Journal of Food and Agriculture 2020, 13(1): 33 - 53
DOI: http://doi.org/10.4038/jfa.v13i1.5230

Identification of an Efficient Deep Learning Architecture for Tomato Disease Classification Using Leaf Images

M.M. Gunarathna*[1], R.M.K.T. Rathnayaka[2] and W. M. W. Kandegama[3]

ABSTRACT

Computer vision technology plays a vital role in studies on plant pathology. The study of plant disease using image processing refers to the study of visually observable patterns on the plants. Recently, various image processing and pattern classification techniques are used to develop digital vision systems that can identify and classify the visual symptoms of plant diseases. As there are so many algorithms for the identification of plant diseases through leaf image classification, it is very critical to know what algorithms provide high accuracy and what type of diseases can be identified by the algorithm. The main objective of this study was to present accurate deep learning architectures that were more efficient in detecting tomato diseases which would eliminate human error in the identification via naked-eye observation. A base model was built with Convolutional Neural Network (CNN) from scratch using 22930 images of tomato plant leaves and compared with architectures like VGG16, MobileNet, and Inceptionv3 by fine-tuning them. The base CNN model has shown a training accuracy of 90% while other models VGG16, MobileNet, and Inceptionv3 have shown 89%, 91%, 87% accuracies, respectively. The computation complexity of the VGG16 model was higher than the other methods because the number of parameters defined in the model was high. Nevertheless, the accuracy of MobileNet was the highest compared with others. It is perfect for the study as it is lightweight, faster and can be easily run on mobile devices. The significant difference between the proposed CNN architecture and the others is that, the proposed CNN is slightly shallower and can be trained on the same dataset much more quickly. This study will help new researchers to conduct further improvements in the context of plant disease classification without any difficulty.

Keywords: Convolutional neural network, Deep learning architectures, Plant disease classification

[1]*Department of Computing and Information Systems, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka, Belihuloya, Sri Lanka.*
[2]*Department of Physical Sciences & Technology, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka, Belihuloya, Sri Lanka.*
[3]*Faculty of Agriculture and Plantation Management, Wayamba University of Sri Lanka, Sri Lanka*

* *mmgunarathna@std.appsc.sab.ac.lk*
https://orcid.org//0000-0002-1333-5542

INTRODUCTION

The agricultural sector remains an important sector in the country's economy and significantly contributes to foreign exchange earnings and GDP. The agricultural sector in Sri Lanka contributes 7 % to the national GDP, which is the highest when compared to other industries like fisheries, livestock accounts (Sri Lanka - Market Overview, 2020). Tomato is a rich source of nutrients, organic acid and vitamins, essential amino acids, and natural fibres. Tomato is one of the most commonly cultivated short duration vegetable crops grown in outdoor and indoor conditions worldwide.

Globally, annual fresh tomato production accounts for about 160 million tonnes, which is three times more than potatoes and six times more than rice worldwide (Background - Tomato News, 2020). According to the Census and Statistics Department, Sri Lanka, tomato production grew by 71% and 80 % in 2000 and 2010, respectively and the production is confined to six districts such as in Badulla, Nuwara-Eliya, Kandy, Matale, Anuradhapura, and Rathnapura. In 2010 tomato was cultivated in an extent of 7,261 ha yielding 75,335mt (Tomato Processing In SL, 2020). This suggests a growing increase over the years in the production of tomatoes. However, unfortunately, many diseases of tomato plants have caused a considerable loss to the quality and the quantity of productivity, which has directly impacted the economic growth (Tripathi and Maktedar, 2017).

Diseases in plants are caused by pathogens such as bacteria, fungi and viruses which can be transmitted through the soil, water, and air (Rath and Meher, 2019). Inexperienced pesticide usage and climate changes can alter the stages and rates of pathogen development. Due to above reasons it has become more difficult to manage diseases as they transmit globally easier than ever before (Sladojevic *et al*., 2016). Leaf mold, early blight, late blight, target spot, Septoria leaf spot, yellow leaf curl virus are some of the common diseases that tomato plants get infected. These diseases affect the complete tomato plant, including leaf, stem, fruit, root, etc. If diseases are not controlled at the appropriate time, they can damage the tomato plant and create a significant loss. Each year farmers have faced severe losses. Thus, protecting the tomato plants from these diseases is

vital to increase the production. The traditional method of identification of plant diseases is from the observation through naked-eye with the help of trained personnel. Nevertheless, it takes excessive vigilant time for continuous and constant observation of the fields (Chouhan *et al*., 2018).

There are many advancements in computer vision technologies that help to recognize and classify plant diseases automatically. These technologies eliminate the need for professional personals for continuous observation of fields, which takes huge time and cost (Verma *et. al.*, 2018). They also help to minimize the probability of human errors. Recent case studies show that soft computing models like Neural Networks (Fuentes *et al*., 2018), Decision trees (Sahith *et al.*, 2019), Support vector machine (Dandawate and Kokare, 2015) and Naïve Bayes (Mohanapriya and Balasubramani, 2019) have been applied for the automatic plant disease classification.

Deep learning has dominated computer vision over the last few years. The combined factors of widespread smartphone penetration, cameras, and high-performance processors in mobile devices and advances in computer vision technology paved by the deep learning have led to a situation where disease diagnosis is based on automated image recognition (Meena *et al*., 2018). A deep neural network is made up of a combination of deep learning and neural network, which mimics the general principles of the brain. Figure 1 shows the high-level architecture of a deep neural network which includes an input layer, number of hidden layers, and an output layer. A simple neural network contains only one hidden layer.

Layers will be trained to figure out best filter weight values (Durmus *et al*., 2017).The input layer is the starting layer of a CNN model, which takes images as inputs. Figure 2 illustrates how an input image looks like. Neural Network input layer takes input with a shape of height, width and depth. Depth represents the number of colour channels to which the images belong to. As an example, if the image is an RGB image, then it has three colour channels Red, Green and Blue. Therefore, the depth of the image is three. If it is a greyscale image, then the depth is one. This is how the depth of the image is calculated.

Hidden layers are located between the input layer and the output layer. They will be used to identify specific features. In the study, hidden layers were used to identify colour and texture features that help to classify tomato plant diseases. The output layer is a type of fully connected layer. It contains the class labels. If we want to classify our images into ten classes, then ten labels must be defined in the fully connected layer.

Convolutional Neural Network had a ground-breaking performance over the past decade in several fields linked to pattern recognition (Albawi *et al*., 2017). Many other deep learning architectures like VGG16, Inceptionv3, MobileNet, AlexNet, DenseNet, and GoogleNet also have been proposed recently, creating a slight difference to the hidden layers (Ferentinos, 2018). Although there were many approaches,

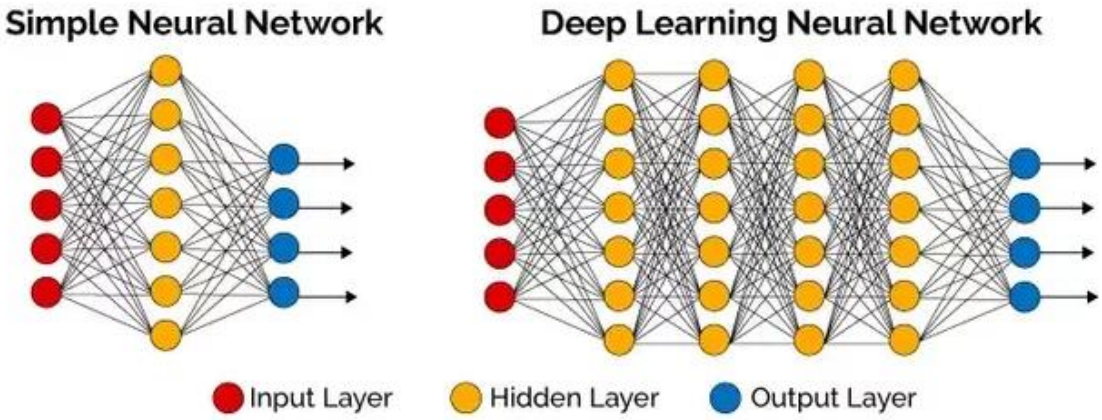

Figure 1. Simple and Deep Learning Neural network (Deep Learning Made Easy with Deep Cognition, 2020).

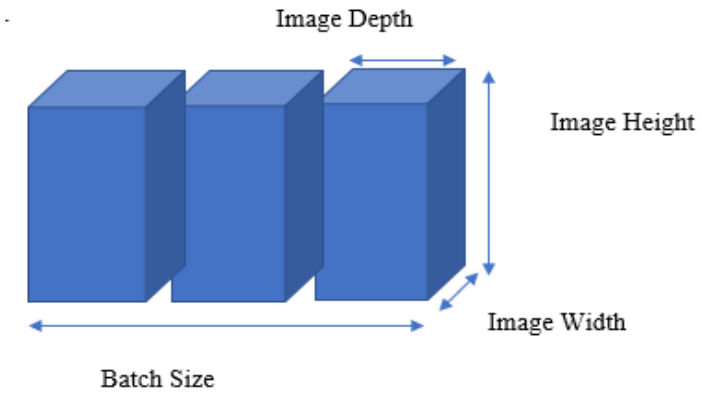

Figure 2. Image shape for the input layer.

there is still a gap in the identification of efficient deep learning architectures in accurate plant disease classification. Each structure has its limitations. As an example, some of the architectures are heavier as they contain a huge number of parameters and require a lot of processing time, although they will help to get high accuracy. The ultimate goal of this study was to identify the most effective way of classifying plant disease for further improvements in this field. Nevertheless, many approaches presented for the plant disease detection were not intended to replace the current way of doing it but rather to supplement them (Mohanty *et al.*, 2016). Although the results reported in many studies were very encouraging, the practical adaptation of such technologies was not thoroughly investigated. Many computer-based applications have been developed and revealed high efficiency, but evaluation of the performance of those architectures was not effectively conducted. Therefore, the main objective of this research was to evaluate the performance of recent deep learning architectures that provide the highest accuracy for plant disease classification. Rather than offering a guess, this study can, at most, give a definite answer that will assist new researchers in choosing the most appropriate recent deep learning algorithm for their future developments in the field of automated plant disease classification.

MATERIALS AND METHODS

Overview of the Methodology

The procedure followed in the study is indicated in Figure 3. There are five main stages: The Data collection stage, where the images were gathered and divided into three main directories as training, validating and testing; Pre-processing stage which ensured the relevant features were emphasized; Architecture selection stage where the suitable architectures were selected; Training and Validating stage where the models were trained and validated to obtain the final model and finally the prediction stage where the classifier was provided with characterized test images to determine its actual performance. Below, the phases are descriptively presented.

Data Collection

Diseased tomato leaf images belonging to 10 different classes with a resolution of 256x256 were collected from the

Plant Village library in Kaggle. The entire dataset was divided into three main directories as training, validating and testing where 17184, 4585, and 1161 images belong to each directory, respectively. The purpose of using two different datasets for testing and validating was to ensure how the model performs with unseen data. Testing data measured the performance of the final model while validating data was used to measure how the model performed during training. Below Table 1 represents the total number of training, testing and validating tomato leaf images belonging to each class.

As hardware; ACER Aspire3 laptop with Intel coreTM i5-8265U 1.66Hz , 8GB DDR4 Memory and NVIDIA GeForce MX230 was used to conduct this study.

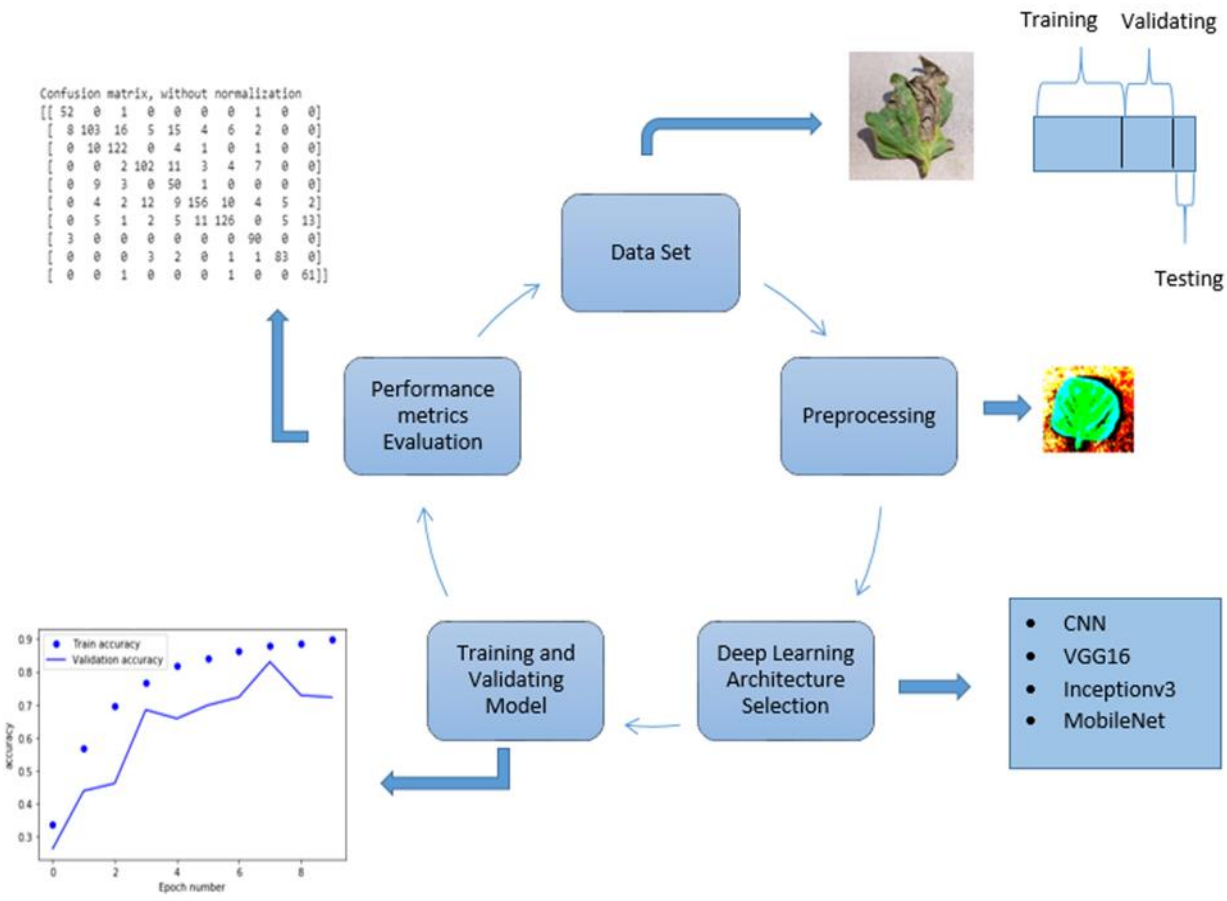

Figure 3. The steps followed in the methodology.

Our approach to classifying diseased tomato leaf images were focused on the automatic identification of marks that appeared on the leaf as spots as shown in Figure 4. Each image represents tomato leaf images belonging to each category.

Image Pre-processing

Image pre-processing is a technique that is applied to enhance the input image for further processing by ensuring the relevant features emphasized for further processing. It helps to remove many issues, including brightness effects, lighting, and issues due to poor contrast. Figure 5a shows the diseased tomato image before processing and Figure 5b shows the diseased tomato leaf image after pre-processing with the Keas input function. Data normalization and data augmentation techniques were applied in the pre-processing stage.

Normalization of data is an important step ensuring that each pixel of the image has a similar distribution of the data. It allows converging faster during network training. In this study, normalization of the data was achieved by subtracting the mean from each pixel and then dividing the output by the standard deviation (Sheela *et al.*, 2019). After this process, each pixel got a value within the range of zero and one. Image augmentation plays a vital role in

Table 1. Tomato leaf images collected for the study (PlantVillage Dataset, 2020).

Disease	**Train dataset**	**Validate dataset**	**Test dataset**
Bacterial spot	1684	425	54
Early blight	1761	480	159
Late blight	1713	463	138
Leaf mold	1753	470	129
Septoria leaf spot	1682	436	63
Spider mites	1537	435	204
Target spot	1659	457	168
Mosaic virus	1700	448	90
Yellow leaf Curl virus	1868	490	93
Healthy leaf	1863	481	63
Total	17184	4585	1161

creating efficient image classifiers (Venkataramanan *et al*., 2019). A large number of different images are needed to construct an accurate classifier. But it is practically difficult to find such an amount of massive data. But by using data augmentation techniques, it is possible to generate new data by making changes to current data. Rotation, shear, zoom, horizontal and vertical flip were the augmentation options applied in this study.

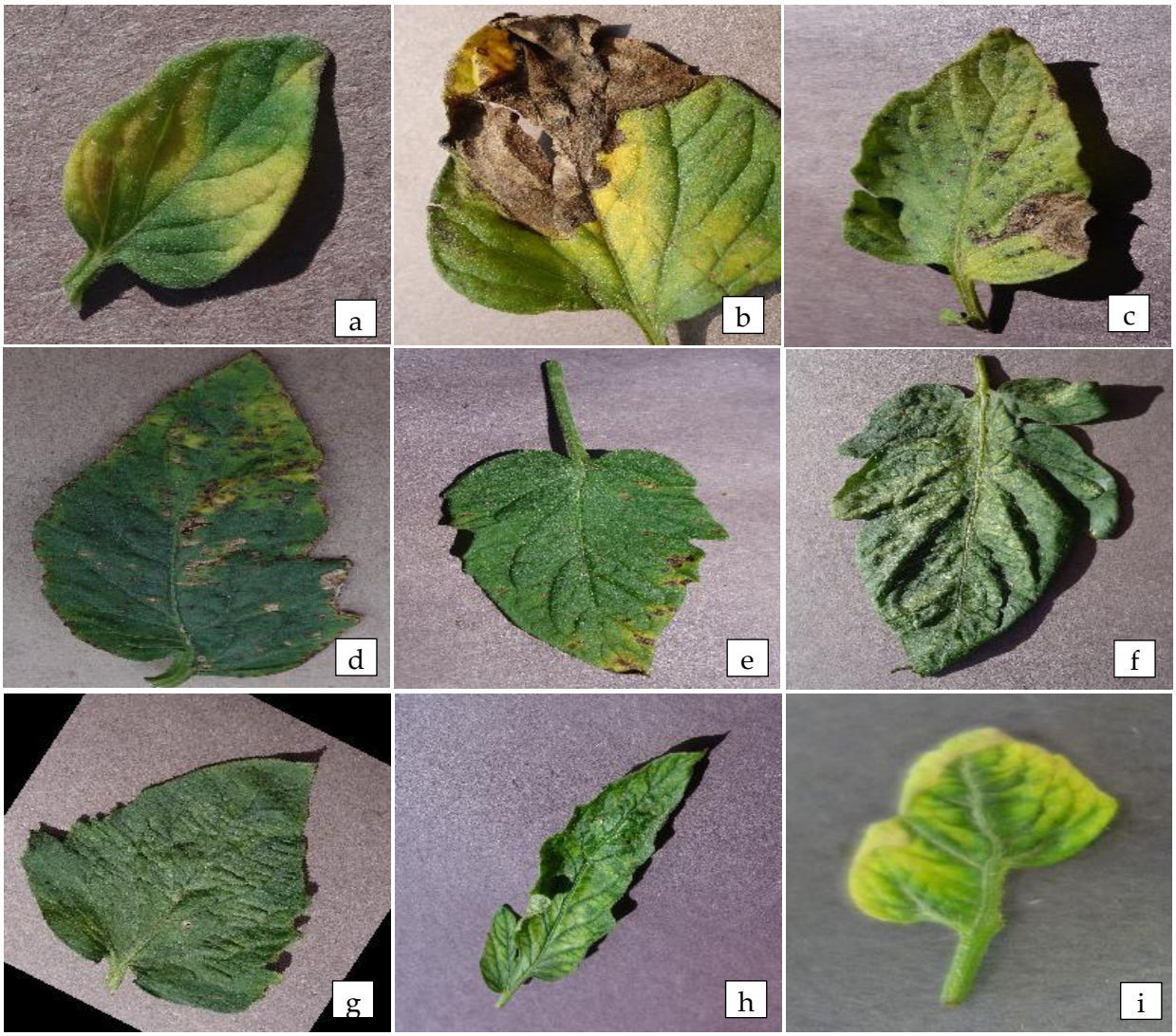

Figure 4. Tomato leaf images infected with: (a) Bacterial Spot, (b) Early Blight, (c) Late Blight, (d) Leaf Mold, (e) Septoria Leaf Spot, (f) Spider Mite, (g) Target Spot, (h) Mosaic Virus, (i) Yellow Leaf Curl Virus.

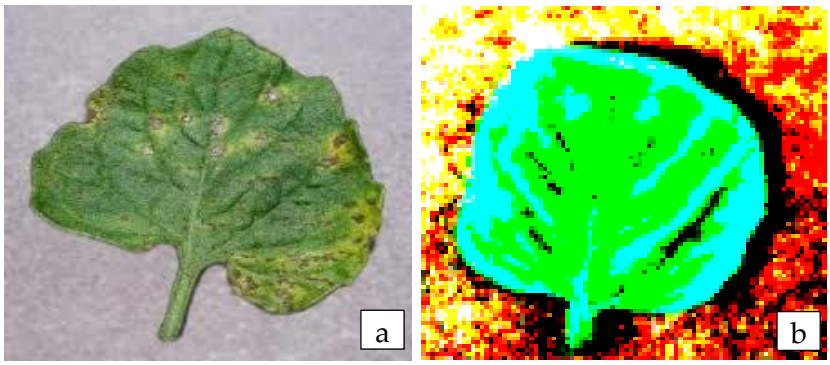

Figure 5. Pre-processing diseased tomato leaf image: (a) before and (b) after.

Suitable Architecture Model Selection

The success of the deep neural networks lies in the well-designed architecture. Therefore, it is critical to identify which design gives the best result and under which circumstances they can be used. An in-depth analysis of convolutional neural networks, Inceptionv3, VGG16, and MobileNet, has been carried out within the context of plant disease classification. All the models were trained with the *Keras* library built on top of the TensorFlow which is an end-to-end open-source platform in Machine Learning.

Convolutional Neural Network (CNN)

A convolutional neural network (CNN) is a type of artificial neural network which is typically designed to extract data features by using high-dimensional data (Bodapati and Veeranjaneyulu, 2019). The selected CNN model comprised with four convolutional blocks followed by batch normalization, max pooling, and dropout layers. Other than that, two dense and flatten layers were also included in the end. In Table 2, the architecture of the network along with the shapes of the layers and the number of trainable parameters is given.

Convolutional layers identify different feature patterns like edges, colours, lines etc. If one layer identifies the edge patterns, then the other layer identifies line patterns. Each has a level of abstraction. The important fact of these layers is that they do not throw the neural network off if they detect a location change of features. Batch normalization was similar to the normalization that we applied in the pre-processing step. The only difference was the type of layers that we

applied. By applying the normalization in the pre-processing stage, we made the images more suitable for the input layer. However, by using batch normalization, we made the images more ideal for the hidden layers. It changed the values that allow learning by itself independently from other layers. Batch normalization improved the stability and the speed of the model. It normalized the output of the previous activation layer by subtracting the mean and dividing the standard deviation of the batch. The reason for adding drop out and pooling layers were to avoid overfitting (Durmus *et al*., 2017). In the model, max-pooling layers were used to reduce the size of the representation that dynamically increases the computational speed. After applying the max-pooling to the inputs, each output was the max from the corresponding data. An important characteristic of max-pooling is that it contains only hyper-parameters, therefore the gradient descent had nothing to learn. As the name implies, drop out layers remove some neurons in training but use them in the validation. As the model got more complicated, it continued to memorize all weights, but drop out layers solved this by enabling the model to learn only random parts of each layer.

Table 2. Summary of the proposed CNN architecture model.

Layer	Output Shape	Parameters
Conv2d	(None , 254, 254,32)	896
Max_pooling2d	(None , 127, 127,32)	0
Conv2d_1	(None , 125, 125,64)	18496
Max_pooling2d_1	(None , 62, 62,64)	0
Conv2d_2	(None , 60, 60,64)	36928
Max_pooling2d_2	(None , 30, 30,64)	0
dropout	(None , 30, 30,64)	0
Conv2d_3	(None , 28, 28,128)	73856
Max_pooling2d_3	(None , 14, 14,128)	0
Flatten	(None, 25088)	0
Dense	(None,128)	3211392
Activation	(None,128)	0
Dropout_1	(None,128)	0
Dense_1	(None,10)	1290
Activation_1	(None,10)	0

Flatten layers usually used in-between pooling and dense layers because it is the function that transforms the pooled feature maps to a single vector that later on passes to a dense layer. For each block, the *Rectified Linear Unit* (*ReLU*) activation function was used, but at the end, after the last dense layer, the *SoftMax* activation function was applied. Activation function checked which nodes have fulfilled defined conditions in order to pass through layers. A 0.001 fixed learning rate was used throughout the training. The number of epochs was set to 10 and batch size was set to 27. The best combination of parameter values were identified by conducting a hyper parameter tuning. After building a model, it needs to be compiled. Adam optimizer and the categorical cross-entropy were used as the optimizer and the loss function for the model compilation, respectively. Finally, the model was trained and saved in the h5 format.

Visual Geometry Group 16 (VGG16)

VGG16 is a famous CNN model submitted by the researchers at the University of Oxford for the ImageNet Large Scale Visual Recognition Challenge (ILSCRC) in 2014 under the topic of image recognition (Khan *et al*., 2019). In this study, a pre-trained VGG16 model was fine-tuned for the identification of plant disease. As the VGG16 model was pre-trained for a variety of image categories, it has already learned features of plant leaves as well. In this study, transfer learning was utilized with fine-tuning to classify diseased plant leaves. Pre-processing was applied before feeding the images to the input layer with the help of *Keras* VGG16 pre-process input function. The VGG16 model containing a total of 138,357,544 parameters was downloaded from the internet with the saved weights. As the model already learned the features, there was no need to update the weights throughout the learning. Therefore, the weights were frozen to avoid the update. Since the weights were frozen, the total parameters of the model decreased to 134, 301,514. The final model was complied with the *Adam* optimizer and categorical *crossentropy* loss function. The learning rate was set to 0. 001, the batch size of the model was set to 27, and the epoch was set to 10. Finally, the model was trained and saved with the weights in the h5 format.

MobileNet

MobileNet belongs to the family of deep neural networks that are lightweight and faster. It can be used for classification and pattern recognition tasks like other convolutional neural networks. Still, the performance of this model is said to be very high because of its small size compared to other models. The size of a particular model mostly depends on the total number of parameters. The model was downloaded from the internet using *Keras* Library, and the images were pre-processed with the help of MobileNet pre-process function. When compared with other models, it usually has applied less data augmentation. The total parameters in the final model after fine-tuning was 3,239,114. The number of epochs was set to 10, batch size as set to 27, and the learning rate was used as 0.001. Finally, the model was trained and saved in the h5 format for further use.

Inceptionv3

Szegedy *et al.* (2016) have found the inception concept which belongs to the family of Deep Neural Network (Szegedy *et al*., 2016). A series of inception models were introduced, and they have been named each as Inception VN where N is referred to the version number (Too *et al*., 2019). In this study, Inceptionv3 was used, which is the third version of the series. The model with 21,802,784 parameters was downloaded with *Keras* library. To the original model, a global average pooling layer, a dropout layer, two dense layers, and the dense output layer with ten nodes were added. The specialty in this model is the addition of the global average pooling layer. It helped to identify the specific features more effectively and efficiently while eliminating the redundant parameters. The number of epochs, learning rate, and batch size were set as 10, 0.001, and 27, respectively. The batch size was set to 27. Here, the weights of the last 172 layers were frozen, and the remained top layers were retrained. The final model was consisted with a total of 23,387,434 parameters, which is slightly higher than the original model. Finally, the model was trained and saved in the h5 format for further use.

Train and Validate Models

The *fit()* function was invoked by feeding the train images, validate images, setting epoch size, and steps for epochs. Finally, the performance

metrics were evaluated, and the final model was predicted with the unseen data. These steps were followed iteratively by changing the parameter values until the best combination of the parameters were found to get the generalized accurate models.

RESULTS AND DISCUSSION

This study has been carried out to evaluate the performance of various deep learning architectures that are commonly used today. The CNN, VGG16, MobileNet, and Inceptionv3 have been built, fined-tuned, and trained successfully. The below Table 3 shows the obtained validation, training accuracy, and the time taken by each model per epoch to get trained.

A confusion matrix was created for each model as below Figure 6. It represents the prediction summary of MobileNet model. Other matrixes for Inceptionv3 and CNN are displayed in Figures 7a and 7b because the available space is minimal. The matrix contains the predicted label on the X-axis and the true label on the Y-axis, where dark colour labels show the amount which were correctly classified. To create the matrix, testing data was used. Matrixes show the prediction accuracy with respect to each class. The last row of the Figure 6, confusion matrix depicts that all the Yellow leaf Curl diseased leaves have classified correctly with a 100 % prediction accuracy. Similar to that, by using this matrix, it is possible to evaluate the prediction accuracy for each diseased class.

Figure 8 and Figure 9 were created with the help of Matplotlib, which is a plotting library in the python. Figure 8 represents the flow of training and validating accuracy and Figure 9 shows flow of training and validating loss of each model across epochs.

Table 3. Accuracy of the Models.

Model	Training Accuracy	Validation Accuracy	Average time per epoch
CNN	0.9018	0.8968	3652s
VGG16	0.8923	0.7894	6341s
MobileNet	0.9112	0.9078	2949s
Inceptionv3	0.8734	0.8628	6068s

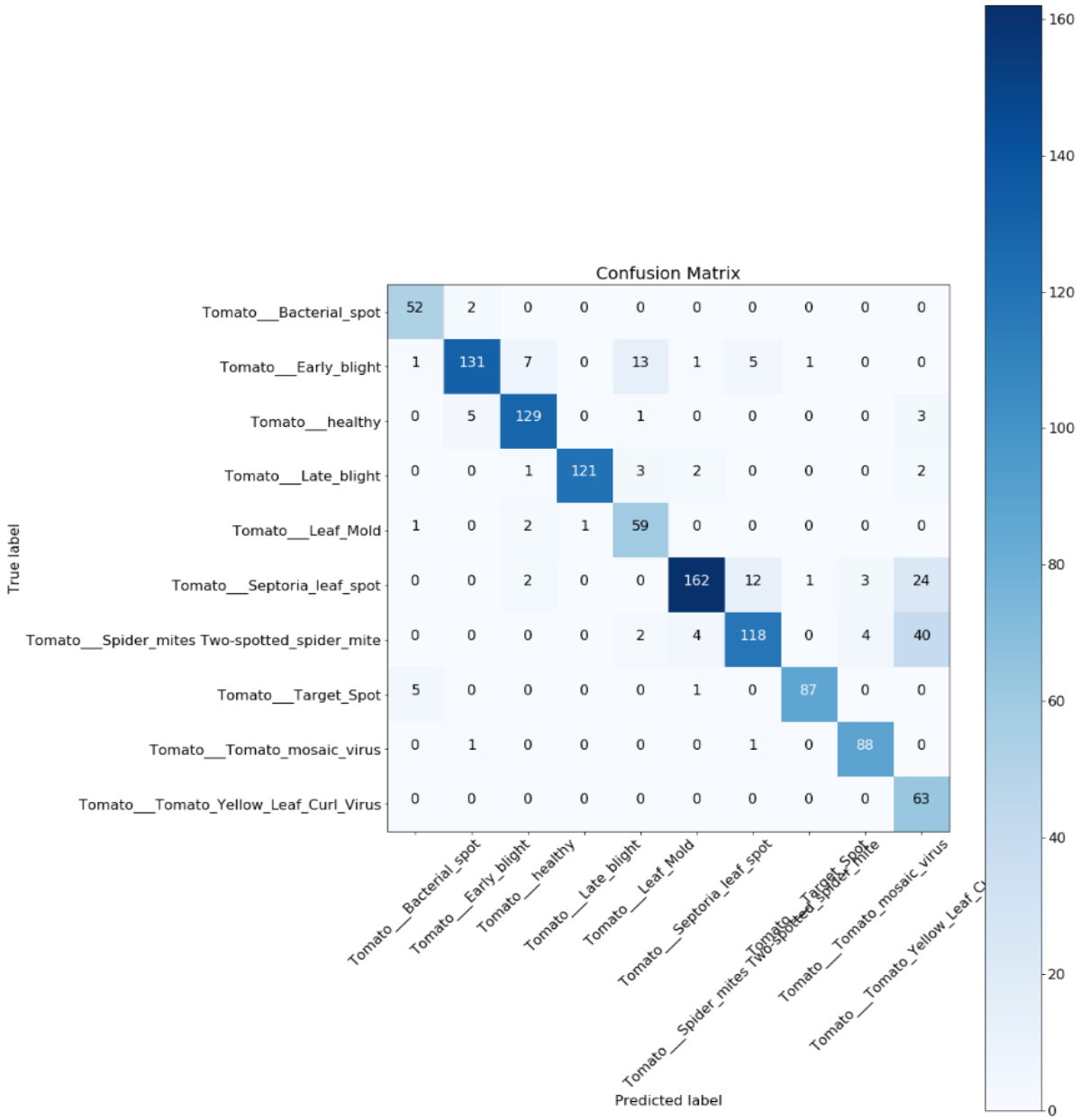

Figure 6. Confusion Matrix of the MobileNet.

By looking at the charts we can get an idea of how validating the accuracy and loss have changed in each epoch.

As shown in Figure 10, the expected class and the confidence can be achieved by randomly feeding an unseen image to the trained model. This has done using a python library called

OpenCV. With the aid of this, we can verify if the predicted outcome with the final trained model has been achieved or not. A leaf image with the disease, Early Blight is shown in Figure 10(a).

Confusion matrix, without normalization

```
[[ 41   0   0   0  13   0   0   0   0   0]
 [ 13  23   0   9  97  14   1   1   1   0]
 [  3   3  30  17  68   7   0   1   8   1]
 [  0   0   0 103  21   2   0   0   3   0]
 [  0   0   0   0  63   0   0   0   0   0]
 [  0   0   0   0   9 192   0   0   3   0]
 [  4   0   0   3  24  42  85   0   6   4]
 [  2   0   0   0   2   1   0  88   0   0]
 [  0   0   0   0   6   0   0   0  84   0]
 [  1   0   0   1   1   4   0   0   0  56]]
```

a

Confusion matrix, without normalization

```
[[ 52   0   1   0   0   0   0   1   0   0]
 [  8 103  16   5  15   4   6   2   0   0]
 [  0  10 122   0   4   1   0   1   0   0]
 [  0   0   2 102  11   3   4   7   0   0]
 [  0   9   3   0  50   1   0   0   0   0]
 [  0   4   2  12   9 156  10   4   5   2]
 [  0   5   1   2   5  11 126   0   5  13]
 [  3   0   0   0   0   0   0  90   0   0]
 [  0   0   0   3   2   0   1   1  83   0]
 [  0   0   1   0   0   0   1   0   0  61]]
```

b

Figure 7. Confusion Matrix of (a) Inceptionv3 and (b) CNN.

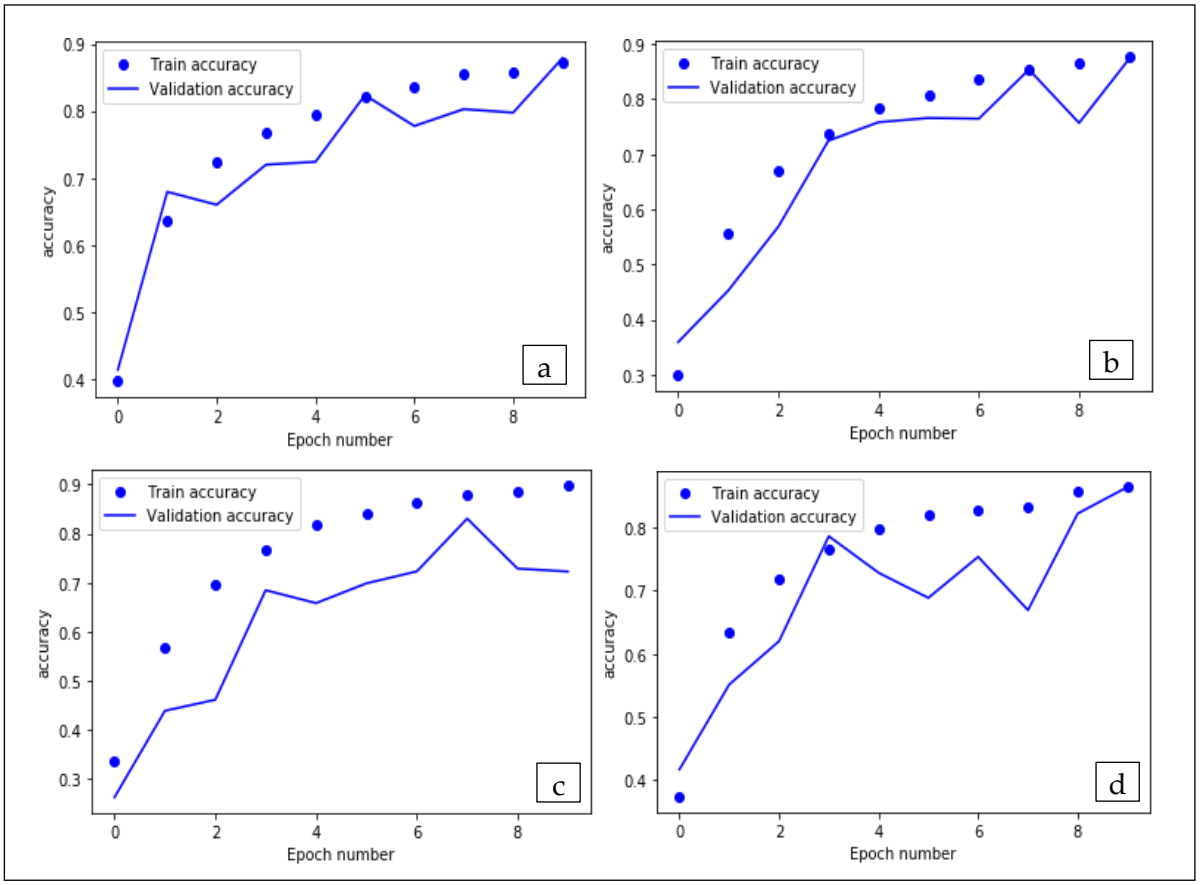

Figure 8. Training and validating accuracy of models: (a) CNN, (b) MobileNet, (c) Vgg16 and (d) Inceptionv3.

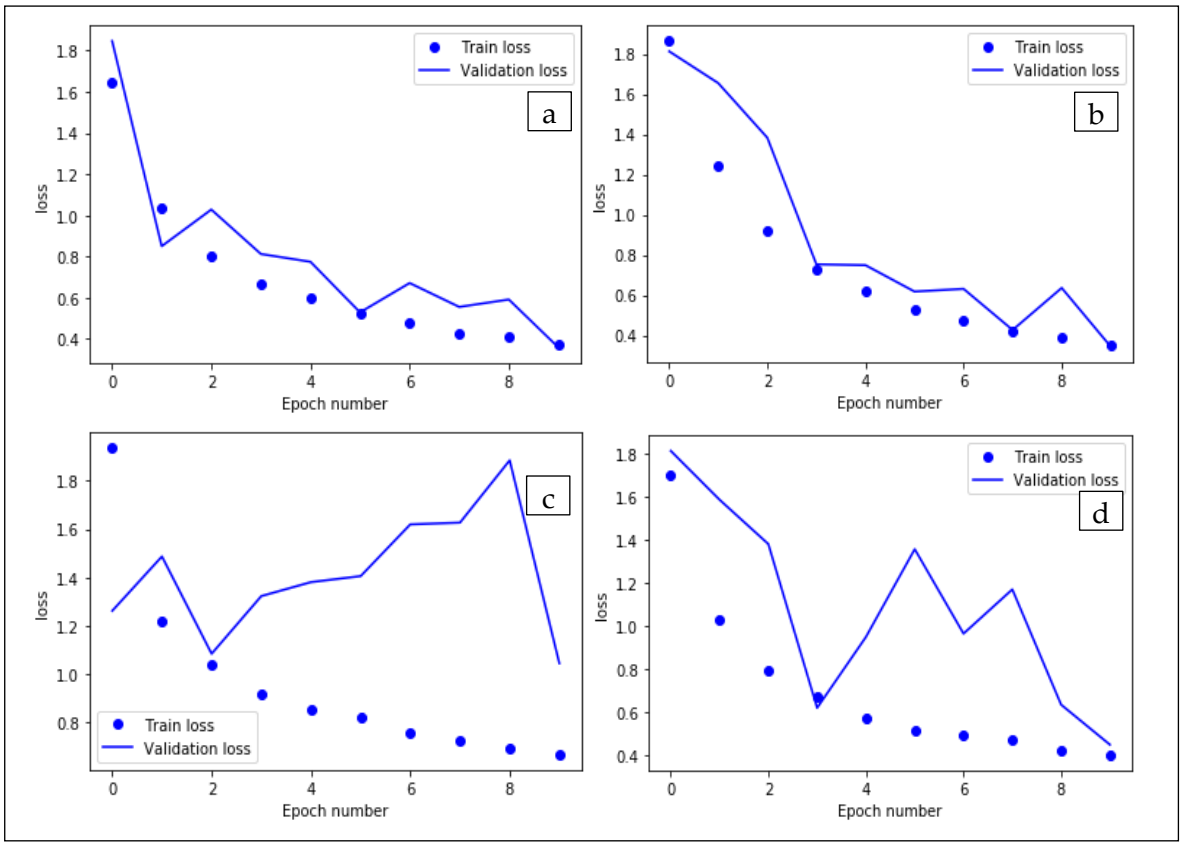

Figure 9. Training and validating loss of models: (a) CNN, (b) MobileNet, (c) Vgg16 and (d) Inceptionv3.

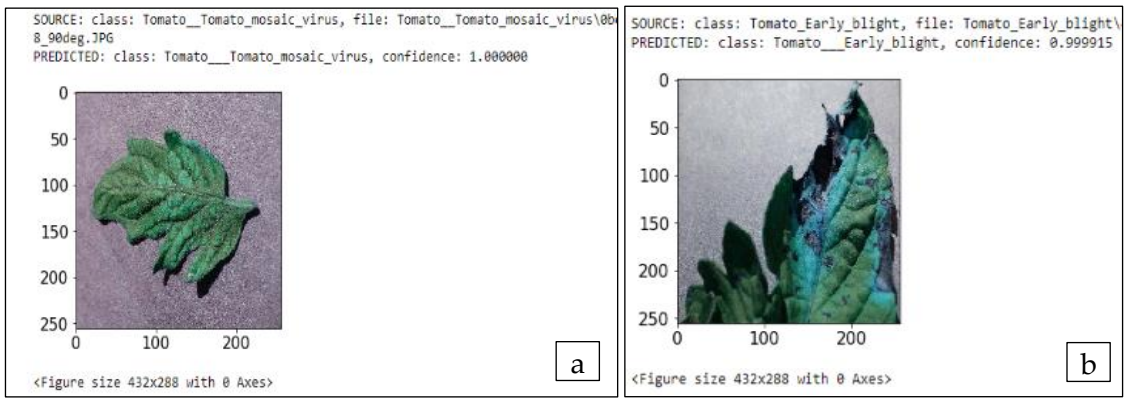

Figure 10. MobileNet model predictions for diseases where: (a) Mosaic Virus and (b) Early Blight.

With the confidence of 100 %, it has been correctly categorized into the appropriate class. Figure 10(b) shows a leaf with the disease, Mosaic Virus. It

also has been correctly classified into the relevant class with the confidence of 100 %.

Table 4 has been created in order to get an overview of the summary of the correctly and incorrectly classified number of images. The first column reflects the class in which the diseases were listed and the second column indicates the total number of images fed from each class to the final MobileNet model. Third and fourth column represents the total number of correctly and incorrectly classified images. Average accuracy for each class was given in the last column. In the last row of the Table 4, we can see that images belonging to the class yellow leaf curl virus has correctly classified with an average accuracy of 100 %. Similar to that, other diseases also can be evaluated with the help of the Table 4.

The deep learning model has shown a significant improvement in the context of pant disease classification. This study has been carried out to test how well the CNN model built from scratch has performed compared to fine-tuned pre-trained models using the Plant Village Dataset from Kaggle.

The VGG16 model has taken a long time to train the model compared to the other two. That is because the total number of parameters used in those models were very high. Therefore, we can conclude that when the parameters increase, time taken to train the model also increases.

Table 4. Summary of the CNN model predictions

Class	**Total images**	**Correctly Classified**	**Incorrectly classified**	**Accuracy %**
Bacterial spot	54	52	2	96
Early blight	159	131	28	82
Healthy	138	129	9	93
Late blight	129	121	8	94
Leaf mold	63	59	4	94
Septoria leaf spot	204	162	42	80
Spider mite	168	118	50	70
Target spot	93	87	6	94
Mosaic virus	90	88	2	98
Yellow leaf curl virus	63	63	0	100

When we consider the training and validation accuracy of VGG16, we can clearly see the training accuracy considerably larger than the validation accuracy. That means this model has worked well with the training data but not with the test data. This problem is referred to as the overfitting. This usually happens when the complexity of the model is very high. Because of the complexity, the model tried to memorize all the information, which results in a poor generalization for unseen data. The MobileNet was not too complicated when compared with the Inceptionv3 and VGG16 models as it contains a smaller number of parameters. We can conclude that to get an effective detection, it is vital to have a model that is not too complicated but complex enough to learn. The prediction accuracy of the MobileNet and CNN model was 0.90 and 0.89, respectively. That means both generalized well for test data than the VGG16 model. When we compare the last class of the matrix in Figure 6, 7a and 7b (Yellow Leaf Curl), we can clearly see Figure 6 has classified 100 % correctly than the other models.

With regards to the results, we can derive that fine-tuning of VGG16 and Inceptionv3 requires more time to identify the layers to freeze and train. The complexity of those models were very high as they were mainly used to identify sensitive patterns, and because of the gradient descent issue, they were hard to train as well. The significant difference between the proposed CNN architecture and the others is that, relative to the others, the proposed CNN is slightly shallower, therefore it can be trained on the same dataset much more quickly. Accuracy always varies with the parameters which need the prior experience to identify correctly. When compared with Inceptionv3, MobileNet works better with size, latency, as well as accuracy. This MobileNet model can be easily used in mobile devices and embedded vision applications in future improvements. The CNN model that has been built in this study was not deeper than the other model, but still, it can generalize well with less computational complexity.

CONCLUSION

This study proves that a simple model with a minimum of four convolution layers is sufficient for extracting features required for the tomato plant disease classification and when compared with other fine-tuned

architectures. MobileNet is perfect for the study as it is lightweight, faster, and can be easily run on mobile devices. Therefore rather than offering a guess, commitment of this study was able to provide a definite answer that will assist new researchers in choosing the best recent deep learning algorithm for their future developments in the field of automatic plant disease classification. This paper has been carried out only for the tomato leaf images, which was collected from the internet. In the future, we are going to test the model with real-time images. By using the models obtained throughout the research, we aim to develop an ensemble deep learning model that will perform better than the proposed models.

REFERENCES

Albawi, S., Mohammed, T. A. M. and Alzawi, S. (2017). A Data-Driven Approach To Precipitation Parameterizations Using Convolutional Encoder-Decoder Neural Networks Pablo. *IEEE*. Available at: https://wiki.tum.de/display/lfdv/Layers+of+a+Convolutional+ Neural+Network

Bodapati, J.D. and Veeranjaneyulu, N. (2019). Feature extraction and classification using Deep convolutional Neural Networks. *Journal of Cyber Security and Mobility*, *8*(2): 261–76.

Background Tomato News (2020). Retrieved 2nd December 2020, from http://www.tomatonews.com/en/background_47.html

Chouhan, S. S., Kaul, A., Singh, U. P. and Jain, S. (2018). Bacterial foraging optimization based radial basis function neural network (BRBFNN) for identification and classification of plant leaf diseases: An automatic approach towards plant pathology. *IEEE Access*, *6*: 8852–8863.

Dandawate, Y. and Kokare, R. (2015). An automated approach for classification of plant diseases towards development of futuristic decision support system in Indian perspective. *International Conference on Advances in Computing, Communications and Informatics* (*ICACCI 2015*), 794–799.

Deep Learning Made Easy with Deep Cognition (2020), KDnuggets, Retrieved 2nd December 2020, from https://www.kdnuggets.com/2017/12/deep-learning-made-easy-deep-cognition.html

Durmus, H., Gunes, E. O. and Kirci, M. (2017). Disease detection on the leaves of the tomato plants by using deep learning, *6th International Conference on Agro-Geoinformatics.*

Ferentinos, K.P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*: 311–

318. DOI: https://doi.org/10.1016/j.compag.2018.01.009

Fuentes, A. F., Yoon, S., Lee, J. and Park, D. S. (2018). High-performance deep neural network-based tomato plant diseases and pests diagnosis system with refinement filter bank. *Frontiers in Plant Science, 9*: 1162. DOI: 10.3389/fpls.2018.01162

Khan, S., N, Meera, Shaikh, A.A., Ansari, H. and Ansari, N. (2019). Disorder Detection in Tomato Plant Using Deep Learning. *SSRN Electronic Journal*, 2154–2160.

Meena, P. R., Saraswathy, G. P., Ramalakshmi, G., Mangaleswari, K. H. and Kaviya, T. (2018). Detection of leaf diseases and classification using digital image processing. *Proceedings of 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS 2017)*, 1–4.

Mohanapriya, K. and Balasubramani, M. (2019). Recognition of Unhealthy Plant Leaves Using Naive Bayes Classifier. *IOP Conference Series: Materials Science and Engineering*, *561*(1).

Mohanty, S. P., Hughes, D. P. and Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*, 1–10.

Plantvillage Dataset (2020). Kaggle, Available online at https://www.kaggle.com/emmarex/plantdisease [Accessed 2nd December 2020].

Rath, A. K. and Meher, J. K. (2019). Disease detection in infected plant leaf by computational method. *Archives of Phytopathology and Plant Protection*, *52*: 19-20, DOI: 10.1080/03235408.2019.1708546

Sahith, R., Reddy, P. V. P. and Nimmala, S. (2019). Decision tree-based machine learning algorithms to classify rice plant diseases. *International Journal of Innovative Technology and Exploring Engineering*, 9(1): 5365–5368.

Sheela, N., Harsha, M., Shastri, N. and Bhat, G. (2019). Image based plant leaf disease recognition and estimation system. *International Journal of Computer Sciences and Engineering*, 7(6): 725–731.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D. and Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *computational intelligence and Neuroscience*, *2016*: 1687-5265. DOI: https://doi.org/ 10.1155 /2016/ 3289801

Sri Lanka - Market Overview. Export.gov. (2020). Retrieved 2nd December 2020. Available at: https://www.export.gov/article?series=a0pt00000000GteAAE&type=Country_Commercial__kav.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision.

Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2818–26.

Tomato Processing In SL: Is It a Viable Remedy to Manage Surplus? (2020), Retrieved 2nd December 2020, Available at: https://www.dailynews.lk/2018/06/20/features/154441/tomato-processing-sl-it-viable-remedy-manage-surplus.

Too, E. C., Yujian, L., Njuki, S. and Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, *161*: 272–279.

Tripathi, M. K. and Maktedar, D. D. (2017). Recent machine learning based approaches for disease detection and classification of agricultural products. *Proceedings - 2nd International Conference on Computing, Communication, Control and Automation.*

Venkataramanan, A., Honakeri, D. K. P. and Agarwal, P. (2019). Plant Disease Detection and Classification Using Deep Neural Networks. *International Journal on Computer Science and Engineering*, *11*(9): 40–6.

Verma, S., Chug, A. and Singh, A. P. (2018). Prediction models for identification and diagnosis of tomato plant diseases. *International Conference on Advances in Computing, Communications and Informatics,* 1557–1563.