IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 25, 2022, accepted March 2, 2022, date of publication March 14, 2022, date of current version March 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3159678

End-to-End Deep Learning Model for Corn Leaf Disease Classification

HASSAN AMIN[1], ASHRAF DARWISH[1,3], (Member, IEEE), ABOUL ELLA HASSANIEN[2,3], AND MONA SOLIMAN[2,3]

[1]Faculty of Science, Helwan University, Cairo 11795, Egypt
[2]Faculty of Computers and Artificial Intelligence, Cairo University, Giza 12613, Egypt
[3]Scientific Research Group in Egypt (SRGE), Cairo 12613, Egypt

Corresponding author: Hassan Amin (hassanmamin994@gmail.com)

ABSTRACT Plant diseases compose a great threat to global food security. However, the rapid identification of plant diseases remains challenging and time-consuming. It requires experts to accurately identify if the plant is healthy or not and identify the type of infection. Deep learning techniques have recently been used to identify and diagnose diseased plants from digital images to help automate plant disease diagnosis and help non-experts identify diseased plants. While many deep learning applications have been used to identify diseased plants and aims to increase the detection rate, the limitation of the large parameter size in the models persist. In this paper, an end-to-end deep learning model is developed to identify healthy and unhealthy corn plant leaves while taking into consideration the number of parameters of the model. The proposed model utilizes two pre-trained convolutional neural networks (CNNs), EfficientNetB0, and DenseNet121, to extract deep features from the corn plant images.The deep features extracted from each CNN are then fused using the concatenation technique to produce a more complex feature set from which the model can learn better about the dataset. In this paper, data augmentation techniques were used to add variations to the images in the dataset used to train the model, increasing the variety and number of the images and enabling the model to learn more complex cases of the data. The obtained result of this work is compared with other pre-trained CNN models, namely ResNet152 and InceptionV3, which have a larger number of parameters than the proposed model and require more processing power. The proposed model is able to achieve a classification accuracy of 98.56% which shows the superiority of the proposed model over ResNet152 and InceptionV3 that achieved a classification accuracy of 98.37% and 96.26% respectively.

INDEX TERMS Colutional neural networks, deep learning, deep features, feature fusion, plant disease.

I. INTRODUCTION

Food security is threatened by many factors, including the decline in pollinators [1], climate change [2], plant diseases [3], and others. Plant diseases compose a threat to global food security and smallholder farmers whose livelihoods depend mainly on agriculture and healthy crops. In developing countries, smallholder farmers produce more than 80% of the agricultural production [4], and reports indicate that more than fifty percent loss in crops due to pests and diseases [5]. The world population is expected to grow to more than 9.7 billion by 2050 [6], making food security a major concern in the upcoming years. Hence, rapid and accurate methods of identifying plant diseases are needed to do the appropriate measures.

Advancements in artificial intelligence and image processing techniques present an opportunity to extend research in agriculture. Deep learning, a class of machine learning techniques, is currently an active research area and has been successfully applied to many fields. Deep learning utilizes deep neural networks to perform feature extraction, pattern analysis, and data classification [7]. Additionally, it has been applied successfully in many sectors such as agriculture, business, automotive industry, communications and networking etc. using object detection and image classification techniques [8]–[11]. In the agriculture sector, traditional methods of detecting plant diseases manually require experts to perform visual inspection and later more in-depth detection in labs which is time-consuming and not

The associate editor coordinating the review of this manuscript and approving it for publication was Paulo Mendes.

always available for smallholder farmers. Hence, researchers explored the application of automated and smart disease detection systems by using artificial intelligence, machine learning, and deep learning techniques in developing such systems. The main goal of deep learning algorithms is to extract features from images and utilize these features to perform either classification or regression, depending on the goal. While each deep learning algorithm can extract features in a specific manner, the fusion of the extracted features of multiple deep learning algorithms can yield better results as the classifier will have more descriptive features to learn from. In this work, a new classification model is proposed to accurately classify corn leaves infected with gray leaf spot, common rust, northern leaf blight, and healthy leaves from digital images. This work uses 2 pre-trained convolutional neural networks (CNNs), namely EfficientNetB0 CNN and DenseNet121 CNN, with reasonable amounts of parameters and uses feature fusion techniques to integrate the two models' predictive power to build an end-to-end classification model. The main contributions of our work can be summarized as follows:

- Apply feature fusion between features extracted from two different CNNs in an end-to-end learning model.
- Increased classification accuracy with a reasonable number of parameters.
- Propose a model for detecting and identifying corn plant diseases.
- Extensive experiments and comparison between the proposed classification model and some of the state-of-the-art models and other authored models that used the same dataset.

The rest of this paper is organized as follows. Section II discusses some related work in machine learning, deep learning, and their use in agriculture. Section III explains some concepts used in this research, whereas Section IV explains the proposed model in detail and the dataset used in the experiment. Section V discusses the proposed models' experimental results and provides a comparative analysis with other models. Finally, Section VI concludes the research results and proposes some future work.

II. RELATED WORK

Researchers have attempted several methods to classify and diagnose plant diseases and extract their features. Deep learning alongside image processing and traditional machine learning techniques have been extensively used in the agricultural field. This section concentrates on some of the previous work that uses deep learning techniques to classify corn leaves diseases from digital images, as this is the primary focus of this study.

In [8], the authors proposed a CNN algorithm for corn leaf disease recognition by using data augmentation for enhancing the training set and transfer learning technique to improve the accuracy of the CNN model. The optimized CNN showed an average accuracy of 97.6% on a subset of the PlantVillage dataset that contained four categories of corn leaves (corn gray leaf spot, corn common rust, corn northern leaf blight, and healthy leaves).

In [12], The authors assess the performance of three state-of-the-art convolutional neural network architectures, namely AlexNet, ResNet50, and SqueezeNet, to classify corn leaf diseases. The authors applied a Bayesian optimization algorithm to fine-tune the values of some of the hyperparameters in their experiment, namely the batch size, learning rate, and momentum values. Additionally, the authors applied data augmentation techniques in random rotation of the images by angles between 0°-360°, vertical and horizontal flips. The authors trained the CNNs on corn leaf images from the PlantVillage dataset using stratified k-fold cross-validation training. They divided the dataset into six stratified sub-datasets. One was kept as the test data of the stratified six-sub-datasets, and five were used for cross-validation. After reaching optimal values for the hyperparameters and training the CNNs, the authors evaluated the CNNs on the test subdataset in which all CNNs reached a similar classification accuracy of 97%.

In [13], the authors developed a CNN consisting of three convolution layers, three max-pooling layers, and two fully connected layers. The authors used a subset of the PlanVillage dataset containing corn leaves with three diseases: corn gray leaf spot, corn common rust, corn northern leaf blight, and a healthy class on which the developed model achieved a classification accuracy of 94%.

In [14], The authors proposed a dense-optimized CNN for classifying four corn leaves classes taken from the PlanVillage dataset. The network consisted of five dense blocks followed by a SoftMax classifier layer. After training the CNN, it achieved a classification accuracy of 98.06% for the four classes on which the experiment was performed.

In [15], The authors proposed a multi-context fusion network employed to concatenate contextual and visual information. The contextual information concerned the plant's environmental factors (e.g., humidity and temperature), which may cause or lead to specific diseases. The categorization of these factors improved the identification phase, where the network achieved a classification accuracy of 97.50%.

In [16], The authors proposed an automated crop disease recognition system using partial least squares (PLS) regression for feature selection from an extracted deep feature set. First, the authors employed a pre-trained VGG19 network to extract deep features from images of tomato, corn, and potato taken from the PlantVillage dataset. Afterward, a PLS parallel fusion method was used to combine the features extracted from the 6th and 7th layers of the VGG19 network. Next, the best features are selected using a PLS projection method. The most discriminate features are then finally plugged into the ensemble baggage tree classifier for final recognition, which achieved a classification accuracy of 90.1%. Table1 highlights the main distinctive characteristics for the reviewed work.

TABLE 1. Main characteristics for the reviewed related work.

Related Work	Methodology
[8]	Transfer learning (GoogleNet), data augmentation
[12]	Bayesian optimization, data augmentation
[13]	Developed a CNN
[14]	Developed CNN based on DenseNet
[15]	Multi-context fusion for contextual and visual features
[16]	PLS fusion applied on the 6th and 7th layer of VGG19

This work proposes a model that uses two different CNNs and uses feature fusion techniques to increase its predictive power as opposed to work in [8], [12], [13], and [14] that uses the features extracted by a single CNN. In comparison, this work utilizes features from two different CNNs. And while in [16], the authors performed feature fusion between two layers of the same network, this work performs fusion between layers from different networks.

III. CONCEPTUAL BACKGROUND

This section describes some main concepts and methods used in the proposed model.

A. CONVOLUTIONAL NEURAL NETWORKS

CNN is a class of deep learning algorithms widely applied in analyzing visual imagery [17]. It utilizes the convolution operation to learn different features from the images. Then, it maps the image into a smaller form that's easily processed without losing much of its features. The initial layers of a CNN learn low-level spatial features that usually correspond to edges, boundaries or simple properties of the objects. At the same time, the deep layers learn more high-level, complex features like complex shapes and objects orientations. A CNN can be divided into two main parts; feature extraction and classification, which can be visualized as shown in Fig. 1.

B. FEATURE EXTRACTION PROCESS

In this part, features are extracted from the image using a group of layers, usually consisting of several convolutional and activation layers followed by a pooling layer. This part takes input as the image's pixel values and produces a feature map sent to the classification part. Convolutional layers are composed of filters applied on the layer's input to learn low/mid/high-level features from the image, such as edges, patterns, and textures. Mathematically, an image can be represented by a tensor with the dimensions as in (1).

$$dim(image) = (n_h, n_w, n_c) \tag{1}$$

where n_h is the size of the height, n_w is the size of the width, and n_c is the number of channels.

A filter is a tensor that usually has an odd dimension and is applied to the input in a sliding-window manner by performing the convolutional operation of multiplying the filter values by the window on which they reside and summing it of the operation. The step size of the sliding window is usually referred to as the stride (s) for the operation. The dimension of the filter tensor is as in (2).

$$dim(filter) = (f, f, n_c) \tag{2}$$

where f is an odd dimension of the filter and n_c is the number of channels in the input. Sometimes, filters do not fit the input image, in which case we pad the image with zeros so that it fits. The amount by which we pad an image is usually denoted by p. The convolutional product between the input and the filter is a 2D matrix which is the sum of the element-wise multiplication of the filter on each window the filter was passed on.

A convolutional product of an input tensor I and a filter tensor K can then be mathematically represented as in (3).

$$Conv(I, K)_{x,y} = \sum_{i=1}^{n_h}\sum_{j=1}^{n_w}\sum_{k=1}^{n_c} K_{i,j,k} \times I_{x+i-1,y+j-1,k} \tag{3}$$

Activation layers are usually defined after the convolutional layers to add non-linearity to the output by performing a non-linear mapping using an activation function Ψ. Hence, for a convolutional layer followed by an activation layer L, we define the following:

- $I^{[L-1]}$: Input of the layer with size $(n_h^{[L-1]}, n_w^{[L-1]}, n_c^{[L-1]})$.
- $n_c^{[L-1]}$: Number of filters where each filter $K^{(n)}$ has the dimensions $(f^{[L]}, f^{[L]}, n_c^{[L-1]})$.
- b_n^L: Bias of the n^{th} convolution.
- $\Psi^{[L]}$: Activation function.
- $I^{[L]}$: Output of the layer with size $(n_h^{[L]}, n_w^{[L]}, n_c^{[L]})$.

and we have for every n between 1 and $n_c^{[L]}$, we have:

$$\begin{aligned} Conv(I^{[L-1]}, K^{(n)})_{x,y} = \sum_{i=1}^{n_h}\sum_{j=1}^{n_w}\sum_{k=1}^{n_c}(K_{i,j,k} \\ \times I_{x+i-1,y+j-1,k} + b_n^{[L]}) \end{aligned} \tag{4}$$

Thus, the output can be computed as in (5).

$$\begin{aligned} I^L = [\Psi^{[L]}(conv(I^{[L-1]}, K^{(1)}), \\ \dots, \Psi^{[L]}(conv(I^{[L-1]}, K^{(n_c^L)})]] \end{aligned} \tag{5}$$

The rectifier linear unit (ReLU) activation function [18] is widely used for its computational efficiency. It has fewer problems with vanishing gradients, where it performs a non-linear conversion to the output of the previous convolutional layer. Mathematically, ReLU can be represented as in (6).

$$F(x) = max(0, x) \tag{6}$$

In order to reduce the size of the feature maps, pooling layers can be added after the activation layer, which creates a down-sampled version of the feature maps and makes the feature maps more resistant to small translations in the input.

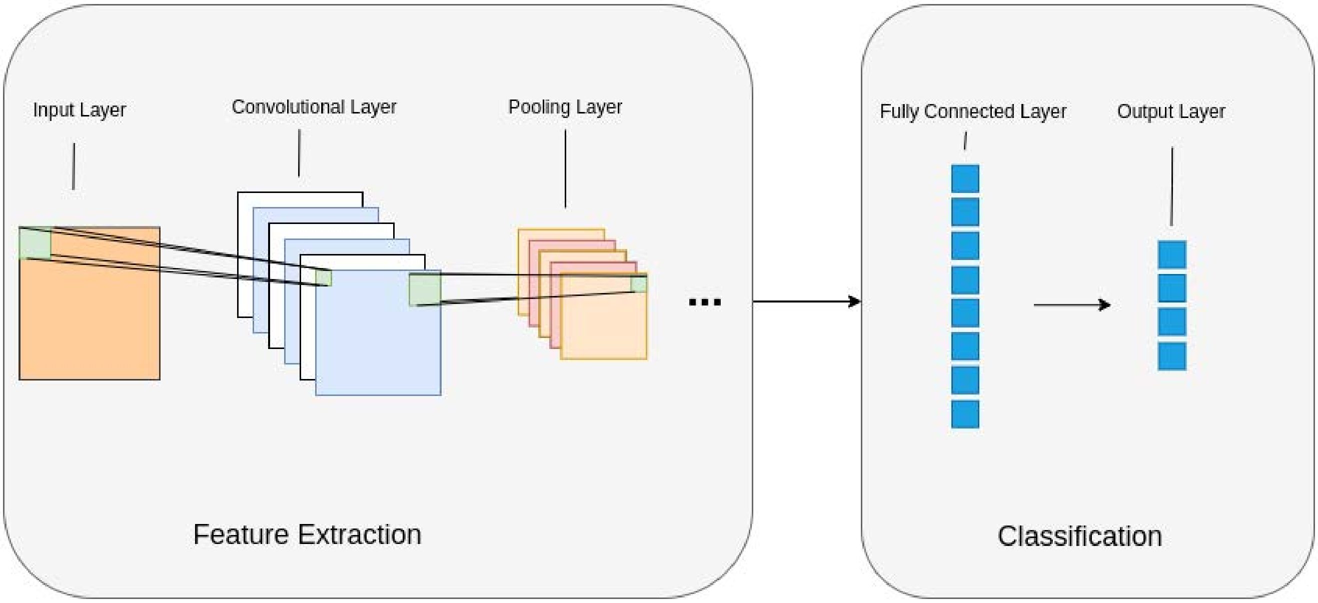


FIGURE 1. The convolutional neural network structure. It consists of two main building blocks: feature extraction and classification.

C. CLASSIFICATION PROCESS

In this part, the extracted feature maps are used to learn a mapping between the features and the required output classes that usually include fully connected layers. Fully connected layers have all the inputs from one layer connected to every activation unit of the next layer. They are used at the network's tail acting as a classifier where a layer is used having several neurons matching the number of the output classes. In multi-class classification problems, the SoftMax activation function is used to normalize the output into decimal values for the probabilities of the input being of a specific class. The class with the highest probability is then denoted as being the target predicted class for the inputs. For a classification problem with K classes, the SoftMax activation function can be formulated mathematically as in (7).

$$\sigma(Z)_i = \frac{e^{z_i}}{\sum_{i=1}^{k} e^{z_i}} \tag{7}$$

where i is a class number from 1 to K, x_i represents the i^{th} dimension output, and $\sigma(Z)_i$ is the probability of the input being the i^{th} class.

IV. TRANSFER LEARNING METHOD

Transfer learning is a technique in which a model is developed and trained on a specific task and then re-used as a starting point for another task [19], reducing the amount of time required to train such models that can take days, if not weeks, on modern hardware. There are two main approaches to doing transfer learning: (1) Develop a model approach. (2) Retrained model approach. Developing a model approach includes selecting a task similar to the task at hand with an abundance of data and developing a source model for this first task. When the model is fitted on the data and converges to acceptable performance, it is then used as a starting point for the second interest task. The pre-trained model approach starts by selecting an existing pre-trained model from available models. For example, the pre-trained weights on the ImageNet dataset [20] are commonly used in image-related tasks and use them as a baseline to train the model on another dataset at hand.

Using transfer learning usually starts by obtaining the model's pre-trained weights, removing the fully connected layers at the top of the model responsible for the classification part on the dataset on which the model was trained, and using the previous layers as feature extractors. Then by adding classification layers to the model and training it, it would learn the mappings from the feature extractor to the output classes on the new dataset. In this study, EfficientNetB0 and DenseNet121 pre-trained CNNs were used.

A. EfficientNetB0

EfficientNetB0 is the baseline network of the EfficientNet family [21] which is developed with considerations of having the highest accuracy and lowest memory and floating-point operations per second (FLOPS) requirements. The authors were able to surpass state-of-the-art accuracy with a much lower number of parameters of around 5.3 million in the baseline network. CNNs can be scaled by adjusting the network depth (number of layers), adjusting the network width (number of channels), or increasing the resolution of the input image by which the network gets to learn more fine-grained features from the image. However, a lot of manual tuning needs to be done to find a good fit for each technique, while using such techniques alone has dimensioning gains as scaling increases. The authors proposed a new compound model scaling technique from

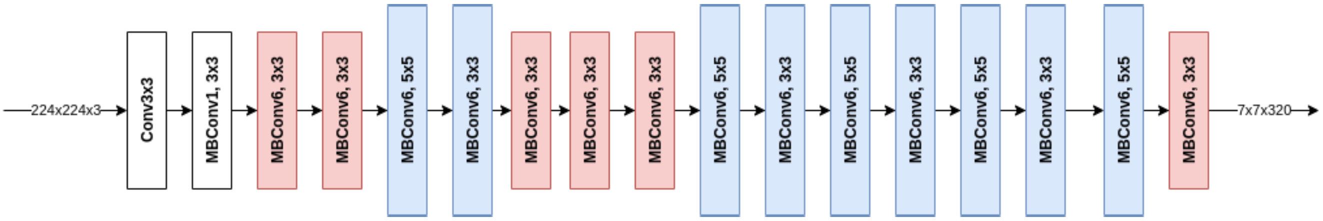


FIGURE 2. The architecture of EfficientNetB0 CNN. EfficientNetB0 uses slightly larger mobile inverted bottleneck convolution.

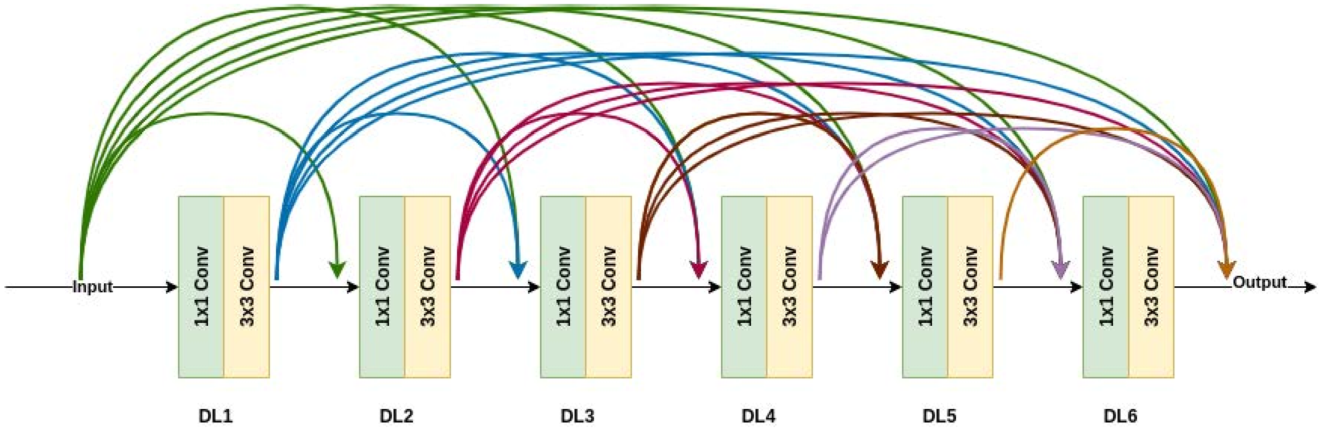


FIGURE 3. A dense block with six dense layers. Each dense layer consists of 1 × 1 convolutional layer followed by 3 × 3 convolutional layer.

which the EfficientNet family has arisen to resolve such issues.

The technique uses a compound coefficient Φ to uniformly scale network depth, width, and resolution in a principled manner as in (8).

$$d = \alpha, w = \beta, r = \gamma$$
$$s.t.\ \alpha \times \beta^2 \approx 2, \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \quad (8)$$

where d is the network depth, w is the network width, r is the resolution. α, β, γ are constants that can be computed using a small grid search. For EfficientNetB0, the authors found the best values α, β, γ to be $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$ on a fixed value of $\Phi = 1$. Additionally, the network architecture uses slightly larger mobile inverted bottleneck convolution (MBConv) which can be viewed in Fig. 2.

B. DenseNet121

DenseNets [22] are convolutional neural networks where each layer in the network is connected to every other subsequent layer in a feed-forward fashion. The input of a layer is the concatenation of the feature maps of all the previous layers. That is, for a network with L layers. For the l^{th} layer in the network, it has l inputs that consist of the concatenation of all previous layers feature maps and passes the resulted feature map to the subsequent $L - 1$ layers. Hence, the network has a total of $\frac{L\times(L+1)}{2}$ connections between the layers. As a subsequent effect for such a technique, the network requires fewer parameters than traditional convolutional neural networks since there's no need to re-learn redundant feature maps. This dense connection pattern also reduces the effect of the vanishing gradient problem during training deeper architectures since each layer has direct access to the gradients from the loss function and the original input signal, which improves the flow of gradients and information through the network.

A DenseNet consist of L layers where each implements a non-linear transformation $H_l(.)$ where l is the index of the layer, $H_l(.)$ is a composite function of batch normalization (BN), followed by a rectified linear unit (ReLU) and a convolution (Conv). The dense connectivity of the input of layer l in the network can be expressed by (9).

$$x_l = H_l([x_0, x_1, \ldots, x_{x-1}]) \quad (9)$$

where x_l is the input of the layer and $[x_0, x_1, \ldots, x_{l-1}]$ is the concatenation of feature maps from 0 to $l - 1$ layers. The authors divided their network into two main building blocks, dense block (DB) and transition block (TB). A dense block consists of multiple dense layers (DL) where each dense layer consists of 1 × 1 Conv and 3 × 3 Conv layers. A dense block can be represented as in Fig. 3. Between the dense blocks, there are transition blocks which consist of a batch normalization layer followed by a 1 × 1 Conv layer and a 2 × 2 average pooling layer. DenseNet121 is one of the implementations of the DenseNet network with four dense blocks, and each dense block consists of 6, 12, 24, 16 dense layers sequentially. The implementation of DenseNet121 architecture is represented in Fig. 4.

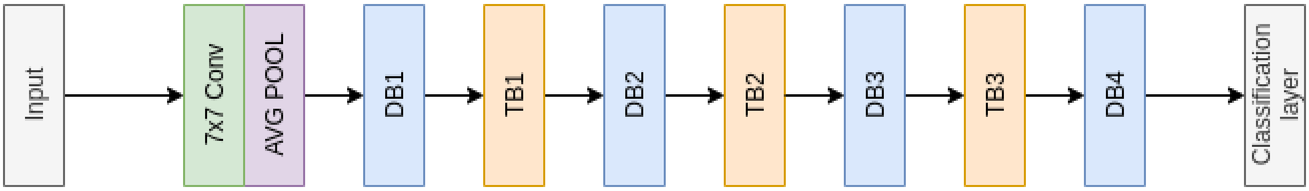


FIGURE 4. The architecture of DenseNet121 CNN. It has 4 dense blocks (DB) where DB1 has 6 dense layers, DB2 has 12 dense layers, DB3 has 24 dense layers, and DB4 has 16 dense layers.

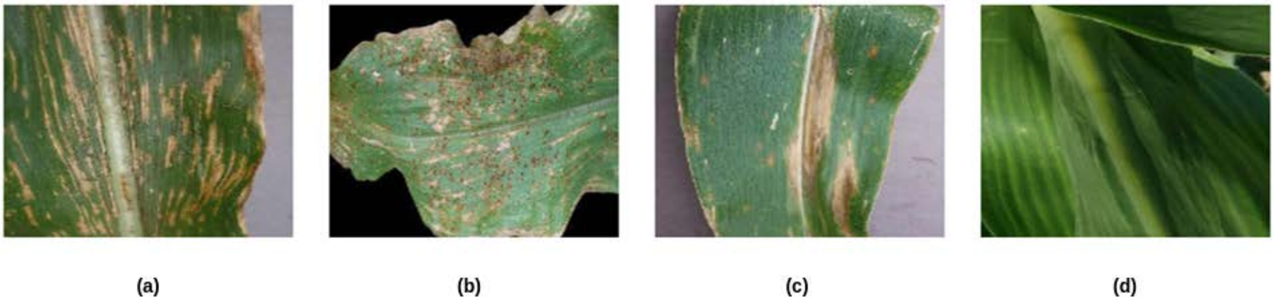


FIGURE 5. Sample images of each category in the dataset. (a) Gray leaf spot, (b) Common rust, (c) Northern leaf blight, and (d) Healthy.

V. FEATURE FUSION

CNN's can extract rich features automatically in each layer. For example, initial layers can extract simple shapes and features from the image like edges and boundaries, while deeper layers can extract more complex high-level features representing complex shapes and complete objects. Since each convolutional layer can extract different sets of features, many researchers extracted such features from different CNN layers and fused them to improve performance. The improved performance can be attributed to having some relevant features being lost in the layered architecture. Additionally, since different CNNs can learn different aspects of the data, fusion of features generated by different CNNs has improved performance. For $a_{i,j,d}$ and $b_{i,j,d}$ input tensors with similar dimensions i, j, d, we consider the following commonly used fusion methods:

A. ADDITION METHOD

The addition method takes a list of tensors of the same shape and outputs a tensor with the same shape after performing addition element-wise for each input tensor. It can be represented by (10).

$$y^{sum}_{i,j,d} = a_{i,j,d} + b_{i,j,d} \tag{10}$$

B. MAXIMUM METHOD

The maximum method takes a list of tensors of the same shape and outputs a tensor with the same shape after taking the maximum value element-wise for each input tensor. It can be represented by (11).

$$y^{max}_{i,j,d} = max(a_{i,j,d}, b_{i,j,d}) \tag{11}$$

where $max(a_{i,j,d}, b_{i,j,d})$ is a maximization function that yields the maximum value being either $a_{i,j,d}$ or $b_{i,j,d}$.

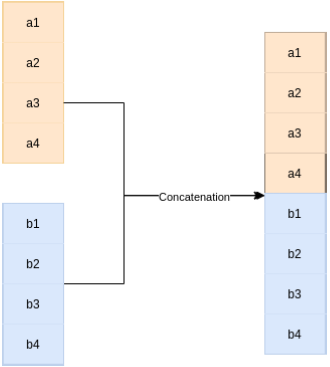


FIGURE 6. Schematic representation for concatenation of two feature vectors.

C. AVERAGE METHOD

Similar to the maximum method, however, instead of using the maximization function, it uses an averaging function as in (12).

$$y^{avg}_{i,j,d} = avg(a_{i,j,d}, b_{i,j,d}) \tag{12}$$

where $avg(a_{i,j,d}, b_{i,j,d})$ is an averaging function that yields the average value of $a_{i,j,d}$ and $b_{i,j,d}$.

D. CONCATENATION METHOD

The concatenation method stacks the input tensors along the concatenation axis together. For $a_{i,j,d}$ and $b_{i,j,d}$ where concatenation is performed on the d dimension, concatenation is performed as in (13). And it can be schematically represented by Fig. 6.

$$y^{concat}_{i,j,2d} = a_{i,j,d} \quad y^{concat}_{i,j,2d-1} = b_{i,j,d} \tag{13}$$

In this study, feature fusion from multiple pre-trained CNNs using the concatenation method improves performance.

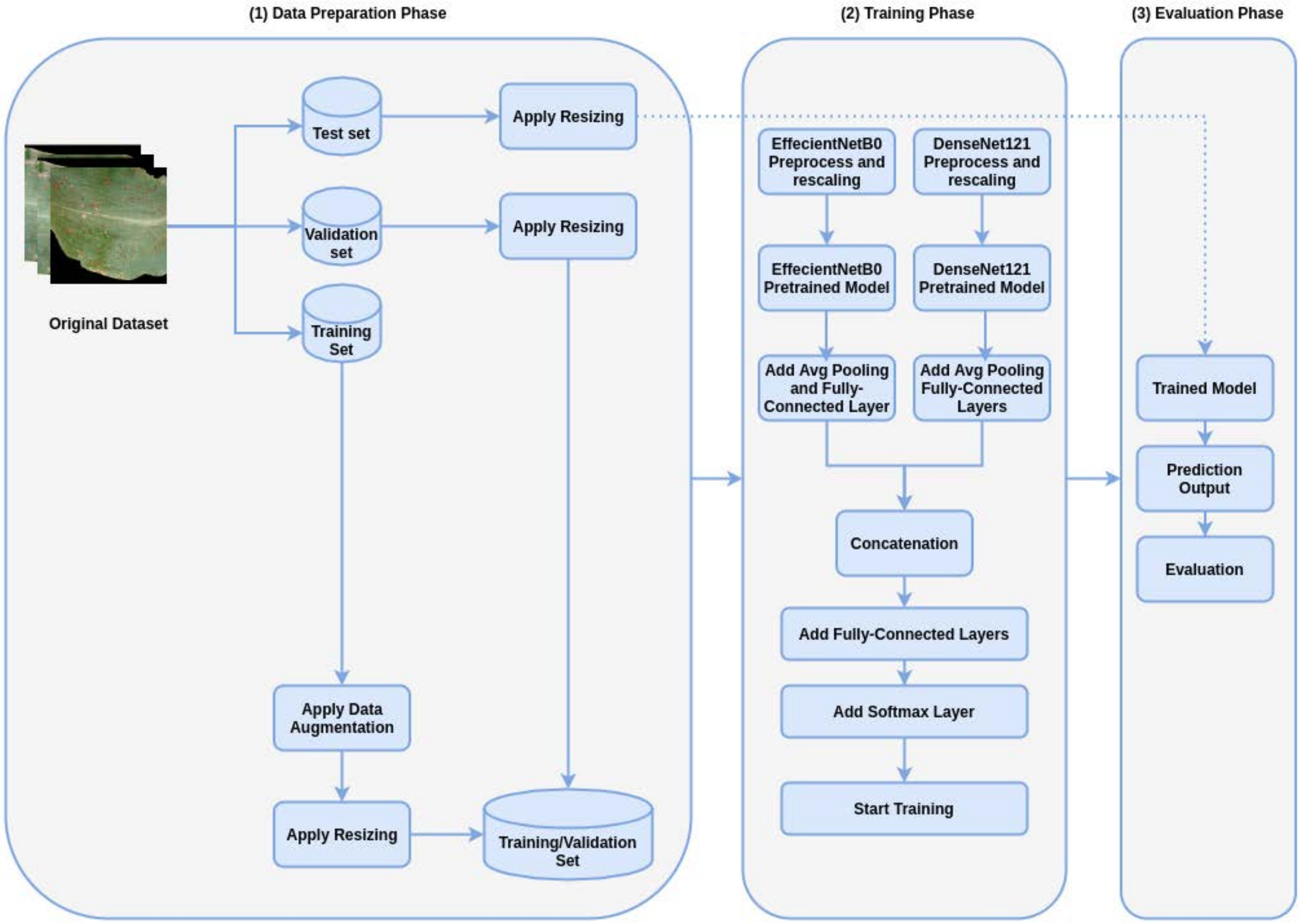

(1) Data Preparation Phase
(2) Training Phase
(3) Evaluation Phase
Original Dataset
Test set
Validation set
Training Set
Apply Resizing
Apply Resizing
Apply Data Augmentation
Apply Resizing
Training/Validation Set
EffecientNetB0 Preprocess and rescaling
DenseNet121 Preprocess and rescaling
EffecientNetB0 Pretrained Model
DenseNet121 Pretrained Model
Add Avg Pooling and Fully-Connected Layer
Add Avg Pooling Fully-Connected Layers
Concatenation
Add Fully-Connected Layers
Add Softmax Layer
Start Training
Trained Model
Prediction Output
Evaluation

FIGURE 7. Proposed classification model framework diagram. It consists of three main phases: Data preparation phase, training phase, and evaluation phase.

VI. THE PROPOSED END-TO-END CORN CLASSIFICATION MODEL

In this section, the proposed classification model is presented, and its different phases are discussed. Fig. 7 shows a visual representation for the proposed framework in which it consists of three main phases: (1) Data preparation phase, (2) Training phase, and (3) Evaluation phase.

A. DATASET DESCRIPTION

The dataset used in this experiment is a subset of a PlantVillage dataset hosted on Kaggle [23]. It consists of approximately 217,000 images consisting of 38 different categories of both healthy and diseased plant images. From this dataset, images of corn plants were chosen for the experiment as it includes a large number of images in four categories. The chosen subset consists of four categories: healthy corn leaves and the other three having infected leaves, namely northern leaf blight, common rust, and gray leaf spot. The number of images in each categories is also presented in Table 2. Sample images of each category in the dataset can be viewed in Fig. 5. Northern leaf blight, also known Turcicum leaf blight, is caused by the fungus Exserohilum turcicum. It is characterised by long lesions of 1 to 6 inches long that are elliptical, gray-green in color. Common rust, which is caused by the fungus Puccinia sorghi, is characterised by dark, reddish-brown small pustules scattered over both the upper and lower surfaces of the corn leave. Gray leaf spot, also know as Cercospora leaf spot, is a fungal diseases cause by Pyricularia grisea. In early stages of the disease it is characterised by small brown leaf spots, which can expand rapidly to a large oval gray spots.

TABLE 2. Number of images in each category of the dataset.

Category	Number of images
Northern leaf blight	3940
Common rust	4768
Gray leaf spot	2052
Healthy	4648
Total	15408

B. DATA PREPARATION PHASE

The dataset was divided into 80% training split to be used for the training process, and a 20% test split to be used for evaluating the model's performance. A validation split was

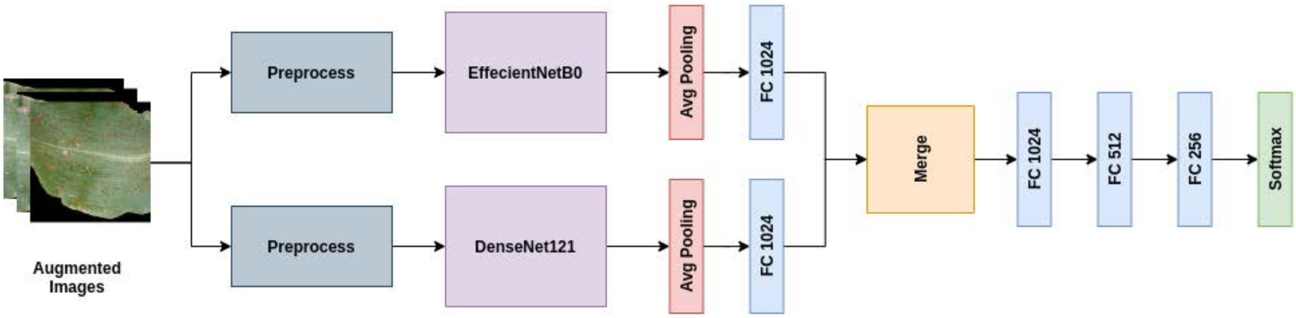
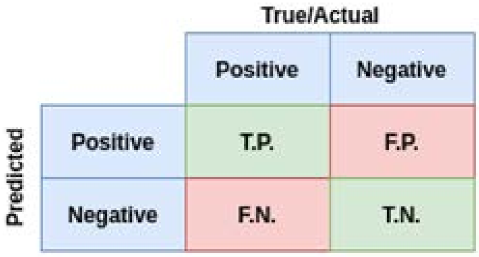


FIGURE 8. Architecture of the proposed CNN. The CNN consists of two branches where each branch has a preprocessing function, a feature extractor, and is followed by an average pooling and a fully connected layer. Afterwards, the results of the fully connected layers are merged using concatenation method and followed by a series of fully connected layers and finally a softmax layer for classification.

TABLE 3. Data augmentation values used for augmentation techniques.

Augmentation Technique	Value
Zoom	20%
Shear	20%
Rotation	20°
Horizontal Flip	True



		True/Actual	
		Positive	Negative
Predicted	Positive	T.P.	F.P.
	Negative	F.N.	T.N.

FIGURE 9. The confusion matrix. The prediction result is classified as either true positive, true negative, false positive, or false negative.

taken from the training subset by taking 20% of the training samples. The training subset is fed to the model to learn the complex features of the images. In contrast, the validation subset is kept separate from the training subset. It is used to monitor the model's performance by feeding it to the model after each epoch and evaluating its performance. The test subset is used after the training has concluded to assess the model's overall performance on data that it didn't see before.

To avoid over-fitting, the dataset was augmented using a combination of the horizontal flip, rotation, shearing, zooming techniques. Table 3 shows the corresponding values for each of the augmentation techniques used. Finally, images were resized to 244 × 244 pixels before the subsets were used in the remaining phases.

C. TRAINING PHASE

The architecture of the CNN used in the experiment can be demonstrated as in Fig. 8. The DenseNet121 and EfficientNetB0 were used as baselines CNNs to construct the new proposed model where the pretrained weights of each model were loaded, and the classification halves of the models were replaced by an average pooling layer and a fully connected layer with 1024 neuron. The weights of the fully connected layers of each branch were then merged via concatenation, and three more fully connected layers of sizes 1024, 512, and 256 neurons respectively were added. Finally, a SoftMax activated layer with four neurons was added for final classification, and the categorical crossentropy loss function was used.

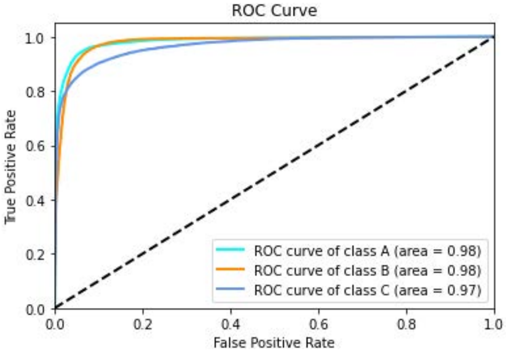


FIGURE 10. The receiver operating characteristic curve for a multi-class classification problem.

Since EfficientNetB0 and DenseNet121 have different preprocessing requirements, preprocessing the augmented images were moved to happen on the fly in the training phase, where a preprocessing layer is added to each branch model before it passes through the feature extraction stage. In the EfficientNetB0 branch, the preprocessing step rescales the image's pixel values to be in the range of 0 to 1, which can be described as in (14).

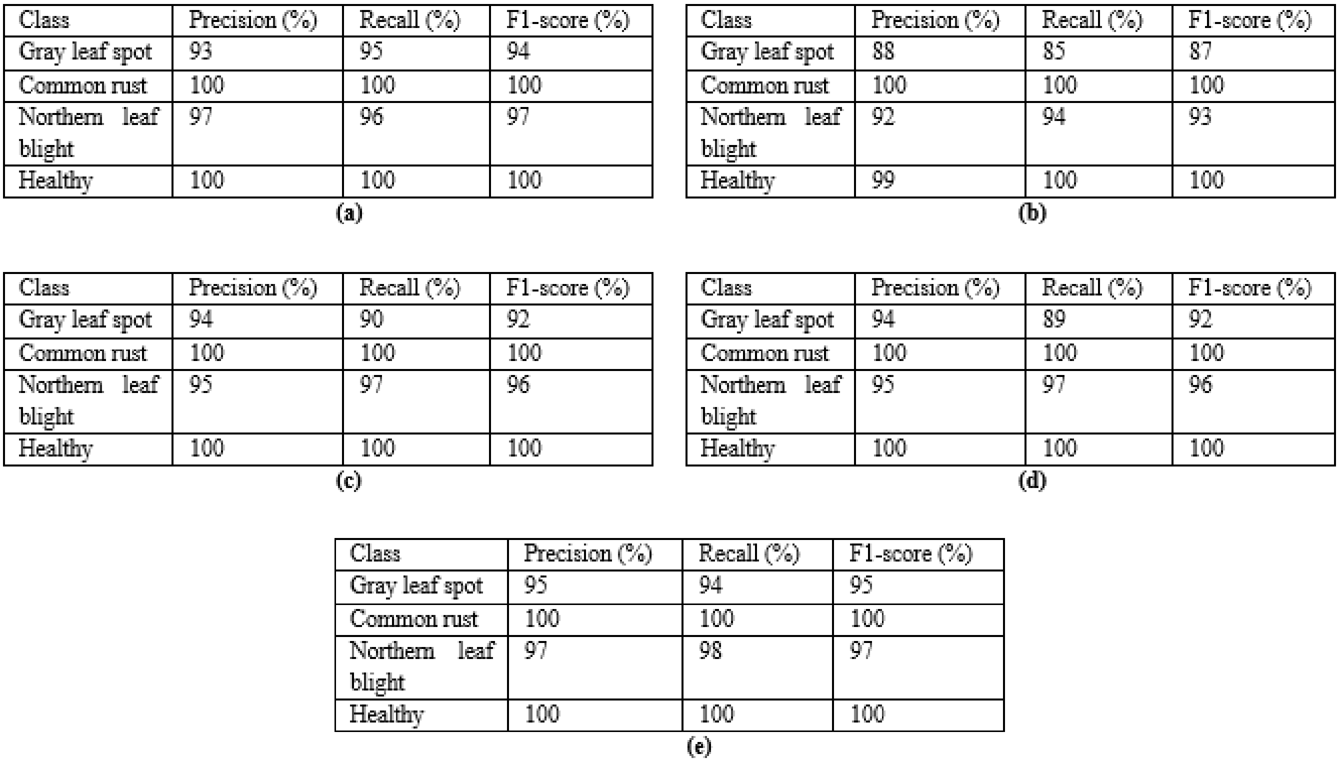
$$y_{i,j,d} = \frac{x_{i,j,d}}{255} \tag{14}$$

While in the DenseNet121 branch, the preprocessing includes the rescaling step in (14) followed by a normalizing step where each channel is normalized concerning the ImageNet means and standard deviation values. This can be mathematically formulated as in (15).

$$y_{i,j,d} = \frac{x_{i,j,d} - \mu}{\sigma} \tag{15}$$

D. EVALUATION PHASE

In the evaluation phase, the test set was used to evaluate the model's performance and ensure that the model doesn't



Class	Precision (%)	Recall (%)	F1-score (%)
Gray leaf spot	93	95	94
Common rust	100	100	100
Northern leaf blight	97	96	97
Healthy	100	100	100

(a)

Class	Precision (%)	Recall (%)	F1-score (%)
Gray leaf spot	88	85	87
Common rust	100	100	100
Northern leaf blight	92	94	93
Healthy	99	100	100

(b)

Class	Precision (%)	Recall (%)	F1-score (%)
Gray leaf spot	94	90	92
Common rust	100	100	100
Northern leaf blight	95	97	96
Healthy	100	100	100

(c)

Class	Precision (%)	Recall (%)	F1-score (%)
Gray leaf spot	94	89	92
Common rust	100	100	100
Northern leaf blight	95	97	96
Healthy	100	100	100

(d)

Class	Precision (%)	Recall (%)	F1-score (%)
Gray leaf spot	95	94	95
Common rust	100	100	100
Northern leaf blight	97	98	97
Healthy	100	100	100

(e)

FIGURE 11. Precision, recall, and f1-score of the models adapted in the experiment. (a) ResNet152, (b) InceptionV3, (c) EfficientNetB0, (d) DenseNet152, (e) Proposed model.

overfit. The proposed model is evaluated on the test set that consists of 20% of the original dataset sample. The evaluation was based on the correct classification of an input image as being one of northern leaf blight, common rust, gray leaf spot, or healthy classes.

VII. RESULTS, ANALYSIS, AND DISCUSSION

This section presents and discusses the results obtained by the proposed approach in details.

A. EVALUATION MEASURES

The trained models are evaluated by computing their accuracy, precision, recall, and f1-score values for their predictions on the test data alongside the receiver operating characteristic (ROC) curve and the area under curve (AUC). Firstly, these values can be explained by denoting the model predictions fit in one of four categories: (1) True positive (T.P.). (2) True negative (T.N.). (3) False positive (F.P.). (4) False negative (F.N.), which is referred to as the confusion matrix.

When there's only two possible classes in the classification problem at hand (e.g., healthy vs unhealthy), the actual class of concern is denoted as the positive class and anything else as the negative class. T.P. is when the models' prediction, for instance, is positive, and the instance is positive. T.N. is when the models' prediction, for instance, is negative, and the instance is negative. F.P. is when the model falsely predicts an instance as positive while it should be negative, and F.N. is when the model predicts an instance as negative while it should be positive. This can be visualized as in Fig. 9.

In multi-class classification problems, T.P., T.N., F.P., and F.N. are treated as a one-vs-all problem. Thus, to a specific class of concern, the positive class is the class of concern, and the negative class is all other classes. In this context, accuracy is the overall predictive accuracy of a model, which is the number of correctly predicted samples divided by the total number of predictions presented by (16).

$$accuracy = \frac{T.P. + T.N.}{T.P. + T.N. + F.P. + F.N.} \tag{16}$$

In contrast, precision is the ratio between the numbers of correctly predicted positives for a class to the total number of positive predictions for that class, calculated by using (17).

$$precision = \frac{T.P.}{T.P. + F.P.} \tag{17}$$

Recall gives a measurement for what fraction of all positive samples are correctly predicted as positive, calculated by (18).

$$recall = \frac{T.P.}{T.P. + F.N.} \tag{18}$$

And the f1-score is computed as the weighted average of Precision and Recall as in (19).

$$f1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{19}$$

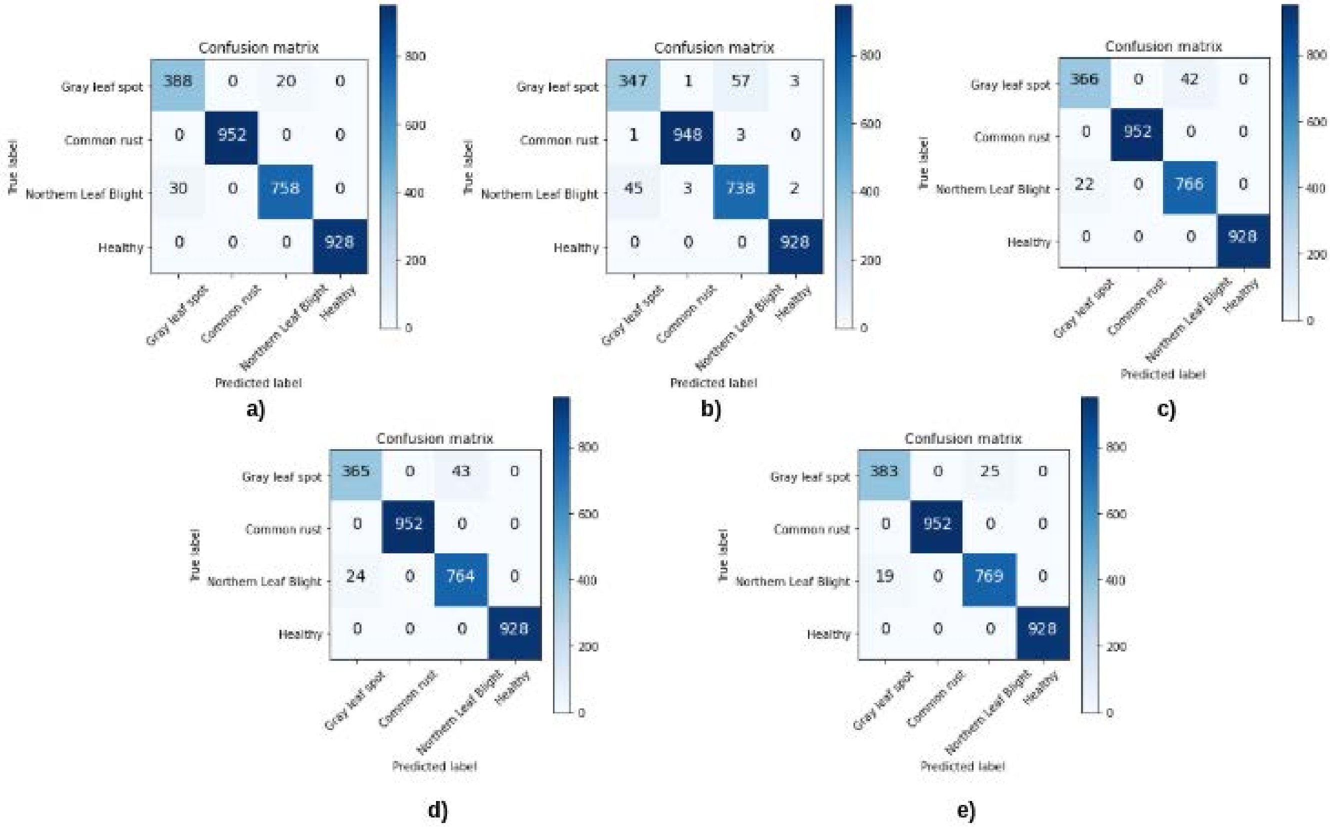


FIGURE 12. Confusion matrix of the models adapted in the experiment. (a) ResNet152, (b) InceptionV3, (c) EfficientNetB0, (d) DenseNet152, (e) Proposed model.

The ROC curve is a graphical plot that illustrates the diagnostic ability of a binary classifier as its discrimination threshold varies. It is created by plotting the recall value against the false positive rate (F.P.R.) value at different threshold levels as in Fig. 10 where the F.P.R. can be represented by (20).

$$F.P.R. = \frac{F.P.}{F.P. + T.N.} \tag{20}$$

In a multi-class classification problem, the ROC curve for each class can be plotted as a one-vs-all problem as well, where the class of concern is considered a positive class, and every other class is considered a negative class.

The AUC is the area under the ROC curve that denotes the probability that a classifier would rank a randomly chosen positive instance higher than a randomly chosen negative one.

B. EXPERIMENT SETUP

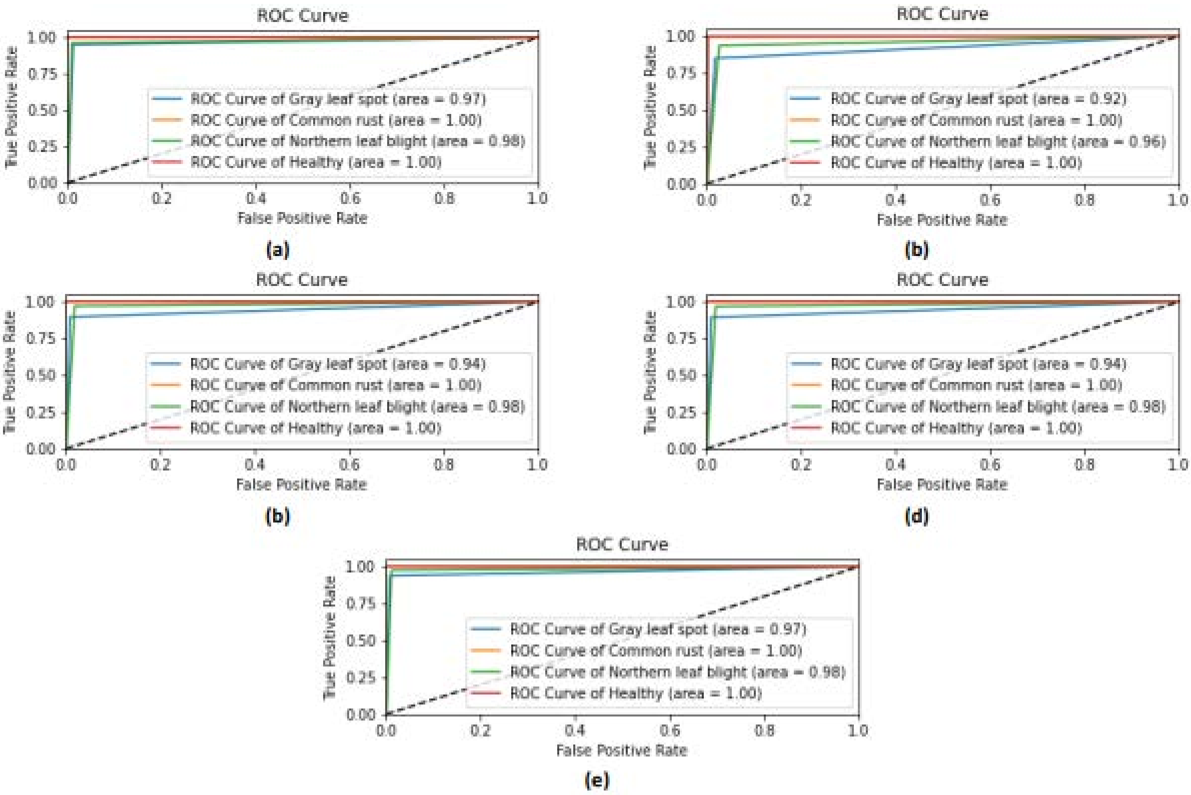
The Keras library was used to build the structure of the proposed approach. Keras is a high-level framework written in Python that enables researchers to build and develop deep learning models rapidly. The Google Colaboratory (Colab) environment was used to build the models, train and test them. The environment provides Nvidia K80 GPU with 12GB VRAM at 0.82GHz, it's also powered by 2 CPU cores and 12GB of RAM.

To minimize the error generated by the CNN, we use a cost function to calculate the difference between the actual output y and the predicted output $\hat{y}$. The cost function used in this study is Cross-Entropy (CE) which is one of the commonly used cost functions that can summarize all aspects of the model into a single value. Hence, improving such value indicates an improvement in the model. For N number of classes, where i is the class number in range $1 - N$, the CE can be represented as in (21).

$$C(y, \hat{y}) = \sum_{i=1}^{N} y_i \times log(\hat{y}_i) \tag{21}$$

To improve the CE function's value, the Adam [24] optimization algorithm is utilized to minimize the cost function by minimizing it indicates a better performance of the model.

After the original training subset was augmented, the CNN illustrated in Fig. 8 was constructed and trained on the training set. The training was initiated with a learning rate of 0.01. It was set to automatically decrease by 0.1 after every four epochs of noimprovement on the validation loss value for a total of 50 epochs. The early stopping technique was used to stop the model's training after 10 epochs of no improvement to avoid overfitting. The same process was used on other CNNs, namely ResNet152, InceptionV3, EfficientNetB0,



(a)

(b)

(b)

(d)

(e)

FIGURE 13. ROC curve of the models adapted in the experiment. (a) ResNet152, (b) InceptionV3, (c) EfficientNetB0, (d) DenseNet152, (e) Proposed model.

and DenseNet121, to compare their performance against the proposed CNN.

C. EVALUATION OF THE MODELS

After training is finished, the models are evaluated by testing them against the test subset.

The proposed model, ResNet152, InceptionV3, EfficientNetB0 and DenseNet121 models achieved 98.56%, 98.37%, 96.26%, 97.91%, and 97.82% classification accuracy respectively on the test subset. Table 4 presents a comparison between the models used in the experiment in terms of accuracy and number of parameters where the proposed model achieved the highest accuracy across all the models used in the experiment. While Table 5 shows the time complexity of the models used in terms of total training/testing time, the number of epochs the model was able to converge in, and the average consumed time per epoch in the training phase. Table 6 also shows a comparison between the classification accuracy of the proposed model with the work of [8], [12], [13] and [14] that used the same dataset where the proposed model outperformed the other methods. The proposed model was able to outperform other models due to having a rich set of feature coming from the different branches of the model where each branch produces a different set of features.

The precision, recall, and f1-scores for the models used in the experiment are reviewed in Fig. 11. For the gray leaf spot class, the proposed model achieved the highest precision and f1-score with precision and f1-score of 95%. At the same time, ResNet152 had the highest recall of 95% as opposed to the proposed model having 94%, while on

TABLE 4. Accuracy of the models used in the experiment.

Model	Accuracy (%)	No. of parameters (millions)
ResNet152	98.37	60.47
InceptionV3	96.26	20.90
EfficientNetB0	97.91	5.36
DenseNet121	97.82	8.09
Proposed Model	**98.56**	16.20

TABLE 5. Time complexity for the models in training and testing phases (time measured in seconds).

Model	Train	Test	No. of epochs	Average per epoch
ResNet152	6583	131	44	149.6
InceptionV3	1740	66	27	64.4
EfficientNetB0	4938	60	50	98.7
DenseNet121	7949	71	50	159
Proposed Model	6183	129	48	128.8

TABLE 6. Comparison between the proposed model and work in [8], [12], [13], and [14] in terms of classification accuracy.

Model	Accuracy (%)
[8]	97.6
[12]	97
[13]	94
[14]	98.06
Proposed Model	**98.56**

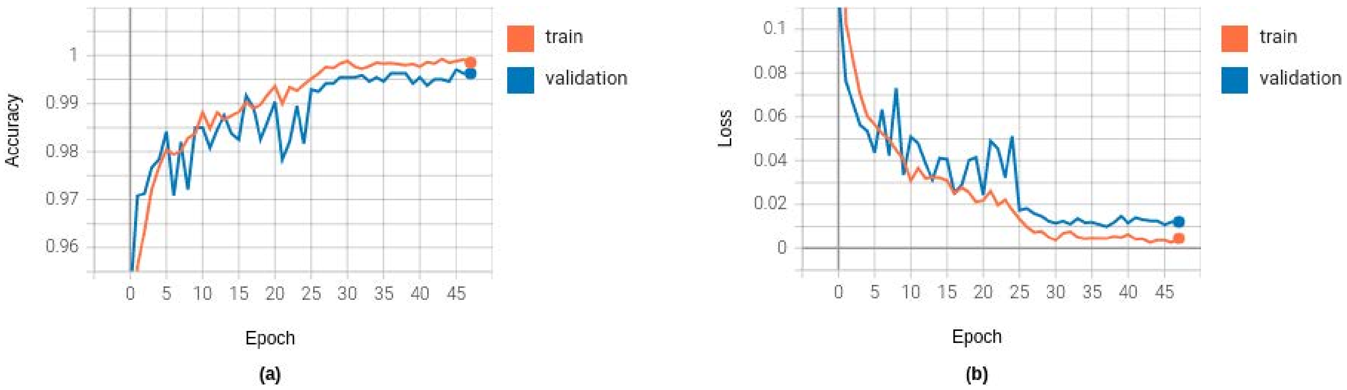


FIGURE 14. Accuracy and loss curves. (a) Accuracy curve for train and validation. (b) Loss curve for train and validation.

common rust, all the models achieved a precision, recall, and f1-score of 100%. For the northern leaf blight class, both the proposed model and the ResNet152 model had the highest precision and f1-score of 97%, while the proposed model had the highest recall in this class with 98%. Lastly for the healthy class, all the models had similar precision, recall and f1-score of 100% except for the InceptionV3 model that had a precision of 99%.

The confusion matrix for the proposed model and other CNNs used in the study is presented in Fig. 12. The figure shows that the proposed model mistakenly classified only 25 gray leaf spot samples as northern leaf blight and 19 northern leaf blight samples as gray leaf spot. Such an error can be attributed to the similarity between the visual symptoms of the two classes. On the other hand, the model correctly classified 100% of the samples in both the common rust and healthy classes. The ROC curve for the proposed model is presented in Fig. 13, which shows that both the proposed model and ResNet152 had similar AUC for all the given classes and outperformed all other CNNs on the gray leaf spot class with an AUC of 97%. In contrast, InceptionV3 had the worst AUC value on both gray leaf spot and northern leaf blight classes with an AUC of 92% and 96%, respectively. On the other hand, all CNNs had similar AUC value on the common rust and healthy classes with an AUC of 100%. The progression of the training and validation accuracies can also be viewed in Fig. 14.

VIII. CONCLUSION AND FUTURE WORK

The results obtained from the experiments in this study allow us to conclude that for corn plant disease classification, deep features extracted from different CNNs and fusing the features to produce a more complex feature set improve classification accuracy. Based on the comparative study, we can conclude that the approach used in this paper that resulted in 98.56% classification accuracy has better results in terms of accuracy scores presented in the literature. Additionally, using CNNs with relatively small parameters to extract features and combining their feature sets produces more robust models that outperform other CNNs with much larger parameters. We can apply the same approach to classify more corn diseases and other plant diseases from digital images as future work. Also, we can explore more augmentation techniques and try different combinations of CNNs as feature extractors. Moreover, the results of this work can be explored in different contexts by using different feature extractors and different fusion methods on any dataset.

REFERENCES

[1] Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. (Mar. 29, 2016). *Report of the Plenary of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services on the Work of Its Fourth Session*. [Online]. Available: https://seea.un.org/content/reportplenary-intergovernmental-science-policy-platformbiodiversity-and-ecosystem-services

[2] A. P. K. Tai, M. V. Martin, and C. L. Heald, "Threat to future global food security from climate change and ozone air pollution," *Nature Climate Change*, vol. 4, no. 9, pp. 817–821, Sep. 2014.

[3] R. N. Strange and P. R. Scott, "Plant disease: A threat to global food security," *Annu. Rev. Phytopathol.*, vol. 43, pp. 83–116, Jul. 2005.

[4] *Smallholders, Food Security and the Environment*, Int. Fund Agricult. Develop., Rome, Italy, 2013, p. 29.

[5] C. A. Harvey, Z. L. Rakotobe, N. S. Rao, R. Dave, H. Razafimahatratra, R. H. Rabarijohn, H. Rajaofara, and J. L. MacKinnon, "Extreme vulnerability of smallholder farmers to agricultural risks and climate change in madagascar," *Phil. Trans. Roy. Soc. B, Biol. Sci.*, vol. 369, no. 1639, Apr. 2014, Art. no. 20130089.

[6] R. Perroy, "World population prospects," *United Nations*, vol. 1, no. 6042, pp. 92–587, 2015.

[7] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, nos. 3–4, pp. 197–387, Jun. 2014.

[8] R. Hu, S. Zhang, P. Wang, G. Xu, D. Wang, and Y. Qian, "The identification of corn leaf diseases based on transfer learning and data augmentation," in *Proc. 3rd Int. Conf. Comput. Sci. Softw. Eng.*, May 2020, pp. 58–65.

[9] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," *Comput. Intell. Neurosci.*, vol. 2016, pp. 1–11, May 2016.

[10] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[11] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 2, pp. 445–458, Jun. 2019.

[12] E. L. da Rocha, L. Rodrigues, and J. F. Mari, "Maize leaf disease classification using convolutional neural networks and hyperparameter optimization," in *Proc. Anais do XVI Workshop Visão Computacional. (SBC)*, 2020, pp. 104–110.

[13] M. Agarwal, V. K. Bohat, M. D. Ansari, A. Sinha, S. K. Gupta, and D. Garg, "A convolution neural network based approach to detect the disease in corn crop," in *Proc. IEEE 9th Int. Conf. Adv. Comput. (IACC)*, Dec. 2019, pp. 176–181.

[14] A. Waheed, M. Goyal, D. Gupta, A. Khanna, A. E. Hassanien, and H. M. Pandey, ''An optimized dense convolutional neural network model for disease recognition and classification in corn leaf,'' *Comput. Electron. Agricult.*, vol. 175, Aug. 2020, Art. no. 105456.

[15] Y. Zhao, L. Liu, C. Xie, R. Wang, F. Wang, Y. Bu, and S. Zhang, ''An effective automatic system deployed in agricultural Internet of Things using multi-context fusion network towards crop disease recognition in the wild,'' *Appl. Soft Comput.*, vol. 89, Apr. 2020, Art. no. 106128.

[16] F. Saeed, M. A. Khan, M. Sharif, M. Mittal, L. M. Goyal, and S. Roy, ''Deep neural network features fusion and selection based on PLS regression with an application for crops diseases classification,'' *Appl. Soft Comput.*, vol. 103, May 2021, Art. no. 107164.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, ''Gradientbased learning applied to document recognition,'' *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[18] A. F. Agarap, ''Deep learning using rectified linear units (ReLU),'' 2018, *arXiv:1803.08375*.

[19] L. Torrey and J. Shavlik, ''Transfer learning,'' in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*. Hershey, PA, USA: IGI Global, 2010, pp. 242–264.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, ''ImageNet: A large-scale hierarchical image database,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[21] M. Tan and Q. Le, ''EfficientNet: Rethinking model scaling for convolutional neural networks,'' in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[22] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, ''DenseNet: Implementing efficient ConvNet descriptor pyramids,'' 2014, *arXiv:1404.1869*.

[23] *Plant Village Dataset*. Accessed: Jul. 9, 2021. [Online]. Available: https://www.kaggle.com/saroz014/plant-diseases

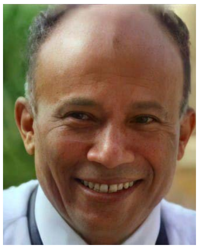
[24] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' 2014, *arXiv:1412.6980*.

HASSAN AMIN received the B.S. degree in statistics and computer science from the Faculty of Science, Helwan University, in 2017. He is currently pursuing the M.Sc. degree. He is also a Teaching Assistant in computer science with the Faculty of Science, Helwan University. He is a member of the Scientific Research Group in Egypt (SRGE). His research interests include deep learning, image processing, and machine learning.

ASHRAF DARWISH (Member, IEEE) received the Ph.D. degree in computer science from the Computer Science Department, Saint Petersburg State University, Russian Federation, in 2006.

He has worked as an Associate Professor and then a Professor in computer science with the Faculty of Science, Helwan University, Cairo. Prior to this, he was an Assistant Professor with the Mathematics and Computer Science Department. From 2017 to 2018, he worked as the Acting Department Chair. He is currently an Adjunct Professor in computer science with Contemporary Technology University, CA, USA. Since 2015, he has been the Vice Chair of the Scientific Research Group in Egypt (SRGE) in the field of computer science and information technology. He has collaborated actively with researchers in this scientific group in several other disciplines of computer science and its applications in medicine, engineering, agriculture, new and renewable energy, nanotechnology, and space science. His research interests span both computer science and information technology. Much of his work has been focusing on artificial intelligence, mainly through the application of data mining, machine learning deep learning, and robotics in different areas of research. His research interests include Internet of Things application, sensor networks, cloud and fog computing, and cyber-physical systems.

Prof. Darwish has a wealth of academic experience and has authored multiple publications, which include research papers, editorial papers, book chapters, essays, and editing books. He is a member of some computing associations, such as the Machine Intelligence Research Laboratories, USA. In addition, he is an Expert and a Reviewer in quality assurance and accreditation systems of education at universities and academic institutes. From 2011 to 2014, he has worked in the Diplomatic Sector as the Cultural and Educational Attaché of Egypt, Embassy of Egypt to Kazakhstan. In 2014, he received the Prestigious Research Distinguished Award in Computer Science and Information Technology. He is an Associate Editor of toughly 25 international journals, and he was a co-editor of some special issues. He has served as the general chair, a program committee, the sessions chair, and a keynote speaker for some of international well-known conferences and workshops.



ABOUL ELLA HASSANIEN is currently the Founder and the Head of the Scientific Research Group in Egypt (SRGE) and a Professor in information technology with the Faculty of Computer and Information, Cairo University. He is also the Ex-Dean of the Faculty of Computers and Information, Beni-Suef University. He has more than 800 scientific research articles published in prestigious international journals and over 40 books covering such diverse topics as data mining, medical images, intelligent systems, social networks, and smart environment. He received several awards, including the Best Researcher of the Youth Award of Astronomy and Geophysics of the National Research Institute, Academy of Scientific Research, Egypt, in 1990, the Scientific Excellence Award in Humanities from the University of Kuwait for the 2004 Award, the Superiority of Scientific—University Award, Cairo University, in 2013, the Islamic Educational, Scientific and Cultural Organization (ISESCO) Prize on Technology, in 2014, the State Award for Excellence in Engineering Sciences 2015, and the Medal of Sciences and Arts of the first class by the President of the Arab Republic of Egypt, in 2017. Also, he honored in Egypt as the Best Researcher with Cairo University, in 2013.

MONA SOLIMAN received the M.Sc. and Ph.D. degrees in information technology from the Faculty of Computers and Information, Cairo University, in 2006 and 2015, respectively. She is currently an Assistant Professor in information technology with the Faculty of Computers and Information. She is a member of the Scientific Research Group in Egypt (SRGE). She also reviewed several articles of *Egyptian Informatics Journal*. Her research interests include pattern recognition, image processing, machine learning, watermarking, and optimization techniques.

• • •