Chapter 3

Crop Diseases Identification

3.1 INTRODUCTION

Crop diseases have an impact on the progress of their particular species, hence early detection is crucial. Aimed at the identification and classification of plant diseases, numerous ML/DL methods have been employed. However, with the development of a subset of Machine Learning (ML) called Deep Learning (DL), this field of study now appears to have significant potential in terms of improved accuracy. In order to identify and categorize the signs of plant diseases, several developed/modified DL architectures are used in conjunction with a number of visualization techniques. Additionally, a number of performance indicators are employed to assess these structures and methodologies. This article offers a thorough justification of the DL models used to depict various plant diseases. Additionally, several study gaps are noted from which to acquire greater transparency for spotting plant illnesses even before they manifest themselves. These days, crop disease diagnosis is completely automated because of the use of digital cameras and computer vision, prepared by clever deep neural network models using various methods. Therefore, the goal of the research is to totally automatically, fast, and flexibly classify plant illnesses (J.G.A. Barbedo 2017).

Plant diseases in the agriculture industry are to blame for significant global economic food losses. Food losses resulting from pathogen-caused crop illnesses, such as bacteria, viruses, and fungus, are ongoing problems. The fact that infections are now spread more readily than ever across the globe complicates the matter even further. Crop protection is essential to reducing disease-induced damage to crops throughout growth. Traditionally, farmers or professionals with some training or experience would identify crop inspection and plant problems. The continuous monitoring required by this human system made it expensive and impractical for larger fields. Even seasoned agronomists and plant pathologists struggle to identify many cultivated plant diseases because of their complexity and variety.

Researchers have come up with numerous strategies to solve the aforementioned issues. Machine learning can classify plant diseases using many different kinds of feature sets. The classic handcrafted and deep learning (DL)-based feature sets among them are the most well-liked feature sets. Before effectively extracting features,

preprocessing, such as picture enhancement, color modification, and segmentation, is necessary. Afterward feature extraction, various classifiers may be employed. K-nearest neighbor (KNN), support vector machine (SVM), decision tree, random forest (RF), naïve-bayes (NB), logistic regression (LR), rule generation, artificial neural networks (ANNs), and Deep CNN are some examples of well-liked classifiers.

Two steps went into the domain's survey analysis: gathering appropriate literature and detailed analysis and review of the work. the literature mentioned in section 2.3.2 was collected with a keyword-based search for reputed journal articles. The main parameters for the search were the "Deep learning" for ["identifying plant diseases" OR ["classification of plant diseases"]. In the following phase, scrutinize the whole study of Several papers and discover the significance of DL and ML techniques in the field of crop disease identification with the below mentioned research queries in mind:

- Which agricultural disease issues were they trying to solve?
- What DL/ML models are used and What type of data is being utilized?
- The notion of pre-processing is applied in which manner?
- How effective is the methodology employed in the suggested research?

3.2 Steps By Step Procedure for Crop diseases Identification

Four phases are primarily involved in the fundamental steps for identifying plant diseases, which are the four processes of image attainment, image preprocessing, feature extraction, and classification as shown in figure 3.1.

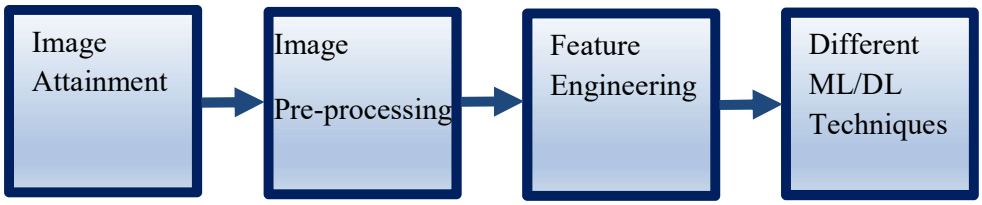


Figure 3.1 Four steps for classification of crop disease using a leaf image

3.2.1 Image Attainment

It is a technique that transforms attained images into the necessary output format for additional processing. To deal with images and before analyzing them the most important thing is to capture the image Scholars may use their own images or any standard dataset, such as the New Plant Village dataset.

3.2.2 Image Pre-processing

With the aim of highlighting the area of interest (the disease-infected area) in plant leaves, image preprocessing is used (Hanson et al. 2017). Image pre-processing frequently entails image segmentation, image augmentation, and color space conversion (Liu and Zhou 2009; Husin et al. 2012; Yao et al. 2009; Liu et al. 2017). First, a filter improves a digital image (Al Hiary et al.2011, Mokhtar et al. 2015, Prasad et al. 2016, Sannakki et al. 2013, Semary et al. 2015).

The background image is filtered to remove the leaves, and RGB colors are transformed into a color space parameter (Dandawate and Kokare 2015). A meaningful portion of the image is further split to make it easier to analyze. This process helps to achieve the desired goal.

3.2.3 Feature Engineering

There are two basic and significant processes involved in feature engineering. Feature selection and feature extraction in the field of digital image processing. However, the employment of feature selection methods is required since the majority of real-world issues can be solved using a large number of photos, and because the data collected from some types of images has a high dimensionality. Semary et al. (2015) has suggested that to create feature vectors, features are taken from the image. This extraction could involve signal processing, statistical analysis, or structural analysis. For instance, color moments are employed to extract color statistics.

Prasad et al. has proved that the combination of Gabor Transform (GT) and Wavelet Transform (WT) is used to recover multiscale features. Many earlier works used the Gray Level Co-occurrence Matrix (GLCM) to extract texture information. GLCM is a 256*256 matrix where the matrix counts the co-occurrences of each line at each position. The image under analysis's color and column colors. Transform using Scale Invariant Features (SIFT) is employed to examine the form characteristics of leaves the benefit automated feature extraction of deep learning holds a positive contribution to greater precision in the end in comparison to other traditional methods.

3.2.4 Classification

The final stage uses a classification model to pinpoint the plant disease that is present in the leaf. Known infection photographs and learning algorithms should be used to train the model. In the section that follows, classification algorithms are explained.

3.2.4.1 Diseases Identification Approaches

ML/DL approaches and image processing-based techniques can be used to categorize disease identification. Some machine learning approaches that may perform on huge datasets must follow image processing techniques for the aim of disease identification. On the other hand, textual attribute-based data that does not require images can also be used with machine learning algorithms, while further data cleaning and pre-processing procedures should be followed, image processing techniques are not required for disease detection which utilize attribute-based tables.

3.2.4.2 Image Processing Approach

Techniques for image processing were successfully applied widely for the precise identification and classification of the plant. M. Egmont et al. (2002) divided different image processing algorithm applications into categories. To increase the effectiveness of illness detection, a variety of pre-processing techniques, such as picture clipping, image smoothing, and image enhancement, are used. Several techniques, including the Otsu method, k-means clustering, and transforming the RGB image into the HIS model, can be used to segmentation of an image. Using a variety of image processing techniques, such as the Gabor filter, a methodology has been developed for the early and accurate diagnosis of plant diseases. Kulkarni and Patil have developed an ANN-based classifier to obtain identification accuracy of up to 91%. In order to ensure that chemicals should only be applied to the diseased chilli plant, Husin et al. (2012) recorded the image of a chilli plant leaf and processed it to identify the health condition of the chilli plant. Revathi et al. (2012) has used Canny and Sobel filters, to detect the edges that support identifying disease spots. They suggested a homogeneous pixel counting method and assert that it outperforms existing algorithms by 98.1% in terms of accuracy in detecting cotton illnesses. To address low-level picture segmentation, Jaware et al. (2012) suggested a fresh and enhanced k-means clustering technique. With the use of the spatial grey-level dependence matrices approach, statistical texture features were recovered (Dhaygude and Kumbhar 2013).

3.2.4.3 ML/DL Approach

According to the purpose, the ML/DL approach attention to procedures that are able to learn on their own from a specified set of input data. The agriculture sector has new prospects thanks to its high-speed computing. Due to its potential to increase the

sensitivity of disease identification and diagnosis, machine learning and statistical pattern recognition has attracted a lot of attention in the agriculture arena (Jain et al. 2005, Sajda 2006). To the best of the literature's knowledge, Jain et al. (2005) were the first to introduce the idea of machine-learning approaches for disease diagnostics in the agricultural domain. Later, utilizing benchmarking dataset available at the UCI repository, Upadhyaya et al. (2006) examined the idea of employing a clustering approach of machine learning for the identification of soybean disease. This method used a data set that wasn't from India. For the dataset gathered in India under real-time conditions, Jain et al. (2009) explored and evaluated the potential of their machine learning models based on Decision Tree (DT) induction utilizing C4.5, Rough Set (RS), and hybridized rough set-based decision tree induction (RDT).

In the proposed methodology, the main focus on the ML/DL approach to identify diseases in the plant, as it has a significant amount of advantage over the traditional image processing methods.

3.3 Some Admired ML/ DL Techniques

There are many ML/DL methods which are very useful in classification problems. As mentioned in section 2.3.1, the use of these techniques is wide and lots of research has been carried out for plant disease classification. A few popular techniques are explained below.

3.3.1 Decision Tree (DT)

Modeling decisions and results using decision trees is one technique to map decisions in a branching structure. Decision trees are used to estimate the likelihood that various iterations of decisions will be successful in achieving a particular goal. As a decision tree can be used to manually model operational decisions like a flowchart, the idea of a decision tree predates machine learning. They are frequently taught and used as a method of analyzing organizational decision-making in the business, economics, and operation management fields. Basic structure of decision tree is shown in figure 3.2.

Decision trees are a type of predictive modelling that can be used to map several options or solutions to a certain result. Different nodes make up decision trees. The decision tree's root node, which in machine learning typically represents the entire dataset, is where it all begins. The leaf node is the branch's termination point or the

result of all previous decisions. From a leaf node, the decision tree won't branch out any further. With decision trees in machine learning, the core nodes represent the data's features, and the leaf nodes represent the results.

In supervised machine learning, which trains models using labeled input and output datasets, decision trees are a method employed. The method is mostly used to address classification issues, which include categorizing or classifying an object using a model. Regression issues are a machine learning application where decision trees are also used to predict outcomes from unobserved data.

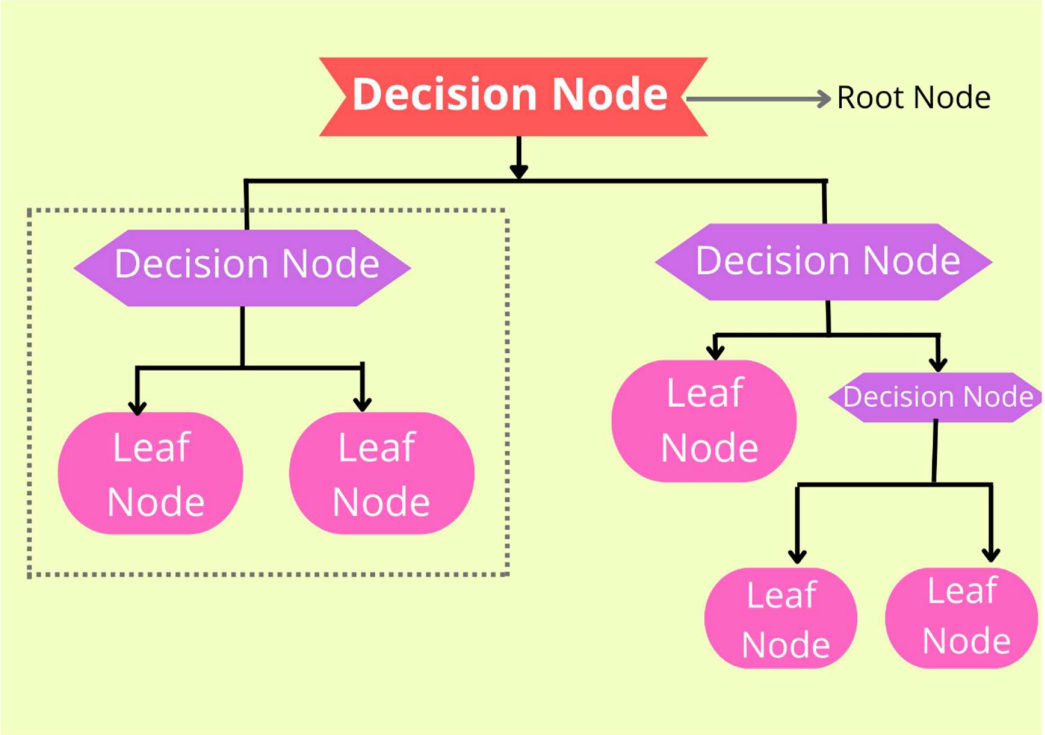


Figure 3.2: Decision Tree Architecture

Due to their ease of use and popularity in machine learning, decision trees are a common model structure. The model's decision-making process is also easily understood because of the tree-like structure. The process of communicating a model's output to a human is a significant consideration in machine learning. It is often challenging to describe a given model's output because machine learning excels at task optimization without direct human input. A decision tree structure makes it easier to understand the reasoning behind a model's decision-making process because each decision branch can be seen. For the dataset gathered in India under real-time

conditions, Jain et al. (2009) explored and evaluated the potential of three machine learning models based on Decision Tree (DT) induction utilizing C4.5, Rough Set (RS), and hybridized rough set -based decision tree induction (RDT). A classification method known as a DT produces a tree and a set of rules that reflect the model of several classes from a given dataset. DT induction was carried out using a Java implementation of C4.5 that the authors refer to as CJP. RJP, an RDT version, combines the advantages of both the RS and DT induction methods.

3.3.2 Support Vector Machine (SVM)

SVM is one of the most well-liked techniques for supervised learning, and it may be applied to both classification and regression issues. However, it is largely employed in Machine Learning Classification issues. The SVM algorithm's objective is to establish the best line or decision boundary that can divide n-dimensional space into classes, allowing us to quickly classify fresh data points in the future. A hyperplane is a name given to this optimal decision boundary. SVM selects the extreme vectors and points that aid in the creation of the hyperplane. Support vectors, which are used to represent these extreme cases, are the basis for the SVM algorithm as shown in figure 3.3.

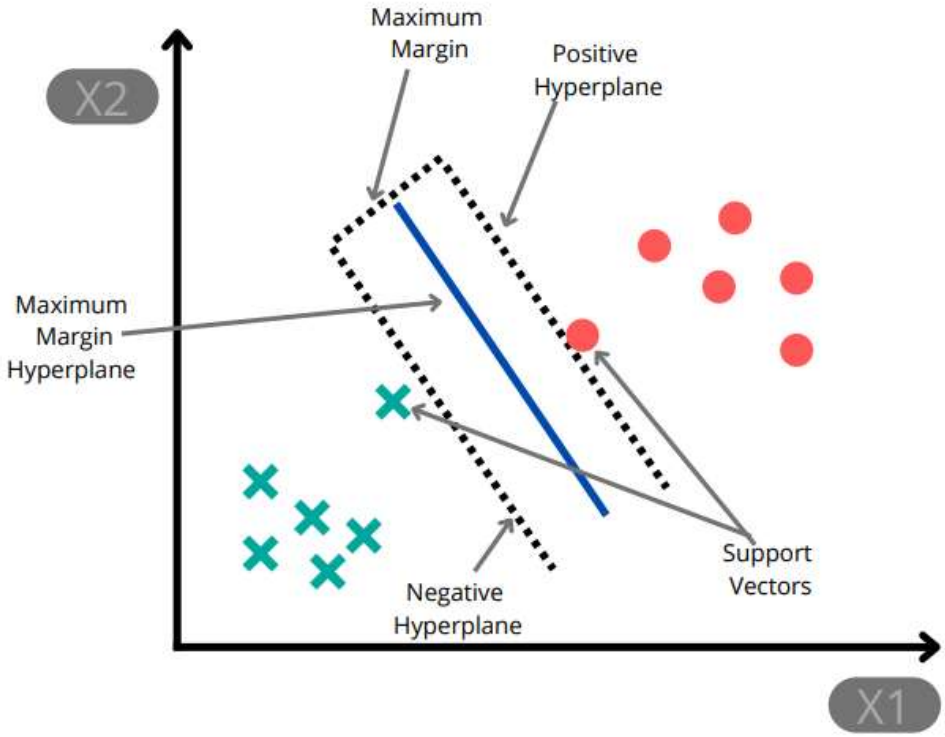


Figure 3.3: SVM Architecture

3.3.3 Convolutional Neural Network

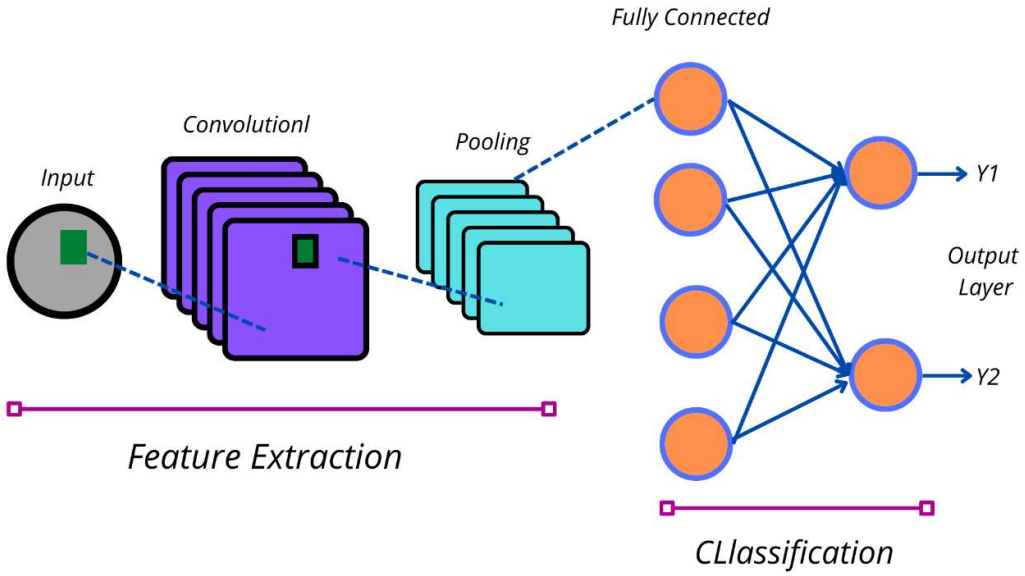


Figure 3.4: Schematic diagram of a basic convolutional neural network (CNN) architecture

Neural networks can be used to extract patterns and detect patterns that are too difficult to be seen by the human brain or computer algorithms because of their exceptional capacity to extract meaning from complex data. Adaptive learning, self-organization, real-time operations, and other benefits of ANNs are also present.

Convolution, pooling, and fully connected layers make up CNN's three primary layers. Convolution's main objective is to automatically extract features from each input image as shown in figure 3.4 It is made up of a number of teachable filters. Sliding windows are used to apply each filter on the image's raw pixel values, computing the dot product between each filter and input pixel as they do so. The feature map, which is a two-dimensional activation map of the filter, is the consequence of this. As a result, the network develops filters (such as edges and curves) that will turn on when certain traits are detected in the input. Throughout the training process, CNN discovers the values of these filters on its own. Subsampling layers come after these convolution layers.

Each sub-sampling layer shrinks the size of the convolution maps and adds invariance to any possible input rotations and translations, no matter how little. The

greatest activation value across all sub-windows in each feature map, which reduces the dimensionality of the feature is the output of the pooling layer. In order to calculate the class scores, the fully connected layer of the model is based on the SoftMax activation function. A vector of features derived from the learning process serves as the input to the SoftMax classifier, and its output is a probability that an image belongs to a certain class. Deep learning has just begun to gain ground in the field of plant pathology, notably in the classification of leaf images and the identification of plant diseases.

In this method, the training phase involves the automatic extraction of characteristics from the data. The suggested research has reviewed studies using small and large datasets with 500 to 87848 images to demonstrate the superiority of deep models over state-of-the-art techniques (Amara et al. 2017, Fuentes et al. 2017, Ferentinos 2018, Brahimi et al. 2017, Lu et al. 2017, Zhang et al. 2017, Liu et al. 2017, DeChant et al. 2017, Oppenheim and Shani, 2017). Section 2.3.2 that deep learning outperforms conventional image processing methods and offers excellent accuracy (Kamilaris and Boldu, 2018). Other approaches based on traditional machine learning and image processing technologies only work in limited and confined environments. Pawara et al. (2017) concluded that CNNs outperform traditional approaches after comparing the effectiveness of several conventional pattern recognition techniques with that of CNN models in the identification of plants using three different datasets. Through deep learning, Nigam et al. (2019) created a convolutional neural network model to identify plant diseases using photos of healthy and yellow rust-infected leaves from wheat crops. A set of 2000 photos that were taken at the ICAR Indian Agricultural Research Institute's experimental field were utilized to train the models. The classification accuracy of the CNN architecture-based model for distinguishing between healthy and yellow rust-infected leaves was 97.37%. The suggested CNN model can accurately and successfully identify plants with yellow rust infection, according to the application to the diagnosis of wheat disease.

3.4 SCRATCH LEARNING AND TRANSFER LEARNING

The comparison of various models for maize crop leaves is done in the proposed scheme. The proposed method first creates a straightforward convolutional architecture from scratch photos, and then, it go on to transfer learning, where the idea of hyper-parameter tuning is used at the top layers of the learned model. Here, the recommended

research makes good use of training data to get the most accurate classification of disease severity.

A few convolutional layers with one or more filters, followed by fully linked layers, make up the simple architectural model. Here, after training straightforward models with 1, 3, 5, and 13 convolutional layers, softmax normalisation is used. 32 filters with 3 X 3 Rectified Linear Units activation make up each convolution layer (ReLU). Now, a 2 X 2 max polling layer follows every other layer. The maximum polling layer has 64 filters. Each of the 64 units in the first completely connected layer has ReLU activation. Just following the initial layers is the dropout layer with a 50% ratio. Transfer learning is a useful method for building a powerful classification network that has been pre-trained on a huge number of entities, like ImageNet. In this situation, determining the severity of a disease is more crucial, and it competes with ImageNet for the best picture classification task. The bottom layers are in charge of encoding only the first information. The proposed research examines the architectures of the VGG16, VGG19, MobileNetv2, Xception, and ResNet152v2 for transfer learning (Xihai et al. 2018, Guan et al. 2017, G. Hinton et al. 2012].

Resnet152v2 has the highest accuracy out of all of them and is ranked top. The VGGNet offers a significant improvement over earlier configurations with a design based on exceedingly tiny convolution filters. It has 16 (VGG16) and 19 (VGG19) weight layers. Batch-wise and ReLU activation are methods used by MobileNet. The depth-wise separable convolutions are used first, followed by the width multiplier and the resolution multiplier.

Mobile Net's processing power is incredibly low, making transfer learning impossible. The precision of the outputs was severely impaired as a result, making it perfect for supporting portable devices, embedded systems, and PCs without a GPU or with low processing capacity. There are 36 convolutional layers in the Xception model. The 14 sets make up the 36 convolutional layers. Every set, with the exception of the first and last set, has linear residual connections everywhere around it.

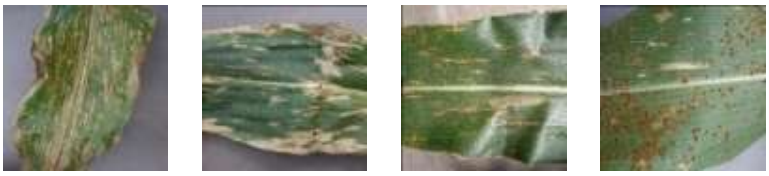
ResNet displays a setup known as a residual learning unit that helps deep neural networks function better. Here, transfer learning is used with all of this to produce the desired results. In the next session, it should clearly explain how the experiment is executed with different kinds of crops with their different kinds of diseases. The model is developed with above mentioned different architecture with hyper-parameter tuning concept. At last an excellent model is prepared.

3.5 EXPERIMENTAL SETTINGS

3.5.1 Material Analysis

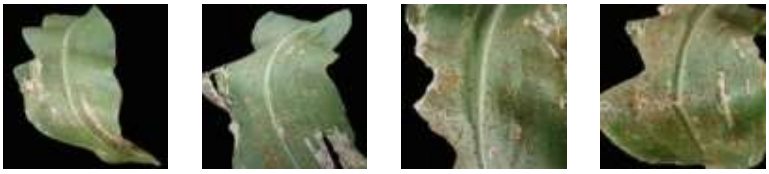
Anyone may use the new plant diseases image dataset because it is a free resource. It comprises over 60000 pictures of both healthy and ill crops spread across 38 directories with various class names. As a first input dataset, we choose photographs of the maize crop leaves. The photos of leaves are divided into four categories: healthy, leaf spot, leaf blight, and rusty. The leaf spot categories include brown spot, round spot, curvularia spot, grey leaf, and dwarf mosaic.

[A] Healthy



[B] Leaf Spot

[C] Leaf Blight



[D] Rust

Figure 3 .5: Original Maize Crop Leaves Images

Each image is a member of a specific class. The healthy class of leaves are spotless and have the structure indicated in figure 3.5(A). The Leaf Spot leaves, as depicted in figure 3.5 (B), have a single circular spot with an approximate radius of 2.5mm. The leaves of the Leaf Blight class have a greater number of spots, which may have combined to form a larger mass of sick tissue as illustrated in figure 3.5 (C). In figure 3.5 (D), rust class leaves are depicted. They can be dropped from trees and are completely infected. 9145 photos of various varieties of maize leaves overall are used in this work to construct a classification method. Here, 7316 photographs are provided for the system's training, and the remaining 1829 images are used to assess the method. We divided the train photos into two portions to test the algorithm. The remaining 20% of images are used to test the model after it has been developed using the other 80% of images. The bifurcations of various class kinds and the proportionate number of photos are displayed in Table 3.1. Here, a fair no. of samples from each class should be used to decrease bias. For every type of class, the suggested method chooses 80% of the images as the training dataset and 20% of the images as the test dataset. In the dataset, validation entities are provided individually. The suggested research avoids any form of grouping in this situation since it worsens approximation loss.

- The Train Dataset has 5851 samples.
- The Validation Dataset has 1829 samples.
- The Test Dataset has 1465 samples.

Class	No. of images for Training	No. of images for Validation	No. of images for Test
Healthy	1487	465	372
Leaf Spot	1313	410	329
Leaf Blight	1526	477	382
Rust	1525	477	382
Total No. of images	5851	1829	1465

Table 3.1 No. of Samples for Training, Validation and Test for Maize Crop Dataset

3.5.2 Model Development

The new plant diseases dataset samples are RGB pictures with a resolution of 256 x 256. First, a generic model for illness prediction is created from scratch using photos.

The suggested approach doesn't use any pre-trained weight at this time. The photos are then resized to 224 × 224 for use with VGG16, VGG19, MobileNetv2, Xception, and ResNet152v2. Here, the proposed method puts the model's optimization and prediction into action. The value of each pixel must be divided by 255 in order to be compatible with the various architectures. Following all of this, normalization is done to enhance end-to-end training. The recommended method calculates the mean value m_i and standard deviation s_i for each input i. The input is then transformed to $i'= (i-m_i)/s_i$. It should be noted that any augmentations, specifically for this dataset, harm model performance and should be disregarded.

3.5.3 CNN Approach

The foundation of CNN's architecture is made up of many Convolutional and pooling layers. Just after they are layers that are fully connected. It determines for any input i from the j^{th} Convolutional layer,

$$I_{jc} = \text{ReLU}(Y_j * i) \quad \text{.....(3.1)}$$

Here, the convolution function is represented by *, while the layer's convolution kernels are represented by Y_j. Y_j is equal to $[Y_{j1}, Y_{j2},...Y_{jw}]$. In this case, w stands for the number of convolution kernels in the layer. With m being the window size and n is the number of channels, all kernels Y_{jw} are m X m X n weight matrices. The rectified linear unit is ReLU (i) = max (0, i). It works for these models as an activation function. Compared to saturated nonlinear functions like Sigmoid, ReLU trains considerably more quickly. The max-pooling layer determines the highest value for all convolution kernel outputs over non-overlapping rectangular regions. Position invariance is made possible over wider local regions by the pooling function, which also helps to reduce output size. The completely connected layers are inserted on top of the final convolution layers. $\text{ReLU}(Y_{fci})$, where i is the input and Y_{fc} is the weight matrix for the fully connected layer.

$$E(Y)=\frac{-1}{n}\sum_{ij=1}^{n}\sum_{w=1}^{w}[y_{jw}\log P(i_j=w) + (1-y_{jw})\log(1-(i_j=w))] \quad \text{..... (3.2)}$$

The loss function is applicable to estimate the difference between the expected outcome and the input's class name. Cross entropy is simply added to create the loss function. The weighting matrices of the convolutional and fully connected layers are displayed here in Y. Here, n stands for the number of training samples, and j and w stand for the order number of training samples and classes, respectively.

If the j^{th} sample comes from the w^{th} class, then $y^{jw} = 1$. P (i_j=w), which is a function of the model's parameters Y, is the likelihood that input i_j will belong to the w^{th} class, as predicted by the model if y_{jw}=0 otherwise. Here, the parameters for the loss function are Y. Finding the value of Y that minimizes the output of the loss function E, is the aim of the network training procedure.

3.5.4 Optimizer Function

Here we select the technique known as Adaptive Moment Estimation (Adam) counts the rates of adaptive learning for all parameters. Both, an exponentially decaying average of previously squared gradients (y_t) and an exponentially decaying average of previous gradients (x_t) are kept in storage. As shown below, the suggested method calculates the decaying averages of the past and past squared gradients, respectively.

$$x_t = \beta_1 x_{t-1} + (1 - \beta_1) g_t \quad \text{..... (3.3)}$$

$$y_t = \beta_2 y_{t-1} + (1 - \beta_2) g_t^2 \quad \text{..... (3.4)}$$

Here, the gradients' first moment (mean) and second moment (variance) are estimated using the notation x_t and y_t, respectively. Additionally, both x_t and y_t have a bias towards zero, especially at early time steps and when the decay rate is low. By calculating the second moment and bias-correction, these biases can be eliminated. It is simple to change other parameters by using x'_t and y'_t.

$$x'_t = \frac{x^t}{1-\beta_1^t} \quad \text{..... (3.5)}$$

$$y'_t = \frac{y^t}{1-\beta_2^t} \quad \text{..... (3.6)}$$

For improved performance, a learning rate equal to 0.001 is employed here. The learning rate is a significant entity since it governs the size of the learning steps. When

the network begins to overfit, the suggested work uses early stopping, which is identical to the close policy, to interrupt the training process. Utilizing the validation loss, we assess the network's performance after each epoch. The network will terminate training once the loss value for the validation data stopped improving. After this, transfer learning is carried out in a manner that overcomes the challenge of over-fitting. All convolution layers employed identical weights throughout the entire network design during training, and top layers were excluded from the architecture. To match the number of categories included in the training dataset, suggested method has replaced the top layers with a single fully connected layer. By using this method, the recommended method is optimizing fully connected layers in order to train a model using its particular training dataset.

3.5.5 MODEL DEVELOPING PROCESS

Input	:	New Plant Village Dataset.
Output	:	Classification of different crop diseases.
STEP 1	:	Get the "New Plant Village" dataset from the "kaggle" machine learning repository.
STEP 2	:	Divide the dataset in half, with 80% going to training and 20% to testing.
STEP 3	:	Data pre-processing is carried out for batch size, image rescaling, and other purposes.
STEP 4	:	Choose an alternative basis model, such as ResNet152V2, MobileNetv2, Xception, VGG16, or VGG19.
STEP 5	:	Initially ‘base_model.trainable = False’ and apply transfer learning as mention below.

- First, incorporate a “GlobalAveragePooling2D()(x)” layer to reduce the number of parameters needed for the network to learn.
- A “Dropout” layer is added to overcome the problem of overfitting.
- Finally, a "Dense" layer is applied to provide the desired results.

STEP 6 : Select ADAM optimizer, epoch =100 and early stopping by monitoring validation loss with patience =5.

STEP 7 : Train the model and find the accuracy.

STEP 8 : Now test the model with test dataset.

STEP 9 : Now apply the concept of hyper parameter tuning, by repeating all the steps from step no.5 except 'base_model.trainable = True'.

STEP 10 : Find the accuracy and confusion matrix.

3.5.6 Test Result Examination for Maize Crop Dataset

The model was initially created from scratch, and no pretrained weight was used. Here, we can see that increasing the convolution layer will result in an improvement in both training and testing accuracy. However, if we continue to add layers after a certain point, both of these accuracies will decline because there won't be enough training data for models with more parameters. With 9 convolution layers, we were able to get the greatest test accuracy of 78.5%. Transfer learning is used with the current deep models to avoid this issue.

Transfer learning has been used, and the model layers have been frozen. The training accuracy is 98.78%, the validation accuracy is 96.44%, and the test accuracy is 97.13%. The idea of hyper parameter tuning is used to increase test accuracy even more. Figure 3.6 displays the outcomes of hyper-parameter adjustment for the ImageNet pretrained models. Each column shows the typical result over 20 epochs. The test set's ultimate accuracy ranges from 87.26% to 98.7%. When compared to models that were trained from scratch, output from fine-tuned models is higher. The ResNet152v2 model, which has an accuracy of 98.7%, produces the best results. The findings show that transfer learning is able to get around the problem of insufficient training data. One ANN model that was trained using the Adadelta optimizer and training data samples was produced for evaluation. We only get 36% test accuracy, which is comparable to accidental prediction. For the ANN to extract basic correlations and search for specific features inside the images, a convolutional feature extractor is necessary (Hinton et al. 2018). The graphs of accuracy and loss for training and validation with hyper-parameter tuning are shown in figures 3.7 (A) and 3.7 (B).

Table 3.2 provides the confusion matrix for the ResNet152v2 model on the current test set. For each of the four classes, the portion of successfully predicted photos is displayed.

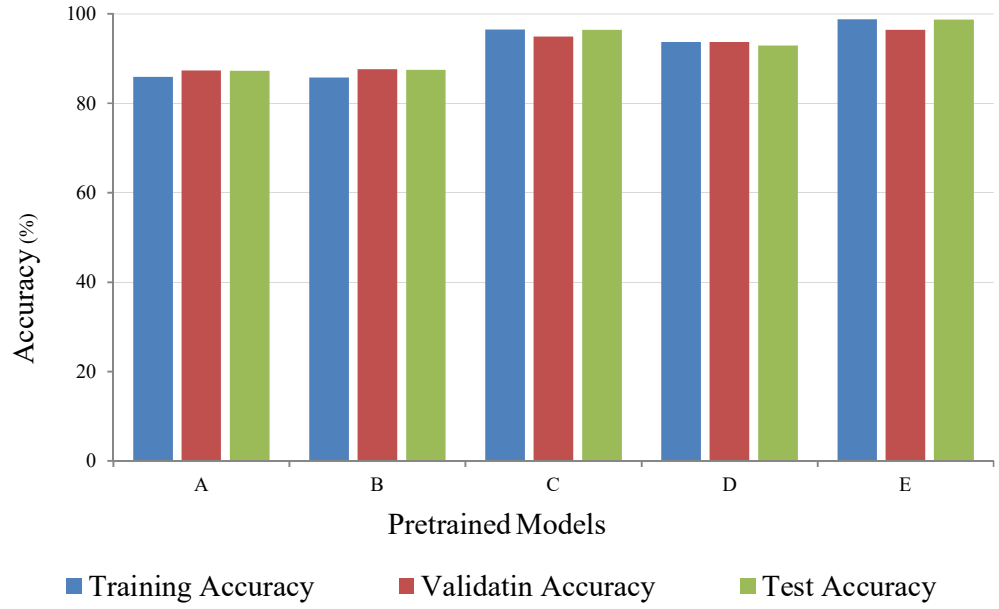
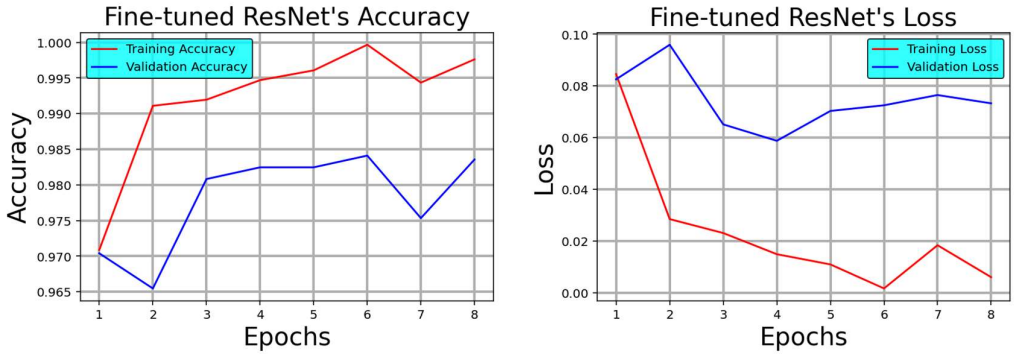


Figure 3.6 Bar Plots of Accuracies of different Deep Learning Models



[A] [B]

Figure. 3.7 (A) Plots of Accuracy and (B) Loss of ResNet152v2 after Hyper Parameter Tuning

The classification accuracy for the Healthy and Rusty classes of leaves is 100%. Leaf-spot class accuracy is 97.59%. Out of 329 photos, only 8 are incorrectly predicted. Images of the Leaf Blight have a 97.35% accuracy rate. Out of 368 photos, only 10 are incorrectly predicted. 1446 out of 1465 total photos were properly predicted. Precision, recall, and f1 score are all model parameters that are all equal to 0.99. In this case, the

neighboring class just mystifies the incorrect classes. In this instance, the classes of leaf blight and leaf spot are confused with one another.

		Predicted Result			
		Healthy	Leaf blight	Leaf spot	Rusty
Actual Class	Healthy	368	0	0	0
	Leaf blight	0	368	10	0
	Leaf spot	0	8	324	0
	Rusty	0	0	0	382

Table 3.2 Confusion Matrix for The Prediction of Resnet152v2 Model Trained with Transfer Learning for Maize Crop

3.5.7 Model Generalization

In order to assess this model's consistency, it is further tested on various plant datasets. To test the model, we use photos of apples, grapes, and tomatoes. We outperformed the maize dataset in terms of performance.

Images of apple leaves are divided into categories such as "Healthy," "Black Rot," "Scab," and "Rusty." Fungus is the cause of the apple scab and apple black rot diseases. The grape leaves are divided into four classes: “Healthy”, “Leaf Blight”, “Black Rot”, and “Esca (Black Measeles)”. Similar to apple, the fungus causes the two disease Black Rot and Esca (Black Measels) in grapes.

The tomato dataset has a lot more classes and is pretty big. There are integrated photos of all 10 different illnesses kinds. It comprises ailments brought on by fungus or by imbalanced weather. These include “bacterial spots”, “early and late blights”, “leaf mold”, “bacterial spots”, “spider mites”, “target spots”, “yellow leaf” “curl viruses”, and “mosaic viruses”. Each class sample for the display of diseases for a distinct crop is shown in Figures 3.8(a), 3.4(b), and 3.4(c).

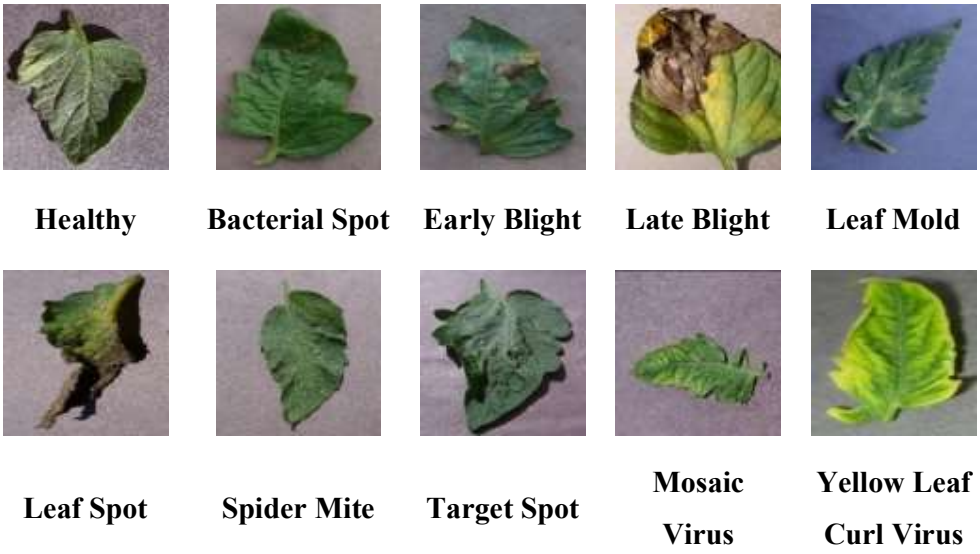
Tables 3.3, 3.4, and Table 3.5 display the bifurcations of various crop-class kinds and the proportional number of photos. Additionally, in the suggested technique, the researcher has collected an adequate number of samples from each class to reduce bias. The recommended method follows the same procedure for training, validating, and testing, that was done for the maize crop.

Healthy **Scab** **Black Rot** **Rusty**

(A) Apple Crop Leaves Images

Healthy **Esca** **Black Rot** **Leaf Blight**

(B) Grapes Crop Leaves Images



Healthy **Bacterial Spot** **Early Blight** **Late Blight** **Leaf Mold**

Leaf Spot **Spider Mite** **Target Spot** **Mosaic Virus** **Yellow Leaf Curl Virus**

(C)Tomato Crop Leaves Images

Figure 3 .8: Original Crop Leaves Images of (A) Apple (B) Grapes (C) Tomato

Apple:

- No. of samples for Train Dataset – 6220
- No. of samples for Validation Dataset-1943
- No. of samples for Test Dataset-1557

Grapes:

- No. of samples for Train Dataset – 5783
- No. of samples for Validation Dataset-1815
- No. of samples for Test Dataset-1448

Tomato:

- No. of samples for Train Dataset – 14686
- No. of samples for Validation Dataset-4585
- No. of samples for Test Dataset-3678

Class	No. of images for Training	No. of images for Validation	No. of images for Test
Rusty	1408	502	352
Scab	1612	497	404
Healthy	1611	504	403
Black Rot	1589	440	398
Total No. of images	6220	1943	1557

Table 3.3 No. Of Samples for Training, Validation and Test for Apple Crop Dataset

Class	No. of images for Training	No. of images for Validation	No. of images for Test
Leaf Blight	1377	430	345
Healthy	1353	423	339
Esca(Black Measles)	1536	490	384
Black Rot	1517	472	380
Total No. of images	5783	1815	1448

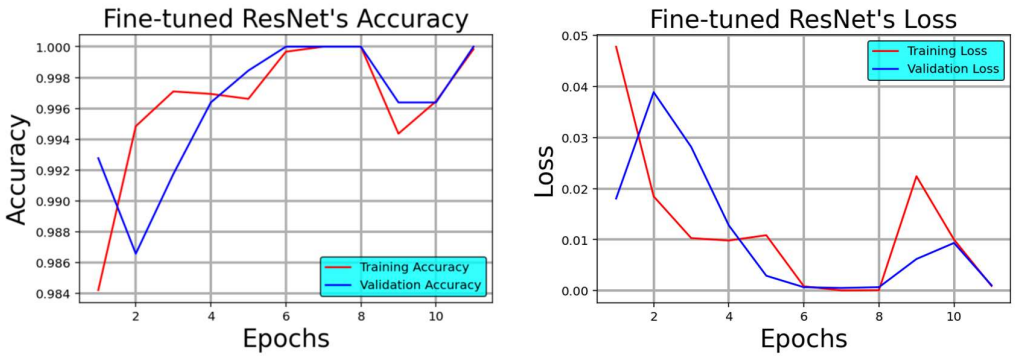
Table 3.4 No. Of Samples for Training, Validation and Test for Grape Crop Dataset

Class	No. of images for Training	No. of images for Validation	No. of images for Test
Bacterial Spot	1368	470	343
Early Blight	1536	448	384
Mosaic Virus	1440	463	360
Late Blight	1480	457	371
Target Spot	1461	425	366
Spider-mites Two-Spotted – Spider Mite	1392	490	349
Leaf Mold	1505	436	377
Yellow Leaf Curl Virus	1568	480	393
SeptoriaLeaf Spot	1396	481	349
Healthy	1540	435	386
Total No. of images	14686	4585	3678

Table 3.5. No. Of Samples for Training, Validation and Test for Tomato Crop Dataset

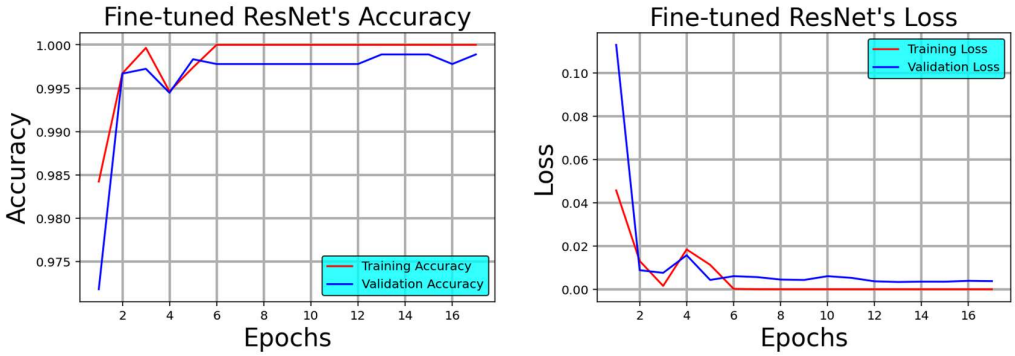
It is entirely acceptable if the accuracy of the model is closer to 100% when the early stopping is triggered based on the graphical analysis and confusion matrix. Since deep learning is data-driven, the test accuracy will rise steadily as we add more data to the training set. Here, it's important to keep in mind that the ResNet152v2 model outperformed the others. The performance of the VGGNet in the PlantCLEF plant identification task is demonstrated in the result, which is significantly superior to that of (S. T.Hang et al. 2016, M. Ghazi et al. 2017).

The performance of the model for different crops are graphically represented in figure 3.9 to 3.11 which shows the plots of, the accuracy and loss of the model for different crops. From the plots, it is pretty understandable that the model performs best with excellent accuracy and minimum loss. These results prove the strength of transfer learning and the ability of deep learning techniques for the classification of different diseases. The confusion matrices of the model for the corresponding crop is shown in table 3.6 to 3.8. The detailed discussion is in the following section.



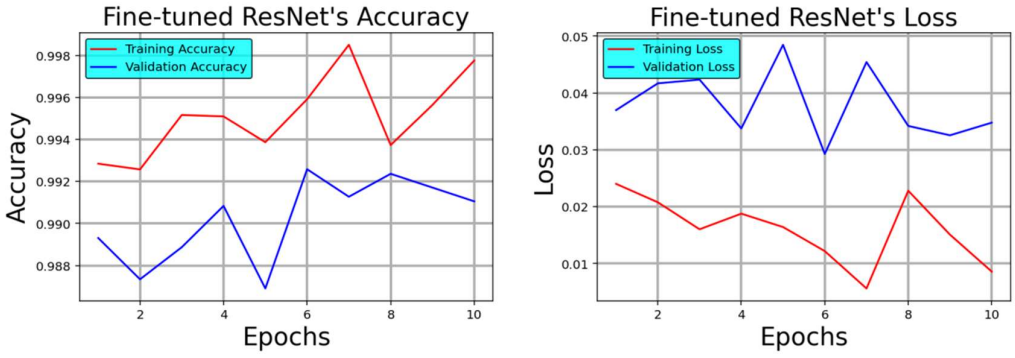
[A] [B]

Figure. 3.9 Plots of (A) Accuracy and (B) Loss of ResNet152v2 after Hyper Parameter Tuning for Apple Crops



[A] [B]

Figure 3.10 Plots of (A) Accuracy and (B) Loss of ResNet152v2 after Hyper Parameter Tuning for Grapes Crop



[A] [B]

Figure 3.11 Plots of (A) Accuracy and (B) Loss of ResNet152v2 after Hyper Parameter Tuning for Tomato Crop

		Predicted Result			
		Scab	Black Rot	Rusty	Healthy
Actual Class	Scab	404	0	0	0
	Black Rot	0	398	0	0
	Rusty	0	0	351	1
	Healthy	0	0	0	403

Table 3.6 Confusion matrix for the prediction of resnet152v2 model trained with transfer learning for apple crop.

		Predicted Result			
		Black Rot	Esca(Black Measles)	Leaf Blight	Healthy
Actual Class	Black Rot	380	0	0	0
	Esca(Black Measles)	2	382	14	0
	Leaf Blight	0	5	345	0
	Healthy	0	0	0	339

Table 3.7. Confusion matrix for the prediction of resnet152v2 model trained with transfer learning for grape crop.

		Predicted Result									
		A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
Actual Class	**A**	342	0	0	0	0	0	1	0	0	0
	B	0	380	2	0	0	1	1	0	0	0
	C	0	1	368	0	2	0	0	0	0	0
	D	0	0	0	376	0	0	0	1	0	0
	E	2	0	0	1	346	0	0	0	0	0
	F	0	1	0	0	0	346	2	0	0	0
	G	2	0	0	0	0	0	363	0	0	1
	H	0	0	0	1	0	0	0	392	0	0
	I	0	0	0	0	0	0	0	0	360	0
	J	0	0	0	0	0	0	0	0	0	386

Table 3.8. Confusion matrix for the prediction of ResNet152v2 model trained with transfer learning for tomato crop.

A:	Bacterial Spot	**B:**	Early Blight
C:	Late Blight	**D:**	Target Spot
E:	Spider-mites Two-Spotted –Spider Mite	**F:**	Leaf Mold
G:	Leaf Mold	**H:**	Yellow Leaf Curl Virus
I:	Septoria Leaf Spot	**J:**	Healthy

3.5.8 Results Analysis and Outcome Discussions

Crop Image	Maize	Apple	Grapes	Tomato
Model	ResNet152v2			
No. of Original Images	9145	9720	9046	22949
Optimizer	ADAM			
Test Accuracy	98.7%	99.93%	99.86%	99.48%
Loss	0.0523	0.0013	0.0024	0.0153
Precision	0.99	1.00	0.99	0.92
Recall	0.99	1.00	0.99	0.92
F1 Score	0.99	1.00	0.99	0.92

Table 3.9 Model Parameters of Different Objects for Proposed DNN Model

	Ref. [3]	Ref. [10]	Proposed work			
Crop Image	Maize	Apple	Maize	Apple	Grape	Tomato
Model	GoogLeNet	VGG 16	ResNet152v2			
No. of Original Images	500	1674	9145	9720	9046	22949
Optimizer	SGD	SGD	ADAM			
Test Accuracy in (%)	98.9	90.4	98.7	99.93	99.86	99.48

Table 3.10 Comparison of Proposed DNN model with Previous Work

In this study, we focused on a comprehensive approach to determining the severity of plant diseases. We create a deep neural network model from scratch and fine-tune five contemporary deep models: VGG16, VGG19 Xception, MobileNetv2,

and ResNet152v2. The model is trained with a small number of training data of varying depths. Contrary to test accuracy, we discovered that hyper-parameter adjustment is crucial for improving deep model performance.

The relevant parameter for each dataset is displayed in Table 3.9. For the dataset of apple, grape, and tomato, the test's accuracy is very close to 100%. In comparison to grape and maize, the precision, recall, and F1 score for the apple crop is 1.00. Because there are many surrounding classes that are comparable and close by, the values for the tomato dataset are a little bit lower, about 0.92.

The ResNet152v2 model that has been tweaked works best and attained a remarkable accuracy of 98.7% to 99.93% on the test set. And it is much superior to Guan Wang et al. (2017). The comparison of the proposed work with earlier research is shown in Table 3.10. The typical accuracy of other methods is 94.71% Xihai Zhang et al. (2018). While comparing the proposed work to the same reference, it should be noticed that the accuracy is nearly identical, but it is clear that our study involved 9165 photographs, which is a far larger dataset than the 500 images used in it. Additionally, they don't just gather photographs from many sources; in order to avoid over-fitting during training, they also expand the dataset, which is in some ways unreliable. On image datasets of apples, grapes, and tomatoes, the model performs better. The equipped model shows great promise for categorization across many datasets. We achieved an excellent 99.93% accuracy for the apple crop dataset. The apple-scab class, apple-healthy class, and apple-black-rot class were all correctly identified. Only one illustration of the rusty class was identified as the healthy class. The outcome is significantly superior to Guan Wang et al. (2017). The accuracy of the suggested method for the grape dataset was 99.86%. With 100% accuracy, the classes of black-rot, leaf blight, and healthy were identified.

Developers can fine-tune a solution to a particular problem by combining an approach from several different models with the use of transfer learning. A significantly more precise and potent model can be created by sharing information between two different models. The method enables the iterative construction of models. Any machine learning approach that uses simulated training must incorporate transfer learning as a crucial component. Digital simulations are a less expensive or time-consuming choice for models that need to be trained in real-world contexts and scenarios. The settings and behaviors of real life can be reflected in simulations. In a simulated world, models can be trained to interact with things. Transfer learning can

assist in applying the precise models created from huge training datasets to smaller sets of images. This includes transferring the model's more universal features, such as the method for spotting objects' edges in pictures. The model's more detailed layer, which is responsible for classifying different kinds of objects or shapes, can then be trained. Although the parameters of the model need to be adjusted and optimized, transfer learning will have already established the model's basic functionality.

An important component of deep learning, a branch of machine learning that aims to emulate and mimic the processes of the human brain, are artificial neural networks. Because of how complicated the models are, it requires a lot of resources to train neural networks. In order to increase process efficiency and decrease resource demand, transfer learning is applied. To speed up the creation of new models, any movable information or features can be transported between networks. A crucial component of creating such a network is the application of knowledge across various jobs or contexts. Transferred learning is typically restricted to universal procedures or tasks that are applicable in a variety of settings.