Journal of Positive School Psychology http://journalppw.com
2022, Vol. 6, No. 5, 193 – 201

Plant Disease Detection using Deep Transfer Learning

[1]Sukhwinder Kaur, [2]Saurabh Sharma

[1]Research Scholar, Sant Baba Bhag Singh University, Jalandhar, sukhwinderkaurasrpb@gmail.com
[2]Assistant Professor, Sant Baba Bhag Singh University, Jalandhar

Abstract

A country's economy largely depends on crops. Crops are the most important factor in food production. Healthy plants lead to healthy crops. If the plants are infected, this can adversely affect the food production. Plant diseases are mainly caused by viral or bacterial organisms. Disease on a plant can be identified by a change in colour of the leaf or even shape. It becomes very important to detect plant diseases as early as possible so that food production doesn't get affected. Detecting plant diseases via technology is a significant step in this direction. Using Deep Learning methods for this task can help us to identify diseases in plants. In this study, we use the VGGNet-19 model that is pre-trained using the weights of the 'ImageNet' dataset. By freezing the top layers and using transfer learning, we add a few layers to the model to try and improve the performance and accuracy of the model. This results in the accuracy of 97.52per cent for apple leaves and 95.75 per cent for grape leaves after running the model for 20 epochs.

Keywords: Deep Learning, Plant diseases, Transfer Learning, VGGNet-19.

1. INTRODUCTION

To Distinguish a plant diseases accurately when it initially appears is crucial stage for efficient disease management. Detecting diseases in plants has become increasingly significant in order to protect the crops. Early detection of diseases on plants is necessary so as not to infect the total crop and also to boost the quality of agricultural products.

Most of the symptoms of plant disease can be seen on the leaves and spotted. Different types of plant disease exist, but the majority of these diseases can be categorized into the three different categories which are bacterial disease, viral disease and fungal disease. Apple trees are vulnerable to numerous diseases. Diseases such as apple scab, black rot canker, powdery mildew can affect apple trees in various ways. Grape trees are also affected by diseases such as black rot, leaf blight and esca, black measle etc.

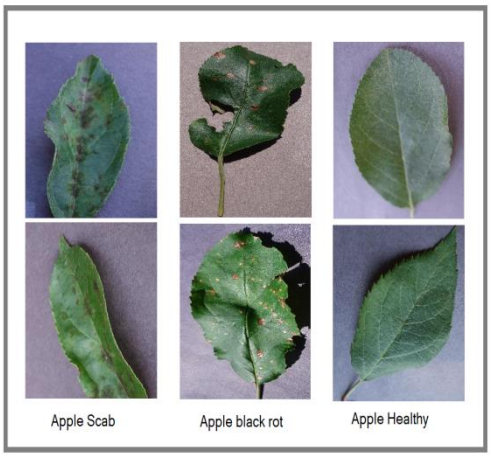


Fig. 1. *Apple leaves in Plant Village dataset*

Manual identification of diseases in plants is a time consuming and inefficient process. Therefore, using Deep Learning helps in reducing the effort required to identify plants with diseases.

1.1 Deep Learning

Deep learning is a machine learning technique that allows computers to learn by example in the same way that humans do [1]. In deep learning, computer models learn to perform classification tasks directly from images, text, or voice. Deep learning models can achieve cutting-edge precision that can exceed human-level performance. The model is trained using a large amount of labelled data and neural network architecture with many layers [2][3]. Models such as CNN (Convolutional Neural Networks) are used to classify images for both large and small datasets. Deep learning frameworks need a lot of information to return accurate outcomes. Data is fed in the forms of datasets. While handling the information, artificial neural networks are able to classify data with the responses received from a progression of parallel true or false inquiries including exceptionally complex numerical estimations.

1.2 Deep Transfer Learning

Transfer learning is a machine learning method in which an already developed task is reused as a continuation point for a novel task. In transfer learning, a new model need not be developed for every task [4]. Instead, we can use pre-trained models and their weights so that computation power requirements are reduced and the target can be achieved quickly as compared to time required in training a new model from scratch. In short, including the pre-trained models in a new model leads to lower training time and lower generalisation error.

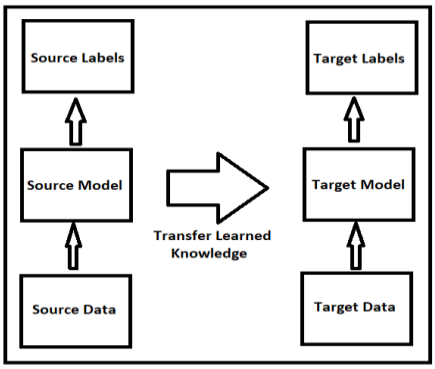


Fig. 2. *Transfer Learning*

2. Related Work

Patil et al. [5] developed a deep CNN model to detect diseases in cotton plants. Augmentation, fine tuning and image processing was performed on the leaves and different test cases gave an efficient outcome for detecting diseases in cotton plants. These methods can help farmers detect diseases as early as possible and prevent their crops from being damaged. Chen et al. [6] studied the transfer learning of deep convolutional neural networks for detecting plant diseases. They selected the VGGNet-19 model pre-trained on ImageNet dataset, combining with the Inception Module. The proposed approach was able to give validation accuracy of around 92% even under complex background conditions. Chohan et al.[7] proposed a Plant Disease detector which uses pictures of plants to detect the diseases. Convolutional Neural Network with multiple convolution and pooling layers is used on PlantVillage dataset with 15 percent data chosen for testing. They proposed that this model could be integrated with drones and other systems for disease detection in plants.Sibiya et al.[8] utilised CNN (GUI) in maize to detect diseases like northern corn leaf blight, grey leaf spot and common rust with accuracies 99.9%, 91% and 87%, respectively. The healthy leaf detection percentage was at 93.5% and the overall accuracy was 92.85%.

Binh et al. [9] proposed a method to detect gray spots on tea leaves using computer vision and image processing algorithms. They used a Neural of Multi-Layer Perceptron (MLP) built with extracted identifying features on the gray spots tea leaves. They combined image processing algorithms, neural networks with computer vision to determine gray spots on tea leaves. Barbedo [10] used individual lesions and spots for detecting plant diseases instead of considering the entire leaf. This was done to increase the variability of data. Multiple diseases could be detected on a single leaf with this approach. Better results were achieved using this approach. Also, each crop had an accuracy of at least 75%.

Türkoğlu et al., [11] compared some of the very common architectures of deep learning to

detect diseases in plants. The experiment was conducted using disease and pest images from Turkey. Deep feature extraction and SVM/ELM classifications produced better results as compared to transfer learning.

Wallelign et al. [12] designed a model using CNN classifier to classify Soybean plant diseases by taking leaf images of four classes from the PlantVillage database. Their model obtained 99.32% accuracy which shows that important features can be extracted and plant diseases classified using CNN .

Park et al. [13] used deep learning to diagnose the disease of strawberry leaf image and with method based on convolution neural networks to classify the disease of strawberry plants into healthy and diseased strawberry images and obtained 92% accuracy for classification .

Pooja et al. [14] proposed a disease detection and classification technique with the help of machine learning mechanisms and image processing tools. To begin with, they captured the infected area of the leaf and then performed image processing. The Support Vector Machine (SVM) classifier was used for classification and it was able to provide better results than previously used techniques.

Shrivastava et al. [15] used transfer learning of deep CNN for classification of rice disease. Images of rice plant disease belonging to the four classes Rice Blast (RB), Bacterial Leaf Blight (BLB) and Sheath Blight (SB) were taken along with healthy (HL) images.They used pre-trained deep convolution neural network(CNN) as a feature extractor and Support Vector Machine (SVM) as a classifier and identified the disease with 91.37% accuracy.

Brahimi et al. [16] proposed a deep models method to create a classifier for detection of disease. They used a large dataset containing 14828 images and nine diseases of tomato. They also showed that Deep Models with pre-training improves accuracy than those without pre-training. Ferentinos [17] applied deep learning classification technique for recognition of diseases in crops by using an open database of 87848 images of 25 different plants in a set of 58 distinct classes of plant disease combination and achieved with accuracy 99.53%.

3. Research Methodology

VGGNet is a convolutional neural network Architecture proposed by Karen Simonyan and Andrew Zisserman from the Visual Geometry Group of Oxford University in 2014[18].

It consists of 16 or more convolution, pooling and fully-connected layers , as shown in Fig. 3. The VGGNet includes two typical models such as the VGGNet-16 and VGGNet-19, which are available as the pretrained models with the 16 and 19 wt layers trained on the ImageNet dataset. 19 layers of VGGNet19 consist of 16 convolutional layers, 3 fully connected layers and 5 pooling layers as shown in Fig. 3.VGGnet19 consists of the following layers:

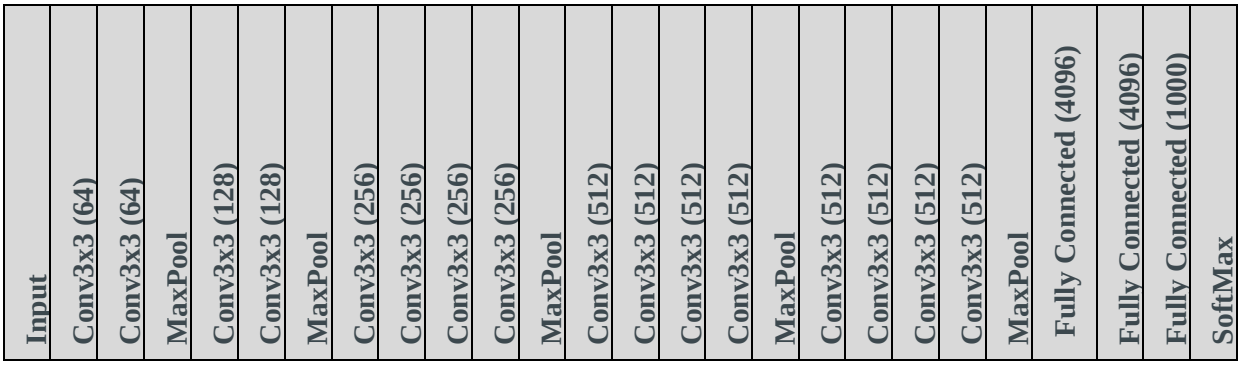


Fig. 3. *The architecture of VGGNet19*

VGGNet19 is pre-trained on ImageNet dataset with 14,197,122 images. VGGNet19 takes an input of size 224*224 RGB image, Using a kernel of 3*3 and a stride of 1 pixel allowed the model to cover an entire image.

Dataset

PlantVillage dataset developed by Hughes & Salathe (2015) consists of 54303 healthy and unhealthy leaf images. This dataset is divided into 38 categories.

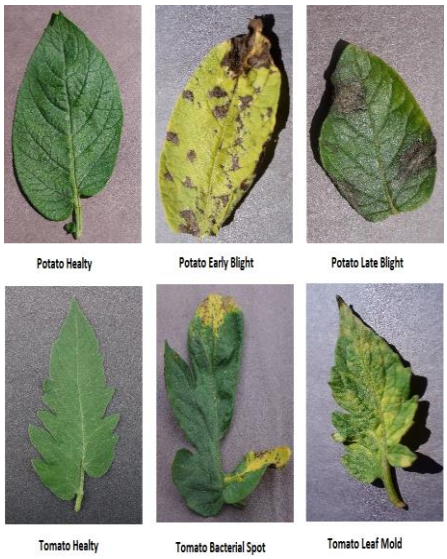


Fig. 4. *Potato and Tomato leaves in PlantVillage dataset*

Around 1600 healthy apple leaf images and 1300 apple leaves with scab and black rot are provided as input to the model with 40 percent data used for testing and 60 percent data used for training. Each image is converted to a size of 224*224, which is the default image size for VGGNet architecture.

We use Google Colab to perform plant disease detection with GPU in order to reduce the time required for training the model.

Following steps are performed to classify healthy and diseased apple leaves using Transfer Learning:

1. Upload the dataset as zip to google drive and mount the drive to the Google colab account.

2. Unzip the dataset in separate train and test folders.

3. Import the necessary and required libraries.

4. Perform data augmentation to increase the variability of data.

5. Load the VGGNet model with ImageNet weights.

6. Freeze the top layers and add new layers for transfer learning.

7. Provide the dataset as a directory to the ImageDataGenerator class.

8. Compile and train the model.

9. Plot the accuracy and loss graphically.

10. Test the model by providing an input image.

The above steps in detail are

The apple dataset is uploaded to Google drive and we mount the drive in Google colab.

```
1 from google.colab import drive
2 drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

Fig 5. *Mounting Google Drive in Colab*

The uploaded zip folder for PlantVillage dataset is extracted to two separate train and test folders.

```
1 #unzip plantvillage apple dataset
2 !unzip apple.zip
```

```
inflating: apple/test/Apple___Black_rot/image (274).JPG
inflating: apple/test/Apple___Black_rot/image (275).JPG
inflating: apple/test/Apple___Black_rot/image (276).JPG
inflating: apple/test/Apple___Black_rot/image (277).JPG
inflating: apple/test/Apple___Black_rot/image (278).JPG
inflating: apple/test/Apple___Black_rot/image (279).JPG
inflating: apple/test/Apple___Black_rot/image (280).JPG
inflating: apple/test/Apple___Black_rot/image (281).JPG
inflating: apple/test/Apple___Black_rot/image (282).JPG
inflating: apple/test/Apple___Black_rot/image (283).JPG
inflating: apple/test/Apple___Black_rot/image (284).JPG
inflating: apple/test/Apple___Black_rot/image (285).JPG
```

Fig 6. *Unzipping the apple leaf images*

The required libraries such as tensorflow, keras, ImageDataGenerator as well as the VGGNet19 model are imported.

```
import tensorflow as tf
from tensorflow import keras
from keras.preprocessing .image import ImageDataGenerator
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras import layers
```

```
#import vgg19 packages
from tensorflow.keras.applications.vgg19 import VGG19
from tensorflow.keras.models import Model
```

Fig 7. *Importing the required libraries*

The layers loaded in the VGGNet19 are frozen and new layers are created to modify the existing model. The base VGGNet19 model is combined with the new layers.

```
#Flatten the output from second last fully connected layer in VGG
flatten_layer = layers.Flatten()(model.output)

#fully connected layer with 512 hidden units and ReLU activation
fattened_fc_layer = layers.Dense(512, activation = 'relu')(flatten_layer)

#The last fully connected sigmoid layer with 5 neurons
flattened_fc_softmax_layer = layers.Dense(3, activation = 'softmax')(fattened_fc_layer)

```

```
#define a new model with base vgg19 model combined with the new layers just created
model = Model(inputs = model.inputs, outputs = flattened_fc_softmax_layer)
```

Fig 8. *Creating new layers and combining them with VGGNet19*

The following image shows the summary of the model just created:

```
model.summary()
```

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv4 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv4 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv4 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12845568
dense_1 (Dense)	(None, 3)	1539

Fig. 9. *Model Summary*

The model is compiled using the adam optimizer and finally the model is trained for 20 epochs. The images in the test folder are used as validation data.

```
model.compile(loss = "categorical_crossentropy", metrics = ['accuracy'], optimizer = 'adam')

history = model.fit(training_iterator, validation_data = testing_iterator, epochs =20)
```

Fig 10. *Model compilation*

4. Results

Training the model for 20 epochs gives an accuracy of around 97 percent. Using Google Colab Pro GPU took an average of 34 seconds for each epoch. The experiment was performed

on the apple leaves with 3 classes: Apple Scab, Apple black rot and healthy apple leaves.

```
Epoch 1/20
31/31 [==============================] - 45s 1s/step - loss: 3.8708 - accuracy: 0.5917 - val_loss: 0.3417 - val_accuracy: 0.8790
Epoch 2/20
31/31 [==============================] - 35s 1s/step - loss: 0.2847 - accuracy: 0.8979 - val_loss: 0.1752 - val_accuracy: 0.9422
Epoch 3/20
31/31 [==============================] - 34s 1s/step - loss: 0.1767 - accuracy: 0.9424 - val_loss: 0.2203 - val_accuracy: 0.9204
Epoch 4/20
31/31 [==============================] - 34s 1s/step - loss: 0.1709 - accuracy: 0.9439 - val_loss: 0.1590 - val_accuracy: 0.9466
Epoch 5/20
31/31 [==============================] - 34s 1s/step - loss: 0.1377 - accuracy: 0.9565 - val_loss: 0.1395 - val_accuracy: 0.9509

Epoch 18/20
31/31 [==============================] - 34s 1s/step - loss: 0.0438 - accuracy: 0.9869 - val_loss: 0.0911 - val_accuracy: 0.9607
Epoch 19/20
31/31 [==============================] - 34s 1s/step - loss: 0.0628 - accuracy: 0.9752 - val_loss: 0.0779 - val_accuracy: 0.9716
Epoch 20/20
31/31 [==============================] - ETA: 0s - loss: 0.0649 - accuracy: 0.9752
```

Fig. 11. *Model training for 20 epochs (apple)*

The following figures show the accuracy and loss graphs which are plotted using the matplotlib library:

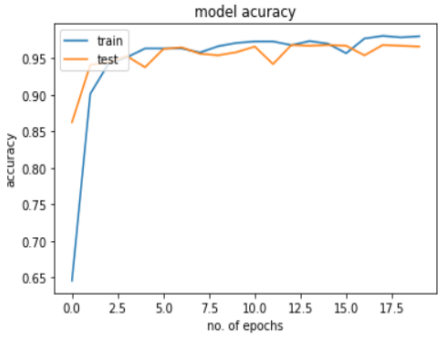


Fig. 12. *Model Accuracy*

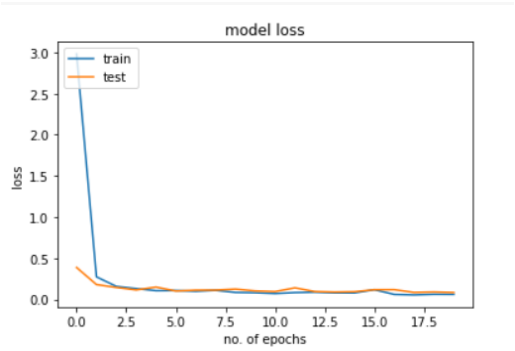
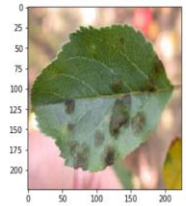


Fig. 13. *Model loss*

Testing the model by uploading a random apple leaf image detects the disease correctly.



```
#convert image to array
img = img_to_array(img)
img = img.reshape(1,224,224,3)
```

```
result = model.predict(img)
```

```
class_labels = {'Apple___Apple_scab': 0, 'Apple___Black_rot': 1, 'Apple___healthy': 2}
```

```
print(result)
```

```
[[1.8543192e-26 1.0000000e+00 0.0000000e+00]]
```

```
print([key for key in class_labels][np.argmax(result)])
```

```
Apple___Black_rot
```

Fig. 14. *Testing the model on an apple black rot leaf*

The same experiment was conducted on grape leaf dataset which has four classes: Grape_Black_rot, Grape_Esca_(Black_Measles), Grape_healthy, Grape_Leaf_blight_(Isariopsis_Leaf_spot)

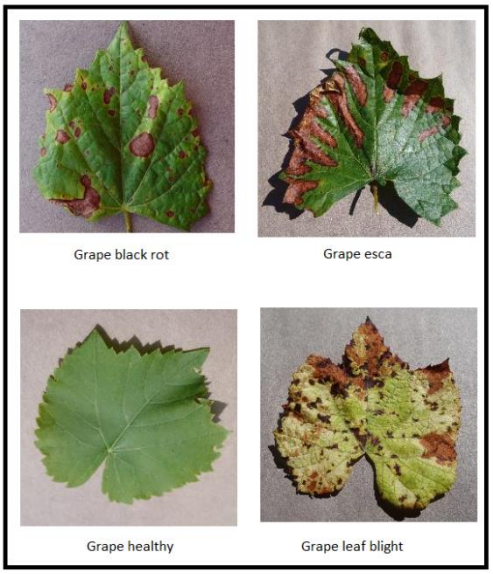


Fig. 15. *Grape leaves in PlantVillage dataset*

Training the model for 20 epochs provides a validation accuracy of around 96 percent.

```
model.compile(loss = "categorical_crossentropy", metrics = ['accuracy'], optimizer = 'adam')

history = model.fit(training_iterator, validation_data = testing_iterator, epochs =20)
```

```
Epoch 1/20
178/178 [==============================] - 53s 292ms/step - loss: 1.5077 - accuracy: 0.7533 - val_loss: 0.3199 - val_accuracy: 0.8456
Epoch 2/20
178/178 [==============================] - 51s 289ms/step - loss: 0.3247 - accuracy: 0.8672 - val_loss: 0.3001 - val_accuracy: 0.8701
Epoch 3/20
178/178 [==============================] - 51s 286ms/step - loss: 0.2608 - accuracy: 0.8957 - val_loss: 0.1811 - val_accuracy: 0.9273
```

```
Epoch 18/20
178/178 [==============================] - 51s 286ms/step - loss: 0.1905 - accuracy: 0.9260 - val_loss: 0.2041 - val_accuracy: 0.9077
Epoch 19/20
178/178 [==============================] - 51s 286ms/step - loss: 0.1976 - accuracy: 0.9221 - val_loss: 0.1730 - val_accuracy: 0.9379
Epoch 20/20
178/178 [==============================] - 51s 285ms/step - loss: 0.1726 - accuracy: 0.9306 - val_loss: 0.1133 - val_accuracy: 0.9575
```

Fig. 16: *Model training for 20 epochs (grapes)*

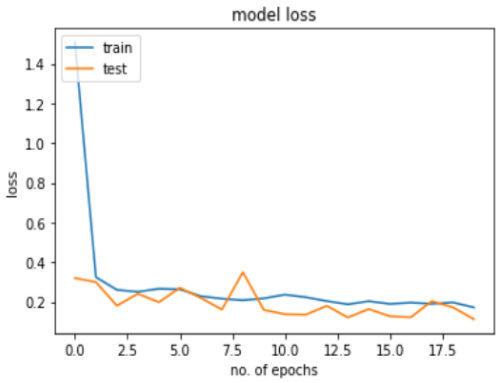


Fig. 17: *Model Loss*

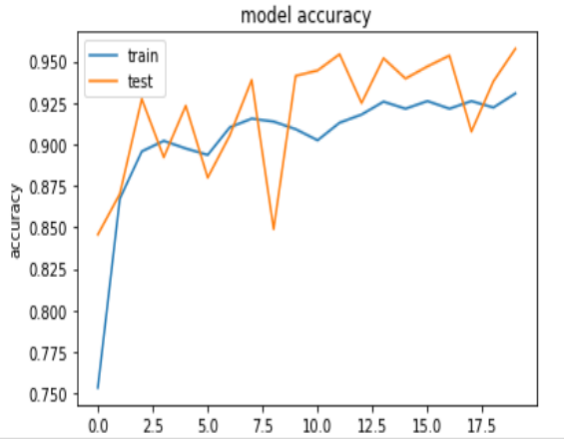


Fig. 18. *Model Accuracy*

The following table shows the comparison of accuracy and loss for apple and grape leaves for 20 epochs:

Table 1: *Comparison of accuracy and loss for apple and grape leaves*

Plant	Accuracy (20 epochs)	Loss (20 epochs)
Apple	97.52 %	0.0649
Grape	95.75%	0.1133

5. Conclusion and Future Scope

Detecting plant diseases using Deep Learning can help farmers protect and minimise the damage to the crops. Using pre-trained models such as VGGNet19 and modifying a few parameters can significantly increase the accuracy in classifying images of plants.

Transfer learning is a great method that can reduce the time and computation power required to train the models. Using VGGNet19 and adding a few layers can provide better results for classifying and detecting diseases in plants. Also, these methods can be used to predict and classify images for a wider range of plants and other crops. Moreover, Deep Transfer Learning can be used in other real world applications using mobile devices for classifications and predictions using images.

Reference

[1] Pouyanfar, S.; Sadiq, S.; Yan, Y.; Tian, H.; Tao, Y.; Reyes, M.P.; Shyu, M.L.; Chen, S.C.; Iyengar, S. A survey on deep learning: Algorithms, techniques, and applications. ACM Comput.Surv. (CSUR) 2018, 51, 1–36.

[2] Liu, J., & Wang, X. (2021). Plant diseases and pests detection based on deep learning: a review. Plant Methods, 17(1), 1-18.

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

[4] Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C.A survey on deep transfer learning. In International Conference on Artificial Neural Networks; Springer: Berlin/Heidelberg, Germany, 2018; pp. 270–279

[5] Patil, B. V., &Patil, P. S. (2021). Computational method for Cotton Plant disease detection of crop management using deep learning and internet of things platforms. In Evolutionary Computing and Mobile Sustainable Networks (pp. 875-885). Springer, Singapore.

[6] Chen, J., Zhang, D., Sun, Y., &Nanehkaran, Y. A. (2020). Using deep transfer learning for image-based plant disease identification. Computers and Electronics in Agriculture, 173, 105393.

[7] Chohan, M., Khan, A., Chohan, R., Katpar, S. H., &Mahar, M. S. (2020). Plant disease detection using deep learning. International Journal of Recent Technology and Engineering, 9(1), 909-914.

[8] Sibiya, M., & Sumbwanyambe, M. (2019). A computational procedure for the recognition and classification of maize leaf diseases out of healthy leaves using convolutional neural networks. AgriEngineering, 1(1), 119-131.

[9] Binh, P. T., Nhung, T. C., & Du, D. H. (2019, December). Detection and Diagnosis Gray Spots on Tea Leaves Using Computer Vision and Multi-layer Perceptron. In International Conference on Engineering Research and Applications (pp. 229-237). Springer, Cham.

[10] Barbedo, J. G. A. (2019). Plant disease identification from individual lesions and spots using deep learning. Biosystems Engineering, 180, 96-107.

[11] Türkoğlu, M., &Hanbay, D. (2019). Plant disease and pest detection using deep learning-based features. Turkish Journal of Electrical Engineering & Computer Sciences, 27(3), 1636-1651.

[12] Wallelign, S., Polceanu, M., &Buche, C. (2018, May). Soybean plant disease identification using convolutional neural networks. In The thirty-first international flairs conference.

[13] Park H., Jeesook E., & Kim S.H., "Crops Disease Diagnosing Using Image Based Deep Learning Mechanism," 2018 International Conference on Computing and Network Communications (CoCoNet), 2018.

[14] Pooja, V., Das, R., &Kanchana, V. (2017, April). Identification of plant leaf diseases using image processing techniques. In 2017 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR) (pp. 130-133). IEEE.

[15] Shrivastava, V. K., Pradhan, M. K., Minz, S., & Thakur, M. P. (2019). Rice Plant Disease Classification Using Transfer Learning of Deep Convolution Neural Network. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 42(3/W6).

[16] Brahimi, M., Boukhalfa, K., & Moussaoui, A. (2017). Deep learning for tomato diseases: classification and symptoms visualization. Applied Artificial Intelligence, 31(4), 299-315.

[17] Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145, 311-318.

[18] Simonyan,K, Zisserman,A.,(2015). Very deep convolutional networks for large-scale image recognition. In:Int. Conf. Learn.Represent.pp1-14.