Advances in Technology Innovation, vol. 8, no. 3, 2023, pp. 192-209

An Improved MobileNet for Disease Detection on Tomato Leaves

Hai Thanh Nguyen[1,*], Huong Hoang Luong[2], Long Bao Huynh[2], Bao Quoc Hoang Le[2], Nhan Hieu Doan[2], Duc Thien Dao Le[2]

[1]College of Information and Communication Technology, Can Tho University, Can Tho, Vietnam

[2]Information Technology Department, FPT University, Can Tho, Vietnam

Received 15 February 2023; received in revised form 02 April 2023; accepted 04 April 2023

DOI: https://doi.org/10.46604/aiti.2023.11568

Abstract

Tomatoes are widely grown vegetables, and farmers face challenges in caring for them, particularly regarding plant diseases. The MobileNet architecture is renowned for its simplicity and compatibility with mobile devices. This study introduces MobileNet as a deep learning model to enhance disease detection efficiency in tomato plants. The model is evaluated on a dataset of 2,064 tomato leaf images, encompassing early blight, leaf spot, yellow curl, and healthy leaves. Results demonstrate promising accuracy, exceeding 0.980 for disease classification and 0.975 for distinguishing between diseases and healthy cases. Moreover, the proposed model outperforms existing approaches in terms of accuracy and training time for plant leaf disease detection.

Keywords: plant diseases, transfer learning, fine-tuning, MobileNet, mobile devices

1. Introduction

Nowadays, the research and application of automatic identification of plant diseases using leaves are fundamental to agricultural needs. Moreover, Hassan et al. [1] reported that early and accurate identification of crop diseases helps farmers to reduce some difficulties and risks and improve the productivity and quality of agricultural products. The tomato plant, one of the easiest fruits to grow, is suitable for many soil types. Additionally, tomatoes bring very high nutritional and economic value to human lives. Tomato contains many antioxidants and vitamins, essential for people's overall health. However, sometimes farmers also need help checking and determining if the tomato fruit quality is up to the standard or if the tomato plant is healthy or not because sometimes diseases of tomato plants usually manifest mainly on their leaves [2].

Moreover, insects and pests that attack tomato plants and produce numerous illnesses can stymie the production of this well-known crop. Therefore, farmers must first understand the illness to treat diseases on tomato leaves manually. They face many problems yearly as they try to raise their healthy harvests to keep the profits from farming and raising livestock. Bhagwat and Dandawate [3] mentioned some countries have a gross domestic product (GDP) of up to 25%.

Insects and other harmful viruses damage production lines and slow them down to the point where they can no longer produce as much as they can. This is a big problem for the industry, especially farmers. Even though farmers use a variety of pesticides and insecticides to protect their tomato plants from disease, they frequently need to gain knowledge of the disease and how to prevent it. Excessive pesticide and insecticide usage endangers human health and life. Crop damage can result from incorrect disease diagnosis and applying too many or too few pesticides. In addition, the improper use of pesticides also seriously affects the surrounding soil and water environment and so on.

* Corresponding author. E-mail address: nthai.cit@ctu.edu.vn

English language proofreader: Chih-Wen Teng

Diagnosing tomato plant diseases is very important to achieve maximum yields. On the other hand, manually detecting disease by carefully examining the crop is time-consuming and complicated. Farmers often find it challenging to contact specialists in remote areas and take steps to prevent unusual diseases. It is easier to detect with helpful information. That is why images can be considered self-contained information helping the system detect diseases. Visual studies of plants without prior information can lead to inaccurate disease diagnoses. As a result, preventive measures are used. Using machines can help locate damaged tomato plants, determine what diseases affect them, and use that knowledge to help the rest of the crop grow more efficiently and with less loss. MobileNet is one of the great models the authors proposed because of its low-latency, low-power models that can be parameterized to meet the resource constraints of a wide range of use cases. It can be used to build classification and detection systems. This study examines the leading convolutional neural network (CNN) MobileNet architecture. They applied transfer learning and refinement to pre-trained data: tomato leaves ranging from healthy to diseases.

The goal is to maximize model accuracy and quickly set up "data augmentation" factors to help in disease categorization and detection on tomato leaves. The following are the key contributions:

(1) This study devised the first transfer learning technique and development stage to detect diseases in tomato leaves. The final stage is to fine-tune the parameters accordingly.

(2) Some machine learning-based architectures were leveraged during the implementation and compared the findings to the CNN models. VGG16, VGG19, MobileNet, DenseNet201, and Xception are other examples.

(3) Three main scenarios were assessed with various metrics to present the predictions for detecting and distinguishing foliar diseases. First, they demonstrated empirically that the author's suggested strategy outperforms alternative CNN and Transformer-based architectures and earlier illness detection and classification models on tomato leaves.

(4) The work obtained promising results using the multilayer classification of the CNN architecture given above, and MobileNet delivered the best results.

The rest of the article is divided into five following sections. Section 1 begins with an introduction and explanation of the problem. Section 2 will then present similar works. Section 3 will next demonstrate the implementation process. The outcomes of the experiments are shown in Section 4. Finally, Section 5 is followed by the conclusion.

2. Related Work

To build automated decontamination procedures on plant detection systems, some researchers have used advanced technology such as machine learning and neural network design including GoogleNet, AlexNet, InceptionV3, VGG16, and SqueezeNet, etc. to perform research and classification of crops based on machine learning. As a result, they use exact methods to detect plant diseases in tomato leaves. Furthermore, researchers have developed various deep learning-based disease detection and classification methods.

Bhagwat and Dandawate [3] presented some ways to detect plant disease for machine learning. The authors provided the traditional machine learning method using various evaluation metrics or the new machine learning on RGB images, which best reported the accuracy at 91.5%.

Fazari et al. [4] proposed a method to detect anthracnose disease on olives using the ResNet101 model. The proposed method was successfully tested, which resulted in an accuracy of 91.8%. This method promises to be one of the proposals to ensure the quality of olives and olive oil is improved and meets the required standards. About the dataset, images were shot on several days and with different lighting to create a dataset with distinct phases of error growth, the data set must satisfy the condition that there must be enough light, and the leaves must see the characteristic manifestations of that disease and the image must not be blurred. On the ground, each image includes a particular leaf object.

Lu et al. [5] introduced and clarified the use of the deep learning (DL) method and suggested that the image set needs to go through the preprocessing stage (increasing contrast, slightly blurring, reducing brightness, etc.) For picture categorization, the majority of these approaches employ ordinary neural networks. For example, using the PlantVillage dataset, Brahimi et al. [6] employed GoogleNet and AlexNet to pinpoint disease zones in the Tomato plant. As a result, they classified nine illnesses with 99.18% accuracy. However, because their model was one of the first proposed, it is gradually becoming obsolete due to processing speed and model size, resulting in a limitation that the research team mentioned that the model could not be applied on mobile devices.

Alruwaili et al. [7] proposed a new method, known as the real-time faster region convolutional neural network (RTF-RCNN) model, for trained & tested with other models using each different parameters such as precision, accuracy, and recall. The final result was 97.42% accuracy. With new technology and support for real-time disease detection, the proposed model outperforms other models and holds great promise for the future. However, learning and acclimating to it can be challenging.

Durmus et al. [8] employed and compared AlexNet and SqueezeNet to classify diseases in real-time. Using the PlantVillage dataset, the authors categorized ten illnesses and a healthy leaf with 95.65% accuracy for AlexNet and 94.3% for SqueezeNet. The SqueezeNet model has shown its strengths due to the AlexNet model in approximate prediction results. However, it is very light and convenient, promising applications with mobile electronic devices.

Saeed et al. [9] and Adhikari et al. [10] presented a pre-trained network model for detecting and categorizing tomato illness has been presented. The authors used ResNetV2 model and increased the dropout coefficient from 5% to 10% to 50%. Each batch of changes demonstrates that these parameters contributed to better model training and fewer errors. Zhang et al. [11] discussed using a DL CNN to identify tomato leaf disease. With an accuracy of 97.19%, the paper used many pre-trained networks such as AlexNet, GoogleNet, and ResNet.

Ishak et al. [12] discussed a method for analyzing plant leaf quality, beginning with image acquisition, image processing, and classification. The images were collected using an 8-megapixel intelligent phone era, and the samples were divided into fifty for healthy and fifty for unhealthy. The image processing method is divided into three steps: contrast enhancement, segmentation, and feature extraction. The classification method was performed using an artificial neural network, which employs a multi-layer feed-forward neural network, followed by a comparison of two network structures, Multilayer perceptron (MLP) and radial basis function (RBF). The RBF network outperformed the MLP network in terms of performance. However, the search only distinguishes between healthy and unhealthy plant leaf images. It cannot determine the type of disease.

A basic CNN model with eight hidden layers was used to identify a tomato plant's circumstances. Compared to other classical models [13-17], the proposed strategies produce the best results. DL approaches are used in the image processing methodology to identify and categorize tomato plant illnesses [13]. The author constructed a comprehensive system using the segmentation technique and CNN. A variant in the CNN model was adopted and used to improve accuracy. In research about the application of disease identification on tomato leaves, the author introduced and used a CNN model developed by Tm et al. [14] called LeNet. This process follows three steps: data collection, preprocessing, and classification. With outstanding and promising results with an accuracy of 94-95%. Trivedi et al. [15] presented this article discussing standard profound learning models and variants. This article discussed biotic diseases caused by fungal and bacterial pathogens, specifically tomato leaf blight, blast, and browning. The proposed model detection rate was 98.49% correct. The proposed model was compared to VGG and ResNet versions using the same dataset.

In another work, Suryanarayana et al. [16] reviewed all CNN variants for plant disease classification. The authors also briefed all DL principles employed for leaf disease diagnosis and classification. The authors concentrated on the most recent CNN models and evaluated their performance. In this paper, the authors summarize several CNN versions, such as VGG16, VGG19, and ResNet, while also examining their benefits, drawbacks, and prospects for use in various applications. Bhagwat

and Dandawate [17] conducted a study on disease identification on crops, the authors applied hand-crafter features (HCF) together with CNN in the form of fusion. The author also mentioned fine-tuning the central coefficient. As a result, the accuracy for the entire leaf dataset is 99.93%, which is impressive. This result is auspicious, demonstrating improved results based on the prominence of feature fusion in crop disease detection. Coulibaly et al. [18] used CNN VGG16 to perform state classification detection of millet crop. The machine learning process is fast, with promising results such as an accuracy of 95.00%, precision of 90.50%, recall of 94.50%, and an F1-score of 91.75%.

Ashwinkumar et al. [19] proposed using MobileNet as a base CNN model and built a new automated model for detecting and classifying plant leaf diseases based on an optimal mobile network-based convolutional neural network (OMNCNN). The proposed OMNCNN model goes through several stages, including preprocessing, segmentation, feature extraction, and classification. With a higher accuracy of 98.7%, the proposed OMNCNN methodology achieved maximum performance. As a result, the CNN model is an effective real-time tool for detecting and classifying plant leaf diseases. Kaya and Gursoy [20] proposed a novel approach based on DL for plant disease detection. The authors fused RGB images with segmentation images, then considered them with a multi-headed DenseNet-based architecture that the authors developed. The accuracy result was 98.17%, which is also promising because the authors mentioned that we could apply this model to the early prediction of diseases on the leaves of plants and reduce costs and losses, which affect the quality of agricultural products.

Thakur et al. [21] introduced a lightweight CNN model called "VGG-ICNN". VGG-ICNN has approximately 6 million parameters, much fewer than most high-performing DL models. The model's performance is evaluated using five public datasets, including various crop kinds. This model produces compelling results with up to 99.16% accuracy. However, because this is the model used to classify crop diversity, it can be inappropriate to compare it with the proposed model.

3. Method

Convolutional neural networks (CNNs) with fine-tuning have shown great promise in the classification of plant diseases. CNN models can efficiently analyze and classify various types of plant diseases, assisting in early detection and efficient management. This is made possible by utilizing the strength of deep learning and transfer learning techniques. First, data collection is important because it ensures clear, bright, and visible images, which leads to better training results. Next, the reference and selection of suitable CNN models are equally important, depending on the degree of accuracy that the model performs, along with the amount of storage that the model occupies in the system. Then comes the stage of decomposing and fine-tuning the hyperparameters. Finally, apply comparison, evaluation, and validation. Remember to periodically update and retrain the model as new data becomes available to ensure its continued accuracy and relevance in classifying tomato leaf diseases.

3.1. Overall workflow

Fig. 1 shows the process of performing machine learning and leaf classification in tomato plants in 8 steps. The following steps will be shown in the order of arrows from the "Start" button to the sequential steps from 1 to 8, and finally the end of the flow. The machine learning model uses a data file containing four picture files of tomato leaves categorized by name to prepare and perform training.

1. The first stage in this process is to gather data. This information was gathered from the PlantVillage dataset† file submitted by the author Charu, who is a user of the Kaggle application. This data set was gathered from a group of tomato plants cultivated together. The data set contains 14,529 JPG photos with a pixel size of 256 px. The authors collected three disease samples and one healthy sample for 2,064 pictures split into four categories: early blight, Septoria, yellow curl, and healthy.

† PlantVillage tomato leaf dataset: https://www.kaggle.com/datasets/charuchaudhry/plantvillage-tomato-leaf-dataset

2. Researchers started to perform image discrimination of tomato leaves, in which there can be three diseases and one healthy type. One of the members used five well-known CNN architectures (MobileNet, DenseNet, VGG16, VGG19, and Xception) to analyze and compare training and testing outcomes for four types of image recognition, as described above. The authors involve partitioning the data set. The obtained data is separated into training, validation, and testing. Specifically, this division is done in the ratio of 60% of the photos used for training, 20% for evaluating, and 20% to test the model's correctness.
3. In the following stage, the authors use several MobileNet models and other CNNs to detect and classify each tomato plant leaf illness as transfer learning models.
4. The authors improve the hyperparameters (including epoch, learning rate, batch size, patch size, weight decay, etc.), volume, project opacity, numeric header, and transform layers and add specific layers to optimize for DL models and get the best prediction results.
5. The authors retrain using the proposed MobileNet and CNN samples from the previous stage's refinement. Following retraining, the next stage is to compute and assess the model's accuracy against the test set.
6. Validation and metric computation: They count how many images were correctly classified and compute metrics to compare the proposed model to other CNN architectures, most notably test accuracy and F1-score results.
7. The authors need to compare the outcomes of the CNN models with the suggested model. Therefore, after training, validating, and testing the proposed model, they compared the results to original CNN architectures like VGG16 [22], VGG19, DenseNet, Xception, and MobileNet [23].
8. The final step is showing the result. After the data has been compared, tables and graphs will be displayed for comparison.

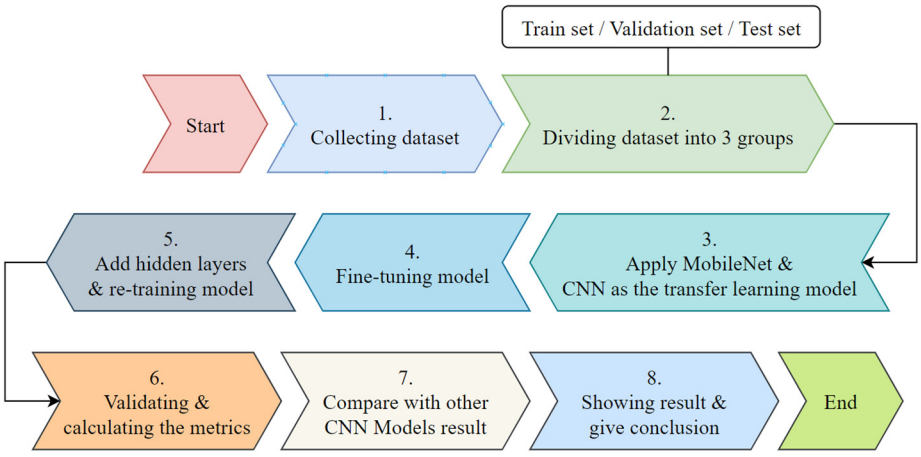


Fig. 1 Implementation process flow

Table 1 shows CNN Models which are used to prepare for the training. There are many CNN models provided, but the models are selected because of the high results in the training process. The table reveals the pros and cons of each model. In addition, the parameter part shows the popularity and usage of that model based on Keras's statistics.

Table 1 List of deep learning models

Deep learning model	#Parameters	Key features with pros and cons
VGG16	138.4 M	Popular, lightweight, and easy to use with high accuracy.
VGG19	143.7 M	Similarity to VGG16 but with 19 layers deep. High accuracy and great understanding of shape, color, and structure.
MobileNet	4.3 M	Consider the depth-wise separable convolution concept. Reduced parameters significantly.
Xception	22.9 M	A depth-wise separable convolution approach.
DenseNet201	20.2 M	Dense connections between the layers. Reduced number of parameters with better accuracy.

3.2. Dataset

The tomato leaf disease dataset is collected from the open source Kaggle platform, the folder named PlantVillage, including 2,064 images taken corresponding to 3 diseases and one healthy, including (early blight, yellow curl, Septoria, and healthy). The data set is divided into three sets with a ratio of 60-20-20 corresponding to the training set-validation set-test set. The dataset has an individual image size of 256 × 256 pixels, as MobileNet takes the form of static square images [21]. A list of image numbers is described in Table 2, and the illustration of the leaf image stands for the leaf status in Fig. 2.

(a) Healthy (b) Early blight (c) Septoria (d) Yellow curl

Fig. 2 Some samples of each status of the tomato leaf

Before implementing section training, the authors pre-processed the image by visualizing, dividing the image size by 255, and then applying the data augmentation method to generate more new images. Data augmentation has seven techniques applied to generate new images, including rotation, zoom, shear, width shift, height shift, horizontal flip, and vertical flip. After generating images, the datasets increase significantly, twice as much as the original dataset, and the difference between datasets is low. Eligible to apply to model and fine-tuning.

Table 2 Classes of Tomato's leaf image and its usage in learning

Class category	Number of images	Images after data augmentation	Images used for the training set (60%)	Images used for validating set (20%)	Images used for the testing set (20%)
Early blight	504	1008	605	201	202
Septoria	528	1056	633	212	211
Yellow curl	507	1014	608	202	204
Healthy	525	1050	630	210	210
Total	2064	4128	2476	825	827

3.3. The proposed MobileNet

This study used MobileNet transfer learning techniques and fine-tuning methodologies to categorize color photos under ideal lighting circumstances, including early blight, Septoria, yellow curl, and healthy leaves. As revealed in Fig. 3, the MobileNet employs depthwise separable convolution (DSC) to reduce the number of computations, the number of parameters, and the ability to perform feature extraction on different channels independently.

Five popular DL models, namely VGG16 [22], VGG19, DenseNet, MobileNet [23], and Xception, were used to build an accurate automated model for general purpose and trained on a diverse set of examples, such as ImageNet with 1000 classes. In addition, in some studies, for example, Coulibaly et al. [18] proposed to use VGG16 to classify and identify many individual plants together on a dataset, PlantVillage dataset for these dilated CNN networks to improve accuracy. Therefore, those models are expected to provide promising performance in disease detection on tomato leaves.

The dilated convolution expands the kernel's field of view while maintaining the same computational complexities by inserting "holes" or zeros between the kernels of each convolutional layer. As a result, it can be used for applications that require a wide field of vision but cannot support larger kernels or many convolutions.

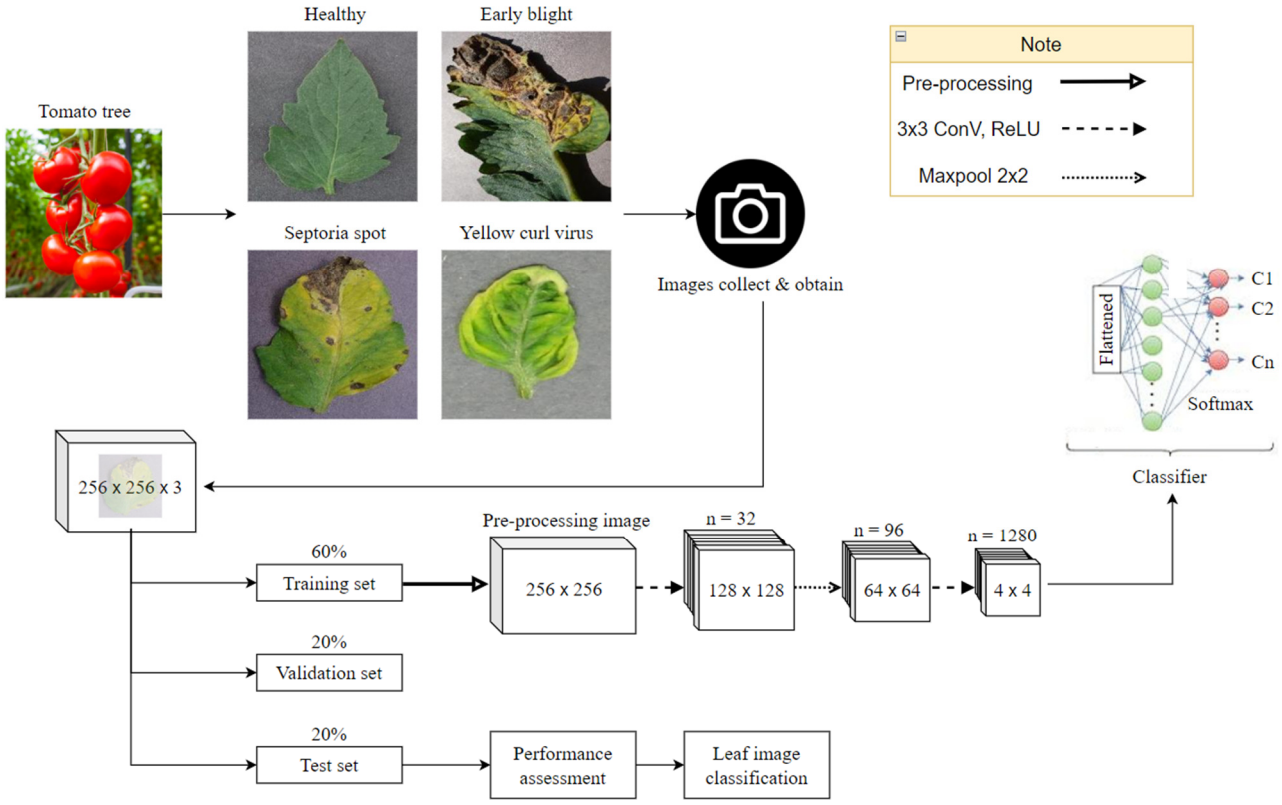


Fig. 3 Transfer learning model to classify diseases

When working with sparse data, dilated convolution can be advantageous because it allows for more expansive receptive fields without introducing extra parameters. In addition, dilated convolution can reduce overfitting in a neural network by reducing the number of parameters that need to be learned. However, dilated convolution can be slower to train than traditional convolution because it requires more computation. Furthermore, dilated convolution may be less effective for small kernel sizes because it decreases the number of parameters learned in a given layer.

The scenario of dilated convolution in the 1D field is as:

$$m[i]=\sum_{h=1}^{h} x[i+r.h]w[h]=y \tag{1}$$

For explanation, in every location, the output is y. Moreover, $x[i]$ is the input signal where x is also referred to as a feature map. Besides, $w[h]$ spatial dimensions were filtered with the length h, r corresponding to the dilation rate with which the authors sample the input signal $x[i]$. In the standard convolution, $r = 1$. The dilated convolution rate is always bigger than 1. An intuitive and straightforward method to comprehend dilated convolution is that push $(r - 1)$ zeros between every two consecutive filters in the standard convolution.

In a standard convolution, the kernel or filter size is $n \times m$, then in the resulting dilated convolution, the filter or kernel size is $n_d \times n_d$ where the value can be found by empirical estimation $n_d = n + (n - 1) \times (r - 1)$. One of the main reasons to use this method is that 1-dilated convolutions allow for exponentially expanding receptive fields without sacrificing coverage or resolution. Dilated convolutions can be used to adjust the effective receptive field of a convolutional layer without changing the kernel size. On the other hand, modify the spacing between the filter's sampling positions within the input. By introducing gaps or skips between the filter elements, dilated convolutions increase the effective receptive field without changing the kernel size.

Fig. 4 depicts the architecture, the first of the pre-trained MobileNet model in a workflow for disease recognition on the tomato tree. MobileNet was designed to provide a small, low-latency, computationally sound model for embedded mobile vision applications. Convolutional operations in MobileNet are classified into three types: standard convolution, pointwise

convolution, and depthwise convolution. To implement the dilated convolution. The authors use five depthwise layers, each with its stride rate (2,2). The first two depthwise layers have a dilation rate of (1,1); however, the third and fourth layers have a dilation rate of (2,1) or (2,2). Furthermore, the authors concatenate three parallel depthwise 2D convolution layers with dilation rates of 4, 8, and 16 for the final depthwise layer. The process continues concatenating these three depthwise convolutional 2D layers to create the fifth depthwise convolutional 2D layer. Initially, every depthwise layer of MobileNet had a dilation rate = 1, but implementing different dilation rates in distinct depthwise layers of MobileNet architecture is novel. Now it is time to start proposing this approach.

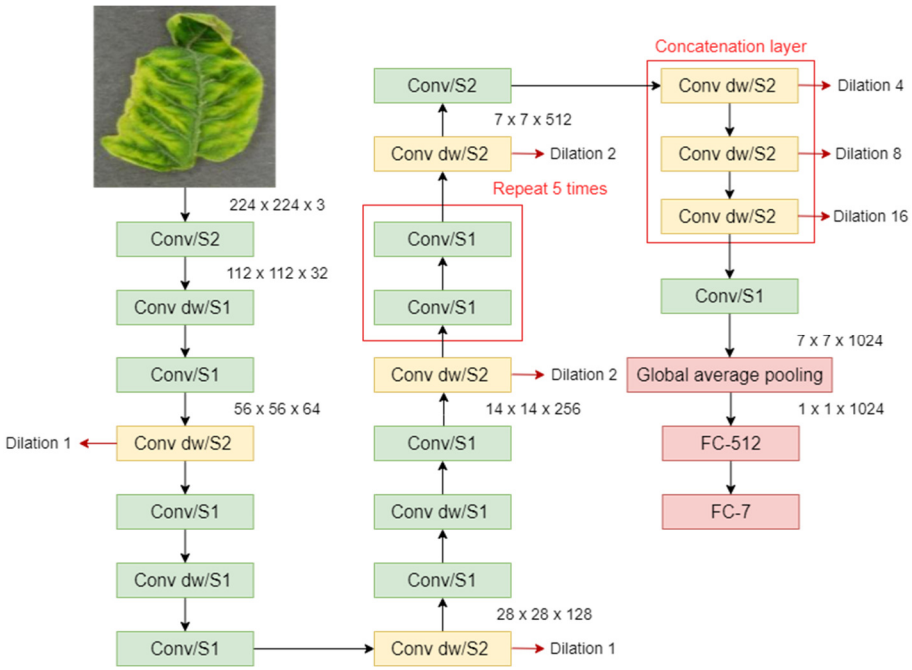


Fig. 4 An illustration of MobileNet architecture to identify and classify diseases

Transformational learning applies past model knowledge to the current situation. The authors apply transformational learning when the provided dataset does not contain enough data to train a full-scale model from the start. Instead, previously trained MobileNet model parameters are utilized in the training procedure. As a result, transfer learning uses the model's existing classes rather than retraining it from the start, boosting its accuracy. The MobileNet model is now enhanced with a new fully connected layer and an output layer with a softmax classification function for diseases in tomato leaf classification.

Fine-tuning: The process continues to refine after using transfer learning, and the results will improve. Fine-tuning uses a trained network model to perform a comparable task for a specific task. Usually, the model's initial layers are frozen (i.e., default and unchanged). The weights of these classes are not changed during training. These layers can perform low-level information extraction; This skill is acquired through prior training. The solution is to continue to tweak the hyperparameters during the fine-tuning phase to help the model achieve the highest accuracy while avoiding overfitting or underfitting.

In this fine-tuning section, the authors have fine-tuned the following hyperparameters:

(i) Dropout: Dropout is a regularization technique used in DL to prevent a model from overfitting. During training, a specific percentage of the nodes in a neural network are randomly dropped out (recommend from 0.2 to 0.5).

(ii) Learning rate: The learning rate is a hyperparameter that controls how quickly the model's parameters are updated during training iterations. It is an essential hyperparameter since it considerably impacts model performance (Should be from 0.00001 to 0.0001).

(iii) Hidden layer: A hidden neural network layer is neurons not directly connected to the input or output layers. Because its values are not directly visible in the input or output data, it is referred to as a "hidden" layer. (The study evaluates the number of units in the hidden layer from 1024 down to 512, 256, 128, etc.)

(iv) Batch size: The batch size is the number of samples used to update the model's parameters during each round of training. It is a crucial hyperparameter that can affect the training speed and the model's performance. (It should be 8, 16, 32, or 64).

(v) Epoch: An epoch is a complete pass of the entire training dataset during the training process. It is a critical hyperparameter that can affect model correctness and training time. (Epochs can be 10 times or 20, or 30 times depending on how long the training is).

The following hyperparameters are used during tuning: First, the number of training intervals is evaluated with a value of 20-25-30 to determine the appropriate time threshold. Next, the batch number size is tested with values ranging from 16-32-64, with the default batch number being 32. Finally, the hidden layer is tested with [1024, 512, 256], [512, 128], and the learning rate is tested with [0.001], [0.0001], and [0.00001]. Each model training implementation will only change one of the hyperparameters described above to avoid overfitting and make them easy to manipulate. Members of the group are working to refine those hyperparameters further. Adjusting the batch size is to use how much data each time has been calculated and update the coefficient. The larger the batch size, the more vectorization the application can be calculated.

When fine-tuning the number of epochs, it only performs more or less traversal in a single train. In case the result is underfitting, which leads to the need to increase the complexity of the model (increase the number of hidden layers and the number of nodes of that layer) along with increasing the number of epochs. If the result is overfitting, the solution is to insert more datasets, perform data augmentation, and refine the dropout coefficient to remove a few percent.

4. Experiments

4.1. Environmental settings

The experiments were implemented on Google Colab with 12 GB of RAM and an NVIDIA K80 GPU with 12 GB of GDDR5 VRAM. The proposed model was trained using 30 epochs and a batch size of 32. The specifications of the Google Colab are shown in Table 3.

Table 3 Google Colab proposed specification

Parameters	Google Colab
CPU model name	Intel(R) Xeon(R)
CPU frequency	2.30 GHz
No. CPU Cores	2
CPU family	Haswell
Available RAM	12 GB
Disk space	25 GB

4.2. Overall evaluation

This study divided the data set into three sets for model evaluation, using a ratio of 60-20-20 for training, validation, and testing, respectively, with accuracy, loss, and F1-score metrics. The authors show that the validation set aims to select the best model to apply to the test set. First, the authors compare the proposed model - MobileNet, with well-known CNNs to compare the results with MobileNet, VGG16, VGG19, DenseNet, and Xception. The research conducted three experiments in this study. Scenario 1 used data from two categories: 1 of tomato leaf diseases (examples below are an early blight, yellow curl, or Septoria with healthy leaves). The proposed model was then trained, evaluated, and tested. Next, compare the results of the proposed model with the results of other MobileNet and CNN models. Then, continue to alternately compare healthy leaves with each other disease, including yellow curl and Septoria. Scenario 2 was performed similarly but with three data types: early blight, Septoria, and yellow curl, and the last thing is to compare the proposed model with the most advanced methods. Finally, the data set is trained using the MobileNet model in two stages: transfer learning and fine-tuning.

Scenario 1 includes the following sequences: First, the study collects data from a pair of types present in the dataset, early blight/Septoria/yellow curl and healthy leaves. Then, the training is performed, analyzed, and tested with the provided model. Finally, the study compares the results of the proposed model to those of different CNN designs. The goal of presenting Scenario 1 aims to evaluate how the proposed model differentiates between healthy leaves and those displaying indications of early blight, Septoria, or yellow curl. The accuracy, F1-score, and confusion matrix metrics are used to evaluate the test results.

Scenario 2 includes the following sequences: The scenario collects four disease types of early blight, Septoria, yellow curl, and healthy leaf classes for multi-class classification tasks. First, the proposed model is modified with the number of outputs to be trained, evaluated, and tested on multi-class classification tasks. Then, the proposed model's results are compared to other CNN designs. The aim of performing this scenario is to predict and classify classes against each other accurately. Therefore, the authors want to test the classification capacity of the proposed model between early blight, Septoria, yellow curl, and healthy leaf.

4.3. Scenario 1 – Discriminate each disease with healthy

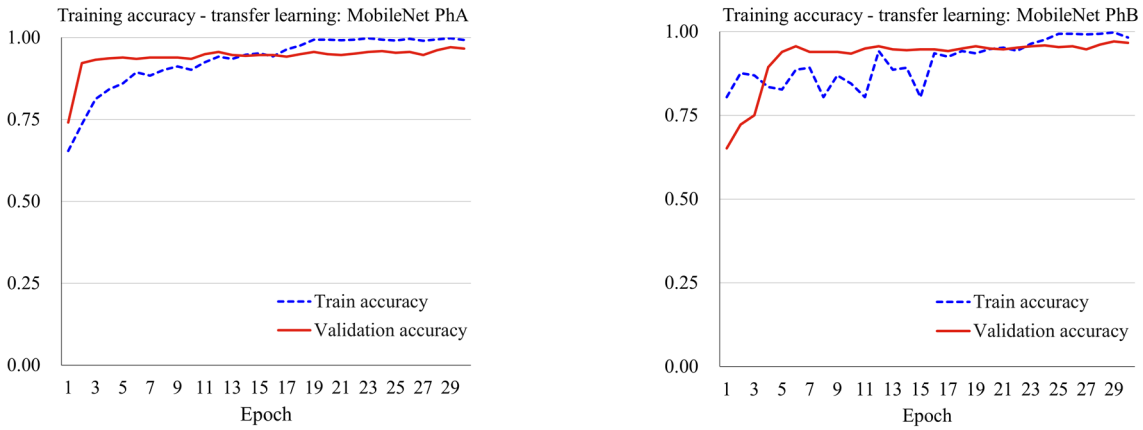
(1) Discriminate early blight and healthy leaves

In this experiment, the study used the same hyperparameters used for all models, including an epoch value of 30, a learning rate value of 0.0001, a batch size of 32, and a weight decay of 0.01. The hidden layer includes 1024 neurons. In addition, four layers, including platten, batch normalization, dense classes with activation Gaussian error linear unit (GELU), and softmax are added. Scenario 1, including training and testing results, is shown in Table 4.

Table 4 The classification results of early blight and healthy leaves

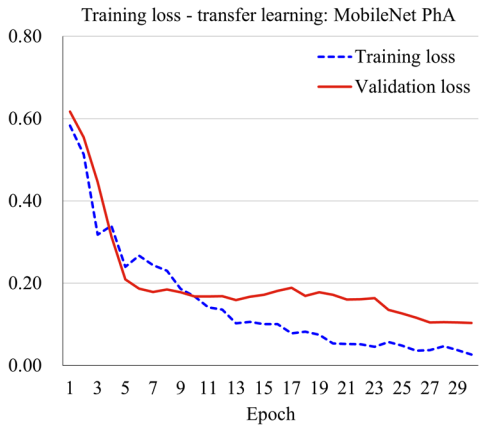
	Training set		Validation set		Test set	
Model	Accuracy	Loss	Accuracy	Loss	Accuracy	F1-score
Proposed model	0.9951	0.0434	1.0000	0.0097	0.9868	0.9873
DenseNet	0.9570	0.0945	0.9516	0.3832	0.9517	0.9516
VGG16	0.9282	0.0227	0.9234	0.1175	0.9287	0.9282
VGG19	0.9145	0.0684	0.9154	0.1542	0.9133	0.9182
Xception	0.8661	0.0807	0.8934	0.1023	0.8662	0.8660
MobileNet	0.7885	0.0354	0.8124	0.0416	0.8647	0.8434

The training and validation curves for accuracy and loss are shown in Fig. 5. Transfer-learning machine-learning techniques provide training accuracy of up to 95 percent, compared to 98 percent for fine-tuning techniques. Following training, it is evaluated with the previously indicated split test set, yielding the confusion matrix model shown in Fig. 6.

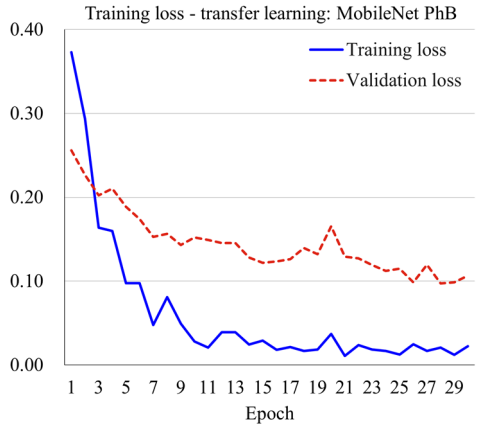


(a) The pre-trained MobileNet without fine-tuning accuracy (b) The improved MobileNet accuracy

Fig. 5 The classification results of Early Blight and Healthy Leaves: Training and validation performance in accuracy and loss during epochs



(c) The pre-trained MobileNet without fine-tuning loss



(d) The improved MobileNet loss

Fig. 5 The classification results of Early Blight and Healthy Leaves: Training and validation performance in accuracy and loss during epochs (continued)

From the illustration of figures and confusion matrix in Scenario 1 (discriminate early blight), the result showed that the training process of the proposed model achieves a promising result. Moreover, the predictive results of the two types in the test kit achieved an accuracy of 98.57% for early blight and an accuracy of 99.5% for healthy leaves. These results prove that the proposed model can classify early blight and healthy leaves. Both results achieved more than 90% accuracy.

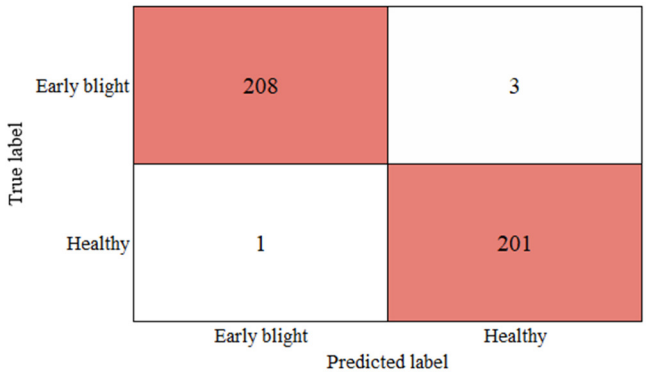


Fig. 6 Confusion matrix of Scenario 1 (discriminate early blight)

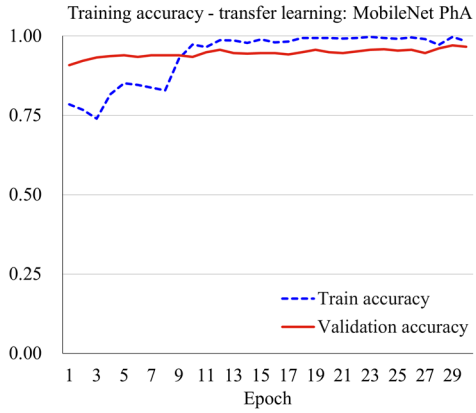
(2) Discriminate yellow curl and healthy leaves

In this experiment, the authors used the same hyperparameters used for all models in it, including epoch's value is 30, learning rates value is [0.0001], batch sizes were also tested at [32], weight decay within the at [0.01] and hidden layers tested value which are [1024]. In addition, four additional layers are included, including platten, batch normalization, dense classes with activation GELU, and software. The training and testing results are shown in Table 5.

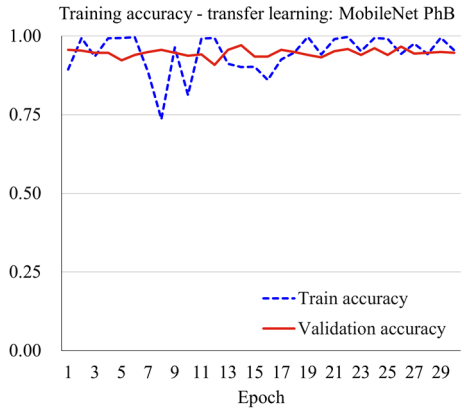
Table 5 The classification results of yellow curl and healthy leaves

	Training set		Validation set		Test set	
Model	Accuracy	Loss	Accuracy	Loss	Accuracy	F1-score
Proposed model	0.9857	0.0024	0.9905	0.0010	0.9857	0.9905
DenseNet	0.9570	0.0945	0.9516	0.0383	0.9517	0.9516
VGG16	0.9282	0.0227	0.9234	0.1175	0.9287	0.9282
VGG19	0.9145	0.0684	0.9154	0.1542	0.9133	0.9182
Xception	0.8661	0.0807	0.8934	0.1023	0.8662	0.8660
MobileNet	1.0000	0.0101	0.7885	0.0416	0.8347	0.8654

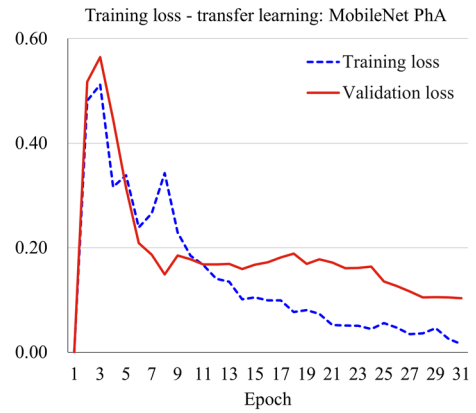
The training and validation curves for accuracy and loss are shown in Fig. 7. Transfer-learning machine-learning techniques provide training accuracy of up to 98 percent, compared to 99 percent for fine-tuning techniques. Following training, it is evaluated with the previously indicated split test set, yielding the confusion matrix model shown in Fig. 8.



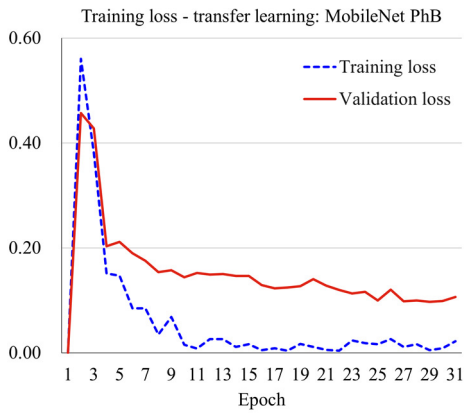
(a) The pre-trained MobileNet without fine-tuning accuracy



(b) The improved fine-tuned MobileNet accuracy



(c) The pre-trained MobileNet without fine-tuning loss



(d) The improved MobileNet loss

Fig. 7 The classification results of yellow curl and healthy leaves in accuracy and loss during epochs

From the illustration of graphs and confusion matrix in Scenario 1 (discriminate yellow curl virus), the training process of the proposed model achieves a promising result. Furthermore, the predictive results of the two types in the test set achieved an accuracy of 99.52% for the yellow curl virus and an accuracy of 99.05% for healthy leaves. These results prove that the proposed model can classify early blight and healthy leaves. Both results achieved more than 90% accuracy.

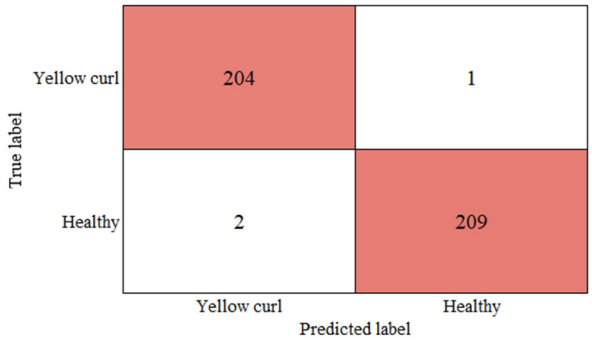


Fig. 8 Confusion matrix of Scenario 1 (discriminate yellow curl virus)

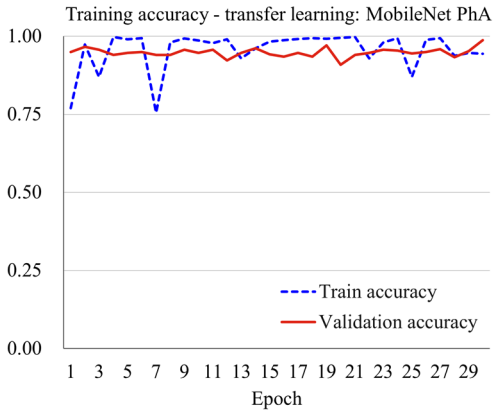
(3) Discriminate Septoria and healthy leaves

In this experiment, the authors used the same hyperparameters used for all models in it, including epoch's value is 30, learning rates value is [0.0001], batch sizes were also tested at [32], weight decay within the at [0.01] and hidden layers tested value which are [1024]. In addition, four additional layers are included, including platten, batch normalization, dense classes with activation GELU, and software. The training and testing results are shown in Table 6.

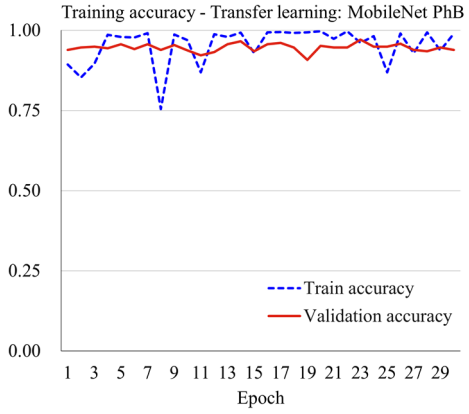
The training and validation curves for accuracy and loss are shown in Fig. 9. Transfer-learning machine-learning techniques provide training accuracy of up to 95 percent, compared to 97 percent for fine-tuning techniques. Following training, it is evaluated with the previously indicated split test set, yielding the confusion matrix model shown in Fig. 10.

Table 6 The classification results of Septoria and healthy leaves

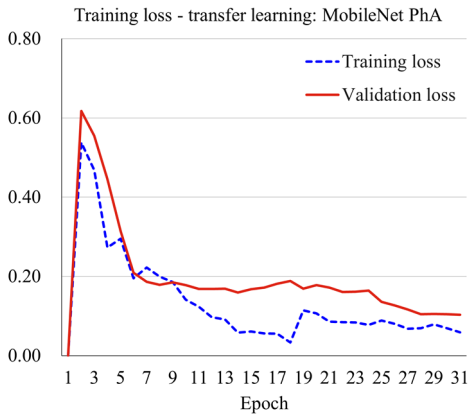
	Training set		Validation set		Test set	
Model	Accuracy	Loss	Accuracy	Loss	Accuracy	F1-score
Proposed model	0.9422	0.0363	0.9443	0.0148	0.9818	0.9905
DenseNet	0.9372	0.0945	0.9383	0.3632	0.9335	0.9316
VGG16	0.9032	0.0227	0.9021	0.5175	0.9076	0.9057
VGG19	0.9279	0.0323	0.9164	0.4789	0.9265	0.9188
Xception	0.8366	0.0607	0.8934	0.1223	0.8536	0.8343
MobileNet	0.9452	0.011	0.9023	0.1024	0.9507	0.9576



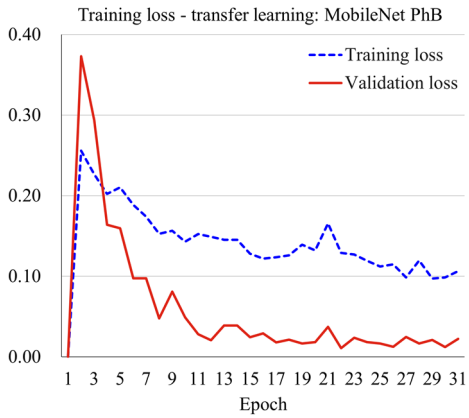
(a) The pre-trained MobileNet without fine-tuning accuracy



(b) The improved MobileNet accuracy



(c) The pre-trained MobileNet without fine-tuning loss



(d) The improved MobileNet loss

Fig. 9 The classification results of Septoria and healthy leaves in accuracy and loss during epochs

From the illustration of graphs and confusion matrix in Scenario 1 (discriminate Septoria), the training process of the proposed model achieves a promising result that archived more than 90% accuracy. In detail, the predictive results of the two types in the test set achieved an accuracy of 99.01% for the yellow curl virus and an accuracy of 99.54% on healthy leaves. These results prove that the proposed model can classify early blight and healthy leaves. Furthermore, both results achieved more than 90% accuracy.

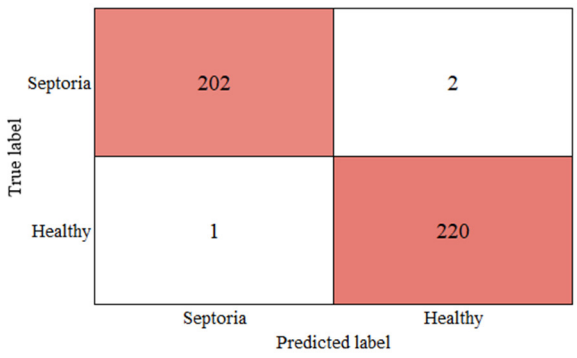


Fig. 10 Confusion matrix of Scenario 1 (discriminate Septoria)

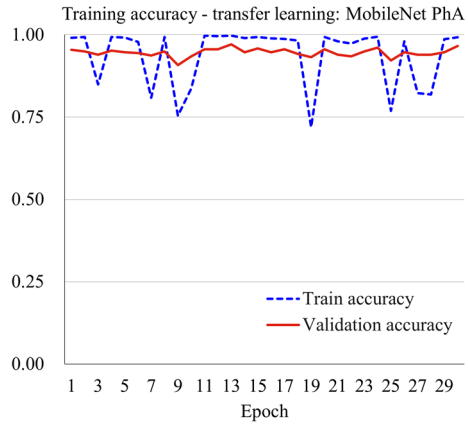
4.4. Scenario 2 – classification of 4 classes: early blight, Septoria, yellow curl, and healthy

This scenario retains the same hyperparameters as in Scenario 1, including adding and deleting dense layers and performing mass normalization. The study trained the model in transfer learning and fine-tuning to compare the MobileNet results to those of other CNN models. The hyperparameters utilized in the models are the same for both stages and models from the same group.

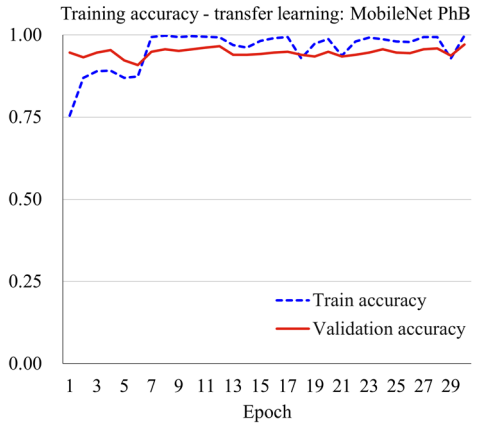
The outcomes of this scenario are presented in Table 7, Fig. 11 and Fig. 12. The results show that the MobileNet model with the proposed architecture beat the other CNN architectures by around 95%. The accuracy and loss during training in Scenario 2 are displayed in Fig. 11. Fig. 12 reveals the confusion matrix for the third case.

Table 7 Classification results of 4 classes: early blight, Septoria, yellow curl, and healthy

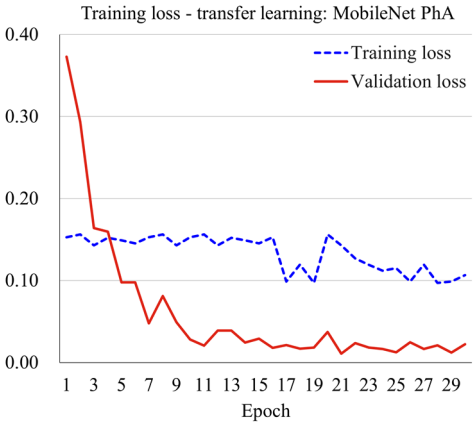
	Training set		Validation set		Test set	
Model	Accuracy	Loss	Accuracy	Loss	Accuracy	F1-score
Proposed model	0.9348	0.0363	0.9443	0.0148	0.9717	0.9442
DenseNet	0.9353	0.0945	0.9383	0.3632	0.9335	0.9316
VGG16	0.9038	0.0227	0.9021	0.5175	0.9076	0.9057
VGG19	0.9279	0.0323	0.9164	0.4789	0.9265	0.9188
Xception	0.8376	0.0607	0.8934	0.1223	0.8536	0.8343
MobileNet	0.9464	0.011	0.9023	0.4404	0.7907	0.7176



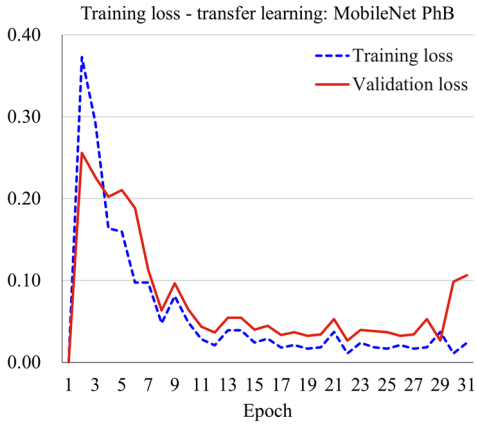
(a) The pre-trained MobileNet without fine-tuning accuracy



(b) The improved MobileNet accuracy



(c) The pre-trained MobileNet without fine-tuning loss



(d) The improved MobileNet loss

Fig. 11 Performance of accuracy and loss training in Scenario 2

In detail, the predictive results of the four types in the test set achieved an accuracy of 95.19% for early blight, 96.78% for yellow curl virus, 97.41% for Septoria, and the accuracy of 97.5% for healthy leaves. These results prove that the proposed model can classify early blight and healthy leaves. Both results achieved more than 90% accuracy.

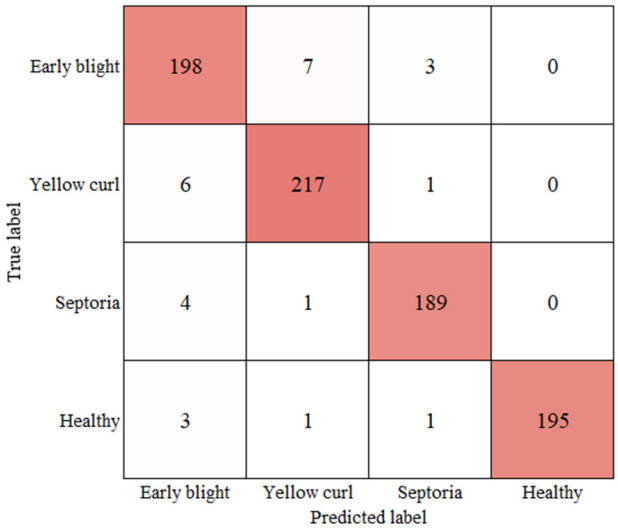


Fig. 12 Confusion matrix of the proposed method in Scenario 2

4.5. Discussion

Through the two scenarios that were run, the results revealed that every prediction case had more accurate results than 90%; this was the level of accuracy that the author desired and was surpassed by the actual outcomes. In Scenario 1, the results of discriminating between early blight and healthy cases are close to 100%. There is almost no confusion in the machine learning process. Furthermore, the classification between yellow curl virus and healthy cases and the tasks of discriminating between Septoria and healthy samples also exhibits similar performance. Because two diseases, including Septoria and yellow curl virus, are more challenging to recognize than early blight, the results are somewhat not as high as early blight.

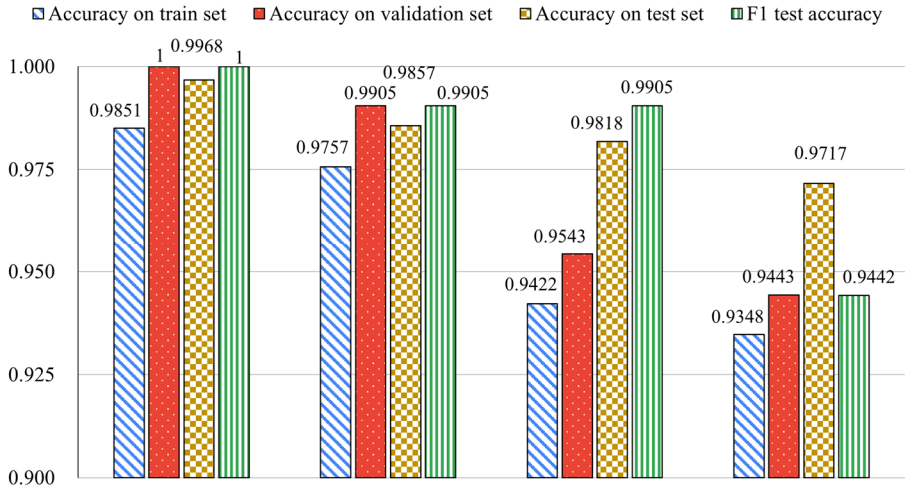


Fig. 13 Transfer learning model results on different scenario experiments with various metrics

In Scenario 2, the implementation of training three diseases and healthy cases can be more complicated, and the implementation time might also be longer. This is the promising result of the proposed model in distinguishing some specific diseases on tomato leaves. All the accuracy, validation accuracy, test accuracy, and F1-score calculated of the proposed model in two scenarios are summarized in Fig. 13.

The experimental results show promising results when training, testing, and evaluating the proposed model in two scenarios suitable for plant disease detection. As observed in Scenario 2, the proposed model achieves 97.17% accuracy when predicting three classifications, and the F1-score accuracy is 0.9442 on the testing set. In addition, the accuracy score achieves higher when there is sufficient accuracy in one classification, with more than 99.68% accuracy in the first scenario (early blight), more than 98.57% in the first (yellow curl), and more than 98.18% in the first scenario (Septoria).

The cumulative match curve (CMC) is also used to illustrate the comparison of the proposed model with other architectures in another view with the x−axis to a constant value of rank and the y−axis for the recognition rate. The curve getting up and then moving to the top of the right side is the best system. The CMC curve plays an essential part in evaluating. It judges the ranking capabilities of an identification system. The results shown in Fig. 14 show that the proposed method gives the best and most stable prediction results.

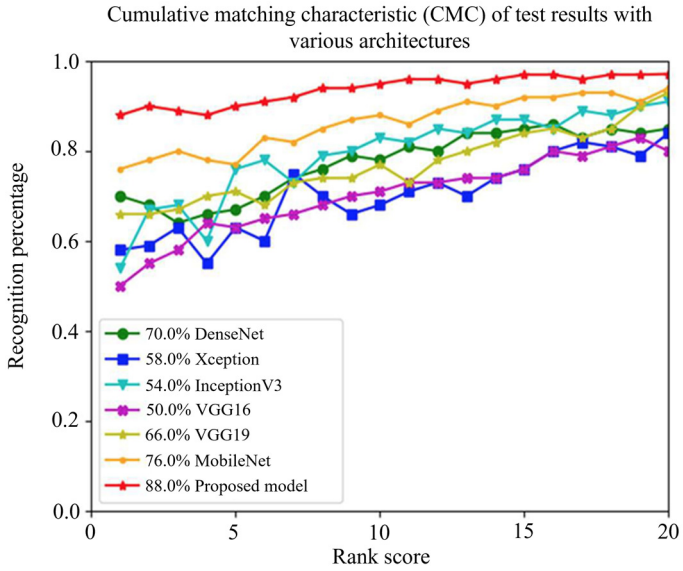


Fig. 14 The CMC in the architecture's comparison on the test set

Based on the experimental findings above, the proposed model is appropriate for diagnosing various illnesses on tomato leaves using color images compared to the previous study as presented in Table 8. The results show that the proposed model (improved MobileNet) outperforms most recently published efforts on identifying disease signals in tomato plants. Furthermore, based on the test results, the machine learning process on the CNN MobileNet model gives a positive and promising result.

Table 8 Comparative analysis of the proposed MobileNet model with state-of-the-art methods

References	Dataset	Model	Result accuracy result
Bhagwat et al. [3]	PlantVillage dataset	Support vector machine	91.5%
Durmus et al. [8]	PlantVillage dataset	Fine-tuned SqueezeNet	94.30%
		Fine-tuned AlexNet	95.65%
Adhikari et al. [10]	ImageNet (Colored images)	Insprired by AlexNet and YOLO	92.61%
Tm et al. [14]	PlantVillage dataset	Fine-tuned LeNet	94-95%
Coulibaly et al. [18]	ImageNet (RGB)	Fine-tuned VGG16	95.00%
Ashwinkumar et al. [19]	PlantVillage dataset	Fine-tuned MobileNet based with OMNCNN improved	98.7%
Kaya and Gursoy [20]	PlantVillage dataset	Fine-tuned DenseNet	98.17%
Proposed model	PlantVillage dataset	Fine-tuned MobileNet	97.17%

5. Conclusion

This study fine-tuned MobileNet to classify several common tomato leaf diseases. There are a few of the significant repercussions of the influence of bacteria, viruses, fungi, etc., that can damage the quality and productivity of tomato plants during the season. The results of the training model have significantly improved in comparison to previous studies. In addition, the authors may determine which combination strategy provides the highest potential performance for the situation. Positive results were obtained with the revised model MobileNet, achieving a result of 98%. The proposed model performs better in both circumstances than the other models.

In future research, the work is expected to adapt and apply different data preparation strategies to improve the prediction model outputs further. Furthermore, research is being conducted to evaluate various preprocessing approaches to tomato leaf pictures. By enhancing the data, the data-enhancement strategy not only improves the model's performance but also expands the range of diseases that the model can predict. The authors realize that this study still has shortcomings, and at the same time, some points have yet to be exploited, in particular, the limitation is in the diversity of tomato leaves diseases dataset because the environment provided is not enough to perform training on many types of tomato. In the future, the research plan will continue to be carried out. This model can be applied in real-time and used on mobile devices.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] S. M. Hassan, A. K. Maji, M. Jasiński, Z. Leonowicz, and E. Jasińska, "Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach," Electronics, vol. 10, no. 12, article no. 1388, June 2021.

[2] E. Moriones and J. Navas-Castillo, "Tomato Yellow Leaf Curl Virus, an Emerging Virus Complex Causing Epidemics Worldwide," Virus Research, vol. 71, no. 1-2, pp. 123-134, November 2000.

[3] R. Bhagwat and Y. Dandawate, "A Review on Advances in Automated Plant Disease Detection," International Journal of Engineering and Technology Innovation, vol. 11, no. 4, pp. 251-264, September 2021.

[4] A. Fazari, O. J. Pellicer-Valero, J. Gómez-Sanchıs, B. Bernardi, S. Cubero, S. Benalia, et al., "Application of Deep Convolutional Neural Networks for the Detection of Anthracnose in Olives Using VIS/NIR Hyperspectral Images," Computers and Electronics in Agriculture, vol. 187, article no. 106252, August 2021.

[5] J. Lu, L. Tan, and H. Jiang, "Review on Convolutional Neural Network (CNN) Applied to Plant Leaf Disease Classification," Agriculture, vol. 11, no. 8, article no. 707, August 2021.

[6] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep Learning for Tomato Diseases: Classification and Symptoms Visualization," Applied Artificial Intelligence, vol. 31, no. 4, pp. 299-315, 2017.

[7] M. Alruwaili, M. H. Siddiqi, A. Khan, M. Azad, A. Khan, and S. Alanazi, "RTF-RCNN: An Architecture for Real-Time Tomato Plant Leaf Diseases Detection in Video Streaming Using Faster-RCNN," Bioengineering, vol. 9, no. 10, article no. 565, October 2022.

[8] H. Durmuş, E. O. Gunes, and M. Kırcı, "Disease Detection on the Leaves of the Tomato Plants by Using Deep Learning," 6th International Conference on Agro-Geoinformatics, pp. 1-5, August 2017.

[9] A. Saeed, A. A. Abdel-Aziz, A. Mossad, M. A. Abdelhamid, A. Y. Alkhaled, and M. Mayhoub, "Smart Detection of Tomato Leaf Diseases Using Transfer Learning-Based Convolutional Neural Networks," Agriculture, vol. 13, no. 1, article no. 139, January 2023.

[10] S. Adhikari, B. Shrestha, B. Baiju, and S. K. KC, "Tomato Plant Diseases Detection System Using Image Processing," 1st KEC Conference Proceedings, vol. 1, pp. 81-86, September 2018.

[11] S. Zhang, H. Zhou, and L. Zhang, "Recent Machine Learning Progress in Image Analysis and Understanding," Advances in Multimedia, vol. 2018, article no. 1685890, 2018.

[12] S. Ishak, M. H. F. Rahiman, S. N. A. M. Kanafiah, and H. Saad, "Leaf Disease Classification Using Artificial Neural Network," Jurnal Teknologi, vol. 77, no. 17, pp. 109-114, December 2015.

[13] Y. Wu, L. Xu, and E. D. Goodman, "Tomato Leaf Disease Identification and Detection Based on Deep Convolutional Neural Network," Intelligent Automation & Soft Computing, vol. 28, no. 2, pp. 561-576, 2021.

[14] P. Tm, A. Pranathi, K. SaiAshritha, N. B. Chittaragi, and S. G. Koolagudi, "Tomato Leaf Disease Detection Using Convolutional Neural Networks," Eleventh International Conference on Contemporary Computing (IC3), pp. 1-5, August 2018.

[15] N. K. Trivedi, V. Gautam, A. Anand, H. M. Aljahdali, S. G. Villar, D. Anand, et al., "Early Detection and Classification of Tomato Leaf Disease Using High-Performance Deep Neural Network," Sensors, vol. 21, no. 23, article no. 7987, December 2021.

[16] G. Suryanarayana, K. Chandran, O. I. Khalaf, Y. Alotaibi, A. Alsufyani, and S. A. Alghamdi, "Accurate Magnetic Resonance Image Super-Resolution Using Deep Networks and Gaussian Filtering in the Stationary Wavelet Domain," IEEE Access, vol. 9, pp. 71406-71417, 2021.

[17] R. Bhagwat and Y. Dandawate, "A Framework for Crop Disease Detection Using Feature Fusion Method," International Journal of Engineering and Technology Innovation, vol. 11, no. 3, pp. 216-228, June 2021.

[18] S. Coulibaly, B. Kamsu-Foguem, D. Kamissoko, and D. Traore, "Deep Neural Networks with Transfer Learning in Millet Crop Images," Computers in Industry, vol. 108, pp. 115-120, June 2019.

[19] S. Ashwinkumar, S. Rajagopal, V. Manimaran, and B. Jegajothi, "Automated Plant Leaf Disease Detection and Classification Using Optimal MobileNet Based Convolutional Neural Networks," Materials Today: Proceedings, vol. 51, no. 1, pp. 480-487, 2021.

[20] Y. Kaya and E. Gürsoy, "A Novel Multi-Head CNN Design to Identify Plant Diseases Using the Fusion of RGB Images," Ecological Informatics, vol. 75, article no. 101998, July 2023.

[21] P. S. Thakur, T. Sheorey, and A. Ojha, "VGG-ICNN: A Lightweight CNN Model for Crop Disease Identification," Multimedia Tools and Applications, vol. 82, no. 1, pp. 497-520, January 2023.

[22] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," https://doi.org/10.48550/arXiv.1409.1556, September 04, 2014.

[23] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," https://doi.org/10.48550/arXiv.1704.04861, April 17, 2017.