease detection in sustainable agriculture. *Sensors*, *25*(5), 1398. https://doi.org/10.3390/s25051398

[36] Thanjaivadivel, M., Gobinath, C., Vellingiri, J., Kaliraj, S., & Femilda Josephin, J. S. (2025). EnConv: Enhanced CNN for leaf disease classification. *Journal of Plant Diseases and Protection*, *132*(1), 32. https://doi.org/10.1007/s41348-024-01033-6

[37] Maijeddah, U. I., Yusuf, S. A., Abdullahi, M., & Hassan, I. H. (2024). A hybrid transfer learning model with optimized SVM using honey badger optimization algorithm for multi-class lung cancer classification. *Science World Journal*, *19*(4), 977–986. https://doi.org/10.4314/swj.v19i4.10

**How to Cite:** Ibrahim, A. T., Hassan, I. H., Abdullahi, M., Mirdad, A. R., Khan, M. I., Bahaj, S. A., & Nayer, F. K. (2026). A Weighted Ensemble of EfficientNetV2 Variants with Optimizer Tuning for Tomato Leaf Disease Classification. *Journal of Computational and Cognitive Engineering*, *5*(2), 292–303. https://doi.org/10.47852/bonviewJCCE62027632