DenseNet Based Model for Plant Diseases Diagnosis

Mahmoud Bakr, Sayed Abdel-Gaber, Mona Nasr, and Maryam Hazman

***Abstract* — The biggest threat to the safety of food is plant diseases. They have the ability to dramatically lower the quantity and quality of agricultural products. Recognizing plant diseases is the biggest issue in the agricultural industries. Convolutional Neural Networks (CNN) are effective in solving image classification problems in computer vision. Numerous deep learning architectures have been used to diagnose plant diseases. This study presents a transfer learning-based model for identifying diseases in plant leaves. In this paper, a CNN classifier based on transfer learning model called DenseNet201 are proposed. An analysis of four deep learning models (VGG16, Inception V3, ResNet152V2, and DenseNet201) done to see which one can detect plant diseases with the greatest degree of accuracy. Web based application developed for plant disease diagnosing from defected leaf image and the proposed model which identify the disease and give the recommended treatment. The used images dataset contains 28310 leaves photos of 3 crops, tomato, potato and pepper divided into 15 different classes, 9 disorders and one healthy class for tomato, 2 disorders and one healthy class for potato and 1 disorder and one healthy for pepper. In our experimental, the results shows that the proposed model achieves the highest training accuracy of 99.44% and validation accuracy of 98.70%.**

***Key words* —Convolutional Neural Network, Deep Learning, Leaf Disease Detection, Transfer Learning.**

I. Introduction

With up to 11.3 percent of the GDP and 28 percent of all jobs, agriculture is a significant part of the Egyptian economy [1]. A key component of the economy is the quality of agricultural products, which is influenced by the weather and other environmental conditions. It is essential to manufacture high-quality goods with a suitable yield because a variety of agricultural products are produced and exported to numerous nations. The human diet contains more than 80% products from plant cultivation. Numerous plant diseases, including bacterial, fungal, and viral infections, affect plants. Food and Agriculture Organization (FAO) estimates that plant pests and diseases cause 20 to 40 percent of the world's food production to be lost (1). Contributing to the solution of this problem is a significant task for Egypt in achieving food security.

Recently, disease prevention has grown in importance when it comes to farming. Identification of plant diseases is crucial in agriculture since plant diseases cannot be prevented. On plant leaves, the majority of disease symptoms can be seen. Therefore, the simplest method to determine whether a plant is sick is to examine the condition of its leaves. Recognizing plant diseases is a difficult task for agriculture experts to take on since it requires the application of scientific processes and a prolonged period of observation. (2).

In Egypt, a significant issue is the lack of agricultural extension specialists who assist farmers with agricultural advice and counseling. Farmers thereby grow reliant on themselves or the Internet to address any issues they encounter in their agricultural operations.

It is necessary to create an automated system that can perform plant disease recognition tasks and offer an efficient solution, given the prevalence of smart phones among farmers, the use of graphics processing units (GPU) in computers and servers, and the quick development of artificial intelligence, computer vision, and deep learning techniques.

CNN's can be used to identify plant diseases (3). One of the most effective methods for identifying patterns in large data sets is CNN. In terms of identifying these diseases, CNN has pretty positive results. Several CNN classification architectures, including VGG16, Inception V3, and DenseNet201, have been employed in the past to detect diseases (3), (4).

Comparison research is used to identify the high accuracy CNN deep learning models in order to construct an automatic plant leaf disease detection system. The transfer learning model that produces the best results is then used as the foundation for our suggested model. The CNN classifier comes after feature extraction in the basic transfer learning model. The research's contributions are: a) A comparison of a few models that rely on transfer learning; b) A proposed model that uses DenseNet transfer learning as a features extractor and a CNN classifier; and c) Web based application utilizes the proposed model to diagnose defected leaf image and identify the disease and give the recommended treatment.

The plantvillage pictures dataset's subset of 3 cropping leaf images is used in the study that is being presented. It contains 28310 images for 15 various classes that were downloaded from the Kaggle portal (5). These classes include tomato bacterial spot, potato early blight, potato late blight, potato healthy, tomato target spot, tomato yellow leaf curl virus, tomato mosaic virus, tomato healthy, tomato leaf mould, tomato septoria leaf spot, tomato spider mites, and pepper bacterial spot. To prepare the dataset of chosen tomato photos for training the classification model, it is first scaled and expanded. We tweak classification models and run the classification models again to improve the classification results. Then, we run tests on several models using a portion of the photos. Our suggested CNN model uses the DenseNet

Submitted on August 14, 2022.
Published on September 21, 2022.
M. Bakr, Agricultural Research Center, Egypt.
(corresponding e-mail: mah.bakr.2005@gmail.com)
S. Abdel-Gaber, Faculty of Computers and Artificial Intelligence, Helwan University, Egypt.
(e-mail: sgaber14@gmail.com)
M. Nasr, Faculty of Computers and Artificial Intelligence, Helwan University, Egypt.
(e-mail: m.nasr@helwan.edu.eg)
M. Hazman, Agricultural Research Center, Egypt.
(e-mail: m.hazman@arc.sci.eg).

model, which provides the maximum accuracy when utilised as a features extractor. Finally, we compared and analysed the results.

The following is the order of the paper: The previous relevant works are described in Section 2. The methodology is described in Section 3. The suggested model is highlighted in Section 4. Experimental results and analyses are given in Section 5. Final thoughts are in Section 6.

II. Literature Review

How to identify plant diseases has been a topic of discussion over the years. Many researchers have developed a range of suitable designs for identifying plant diseases using machine learning techniques.

Transfer learning was utilised by the authors in (6) to decrease the amount of training data needed, the amount of time needed, and the cost of computation. They divide diseased leaves into nine categories, which includes healthy tomato leaves. The feature extraction was carried out using five deep network structures: Resnet50, Xception, MobileNet, ShuffleNet, and Densenet121 Xception. In an experiment, network architectures with various rates of learning were compared. Test those networks after making the necessary training parameter adjustments. The parameters and average accuracy of the five convolutional neural networks varied. Densenet Xception has a best recognition accuracy of 97.10 percent, however its parameters are at most. ShuffleNet has a recognition accuracy of 83.68 percent, and its paramenters are minimal.

For the identification and categorization of tomato disease, the authors in (7) used a CNN-based method. The experimental findings demonstrate the proposed model's superior performance to VGG16, InceptionV3, and MobileNet, three pre-trained models. The suggested model has an average accuracy of 91.2 percent for the nine disease classes and one healthy class, ranging from 76 to 100 percent for each class.

The authors of (8) looked into a model for identifying plant diseases that combined different plant diagnoses. The data they used comprised pictures of the leaves from six different plants, including tomato, potato, rice, corn, grape, and apple. It was gathered from several web sources. Many well-known convolutional neural network (CNN) architectures were put into practise. They discovered that for multi-label plant disease classification tasks, both the Xception and DenseNet architectures outperform the competition.

The authors in (9) classified six distinct tomato diseases as well as a healthy class using AlexNet and VGG 16. By adjusting the number of images, batch sizes, and weight and bias learning rates, the performance was evaluated. They found that AlexNet outperformed VGG 16 in terms of execution speed and accuracy. Given that this effort also aims to classify the illnesses identified in tomato plants, it should be mentioned that. Based on the findings of this comparison, they established their suggested methodology, which supported them in defining their scope of work and choosing which architectures to use. Nevertheless, it is possible to disregard the VGG 16 implementation due to the shortcomings it offers in comparison to AlexNet, particularly in the computational area.

The authors of (10) suggested using a CNN model to categorize grape diseases into four groups based on photographs. The suggested model is a combined CNNs architecture named UnitedModel that is built on Google InceptionV3 and ResNet50. Utilizing the depth of ResNet50 and the width of InceptionV3, UnitedModel learns from the output feature layers from both models. The accuracy of the suggested UnitedModel is 99.17%.

The authors of (11) created and assessed a number of techniques for diagnosing plant diseases in the absence of sufficient data. They created a Triplet network and a deep adversarial metric learning (DAML) strategy using three CNN architectures (ResNet18, ResNet34, and ResNet50). These techniques were developed to identify novel diseases from few photos, ranging from 5 to 50 photographs per disease, after being trained on a large source domain dataset. Their suggested strategies were assessed in the event that the illness and plant species were recognized simultaneously or only in the event that the disease was identified, regardless of the affected plant. The results reveal that the baseline model beat all other competitive techniques and obtained an accuracy of 99 percent when the change from the source domain to the target domain was minimal and 81 percent when that shift was substantial.

In this study, we examine four distinct transfer learning-based deep learning models. We mostly used the CNN architecture known as the DenseNet201 network as our pre-trained model in Transfer Learning. A few other well-known pre-trained models (VGG16, Inception V3, and ResNet152V2) were also studied along with a few Transfer Learning architectures, and they were compared to DenseNet201. Furthermore, fine-tuning has been done to increase the detection's precision. Images of healthy plants as well as nine distinct diseases are included in the dataset for our investigation. The section below describes our system for identifying plant diseases.

III. Methodology

This study uses a variety of supervised deep learning algorithms to find diseases in plant leaves. We want to investigate how well they did in identifying the 15 plant diseases that were thought to exist and determine which one was the best. The best model will then serve as the foundation for our suggested model.

Deep learning model implementation involves a number of processes. After gathering the data, it is split into two halves, typically 80 percent training and 20 percent validation. The relevance of the models is then determined by training/validation plots, which can be produced by starting from scratch or by applying the transfer learning technique. The photos are then categorized using performance metrics, and classification is completed by applying visualization techniques and mappings (12).

It has been shown that CNNs can do image recognition without pre-processing, feature extraction, or feature classification. On the other hand, the trained model can quickly classify the image. It takes a lengthy time and a big number of data sets to train a large-scale neural network. Additionally, manually labelling data in accordance with predetermined selection criteria is time-consuming and expensive(13).

Transfer learning is a knowledge-sharing technique that reduces the volume of training data, training time, and processing costs for creating deep learning models. Transfer learning enables the learning from one model to be applied to another. In the transfer learning method of machine learning, CNNs that have been trained for one job are utilized as the foundation for a model for another task (4) (13).

We adhere to the procedures depicted in Fig. in order to compare various models. 1. First, we choose the subset of the plantvillage images dataset called the tomato images dataset. The necessary image preparation, such as resizing, is then applied. After enhancing the photos, we use the chosen classification models to generate the training model. We tweak the classification models and retrain them to produce better results in order to enhance the performance of the currently employed models. The chosen models are then put to the test against a subset of photos. The outcomes and findings are then analyzed.

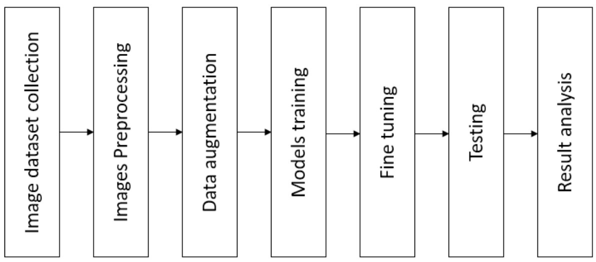


Fig. 1. Steps of Detection and classification process for plant leaf diseases.

A. Plant Leaves Images Dataset

The utilized dataset is a portion of the bigger plantvillage dataset (5). It has 22930 images total, which is broken up into three groups with 75% for training, 20% for validation, and 5% for testing. Every image in the data set contains a leaf of a plant, which occupies the bulk of the image's area and acts as the image's background virtually continuously. The data set was divided into 15 classes, 9 classes of tomato diseases besides the tomato healthy class, 2 classes of potato diseases besides the potato healthy class, and one class of pepper disease besides the pepper healthy class. The 15 classes were as follows: tomato Bacterial spot, tomato early blight, tomato late blight, tomato leaf mold, tomato septoria leaf spot, tomato spider mites two spotted spider mite, tomato target spot, tomato yellow leaf curl virus, tomato mosaic virus, tomato healthy, potato early blight, potato late blight, potato healthy, pepper bacterial spot and pepper healthy. Fig. 2. shows example image of each class. The number of photos for each disease is displayed in Table I.

B. Image Preprocessing

The quality of the image data required for image classification is improved through image preprocessing. Preprocessing techniques employ geometric adjustments of images, such as image rotation, scaling, and translation. In this step, we reduced all of the photos' original 256×256 pixel resolutions throughout the preprocessing stages to 224×224 pixels. All photos must have the same size and resolution, according to this requirement.

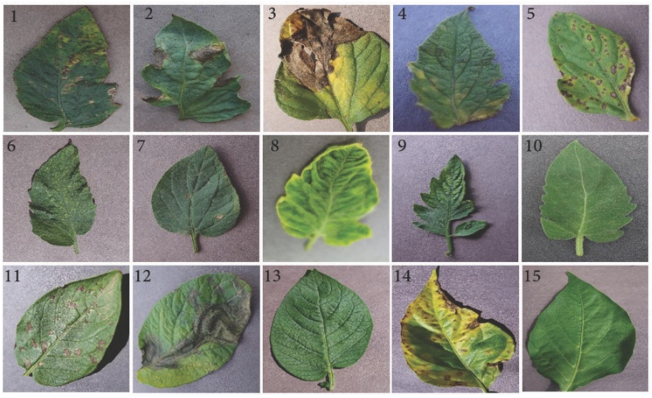


Fig. 2. Examples of tomato 10 classes. (1) tomato bacterial spot, (2) tomato early blight, (3) tomato late blight, (4) tomato leaf mold, (5) tomato septoria leaf spot, (6) tomato spider mites two spotted, (7) tomato target spot, (8) tomato yellow leaf curl virus, (9) tomato mosaic virus, (10) tomato healthy, (11) potato early blight, (12) potato late blight, (13) potato healthy, (14) pepper bacterial spot and (15) pepper healthy.

TABLE I: NUMBER OF IMAGES FOR EACH CLASS OF USED DATASET

Tomato Class	Training Images	Validating Images	Testing Images
Tomato Bacterial spot	1,617	425	85
Tomato Early blight	1,824	480	96
Tomato Late blight	1,758	463	93
Tomato Leaf mold	1,788	470	94
Tomato Septoria leaf spot	1,658	436	87
Tomato Spider mites	1,654	435	87
Tomato Target spot	1,736	457	91
Tomato Yellow leaf virus	1,863	490	98
Tomato Mosaic virus	1,700	448	90
Tomato Healthy	1,830	481	96
Potato Early blight	750	200	50
Potato Late blight	750	200	50
Potato Healthy	684	182	46
Pepper Bacterial spot	747	199	51
Pepper Healthy	1,108	295	75

C. Augmentation Process

For CNN to produce better results, a lot of training data is necessary (14). To develop the best deep CNN model with minimal training data, image augmentation is frequently necessary to enhance the model's performance. By include a few distorted images in the training data, image augmentation expands the number of images in the dataset and decreases overfitting. Overfitting occurs when the network learns the data itself rather than the broader pattern of the dataset. Using a variety of processing techniques or a combination of processing techniques, such as picture flipping, rotation, blur, relighting, and random cropping, image augmentation artificially builds training images (13). In this study, we scale, shear, zoom, and horizontally flip the photos as part of the image augmentation process.

D. Fine-Tuning

A method for increasing a function's effectiveness is fine-tuning. It makes little tweaks to enhance the result. Due to the significance of the adjustment process, even little adjustments can have a big influence on training in terms of computation time, convergence speed, and the number of processing units required (15). To increase the precision of our model, this fine-tuning procedure was performed multiple times. Table II lists the training and fine-tuning variables that generate the best outcomes.

TABLE II: FINE-TUNING PARAMETERS AND VALUES USED THROUGH TRAINING MODELS

Parameter	Value
Batch size	32
Steps per epoch	885
Epoch	50
Validation steps	1
Optimizer	Adam
Activation function	Softmax

E. Training The Models

In this step, the chosen CNN models were trained using the plant disease identification data set. The fixed low-level network parameters remain intact throughout the training process while the high-level network parameters are adjusted. The network's high-level parameters are trained using the plant disease image, and the learned model is then used to categorise the 15 different types of plant leaves. VGG16, Inception V3, Resnet152V2, and DenseNet201 are the chosen CNN.

Simonyan and Zisserman of Google DeepMind and Oxford University's Visual Geometry Group introduced the VGG architecture in 2014. With only 16 convolutional layers stacked on top of one another, it is simple and therefore popular. It has max-pooling layers that aid in reducing volume size, two fully connected layers with 4096 nodes each, and a softmax classifier (16). According to their research, VGG-16 consists of thirteen convolution layers, including three fully connected layers and five coupled max-pooling layers. The second completely connected dense layer comes before the rectified linear unit (ReLU) function. A softmax regression classifier, the final layer of the network, employs probability to categorise the input images.

The picture input size for the VGG-16 architecture is set to 224 × 224 × 3. The architecture of the VGG16 model is shown in Fig. 3 (a). In our work, the last layer with 1000 output classes was removed, the model output was flattened, and then a dense layer with 15 outputs—the plant classes—was introduced to the model for transfer learning of VGG16.

Convolutional deep architecture Inception V3 is frequently used to solve classification issues. Based on the GoogleNet design, Szegedy and his colleagues presented their model concept (17). The inception module was altered to produce Inception V3. The Inception V3 network's building blocks are both symmetric and asymmetric, and each block also includes a variety of convolutional branching, average pooling, max pooling, concatenated, dropout, and fully-connected layer types. The computational cost of this network, which has 42 layers and 29.3 million parameters, is just 2.5 times that of GoogleNet. Finally, the researchers found that they could train a high-quality network on small training sets by lowering the number of parameters and further regularising the network with batch normalised auxiliary classifiers label smoothing (17). The Inception V3 model's architecture is depicted in Fig. 3b. In our work, the top layer was removed, the model output was flattened, and then a dense layer with 15 outputs—plant classes—was built on top of the model for transfer learning of Inception V3.

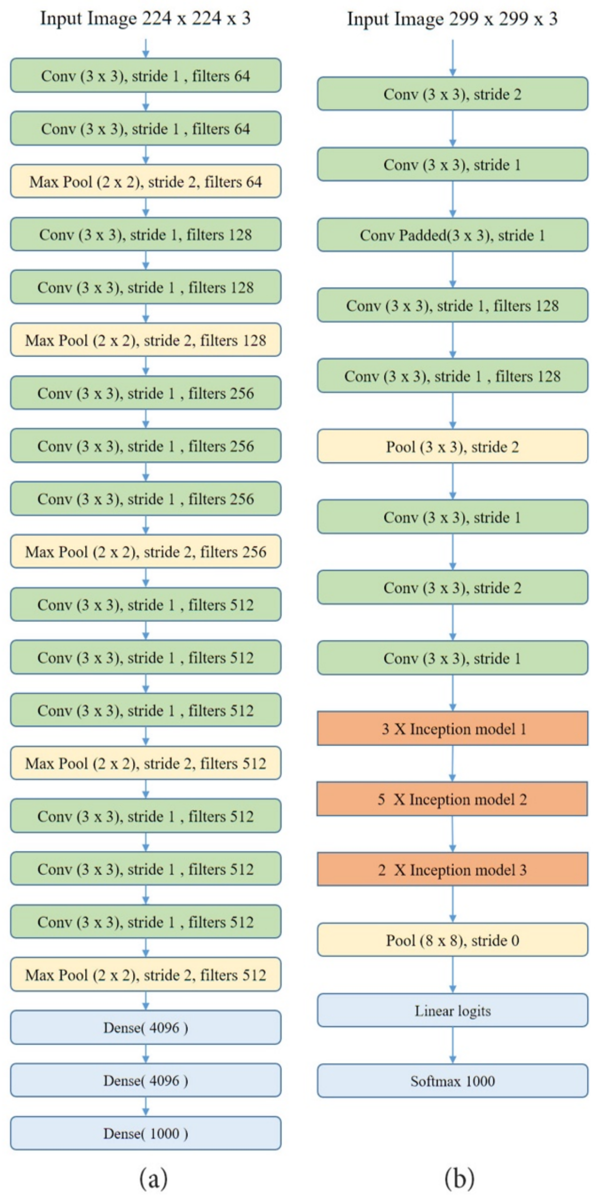


Fig. 3. a) VGG16 architecture, b) Inception V3 architecture.

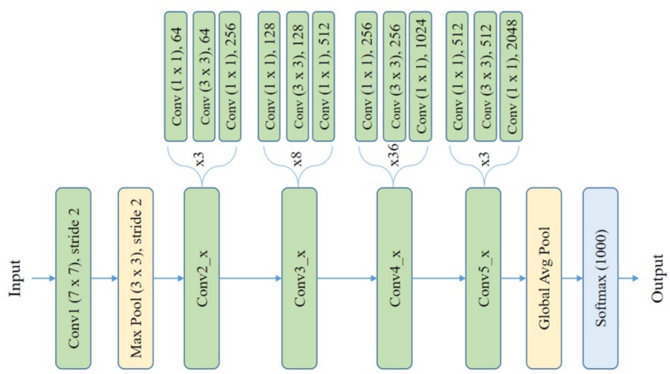


Fig. 4. ResNet152V2 architecture.

A Residual Network (ResNet) is a CNN architecture with hundreds or thousands of convolutional layers (18). Previous CNN designs limited the effectiveness of additional layers. ResNet is highly quick and has a lot of layers. The primary distinction between ResNetV2 and the original (V1) is that each weight layer is first subjected to batch normalization in V2. ResNet performs admirably in picture localization and recognition tests, highlighting the significance of many visual recognition tasks (19). The architecture of the Resnet152V2 model, which has 152 layers overall and was mostly constructed from 3-layer blocks, is shown in Fig. 4.

In their study known as DenseNet, authors in (20) suggested a highly coupled convolutional network design. For maximal information flow between layers, the network's connections are made directly and in a feed-forward fashion. Each layer uses its own feature-map as an input, and all previous levels' feature-maps are used as inputs into all subsequent layers. DenseNets effectively lowers the number of parameters while resolving the vanishing-gradient issue. The architecture of the DenseNet201 model is depicted in Fig. 7 and is further explored in the section below.

IV. Proposed Model

For the purpose of predicting and classifying plant diseases from images of infected leaves, a proposed model based on a pre-trained model and CNN classifier is presented in this section. DenseNet201, which offers the highest accuracy among other models, is the pre-trained architecture used in the suggested model. A CNN is utilised to classify the features that were extracted using DenseNet201. In order to assess the proposed model, the test set and validation set are then used. The proposed model has five phases, which are depicted in Fig. 5. Data pre-processing is the first stage. Data augmentation is the second stage. The third stage is feature extraction, which employs transfer learning and the pre-trained architecture DenseNet201. The fourth stage is categorizing plant leaf diseases using the obtained features and CNN classifier. Performance analysis and measurement make up the last stage. The first two stages of data augmentation and image pre-processing are carried out in the same manner as described in Section III.

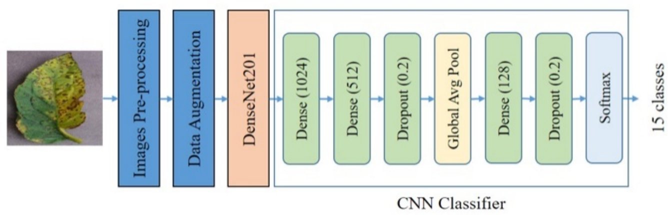


Fig. 5. The proposed model.

The DenseNet201 model, which employs transfer learning to automatically extract features and leverages their weights learned on the ImageNet dataset to reduce calculation workload, was proposed in the third phase. The architecture of DenseNet201 enables the construction of simple and straightforward models. Additionally, it is possible to reuse features across layers, increasing the efficiency of the architecture's parameters and allowing for greater variation and better performance in later layers. The architecture uses a feed-forward method to connect every layer to every other layer. The DenseNet201 model also makes use of a pooling layer and a bottleneck structure. This architecture becomes more effective as a result of reducing model complexity and property parameters. Convolution (Conv), pooling, rectified linear units (ReLU), and batch normalization are among the nonlinear transformations that are implemented in each layer of the DenseNet201 network (BN) (20). There are L(L+1)/2 connections in an L-layer DenseNet201 network because, unlike other networks, the output of each layer is used as the input for each succeeding layer (i.e., X0, X1, X2, X3 and X4) (20). The DenseNet201 architecture used in this work has 707 layers and around 20 million parameters. Table 3 displays the parameter counts of the various models employed in this investigation. The input layer's image dimensions are set to 224×224×3. Fig.6 shows the architecture of DenseNet201.

Six layers are suggested for the classification task in the fourth phase, which removes the DenseNet201 network's classification output layers. The architecture of the proposed model based on DenseNet201 is shown in Fig. 6. A Rectified Linear Unit (ReLu) serves as the activation function in the first layer's dense layer, which has 1024 neurons. The second layer, which has 512 neurons and activation ReLu, is also dense. The third layer is a dropout layer with a dropout rate of 0.2, which means that 20% of the neurons will output 0. This is done to prevent overfitting. A global average pooling layer for shrinking the size of feature maps makes up the fourth layer. A dense layer with 128 neurons and activation ReLu is in the fifth layer. A dropout layer with a dropout rate of 0.2 makes up the sixth layer. With 15 neurons and Softmax activation function, the final layer is dense. The 15 classes of plant diseases were output by the final layer. In the section that follows, we examine the outcomes of our suggested model in further detail and evaluate it against other transfer learning-based models.

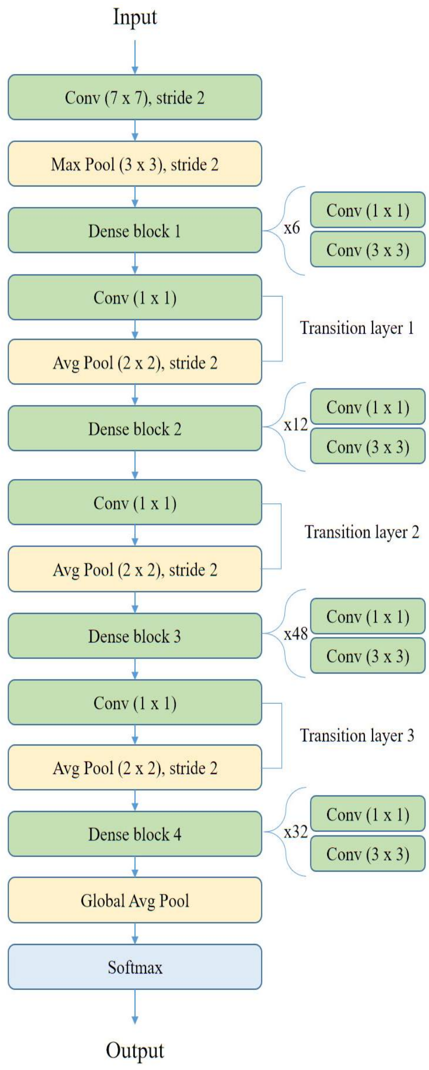


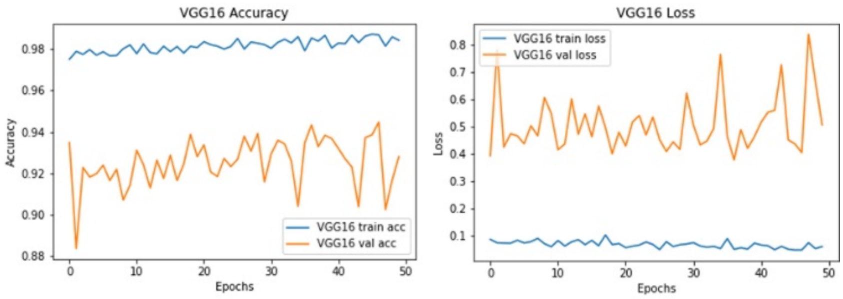
Fig. 6. DenseNet201 architecture.

V. Experimental and Discussions

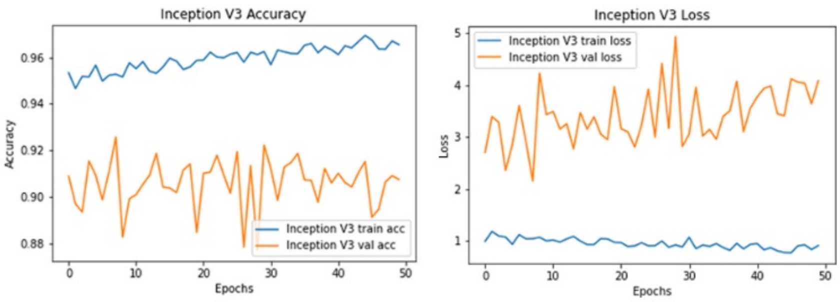
The open-source web programme Jupyter Notebook (21) is used to carry out our experiment. It contains the coding for numerous algorithms for both feature extraction and classification. It also contains code for machine learning, statistical modelling, data visualisation, data cleaning and transformation, and much more. The computer used to carry out this research has an NVIDIA GeForce RTX 2060 graphics card with a dedicated 6.0 GB of RAM and 1920 CUDA Cores. Processor: 2.60GHz 2.59GHz Intel(R) Core(TM) i7-10750H CPU. 16.0 GB of memory.

In our experiment, we first downsized plant leaf photos from the plantvillage collection to 24 x 24. The augmentation was then carried out. For faster training and improved accuracy, we employed the weights from ImageNet (20). We employed the Adam optimizer, the softmax activation function, and a 32-batch size. The learning rate and other variables were left at their default settings.

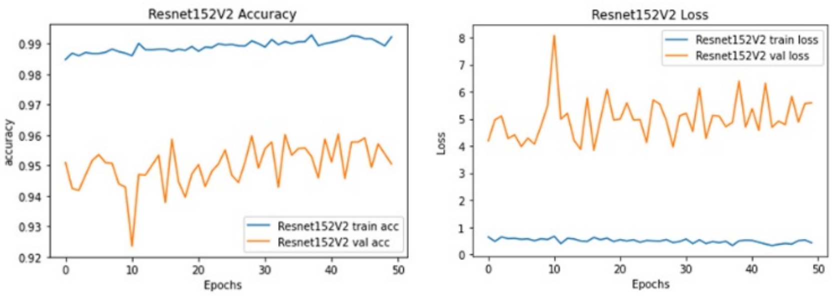
Then, we classified plant diseases using four CNN models—VGG16, Inception V3, ResNet152V2, and DenseNet201—along with a transfer learning technique, and we contrasted them with our suggested model. The accuracy and loss of several models are displayed in Fig. 7.



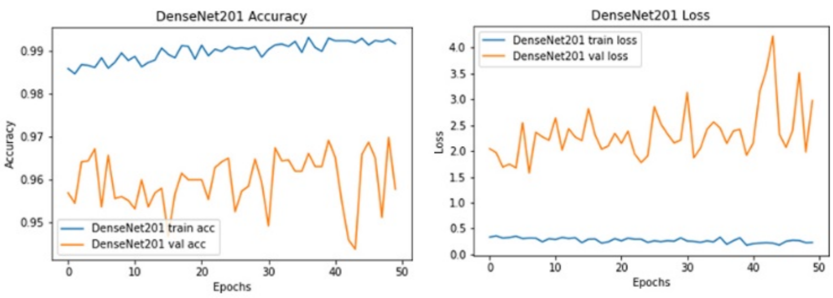
(a) VGG16 Accuracy & Loss



(b) Inception V3 Accuracy & Loss



(c) ResNet152V2 Accuracy & Loss



(d) DenseNet201 Accuracy & Loss

Fig. 7. Accuracy and loss of different models.

TABLE III: RESULTS OF DIFFERENT USED MODELS

Model	Parameters (M)	Train. Accuracy	Val. Accuracy	Train. Loss	Val. Loss
VGG16	15	0.9869	0.9447	0.0466	0.4049
Inception V3	22.3	0.9560	0.9258	0.9437	2.1475
Resnet152V2	59.34	0.9909	0.9603	0.4428	4.5758
DenseNet201	19.30	0.9910	0.9698	0.3089	1.9804
Proposed Model	20.88	0.9932	0.9797	0.0230	0.0898

Table III displays the outcomes of these tests. It demonstrates the remarkable correctness of the model we've suggested. It earned the best validation accuracy of 97.97% and the highest training accuracy of 99.32%.

To improve the findings, the five models' acquired results are examined. However, two factors—validation accuracy and confusion matrix—are used to evaluate the architecture. How precisely the trained model follows the trained data is referred to as validation accuracy. On the other hand, the total of each column in a confusion matrix represents the false positive rate (FP), and the amount for each row is the false-negative rate (FN) for each class. The sum of the other diagonal numbers represents the precise negative rate (TN), while the diagonal numbers represent the exact positive rate (TP). However, (1), (2), (3) and (4) are used to calculate an architecture's accuracy, precision, recall, and F1 score.

$$Accuracy = \frac{T_P+T_N}{T_P+T_N+F_P+F_N} \quad (1)$$

$$Precision = \frac{T_P}{T_P+F_P} \quad (2)$$

$$Recall = \frac{T_P}{T_P+F_N} \quad (3)$$

$$F1\_Score = \frac{2 \times Recall \times Precision}{Recall+Precision} \quad (4)$$

An established performance metric that is used to assess the effectiveness of the classifier is classification accuracy. Where FP (false positive)-misclassified negative samples, FN (false negative)-misclassified positive samples, TN (true negative)-misclassified negative samples, and TP (true positive)-correctly classified positive samples.

Performance metrics including Classification Accuracy, Precision, Recall, and F1-Score are produced to assess the effectiveness of various models, as shown in Table IV. The accuracy and loss of the suggested model are displayed in Fig. 8. The test data set's results are essentially identical to those of the validation data set.

We employed the DenseNet201 model, which outperformed other models in terms of transfer learning classification accuracy, as well as other characteristics of DenseNet stated in section 4 to extract features for our suggested model for tomato disease identification. According to Table III, the DenseNet201 model has fewer training parameters than the Resnet152V2 model, which results in improved accuracy and has an impact on the model's size and training time.

DenseNet201 in our instance has 707 layers. The ability to train all layers, some layers, or just the top layer is one advantage of the DenseNet201 model architecture. As we did in the initial phase, the outcomes are displayed in Table III.

We improved the training accuracy to 99.44%, the validation accuracy to 98.70%, and the testing accuracy to 99% by retraining almost half of the DenseNet201 model layers—300 layers—during the features extraction phase. Training time per step was 309.098 seconds, and validation loss was 0.0866. The outcomes of the proposed model after retraining various layers are shown in Table V.

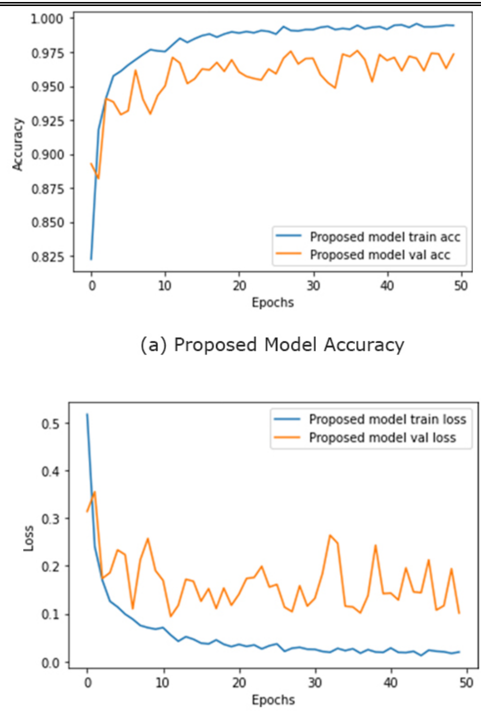
TABLE IV: ACCURACY, PRECISION, RECALL AND F1 SCORE FOR DIFFERENT USED MODELS

Model	Accuracy	Precision	Recall	F1 Score
VGG16	0.9447	0.94	0.93	0.93
Inception V3	0.9258	0.93	0.92	0.92
Resnet152V2	0.9603	0.95	0.95	0.95
DenseNet201	0.9698	0.96	0.96	0.96
Proposed Model	0.9797	0.97	0.97	0.97

The final trained and tested model was saved as h5 file format, which is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data, for later usage in prediction application (22). We developed a web based application that load the detection model h5 file and provide the farmer with the capability to upload image of defected leaf and the application convert that image into the appropriate format and scaling the image to 224×224. Then the model predicts the image and gives a class of the image that represent the diseases detected.

TABLE V: PROPOSED MODEL PERFORMANCE VALUES AFTER TRAINING SOME LAYERS OF BASE MODEL

Performance Metrics	Value
Training Accuracy	99.44 %
Validation Accuracy	98.70 %
Validation Loss	0.0866
Training time per step	309.098 seconds
Testing Accuracy	99.0 %
Precision	0.99
Recall	0.99
F1 Score	0.99



(a) Proposed Model Accuracy

(b) Proposed Model Loss

Fig. 8. Accuracy and loss of Proposed Model.

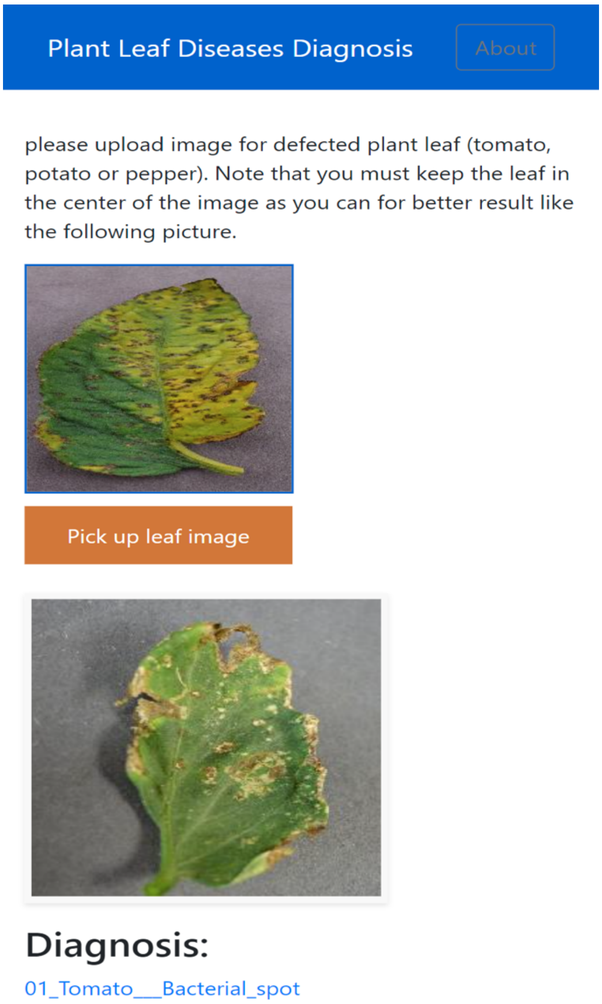


Fig. 9. Plant diagnosis application home page.

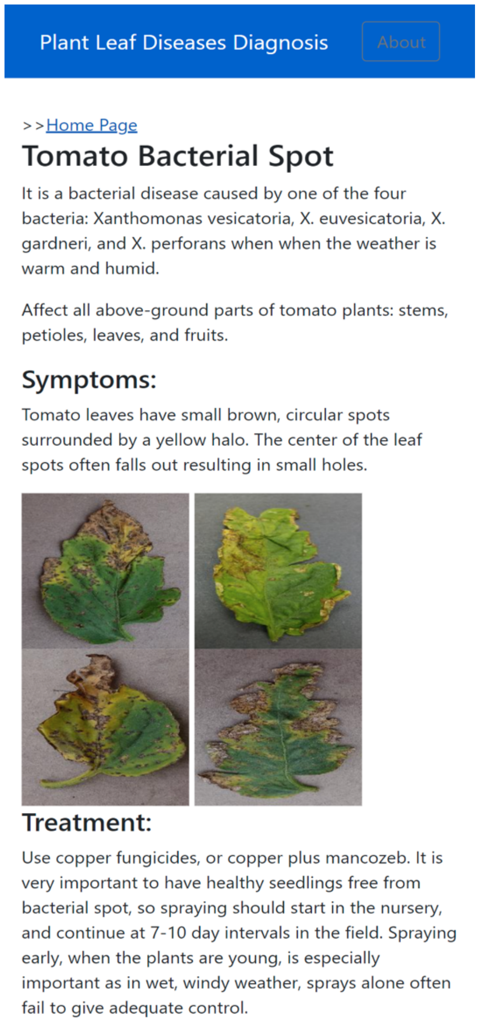


Fig. 10. Plant diagnosis application results and treatment recommendation.

The application after that search a database of diseases and get the information of the detected disease and show them to the farmer. The information about detected diseases that shown to the farmer by the application include the name – title- of the disease, causes, symptoms, type of the disease (fungal, bacterial, viral, others), some images of disease and treatment of that disease. The web based application developed by python library named flask and the database of diseases information is a collection of html pages enhanced with images of disease symptoms. The treatment of disease may be chemical spray and/or some other recommendations. Fig.9 show the main page of the application while Fig. 10 shows the result after some image detection done.

VI. Conclusion

To choose the best deep CNN model to use in the detection of plant leaf diseases, comparison research has been done in this paper. Utilizing the plant leaf disease data set as training and testing data, four deep CNN models—DenseNet201, VGG16, Inception V3, and ResNet152V2—were used. To save time and effort when training these models, transfer learning techniques are used. The data set of 3 crops, which included photos of tomato, potato, and pepper leaves, was divided into three sections: training, validation, and testing, with each section receiving 25% of the data set. Table III displays the findings for every instance. On the basis of transfer learning and DenseNet201, we also suggested a classification model. The DenseNet201 model functions in our suggested model as the features extraction phase, which is followed by a CNN classifier. The outcomes demonstrate that the suggested model provides the greatest accuracy. By training certain additional model layers during transfer learning in addition to the top layer, we apply additional fine-tuning. The results also demonstrate that there are differences between the parameters and average accuracy of the five convolutional neural networks. The most accurate model is the one we've suggested. After that we developed a web based application that uses the proposed model to help farmers in diagnosing defected plant by uploading image and get the right diagnosis of the disease and also the recommended treatment in addition to more information about the disease.

We intend to broaden our research in the future by using other CNNs that have already undergone multi-classification training. Adapt our suggested model to a wider range of plants and diseases. Update the developed tool to be more accurate in the treatment section and take in consideration some crop conditions and weather data for better diagnosing plant diseases that will benefit Egyptian farmers.

References

[1] Plant health and food security [Internet]. FAO; 2017. Available from: http://www.fao.org/3/a-i7829e.pdf.

[2] Mohamed A, Abdel-Gaber S, Nasr M, Hazman M. An Intelligent Approach to Mitigate Effects of Climate Change and Insects on Crops. *International Journal of Computer Science and Information Security (IJCSIS)*. 2020; 18(3).

[3] Venkatesh, Y N, S ST, S S, Hegde SU. Transfer Learning based Convolutional Neural Network Model for Classification of Mango Leaves Infected by Anthracnose. In: *2020 IEEE International Conference for Innovation in Technology (INOCON)*. 2020: 1–7.

[4] Peyal HI, Shahriar SM, Sultana A, Jahan I, Mondol MdH. Detection of Tomato Leaf Diseases Using Transfer Learning Architectures: A Comparative Analysis. In: *2021 International Conference on Automation, Control and Mechatronics for Industry 40 (ACMI).* 2021: 1–6.

[5] kaggle [Internet]. 2018. Available from: https://www.kaggle.com/noulam/tomato/download.

[6] Hong H, Lin J, Huang F. Tomato Disease Detection and Classification by Deep Learning. In: *2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE).* 2020: 25–9.

[7] Agarwal M, Singh A, Arjaria S, Sinha A, Gupta S. ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network. *Procedia Computer Science.* 2020;167:293–301.

[8] Kabir MM, Ohi AQ, Mridha MF. *A Multi-Plant Disease Diagnosis Method using Convolutional Neural Network. CoRR* [Internet]. 2020;abs/2011.05151. Available from: https://arxiv.org/abs/2011.05151.

[9] Rangarajan AK, Purushothaman R, Ramesh A. Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia Computer Science.* 2018;133:1040–7.

[10] Ji M, Zhang L, Wu Q. Automatic grape leaf diseases identification via UnitedModel based on multiple convolutional neural networks. *Information Processing in Agriculture.* 2020;7(3):418–26.

[11] Afifi A, Alhumam A, Abdelwahab A. *Convolutional Neural Network for Automatic Identification of Plant Diseases with Limited Data. Plants* [Internet]. 2021;10(1). Available from: https://www.mdpi.com/2223-7747/10/1/28.

[12] Saleem MH, Potgieter J, Arif KM. *Plant Disease Detection and Classification by Deep Learning. Plants* [Internet]. 2019;8(11). Available from: https://www.mdpi.com/2223-7747/8/11/468.

[13] Chen J, Chen J, Zhang D, Sun Y, Nanehkaran YA. Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture.* 2020;173:105393.

[14] Shorten C, Khoshgoftaar TM. A Survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*. 2019 Jul 6;6(1):60.

[15] Too EC, Yujian L, Njuki S, Yingchun L. A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture.* 2019;161:272–9.

[16] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: Bengio Y, LeCun Y, editors. 3rd *International Conference on Learning Representations, ICLR*. 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings [Internet]. 2015. Available from: http://arxiv.org/abs/1409.1556.

[17] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016. p. 2818–26.

[18] Gulli A, Pal S. *Deep Learning with Keras. Birmingham,* United Kingdom: Packt; 2017: 318.

[19] Kumar V, Arora H, Harsh, Sisodia J. ResNet-based approach for Detection and Classification of Plant Leaf Diseases. In: *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC).* 2020. p. 495–502.

[20] Huang G, Liu Z, Weinberger KQ. *Densely Connected Convolutional Networks.* CoRR [Internet]. 2016;abs/1608.06993. Available from: http://arxiv.org/abs/1608.06993

[21] jupyter.org [Internet]. 2021. Available from: https://jupyter.org.

[22] Aquiles C. *How to use HDF5 files in Python* [Internet]. python for the lab. [cited 2022 Aug 1]. Available from: https://www.pythonforthelab.com/blog/how-to-use-hdf5-files-in-python/.