


Identification of Tomato Leaf Diseases Using Deep Convolutional Neural Networks


Ganesh Bahadur Singh, National Institute of Technology, Jalandhar, India

Rajneesh Rani, National Institute of Technology, Jalandhar, India

 <https://orcid.org/0000-0003-2104-227X>

Nonita Sharma, National Institute of Technology, Jalandhar, India

Deepti Kakkar, National Institute of Technology, Jalandhar, India

 <https://orcid.org/0000-0002-9681-1291>

ABSTRACT

Crop disease is a major issue as it drastically reduces food production rate. The tomato is cultivated in most of the world. The most common diseases that affect tomato crops are bacterial spot, early blight, Septoria leaf spot, late blight, leaf mold, target spot, etc. In order to increase the production rate of tomato, early identification of diseases is required. The existing work contains a less accurate system for identification of tomato crop diseases. The goal of the work is to propose a cost effective and efficient deep learning model inspired from Alexnet for identification of tomato crop diseases. To validate the performance of proposed model, experiments have also been done on standard pretrained models. The plantVillage dataset is used for the same, which contains 18,160 images of diseased and non-diseased tomato leaf. The disease identification accuracy of the proposed model is compared with standard pretrained models, and it is found that proposed model gave more promising results for tomato crop disease identification.

KEYWORDS

Computer Vision, Convolutional Neural Networks, Image Augmentation, Pretrained Models, Tomato Leaf Diseases

1. INTRODUCTION

The crop of Tomato is mostly cultivated in a wide area of the globe as it contains three major antioxidants vitamin E, beta-carotene, and vitamin C. In India, the area of cultivation spans approximately around 3,50,000 hectares and became the third-largest producer in the globe (Tm, Prajwala, et al., 2018; Ireri, David, et al., 2019). Tomato leaf diseases cause a major loss in quality and production rates. The diseases in tomato crops are mainly in leaves, stems, and roots. Commonly, the diseases in crops are due to fungi, viruses, and bacteria. Some common diseases that affect tomato crops are early blight, septoria leaf spot, two-spotted spider mite, target spot, bacterial spot, mosaic virus, late blight, curl virus, etc. (Durmus, Halil, et al., 2017). It is very difficult for a human to recognize an accurate class of diseases. Incorrect prediction of diseases causes inaccurate usage of pesticides, which causes loss of quality and production of food. Hence, the recognition of diseases plays a major role in the field of agriculture.

DOI: 10.4018/IJAEIS.20211001.0a3

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

There are various techniques available for crop disease recognition. One of the methods is recognition by the farmer under the supervision of an agricultural expert, which is a time-consuming and expensive methodology (Kamilaris, Andreas, et al., 2018). With in time various technologies like machine learning, computer vision and artificial intelligence has been used for crop disease recognition. Machine learning-based recognition algorithms involve major two steps that are feature extraction and classification (Agarwal, Mohit, et al., 2020). The feature from an image is extracted using an appropriate feature extractor. In classification problems mostly supervised learning classification algorithm is used. Machine learning techniques are applied in various area of the agricultural field like, classification of guava, jamun, tomato, mango, grapes, and apple plants using random forest and support vector machine (SVM) algorithms through its leaf images (Kour Vippon Preet et al., 2019), potato crop diseases identification using multiclass support vector machine (Islam, Monzurul et al., 2017), grape leaf diseases recognition using k-nearest neighbor, support vector machine, and random forest (Krithika N. et al. 2017; Sandika Biswas et al., 2016; Padol, Pranjali, et al. 2016), and recognition of wheat leaf diseases using support vector machine (Nema et al. 2018).

Actually, Machine learning techniques have various shortcomings in image classification. One of the major shortcomings is manually feature extraction from an image and thereafter, classification using extracted features (Rangarajan, Aravind, et al., 2018; Kaur, Sukhvir, et al., 2019). For a particular problem, selecting feature extractor and classifier separately is a time consuming and tough work. Hence, the concept of deep learning techniques comes into the picture to overcome the above-mentioned issue. The popularity of deep learning techniques is due to the effort required for developing, maintaining, and controlling these models is less (Meng, Xiangyan, et al., 2020). Automatic feature extraction is the most important feature of the deep learning model. Deep learning model is composed of the main four processing layers namely convolution, pooling, flattening, and fully connected layer. The deep learning has been research hotspot in various areas like agriculture, medical imaging, object detection, etc. In the field of agriculture, researchers have applied in various applications such as for, apple plant diseases identification through its leaf images (Jiang, Peng, et al. 2019; Jiang, Bo, et al. 2019), automatic recognition and severity prediction of pepper bacterial spot disease(Wu, Qiufeng, et al. 2020), detection of cherry plant leaf diseases (Zhang, Keke, et al. 2019), on-field detection of weed (Fu, Lifang et al. 2020), fungus recognition in crops (Tahir, Muhammad Waseem et al. 2018), detection of tomato crop disease (Agrawal, Mohit, et al. 2020), and identification of maize crop diseases through leaf image (Priyadharshini Ramar Ahila et al. 2019; Zhang, Xihai, et al. 2018).

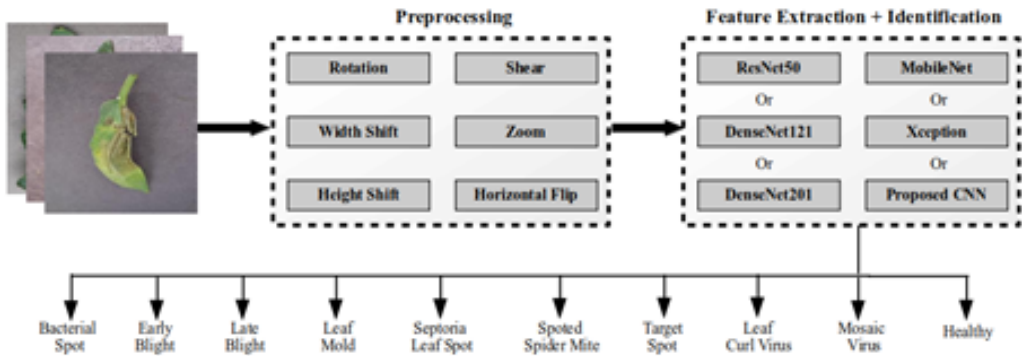
In tomato, common diseases that affect leaves are bacterial spot, curl virus, late blight, early blight, mosaic virus, spotted spider mite, septoria leaf spot, leaf mold, and target spot. For developing a diseases identification system following factors makes this task challenging: first, diseases spot size in the leaf may differ among diseases or for the same disease. Furthermore, the same leaf can have multiple diseases. Also, some disease spots in tomato leaves are too small. At last, the atmosphere component also intercepts with the tomato crop disease recognition system. In previous research work, researchers have used basic convolutional neural networks, which identification accuracy is very low. Hence, this research work proposed a deep convolutional neural network-based model for the identification of tomato leaf diseases. The objective of this paper is to develop an efficient and accurate deep convolutional network for tomato crop disease identification. It will help farmers to identify diseases in tomato crops at an early stage in an efficient manner. Hence, diseases can be recovered at an early stage by using suitable pesticides.

The remaining section of this article is arranged as follows: The data preprocessing and deep convolutional neural network-based models are briefly discussed in section 2. In section 3 the proposed approach is evaluated and the experimental results are explored. Finally, the work is summarized in section 4.

2. METHODOLOGY

There are various techniques available for crop disease identification, but deep learning techniques are mostly used now a day as discussed in the introduction section. Therefore, this study uses a deep convolutional neural network for tomato crop diseases identification. The overall system for tomato crop disease identification involves three major steps namely data preprocessing, feature extraction, and identification. Each step is briefly discussed in this section. Figure 1 shows the workflow diagram of this work.

Figure 1. Workflow diagram



2.1 Data Preprocessing

The preprocessing technique is used to enhance the image data. The deep learning model trained on enhanced images can improve the identification accuracy of the model and overcome it from overfitting (Jiang, Peng, et al., 2019). If every image has different variants then, the model can learn more irrelevant patterns during the training phase. The expansion of the dataset by creating different variants of the existing image is called data augmentation. In this step; we have used width shift, height shift, rotation, horizontal flip, shear, and zoom as a data augmentation parameter. We have chosen a parameter range between 0.1-0.5 that is maximum 5% by assuming that this range will not actually change the shape of the image. Then we tested the performance of the model with a different combination of values in the given range. And the values of each parameter on which the model performed better are listed in Table 1.

A shift in an image refers to moving pixels in one direction either towards height or width, while the dimension of an image remains the same. A horizontal flip means exchanging row pixels with column pixels and vice-versa. Images are randomly rotated in a clockwise direction using a rotation parameter. Zoom parameter in an augmentation process randomly zoom in an image or insert new pixels around it. The original images are resized to 128×128 pixels to minimize training time. Original images are in RGB format. RGB coefficients range from 0 to 255, which will be higher and complex for our model to learn. Therefore, coefficients are rescaled between 0 and 1 by multiplying each pixel with 1/255.

2.2 Pretrained Deep CNNs

This section discusses standard pre-trained deep convolutional neural networks, which have been used to compare the performance of the proposed deep convolutional neural network. The pre-trained models are already trained on a large benchmark dataset 'ImageNet'. These models don't need to train

Table 1. Data augmentation parameter details

Parameter	Value
Rotation range	5
Width shift range	0.1
Height shift range	0.1
Shear range	0.2
Zoom range	0.1
Fill mode	'nearest'
Horizontal flip	True

from scratch. Some of the well-known pre-trained DCNNs models are AlexNet, VGG, DenseNet, MobileNet, and Xception. These models are differing by their configurations, kernel size, depth, and a number of neurons. This study uses top-5 DCNNs models having higher identification accuracy on the tomato leaf dataset. The top-5 DCNNs having higher identification accuracy are ResNet50, DenseNet121, DenseNet201, MobileNet, and Xception, which are discussed in detail in this section.

2.2.1 ResNet50

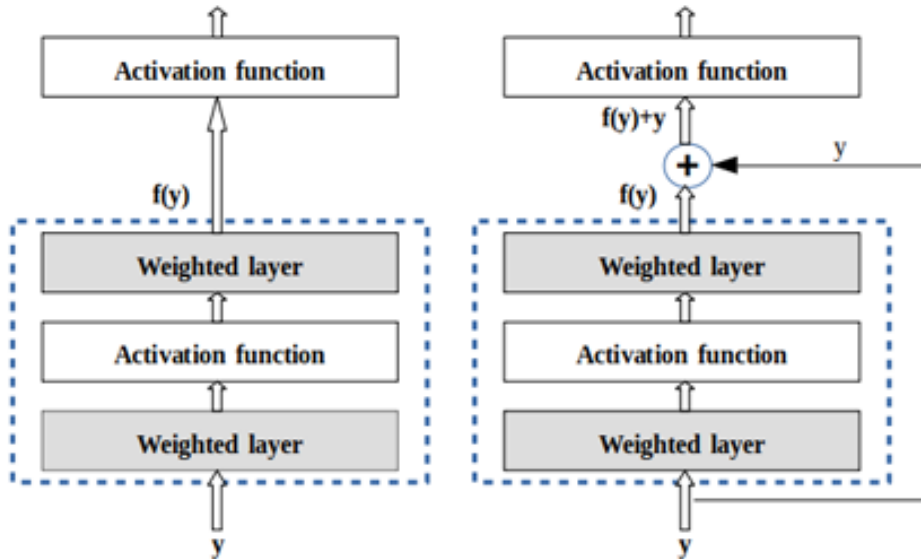
ResNet was introduced in 2015 by the Microsoft team. In deep networks vanishing gradient problem is major issue while training the model (He, Kaiming, et al., 2016). As the network becomes deeper, it starts converging. Hence, after some instant of time accuracy becomes saturated and starts decreasing. Also, training error starts increasing after some instant of a point. To overcome all issues, the activation unit is fed into a deeper layer, known as skip connection or shortcut connection. ResNet is based on a skip connection. Figure 2 shows the difference between the basic block and residual block of a deep convolutional neural network.

The residual network was evaluated on the imageNet dataset. The residual networks up to 152 layers were evaluated. The deepest residual network has lower complexity than the VGG network. The network achieved less than a 4% error on test data, which is less than human visualization error. A ResNet50 network is divided into five stages. The first stage contains convolution, batch normalization, a ReLu, and a max-pooling layer (Koay, Kah Leong, et al., 2020). The remaining four stages consist of one convolution block and one identity block. The convolution block contains three convolution layers, each followed by a batch normalization layer. There is a skip connection from the input of the first convolution layer to the ReLu layer of the third convolution layer. The identity block also contains three convolution layers, only difference is in skip connection. The architecture begins with zero padding layers. The zero-padded input is then fed into the first stage of the architecture. The ResNet50 architecture has been used in the various field for image classification like brain tumor disease classification, wall break classification, plant disease classification, blood cell classification, face disease classification, skin cancer detection, etc (Kumar, Ashnil, et al., 2016).

2.2.2 DenseNet

DenseNet stands for dense convolutional network. It was presented by Gao Huang in 2016. In the deeper network information passes through many layers so, they can vanish before reaching the end of the network. In comparison to ResNet, DenseNet uses the concatenation of features instead of summation before passing it to layer (Huang, Gao, et al., 2017). Each layer feature-maps are passed into all successive layers. In N-layer dense network n^{th} layer holds feature-matrix of all n previous convolution layers and its own features are forwarded to all $N-n$ successive layers. Therefore, the total number of connections in the N-layer network is $N(N+1)/2$. DenseNet has fewer parameters than

Figure 2. Difference between basic block(left) and residual block(right) of CNN (He et al., 2016)



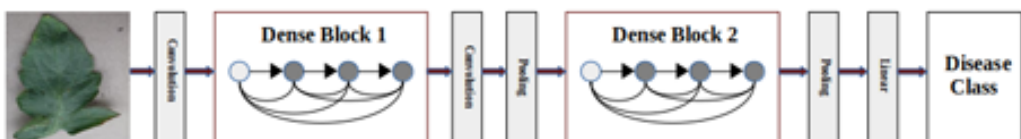
ResNet as it doesn't use the redundant feature map. The DenseNet network uses 1×1 convolutions called the bottleneck layer before each 3×3 convolution to reduce the feature-map size and enhance computational efficiency (Kamilaris, Andreas, et al., 2018). The DenseNet architecture is divided into adjacent dense blocks and transition blocks. The transition layer contains a batch normalization layer and convolution layer of size 1×1 followed by an average pooling layer of size 2×2 . The DenseNet with two dense blocks is shown in Figure 3.

The major advantages of DenseNet are improvement in the flow of gradient and feature-map inside the networks. It also reduces the number of parameter by reusing features. This paper uses two versions of dense networks namely DenseNet121 and DenseNet201.

2.2.3 MobileNet

MobileNet architecture was introduced by Google which is appropriate for mobile vision applications like text recognition, object detection, object tracking, etc. (Howard et al., 2017). It is a lightweight architecture and uses depth-wise separable convolutions. The depth-wise separable convolutions consist of two convolution operations namely depth-wise convolution and pointwise convolution. The Depth-wise convolution performs a single convolution operation per input channel. It uses filter

Figure 3. The Dense Net containing two dense blocks



size of 3x3. Depth-wise convolution operation with a single convolution per input channel can be represented using the following equation (1):

$$O_{a,b,c} = \sum_{i,j} L_{i,j,c} \cdot G_{a+i-1,b+j-1,c} \quad (1)$$

Where L represents the depth-wise convolution kernel of size $S_a \times S_a \times C$ and G is the input feature-map of the convolution layer. The above equation shows that the c^{th} filter of L is applied to c^{th} channel in G to create c^{th} channel feature map O . The pointwise convolution is a simple 1×1 convolution, which generates new features by combining the output feature matrix of a depth-wise convolution linearly. Both depth-wise convolution and pointwise convolution is followed by batch normalization and ReLu activation function (Elhassouny, Azeddine, et al. 2019). The MobileNet network was assembled using depth-wise separable convolution except for the first layer which uses full convolution. Considering depth-wise and pointwise convolution as separate layers, MobileNet contains 28 layers. MobileNet is used in various applications like image classification, object detection, etc (Alarifi, Jhan, et al., 2017).

2.2.4 Xception

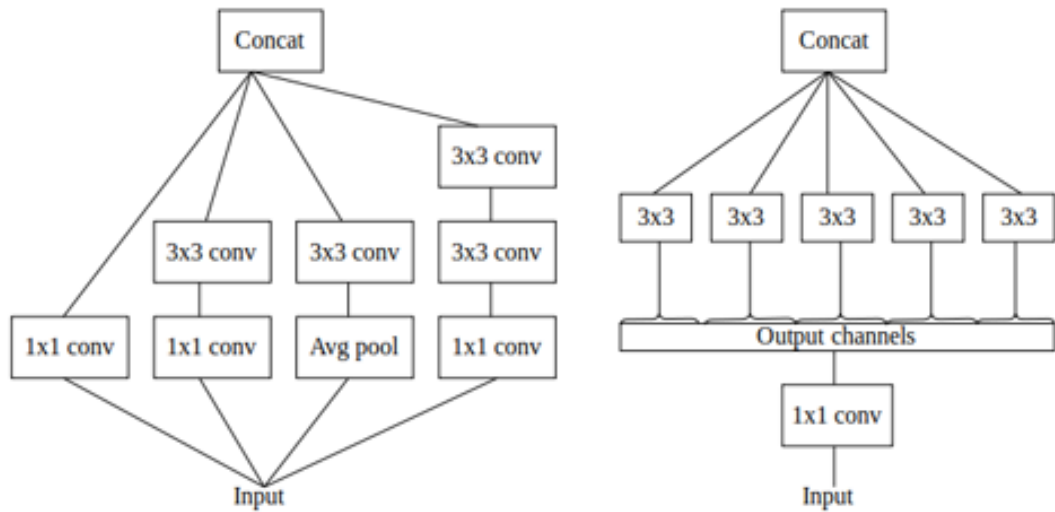
Xception architecture is a modified version of InceptionV3, which replaces the Inception module with a modified depth-wise separable convolution called extreme inception. It was introduced by Francois Chollet in 2017. The numbers of parameters in xception architecture are approximately similar to InceptionV3 (Chollet, Francois, et al., 2017). The word xception is a short form of extreme inception. The performance improvement of xception module is due to the systematic use of model parameters. The extreme inception is similar to depth-wise separable convolution with minor two changes. The first changes is in a sequence of convolution operation, depth-wise separable convolution block first performs channel-wise 3x3 convolution and after that 1x1 pointwise convolution is performed, while extreme inception first performs 1x1 pointwise convolution. And the second difference is, in inception both channel-wise and point operation is followed by the ReLu activation function, while depth-wise convolution generally not uses any activation function (Krisnandi, D. et al., 2019; Kamal, K.C., et al., 2019). Figure 4 shows the difference between the InceptionV3 module and the extreme inception block.

The Xception architecture contains 36 convolutional layers, which are splitted into 14 modules. Each module are connected through linear residual connection except the first and last modules. The performance of the xception model is evaluated on ImageNet and JFT dataset. As compare to Inception V3, the Xception model shows better classification accuracy on both ImageNet and JFT dataset. It has better accuracy gain on the JFT dataset as compared to the ImageNet dataset. It also performs better than ResNet-50, ResNet101, and ResNet152.

2.3 Proposed Work

This research work proposed a deep convolutional neural network model to recognize different types of diseases in tomato crops. The proposed model is influenced by the standard pre-trained model AlexNet. According to a review of the state of the art model for tomato disease identification, AlexNet has higher identification accuracy, but the number of parameter is more. Hence, AlexNet is considered as the basic network for this work. This work improves the recognition accuracy of model and minimizes the network parameter. The AlexNet model is less deep than ResNet, DenseNet, Inception, MobileNet, and Xception, but the number of parameters is much higher than these models. So, the proposed model minimizes the number of parameters by changing number of neurons, kernel size and number of kernels, and rearranging max-pooling and convolution layers. The AlexNet and proposed model have been briefly discussed in this section.

Figure 4. Difference between Inception V3 block (left) and extreme inception block (right)



2.3.1 AlexNet

The AlexNet model was proposed by Krizhevsky in 2012. The AlexNet was the starting point of the craze of convolutional neural networks (Krizhevsky, Alex, et al. 2012). It won the ImageNet challenge with a large difference in error rate, which majorly impacts the machine learning techniques for image classification. Figure 5 shows the architecture of the AlexNet network.

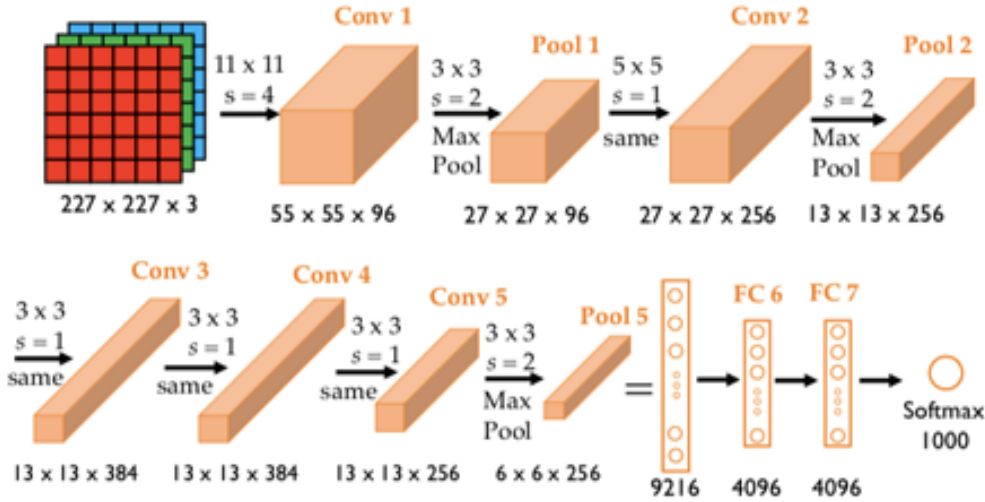
It consists of convolution, max-pooling, and ReLu activations. It uses a kernel size of 11×11 , 5×5 , and 3×3 . The architecture of AlexNet contains a total of eight weight layers, which includes five convolution and three dense weighted layers. The resultant feature-map of the fifth convolution layer becomes an input to the first dense layer. Every fully connected layer is followed by a dropout layer to overcome overfitting. Each convolution and a max-pooling layer is followed by the ReLu activation function. The last dense layer uses a softmax activation function which calculates the probability for each class label. The AlexNet model uses a stochastic gradient descent (SGD) optimizer with momentum.

2.3.2 Proposed Deep CNN

The proposed model consists of convolution, Maxpooling, batch normalization, flattening, dropout, and fully connected layer. Figure 6 shows the detailed configuration of the proposed deep convolutional neural network.

The main aim of convolution operation is to extract features like corners, edges, and colors from an image. The convolution operation is performed by continuous sliding of filter (kernel) over image pixels and taking the dot product of the corresponding pixel of filter and input image pixel. The proposed model contains six convolution layers with a kernel size of 3×3 and each followed by a rectified linear unit (ReLU) activation function. The ReLU activation function has been used to make the input neuron capable of learning more complex and complicated features. The ReLU activation function also rectifies the vanishing gradient problem. Due to the increase in number of convolution layers, the network parameter increases exponentially. So, pooling is performed to decrease the dimension of the feature-map. It extracts essential features from the feature map by removing non-essential features. The proposed model uses max-pooling, due to its better performance and greater convergence. The

Figure 5. AlexNet architecture (Krizhevsky, Alex, et al. 2012)



max-pooling is done by simply taking the max value in the pooling window. The training of a deep convolutional neural network is a challenging task due to the overfitting problem. There are mainly two ways to overcome from over fitting namely regularization and dropout operation. Therefore, batch normalization has been used to include the regularization effects in the network. It also boosts the training speed and performance of the model. The batch normalization mainly standardizes the input by scaling it in a similar range. In the proposed network, batch normalization operation has been performed after every activation operation and a dense layer. The model also includes dropout operations with a dropout rate of 25%. Dropout operation refers to the deactivating some randomly chosen neurons during training. It means, temporarily deactivates neurons from the network and also its incoming and outgoing edges.

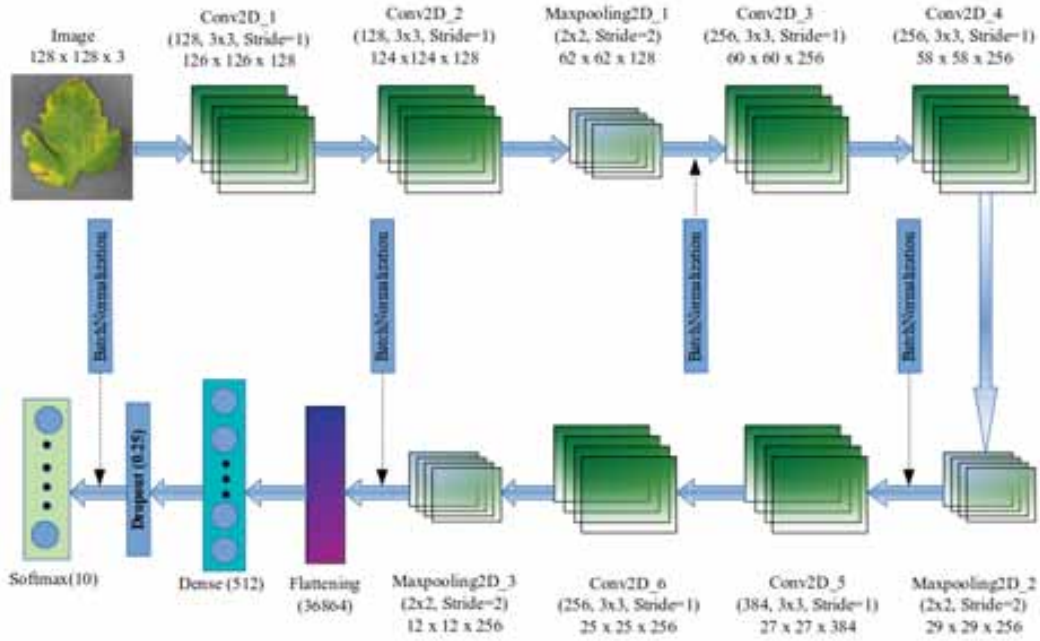
At last, the resultant feature matrix of the final pooling layer is flattened into a 1-d feature vector. Our proposed model consists of two dense layers. The first dense layer takes input as a 1-d feature vector and contains 512 neurons. It is followed by batch normalization and dropout layer. The output of this layer is passed to the second dense layer having 10 neurons and softmax as activation function, which acts as an output layer.

The proposed model contains a total of 21,691,146 parameters. Among total parameters, 21,688,842 parameters are trainable and the remaining 2304 parameters are non-trainable. The non-trainable parameters came from batch normalization. The batch normalization parameters are considered non-trainable because its mean and variance values are updated during layer updates instead of the back propagation process. For the input layer, the number of parameters is zero as it contains only the shape of input images. The number of parameters inside the convolution layer defined using filter size and count of filters used for the convolution process. Mathematically, the count of parameters for the convolution layer can be calculated using the following formula-

$$No.ofParameters = (W \times H \times N_p + 1) \times N_c$$

Where W indicates the width of filter, H indicates the height of filter, N_p shows several filters used in the previous layer, and N_c indicates the count of filters in the current layer. For each filter, 1 is added as a bias in the formula.

Figure 6. Detailed architecture of the proposed deep convolutional neural network



For the pooling layer number of parameters is zero as it doesn't involve in any backpropagation process. In comparison to the convolution layer, the count of parameters in a fully connected layer (Dense layer) is higher because each neuron is attached with all neurons in the next layers. Hence, when the count of the fully connected layer increases, then the count of parameters of the model increases rapidly. In a fully connected layer count of parameters depends on the count of neurons in the current layer and the previous layer. The number of parameter in a fully connected layer can be determined using the following formula-

$$No.ofparameters = R_c \times R_p + (1 \times R_c)$$

Where R_c indicates the count of neurons in the current layer and R_p represents the count of neurons in the previous layer. The value of R_p for the first fully connected layer will be the product of its previous layer output size. The value 1 indicates bias term in a given formula. The batch normalization by default takes 4 parameters per feature maps. Hence its parameter is calculated by multiplying 4 with a count of filters for the convolution layer and count of neurons for a fully connected layer. For first fully connected layer, count of parameters varies as the output shape of the previous layer changes. Hence the number of parameters also changes as the size of the input image changes. Table 2 detailed the parameter details of each layer for our proposed model. It also shows the input and output shape for each layer. For the convolution and pooling layer, output shape is calculated using a different method. The output shape of the convolution layer and pooling layer is determined using the following formula-

$$O_c = ((I - K + 2P) / S) + 1$$

$$O_p = ((I - P_s) / S) + 1$$

Where O_c represents output shape of the convolution layer, O_p indicates output shape of the pooling layer, I indicates input size, K indicates kernel size, N shows the count value of kernels, S shows the size of strides, and P determines types of padding used. In the stated deep convolutional

Table 2. Parameter details of the proposed model for each layer

Layer Name	Input Shape	Output Shape	# Parameters
Conv2D_1	(128, 128, 3)	(126, 126, 12)	3584
Conv2D_2	(126, 126, 128)	(124, 124, 128)	147584
Maxpooling2D_1	(124, 124, 128)	(62,62, 128)	0
Batch_Normalization_1	(62, 62, 128)	(62, 62, 128)	512
Conv2D_3	(62, 62, 128)	(60, 60, 256)	295168
Conv2D_4	(60, 60, 256)	(58, 58, 256)	590080
Maxpooling2D_2	(58, 58, 256)	(29, 29, 256)	0
Batch_Normalization_2	(29, 29, 256)	(29, 29, 256)	1024
Conv2D_5	(29, 29, 256)	(27, 27, 384)	885120
Conv2D_6	(27, 27, 384)	(25, 25, 256)	884992
Maxpooling2D_3	(25, 25, 256)	(12, 12, 256)	0
Batch_Normalization_3	(12, 12, 256)	(12, 12, 256)	1024
Flatten_1	(12, 12, 256)	36864	0
Dense_1	36864	512	18874880
Dropout_1	512	512	0
Batch_Normalization_4	512	512	2048
Dense_2	512	10	5130

neural network zero padding are used in convolution operation. The count of neurons in a dense layer is equal to its size. The shape of the batch normalization layer is equal to the previous layer size. And the shape of the dropout layer is equivalent to the number of neurons in the previous layer.

2.3.3 Proposed deep CNN advantages

The advantages of the proposed model over the standard pre-trained models are as follows-

- It contains less number of parameters, hence requires less time for training.
- There is very little chance of vanishing information before reaching to the output layer of the network as the proposed architecture is less deep.
- The batch normalization feature prevents over fitting and also makes the network training process faster.
- The error rate for prediction of disease is also very less as compare to other networks, while it has a lesser number of parameters.

3. RESULTS AND DISCUSSION

The plantVillage dataset is used for this research work to evaluate the performance of the proposed model. This section presents the dataset and experimental setup for this study. Thereafter, the results of models are compared using various plots. At last, the performance of models is compared and discussed.

3.1 Dataset

We need a suitable dataset at every step of our research work. So, we retrieved the plantVillage dataset from the publicly available repository plantVillage organization (<https://plantvillage.psu.edu/>). The plantVillage organization helps farmers working in a remote area. The dataset contains 18,160 images of diseased and healthy tomato leaves. Each image is of size 256×256 pixels. The images are labeled under the supervision of an agricultural expert, according to their disease categories (TM, Prajwala, et al., 2018). Images are divided into 10 classes, in which nine classes contain diseased leaf images and one class contains healthy images. Classes representing nine categories of diseases are bacterial spot, early blight, curl virus, septoria leaf spot, mosaic virus, target spot, spotted spider mite, late blight, and leaf mold. Figure 7 shows the sample images of various tomato leaf diseases.

3.2 Experimental Setup

The proposed methodology is implemented on the google cloud platform. Google provides free GPU resources for AI developers, which is known by Google Colab. Google Colab uses a jupyter notebook environment. The hardware specification of Google Colab is as follows: uses 1×Tesla K80 GPU, 13 GB GDDR5 VRam, and 33 GB disk space. The models are implemented using the Keras library and Tensorflow framework. The dataset used for this research work contains 18,160 images of 9 common tomato leaf diseases and a healthier one. The dataset is split into training and validation sample in the percentage split ratio of 75:25. Table 3 lists the count of training and testing sample details of each class label.

3.3 Accuracy and Loss Comparison

To compare the performance of the proposed deep convolutional neural network various standard pre-trained deep convolutional neural networks, ResNet50, DenseNet121, DenseNet201, MobileNet, and Xception are applied for tomato crop diseases identification. This study mainly uses the accuracy and loss parameter for the evaluation of models.

The models are trained on training image samples and tested on test image samples to compare the performance of models. Throughout the training process, Adam optimizer is used for the pre-trained model and stochastic gradient descent (SGD) with momentum is used for the proposed deep convolutional neural network.

The Adam optimizer is basically considered as a combination of stochastic gradient descent (SGD) and RMSprop with momentum. Like RMSprop, it uses a squared gradient to map the learning rate. And like stochastic gradient descent it uses the moving average of the gradient in place of using the gradient itself. The optimizer randomly selects a group of images for training, which is known as batch size. The batch size value depends on the capacity of resources used for training. This work uses a batch size value of 32. The proposed model uses a learning rate value of 0.01 with a momentum value of 0.09. And pre-trained models use learning rate value as 0.001. The momentum defines how fast gradients move towards the optimum point. To find the appropriate weight for the network, weight is updated using a back propagation algorithm.

Each model is trained for 200 epochs. The epoch defines the number of times the model will learn on training samples. The training and validation accuracy curve is plotted to visualize the performance of the model during training. Figure 8 represents the accuracy (training and testing) comparison of ResNet50, DenseNet121, DenseNet201, MobileNet, Xception, and proposed model for each epoch.

The X-axis of the accuracy plot represents epoch number (1-200) and Y-axis represents the accuracy values corresponding to each epoch. The loss plot of each model is shown in Figure 9. The X-axis of the loss plot represents the epoch number (1-200) and Y-axis shows the loss value corresponding epochs. When validation loss is much higher than training loss then, the network is considered as over fitting. And if training loss value is much greater than validation loss then, the network is considered as under fitting.

Figure 7. Sample images of the tomato leaf dataset

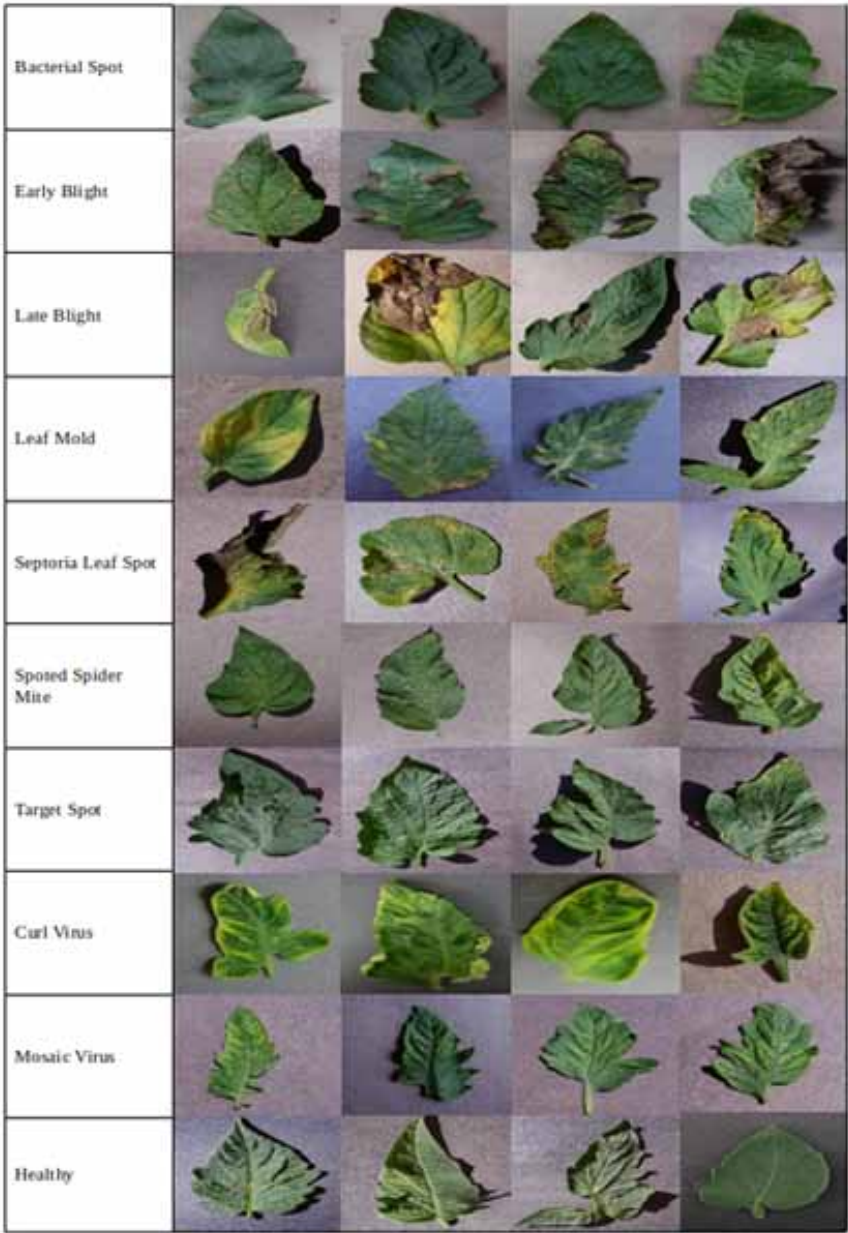


Table 3. Details of training and validation samples of tomato leaf diseases

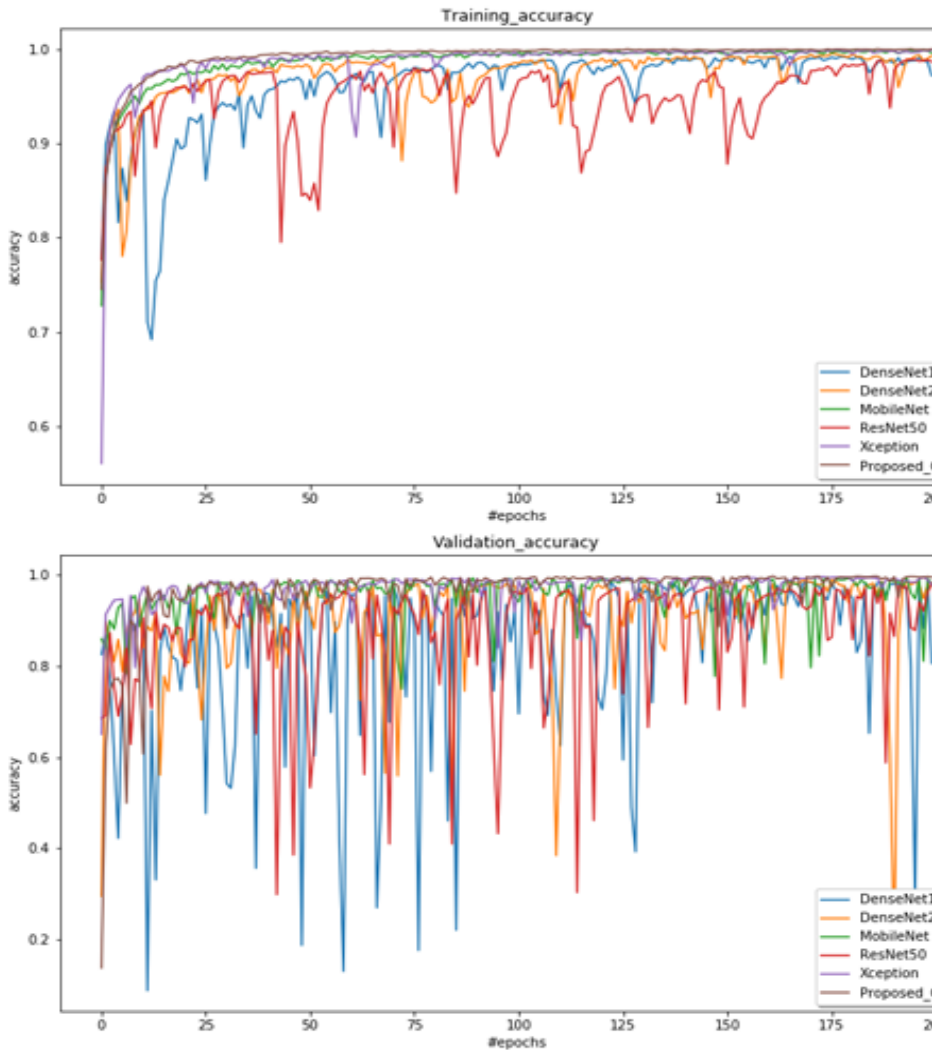
Disease	Class Label	Training Images	Validation Images	Total Images
Bacterial Spot	A	1596	531	2127
Early Blight	B	750	250	1000
Late Blight	C	1432	477	1909
Leaf Mold	D	714	238	952
Septoria Leaf Spot	E	1329	442	1771
Spoted Spider Mite	F	1257	419	1676
Target Spot	G	1053	351	1404
Leaf Curl Virus	H	4018	1339	5357
Mosaic Virus	I	280	93	373
Healthy	J	1194	397	1591

Table 4 list the performance measure of pre-trained models and the proposed model. The performance parameter includes training accuracy, validation accuracy, validation loss, Precision, Recall, and F1-Score. The training accuracy is the number of images correctly classified on which it was trained. And the validation accuracy is the number of unseen images correctly classified. The Precision is proportion of positive identifications was actually correct. Recall is proportion of actual positives was identified correctly. And F1-Score is the weighted average of Precision and Recall. As observed from Table 4, among all pre-trained deep convolutional neural networks, Xception outperformed ResNet50, DenseNet121, DenseNet201, and MobileNet with identification accuracy and loss of 99.60% and 1.73% respectively. But MobileNet models have lesser network parameters among all models. The proposed deep convolutional neural network realizes higher performance in terms of identification accuracy and loss. And also it has lesser parameters among all pre-trained models except DenseNet121 and MobileNet.

Table 4. Performance comparison of models

Model	#Parameters	Training Accuracy (%)	Validation Accuracy (%)	Validation Loss	Precision	Recall	F1-Score	Prediction Time for 3634 Samples (Sec.)
ResNet50	40,370,570	98.63	98.15	0.0670	0.98	0.98	0.98	11
DenseNet121	15,431,754	98.96	98.54	0.0475	0.99	0.98	0.98	12
DenseNet201	34,056,266	99.50	98.95	0.0466	0.99	0.99	0.99	15
MobileNet	11,623,114	99.84	99.40	0.0313	0.99	0.99	0.99	09
Xception	37,644,338	99.83	99.60	0.0173	0.99	1.0	0.99	17
Proposed model	21,688,842	99.85	99.71	0.0005	1.0	1.0	1.0	17

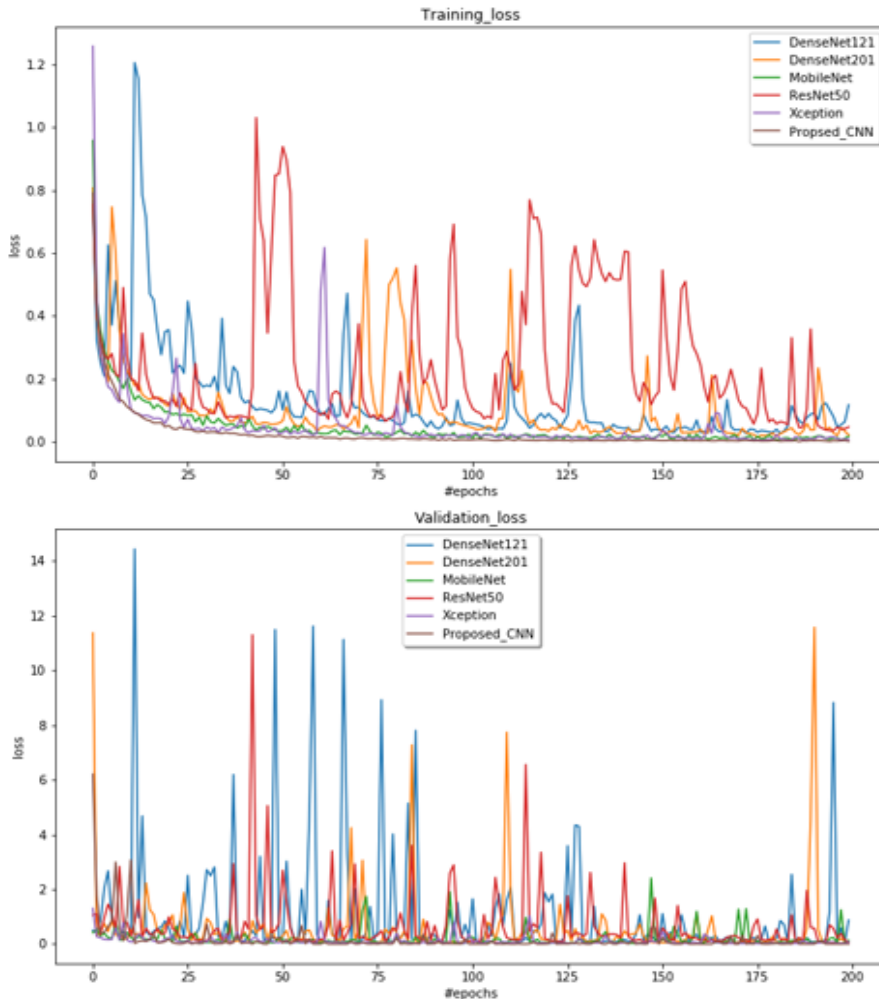
Figure 8. Training and validation accuracy plot of deep convolutional neural network models



Now, the proposed model is analyzed using a re-sampling technique known as k-fold cross-validation. The value k represents number of folds dataset to be split. And also it represents the number of groups. In each group, one fold is used for testing and the remaining k-1 folds involve in training. In this approach, we have used 5-fold cross-validation. The dataset is split into 5-folds. Each folds get a chance to train for 4 times. Each group is trained for 100 epochs. The result obtained from each group and the average result is shown in Table 5. The model achieves average validation accuracy and loss of 99.42% and 0.0220.

Finally, we have also tested the performance of the proposed model on splitting the dataset into the train, validate, and test. The dataset is split in to the train, validation, and test ratio of 70:15:15. The model is trained for 100 epochs. The performance of the proposed model on the given split is listed in Table 6.

Figure 9. Training and validation loss plot of deep convolutional neural network models



3.4 Confusion Matrix

The confusion matrix is a table in which rows represent the true class labels and columns correspond to the predicted class label. Hence, the confusion matrix helps to visualize the recognition accuracy of the system (Jiang, Peng, et al., 2019). The diagonal elements in the confusion matrix show the number of samples correctly classified and the remaining elements shows the number of samples incorrectly classified in a given set of samples. Figure 10 shows the confusion matrix of the proposed deep convolutional neural network on a given test image samples. The confusion matrix consists of 10 rows and 10 columns. Each row represents the class label of tomato crop diseases. The deeper color represents the higher recognition accuracy of corresponding classes.

According to the confusion matrix, among 4537 test images of ten classes, the model correctly classifies 4524 images and incorrectly classifies 13 images. Using this result, we can visually calculate the performance of the model. We can also calculate the recognition accuracy for each class label. Based on the result of the confusion matrix we visualize that, the proposed deep convolutional neural network has 100% recognition accuracy for diseases bacterial spot(A), leaf mold(D), leaf curl virus(H),

Table 5. Performance of each group in 5-fold cross validation experiment

Group No.	Training Accuracy	Validation Accuracy	Validation Loss	Precision	Recall	F1-Score
1.	99.77	99.27	0.0249	0.99	0.99	0.99
2.	99.94	99.38	0.0291	0.99	0.99	0.99
3.	99.86	99.59	0.0132	0.99	0.99	0.99
4.	99.81	99.45	0.0164	0.99	0.99	0.99
5.	99.75	99.42	0.0266	0.99	0.99	0.99
Average:	99.83	99.42	0.0220	0.99	0.99	0.99

Table 6. Performance of proposed model on training, validation, and test sample

Training Accuracy	Validation Accuracy	Test Accuracy	Validation Loss	Test Loss	Precision	Recall	F1-Score
99.74	99.23	99.12	0.0269	0.0315	0.99	0.99	0.99

and mosaic virus(I). Figure 11 shows the confusion matrix on test samples. For disease Leaf mold(D), Leaf curl virus(H), and Mosaic virus(I) model shows 100% identification accuracy on test samples.

3.5 Comparison With Existing Work

In this section, the performance of our proposed work is compared with the existing work for the tomato crop disease identification model on the same dataset and is represented in Table 7. From this table, it is clear that our proposed model performed better than existing work. Shijie, Jia, et al., 2017 has used VGG16 for feature extraction, and extracted features are fed to SVM classifier, which is time-consuming and required more resources. And the identification accuracy of this method is still low as compared to the proposed model. AlexNet has significantly better identification accuracy than the other identification model. But, when we compare the number of parameters in AlexNet and LeNet-5, LeNet-5 has a much lesser number of parameters than AlexNet. The proposed model identification accuracy is 99.71%, which is better than all other models.

Table 7. Proposed model comparison with state of art models

Model	#Parameters	Identification Accuracy (%)
AlexNet (Durmus, Halil, et al., 2017)	62,378,344	95.65
VGG16 + SVM (Shijie, Jia, et al., 2017)	138,423,208	89.00
LeNet-5 (Tm, Prajwala, et al., 2018)	376,046	94.85
Proposed CNN (Agarwal, Mohit, et al., 2020)	-----	98.40
Our proposed model	21,688,842	99.71

Figure 10. Confusion matrix results of the proposed model on tomato leaf validation samples

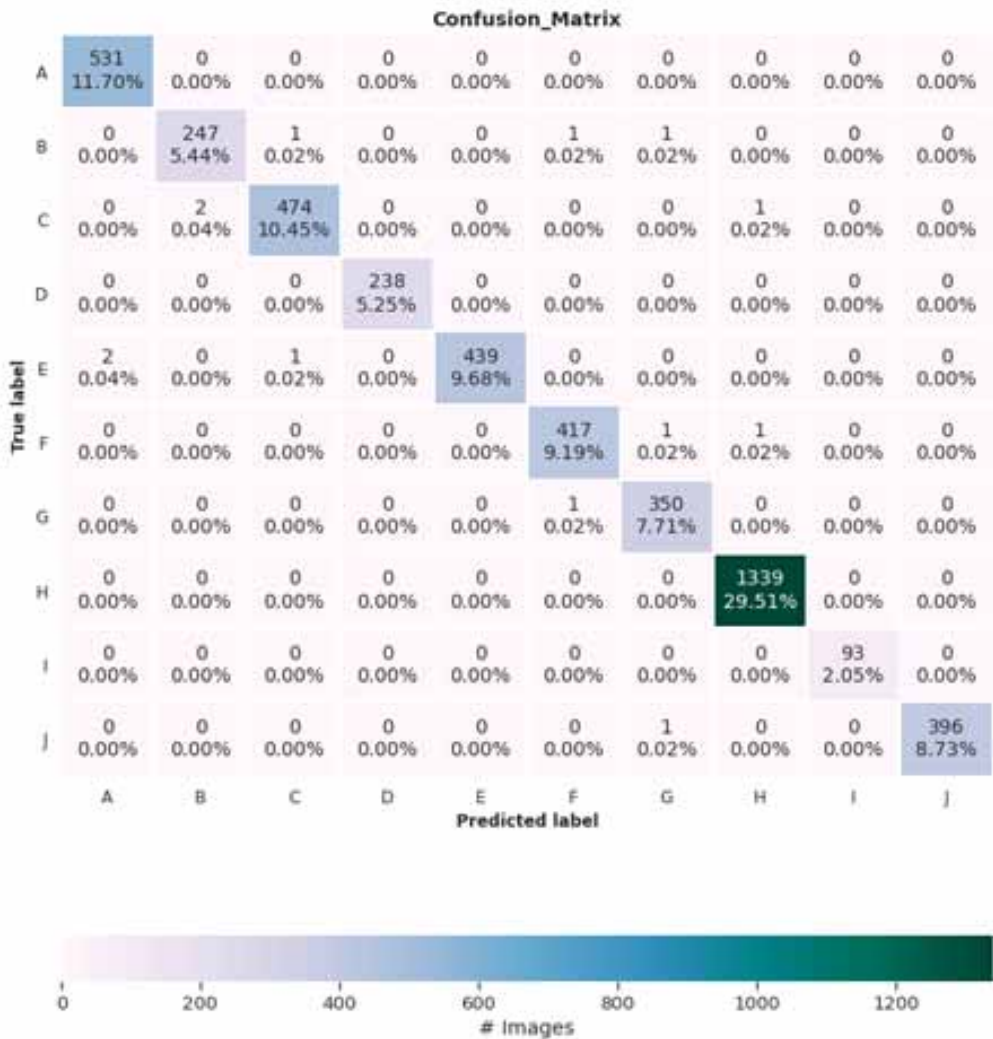
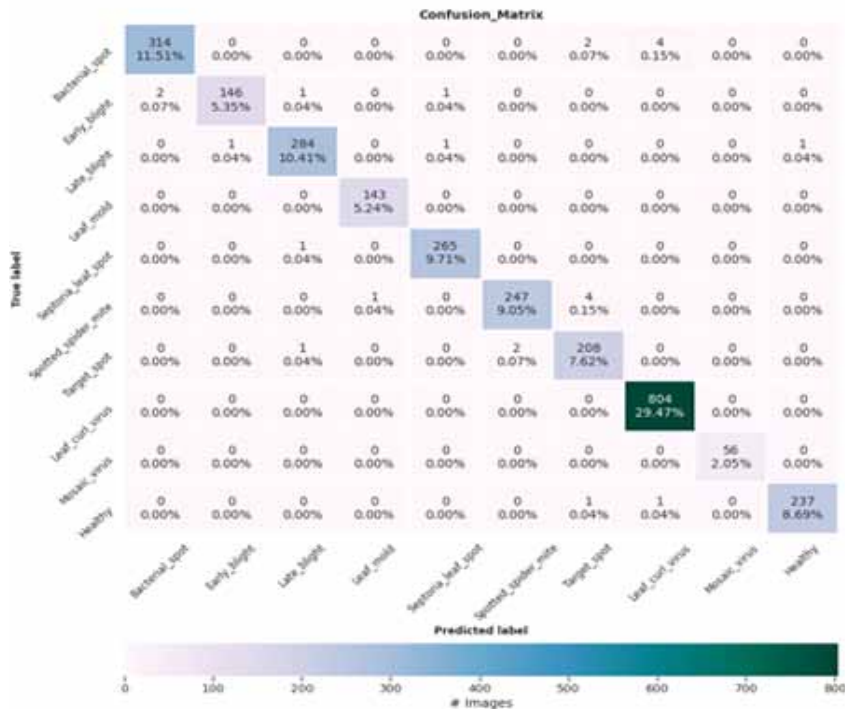


Figure 11. Confusion matrix results of the proposed model on tomato leaf test samples



3.6 Contribution

The major contribution of this paper are as follows-

- We proposed light weight deep convolutional neural network for identification of various tomato crop disease. In the proposed deep convolutional neural network number of parameter is minimized by reducing the kernel size and number of neurons. The batch normalization feature is added to overcome from overfitting and to make the training process faster.
- We presented suitable set of augmentation parameter and its value in order to improve the identification accuracy of tomato crop disease identification model. The parameter and its value is selected by testing on different set of values in given range.
- The proposed model has been compared with five state-of-the-art deep learning-based models ResNet50, DenseNet121, DenseNet201, MobileNet, and Xception. These models were trained in similar environmental conditions and evaluated on same public benchmark dataset with same training and validation images.

4. CONCLUSION

This article has proposed the most efficient and accurate model for the identification of tomato crop diseases. Because of tomato leaf diseases, there is a huge loss in the production of tomato. The proposed method classifies nine types of tomato crop diseases and a healthier one to gain the productivity and quality of tomato crops. Recently, the deep learning-based approach has become popular for plant disease classification as it automatically extracts features from images. Hence in this study, the deep

convolutional network model has been proposed for the identification of tomato crop diseases. This study also implements the standard pre-trained deep convolutional network-based model ResNet50, DenseNet121, DenseNet201, MobileNet, and Xception to compare the performance of the proposed model. To evaluate the performance of the proposed model, the diseased and healthy leaf images of tomato crops were collected from the plantVillage repository. The dataset contains a total of 18,160 images in RGB format. The dataset is preprocessed and split into a train-test set of 75-25 for this experiment. Finally, the models are trained on training samples and tested the performance on the test samples. Two parameters namely accuracy and loss have been used to compare the performance of models. The experiment results show the proposed deep convolutional neural network outperformed all pre-trained model and the state of art identification model with identification accuracy and loss of 99.71% and 0.05% respectively. Along with, loss and accuracy plot is presented, which shows the loss and accuracy of each model per epochs. This study also presents a confusion matrix, which can be used to visualize the performance of the model for each class of diseases. The performance of the proposed model is also analyzed using a cross-validation technique. The model is trained for 100 epochs and achieves an identification accuracy of 99.42%. Finally, the proposed model performance is tested by splitting the dataset into train, validation, and test set. The model achieves validation and test accuracy of 99.23% and 99.12% respectively.

REFERENCES

- Agarwal, M., Gupta, S. K., & Biswas, K. K. (2020). Development of Efficient CNN model for Tomato crop disease identification. *Sustainable Computing: Informatics and Systems*, 28, 100407. doi:10.1016/j.suscom.2020.100407
- Agarwal, M., Singh, A., Arjaria, S., Sinha, A., & Gupta, S. (2020). ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network. *Procedia Computer Science*, 167, 293–301. doi:10.1016/j.procs.2020.03.225
- Alarifi, J. S., Goyal, M., Davison, A. K., Dancey, D., Khan, R., & Yap, M. H. (2017, July). Facial skin classification using convolutional neural networks. In *International Conference Image Analysis and Recognition* (pp. 479–485). Springer. doi:10.1007/978-3-319-59876-5_53
- Chollet, F. (2017). Xception: Deep learning with depth-wise separable convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*. doi:10.1109/CVPR.2017.195
- Durmuş, H., Güneş, E. O., & Kırıcı, M. (2017). Disease detection on the leaves of the tomato plants by using deep learning. In *2017 6th International Conference on Agro-Geoinformatics*. IEEE. doi:10.1109/Agro-Geoinformatics.2017.8047016
- Elhassouny, A., & Smarandache, F. (2019). Smart mobile application to recognize tomato leaf diseases using Convolutional Neural Networks. In *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*. IEEE. doi:10.1109/ICCSRE.2019.8807737
- Fu, L., Lv, X., Wu, Q., & Pei, C. (2020). Field Weed Recognition Based on an Improved VGG With Inception Module. *International Journal of Agricultural and Environmental Information Systems*, 11(2), 1–13. doi:10.4018/IJAEIS.2020040101
- He, K. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*.
- Howard, A. G. (2017) Mobilenets: *Efficient convolutional neural networks for mobile vision applications*. arXiv preprint arXiv:1704.861.
- Huang, G. (2017). Densely connected convolutional networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*.
- Ileri, D., Belal, E., Okinda, C., Makange, N., & Ji, C. (2019). A computer vision system for defect discrimination and grading in tomatoes using machine learning and image processing. *Artificial Intelligence in Agriculture*, 2, 28–37. doi:10.1016/j.aiia.2019.06.001
- Islam, M. (2017). Detection of potato diseases using image segmentation and multiclass support vector machine. In *2017 IEEE 30th canadian conference on electrical and computer engineering (CCECE)*. IEEE. doi:10.1109/CCECE.2017.7946594
- Jiang, B., He, J., Yang, S., Fu, H., Li, T., Song, H., & He, D. (2019). Fusion of machine vision technology and AlexNet-CNNs deep learning network for the detection of postharvest apple pesticide residues. *Artificial Intelligence in Agriculture*, 1, 1–8. doi:10.1016/j.aiia.2019.02.001
- Jiang, P., Chen, Y., Liu, B., He, D., & Liang, C. (2019). Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks. *IEEE Access: Practical Innovations, Open Solutions*, 7, 59069–59080. doi:10.1109/ACCESS.2019.2914929
- Kamal, K. C., Yin, Z., Wu, M., & Wu, Z. (2019). Depth-wise separable convolution architectures for plant disease classification. *Computers and Electronics in Agriculture*, 165, 104948. doi:10.1016/j.compag.2019.104948
- Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147, 70–90. doi:10.1016/j.compag.2018.02.016
- Kaur, S., Pandey, S., & Goel, S. (2019). Plants disease identification and classification through leaf images: A survey. *Archives of Computational Methods in Engineering*, 26(2), 507–530. doi:10.1007/s11831-018-9255-6
- Koay, K. L. (2020). *Detection of Plant Leaf Diseases using Image Processing* (Diss.). Tunku Abdul Rahman University College.

- Kour, V. P., & Arora, S. (2019). Particle swarm optimization based support vector machine (P-SVM) for the segmentation and classification of plants. *IEEE Access: Practical Innovations, Open Solutions*, 7, 29374–29385. doi:10.1109/ACCESS.2019.2901900
- Krisnandi, D., Pardede, H. F., Yuwana, R. S., Zilvan, V., Heryana, A., Fauziah, F., & Rahadi, V. P. (2019). Diseases Classification for Tea Plant Using Concatenated Convolution Neural Network. *Communication and Information Technology Journal*, 13(2).
- Krithika, N., & Grace Selvarani, A. (2017). An individual grape leaf disease identification using leaf skeletons and knn classification. In *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE. doi:10.1109/ICIIECS.2017.8275951
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.
- Kumar, A., Kim, J., Lyndon, D., Fulham, M., & Feng, D. (2016). An ensemble of fine-tuned convolutional neural networks for medical image classification. *IEEE Journal of Biomedical and Health Informatics*, 21(1), 31–40. doi:10.1109/JBHI.2016.2635663 PMID:28114041
- Meng, X., Liu, M., & Wu, Q. (2020). Prediction of Rice Yield via Stacked LSTM. *International Journal of Agricultural and Environmental Information Systems*, 11(1), 86–95. doi:10.4018/IJAEIS.2020010105
- Nema, S., & Dixit, A. (2018). Wheat Leaf Detection and Prevention Using Support Vector Machine. In *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*. IEEE. doi:10.1109/ICCSDET.2018.8821098
- Padol, P. B., & Yadav, A. A. (2016). SVM classifier based grape leaf disease detection. In *2016 Conference on advances in signal processing (CASP)*. IEEE. doi:10.1109/CASP.2016.7746160
- Priyadharshini, R. A. (2019). Maize leaf disease classification using deep convolutional neural networks. *Neural Computing & Applications*, 31(12), 8887–8895. doi:10.1007/s00521-019-04228-3
- Rangarajan, A. K., Purushothaman, R., & Ramesh, A. (2018). Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia Computer Science*, 133, 1040–1047. doi:10.1016/j.procs.2018.07.070
- Sandika, B. (2016). Random forest based classification of diseases in grapes from images captured in uncontrolled environments. In *2016 IEEE 13th International Conference on Signal Processing (ICSP)*. IEEE. doi:10.1109/ICSP.2016.7878133
- Shijie, J., Jia, P., & Hu, S. (2017). Automatic detection of tomato diseases and pests based on leaf images. In *2017 Chinese Automation Congress (CAC)*. IEEE. doi:10.1109/CAC.2017.8243388
- Tahir, M. W., Zaidi, N. A., Rao, A. A., Blank, R., Vellekoop, M. J., & Lang, W. (2018). A fungus spores dataset and a convolutional neural network based approach for fungus detection. *IEEE Transactions on Nanobioscience*, 17(3), 281–290. doi:10.1109/TNB.2018.2839585 PMID:29994314
- Tm, P. (2018). Tomato leaf disease detection using convolutional neural networks. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*. IEEE. doi:10.1109/IC3.2018.8530532
- Wu, Q., Ji, M., & Deng, Z. (2020). Automatic Detection and Severity Assessment of Pepper Bacterial Spot Disease via MultiModels Based on Convolutional Neural Networks. *International Journal of Agricultural and Environmental Information Systems*, 11(2), 29–43. doi:10.4018/IJAEIS.2020040103
- Zhang, K., Zhang, L., & Wu, Q. (2019). Identification of Cherry Leaf Disease Infected by Podosphaera Pannosa via Convolutional Neural Network. *International Journal of Agricultural and Environmental Information Systems*, 10(2), 98–110. doi:10.4018/IJAEIS.2019040105
- Zhang, X., Qiao, Y., Meng, F., Fan, C., & Zhang, M. (2018). Identification of maize leaf diseases using improved deep convolutional neural networks. *IEEE Access: Practical Innovations, Open Solutions*, 6, 30370–30377. doi:10.1109/ACCESS.2018.2844405

Ganesh Bahadur Singh was born in Madhubani (Bihar) in 1996. He is currently Member (Research Staff) in Bharat Electronics Limited- Central Research Laboratory, Ghaziabad. He received B. Tech degree in Computer Science and Engineering from NIT Patna, in 2018 and the M. Tech degree in Computer Science and Engineering from NIT Jalandhar, in 2020. His research interests include Deep Learning and Computer Vision.

Rajneesh Rani (PhD) has received the B.Tech and M.Tech degrees, both in Computer Science and Engineering, from Punjab Technical University, Jalandhar, India in 2001 and Punjabi University Patiala, India in 2003 respectively. She has done her Ph.D in computer Science and Engineering from NIT Jalandhar in 2015. From 2003 to 2005, she was a lecturer in Guru Nanak Dev Engineering College, Ludhiana. Currently, she has been working as an assistant professor in NIT Jalandhar since 2007. Her teaching and research include areas like Image Processing, Pattern Recognition, Machine Learning, Computer Programming and Document Analysis and Recognition.

Nonita Sharma (PhD) is working as Assistant Professor, National Institute of Technology, Jalandhar. She has more than 10 years of teaching experience. Her major area of interest includes data mining, bioinformatics, time series forecasting and wireless sensor networks. She has published several papers in the International/National Journals/Conferences and book chapters. She received best paper award for her research paper in Mid-Term Symposium organized by CSIR, Chandigarh. She has authored a book titled- "XGBoost- The Extreme Gradient Boosting for Mining Applications".

Deepti Kakkar was born in 1982, in Jalandhar, Punjab, India. She did her Bachelor of Technology in Electronics and Communication Engineering from Himachal Pradesh University, India in 2003 and Masters of Engineering in electronics product design and technology from Punjab University, Chandigarh. Deepti did her PhD in Cognitive Radios from Dr. B.R. Ambedkar National Institute of Technology, Jalandhar, India. She has a total academic experience of 11 years and at present she is Assistant Professor in Electronics and Communication department with Dr. B. R. Ambedkar National Institute of Technology, Jalandhar, India. Earlier, she had worked as lecturer in Electronics and Communication department with DAV Institute of Engineering and Technology, Jalandhar, Punjab. She has guided more than 40 post graduate engineering dissertations and several projects. She is currently guiding 3 PhD theses. She has more than 30 papers in the proceedings of various International Journals and Conferences. Her recent research interests include cognitive neuroscience, neuro-developmental disorder, dynamic spectrum allocation, spectrum sensing, and Cognitive Radios.