

TEA LEAF DISEASE DETECTION USING ML/DL

A Minor project report submitted in partial fulfillment of the requirements for the degree

of

Master Of Computer Application

Submitted by

NANDANA MEDHI (CSM21038)

SWEATA DAS (CSM21049)

Under the supervision of

Dr. JYOTISMITA TALUKDAR

Assistant Professor



School of Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

TEZPUR UNIVERSITY

NAPAAM- 784028, ASSAM

19-12-2023



Department Of Computer Science & Engineering

TEZPUR UNIVERSITY

Certificate By HoD

This is to certify that the dissertation entitled “**Tea Leaf Disease Detection Using ML/DL**” is submitted by **Nandana Medhi** bearing Roll no: **CSM21038** and **Sweata Das** bearing Roll no: **CSM21049**. They have completed their project work successfully as needed for partial fulfillment of the requirements and the regulations for the award of the degree of Master of Computer Application during the session 2021-2024 at Tezpur University. To the best of my knowledge, the matter embodied in the dissertation has not been submitted to any other university/institute for the award of any Degree or Diploma.

Date:

Place:

Head of the Department
Department of Computer Sc & Engineering
Tezpur University



Department of Computer Science and Engineering

Tezpur University

CERTIFICATE

This is to certify that the dissertation entitled “**Tea Leaf Disease Detection Using ML/AI**” submitted by **Nandana Medhi** bearing Roll no: **CSM21038** and **Sweata Das** bearing Roll no: **CSM21049** is carried out by them under my supervision and guidance for partial fulfillment of the requirements and the regulations for the award of the degree of Master of Computer Application during the session 2021-2024 at Tezpur University. To the best of my knowledge, the matter embodied in the dissertation has not been submitted to any other university/institute for the award of any Degree or Diploma.

Date:

Place:

Dr. Jyotismita Talukdar

Assistant Professor

Department of Computer Sc & Engineering
Tezpur University



Department of Computer Science and Engineering Tezpur University

DECLARATION

We hereby declare that the dissertation work titled “**Tea Leaf Disease Detection**” submitted to the Department of Computer Science & Engineering, Tezpur University was prepared by us and was not submitted to any other institution for the award of any other degree.

Date:

Date:

Place:

Place:

Nandana Medhi
CSM21038
Department of CSE
Tezpur University

Sweata Das
CSM21049
Department of CSE
Tezpur University

Acknowledgment

We would like to extend my heartfelt gratitude to our project guide **Dr. Jyotisma Talukar**, Assistant Professor, Department of CSE, Tezpur University, for allowing us to work under her and providing us ample guidance and support through the course of the project. We also express our sincere gratitude to all the teachers and friends who directly or indirectly have helped us to complete this project work.

Finally, we are grateful to our parents and all mighty God for providing me with good health, and support and making me stay motivated during this project.

(Nandana Medhi, Sweata Das)

Abstract

Numerous illnesses can affect tea leaves, which can have a substantial effect on the quantity and quality of tea produced. For prompt treatments and efficient illness management, early detection and diagnosis of these conditions are essential. Deep learning methods have produced encouraging results in automated image-based illness diagnosis systems in recent years. This study looks at the use of VGG16, ResNet, and Sequential—three pretrained convolutional neural networks (CNNs)—for the diagnosis of tea leaf illness.

Contents

1. Introduction
2. Initial system Study
 - 2.1 Background Study
 - 2.2 Problem Definition
 - 2.3 Objectives
 - 2.4 Scope of the Project
3. Feasibility Analysis
 - 3.1 Economical feasibility
 - 3.2 Technical feasibility
 - 3.3 Behavioral feasibility
4. Software Requirement Specification
5. System Methodology
 - 5.1 Dataset Collection
 - 5.2 Data Preprocessing
 - 5.3 Model Selection
 - 5.4 Train the Model
 - 5.5 Validate the Model
 - 5.6 Test the Model
 - 5.7 Evaluate the Model
 - 5.8 Convolutional Neural Network
 - 5.8.1 Convolutional Layer
 - 5.8.2 Pooling Layer
 - 5.8.3 Fully Connected Layer
 - 5.8.4 DropOut Layer
 - 5.8.5 ReLu Activation Function
6. Experimental results
7. Future Scope and Conclusion
8. Reference

Chapter 1

INTRODUCTION

A vital concern in the rapidly changing fields of agriculture and technology is maintaining the health and production of crops. The preservation of tea plants presents a number of difficulties for the tea business, which is an essential part of the world's agricultural economy. The **"Tea leaf Disease Detection"** initiative is a trailblazing attempt to use cutting-edge technology in agriculture in response to these difficulties.

One of the most popular drinks in the world, tea, is made from tea plants, which can suffer from a variety of illnesses that can have a big influence on productivity and quality. Ensuring the sustainability of tea crop and executing appropriate treatments depend on the prompt and precise detection of these illnesses. The amalgamation of contemporary technologies, namely in the domains of image processing and machine learning, has unparalleled prospects to transform the conventional techniques of disease identification in tea plants.

The goal of this research is to create a reliable and effective system for the early identification of illnesses affecting tea plants by utilizing the capabilities of artificial intelligence and computer vision. The goal is to develop a tool that can analyze visual data, such pictures of tea plant leaves, and accurately identify illness symptoms by using complex algorithms and trained models. In addition to increasing the timeliness of identification, this proactive strategy gives tea producers the ability to take focused, well-informed action to lessen the impact of illnesses on their harvests.

We shall examine the methods, tools, and anticipated results of the **"Tea leaf Disease Detection"** study in the parts that follow. This effort aims to support the sustainability and resilience of tea growing by embracing innovation at the nexus of agriculture and technology, assuring a prosperous future for this well-respected sector.

Chapter 2

INITIAL SYSTEM STUDY

2.1 Background Study

A survey on several classification methods that may be applied to the categorization of plant leaf diseases was provided by Savita N. Ghaiwat. Using the test case provided, k-nearest neighbor approach appears to be the most appropriate and straightforward procedure for class prediction. One of the problems with SVM is that it might be challenging to find the ideal parameters if the training data is not linearly separable.

The effectiveness of a pre-trained ResNet34 model in identifying crop illness is examined in the paper "Plant Disease Detection using CNN" from Emma Harte School of Computing, National College of Ireland, Mayor Street, IFSC, Dublin 1 Dublin, Ireland. The created model can distinguish seven plant illnesses from healthy leaf tissue and is implemented as a web application. For the purpose of training and verifying the model, a dataset of 8,685 leaf photos that were taken under controlled conditions is created. The suggested approach may get an accuracy of 97.2% and an F1 score of more than 96.5%, according to validation data. This shows that identifying plant diseases with CNNs is technically feasible and offers a way forward for small-holder farmers to use AI solutions.

"Identifying plant diseases with a shallow convolution neural network: The two approaches in the proposed methodology are shallow VGG with RF and VGG with Xgboost. The suggested model's ability to perform better with less parameters when compared to VGG19 and other deep learning models is one of its main features.

"Deep learning-based automatic recognition of tea diseases: After acquiring the local feature vectors of every sample picture, the proposed technique detailed the extraction and description of image features and built a visual language. Count the instances of each visual word in the image using vocabulary as a guide.

For picture recognition, Sumit Kumar, Veerendra Chaudhary, and Ms. Supriya Khaitan Chandra have presented a deep learning-based method. They have looked at the three primary neural network architectures: Single shot Multibook Detector (SSD), Region-based Fully CNN (R-CNN), and Faster Region-based Convolution Neural Network (Faster R-CNN). The system described in the study has the capacity to handle complicated situations and is capable of effectively detecting various illness kinds. The correctness of the validation result, which is 94.6%, illustrates the viability of the convolution neural network and points the way for an AI-based deep learning solution to this challenging issue.

For farmers, identifying illness signs with the unaided eye can be challenging. Using computerized image processing techniques that can identify damaged leaves based on leaf color information, crop protection is achieved, particularly in big farms. Numerous image processing approaches have been established to address difficulties using automatic categorization tools and pattern recognition, depending on the application.

2.4 Problem Definition

To find out the highest accuracy among four models. The input will be given as five random samples of healthy and diseased leaf using random() function and the project and it will show us the accuracy of four models of each diseases. The highest accurate model will be taken into consideration for any further work to concrete the limitations and reduce human labour.

2.5 Objectives

- Determine the impacted area using augmentation.
- Determine whether leaf parts are healthy and afflicted by using feature extraction and classification.
- For reliable results, train the models using testing data.
- Increased accuracy and reliability.
- Reducing human efforts.

2.6 Scope of the Project

The scope of a "Tea Plant Disease Detection" project can be extensive and may involve various aspects of technology, agriculture, and data analysis. Here are key components to be considered when defining the scope of such a project: Disease identification, Data collection, Image Processing, Machine learning models, Real time detection, accuracy etc.

Chapter 3

FEASIBILITY ANALYSIS

The assessment of potential effects on an organization caused by system development is known as feasibility. There might be a good or negative effect. The system is deemed viable when the positive indicates the negative. There are four techniques to carry out the feasibility study in this case: Operational, behavioral, economical, and technical feasibility.

3.1 Economic Feasibility:

Economic analysis is most frequently used for evaluation of the effectiveness of the system, more widely referred to as cost/benefit analysis, is figuring out the savings and benefits that may be anticipated from a system and contrasting them with the cost. Since all that is needed for the project, we built to function is a computer and a reliable internet connection, it is economically feasible.

3.2 Technical Feasibility:

The focus of technical feasibility is on how well the test management process's current manual system functions technically and to what degree. The technical requirements and the system's technical feasibility are examined in accordance with the feasibility analysis technique. such as inputs, procedures, and software facilities are recognized.

The system was designed with the latest and best technologies available on the market. This project uses the modern technologies like Machine Learning, Deep learning, Image processing which are common technologies to work with now-a-days. Technical staff should have no problem modifying and maintaining the project as and when required.

3.3 Behavioral Feasibility:

Behavioral feasibility, in the context of a project like "Tea Plant Disease Detection," assesses whether the intended users (farmers, plantation managers, etc.) are likely to accept and adopt the proposed solution. As the designed is simplest, it will be easy to use for all the end users, also there is no harm included in using the system. So the proposed method is be Behavioral feasible.

Chapter 4

Software Requirement Specification

General Description:

Purpose:

The purpose of the Tea Plant Disease Detection System is to develop a software solution that aids in the early detection of diseases affecting tea plants.

Scope:

The system will analyze images of tea plants to identify and classify potential diseases, providing timely information for farmers to take corrective actions.

Functional Requirements:

Image Input:

The given dataset will contain images with different diseases.

Image Processing:

Implementation of the image processing algorithms to identify regions of interest related to potential diseases.

Disease Classification:

Developed classification model will be capable of identifying various diseases affecting tea plants.

Accuracy Display:

Display of the results of the analysis, indicating the presence of diseases.

Non-functional Requirements:

Performance:

The system processes images within a reasonable time frame to ensure timely detection.

Accuracy:

The classification models have achieved a high level of accuracy in disease identification.

System Constraints:

Hardware Components

- **Device:** Lenovo IdeaPad Flex 5
- **Processor:** AMD Ryzen™ 7 5700U Processor (1.80 GHz up to 4.3GHz)
- **RAM:** 16 GB Soldered LPDDR4x 4266MHz
- **Storage:** Drive 512 GB SSD M.2 2280

Software Environment

The software environment includes the operating system, development tools, and libraries used for building and deploying the system.

1. **Operating System:** Windows 11
2. **Development Environment:** Visual Studio Code (VS Code)
3. **Programming Language:** Python

System Development Platform

1. **Development Language:** Python 3.11.7
2. **Library and Framework:**
 - TensorFlow, Version: 2.12.0
 - Keras, Version: 2.12.0
 - OpenCV
 - Matplotlib
 - Scikit-learn
 - NumPy

Chapter 5

System Methodology

To understand the methods to be used we gathered some information from already published papers. We have seen the types of diseases that have been detected and also the method and the model. Some models gave lesser accuracy and some gave higher. From the gathered information we have found following diseases that we are going to work on.

Few common tea leaf diseases:

- Gray blight
- Brown blight
- White spot
- Algal leaf
- Red leaf spot
- Healthy

We have also collected the information on accuracy from the already published papers to see which model would be appropriate to use. Here are some comparisons of papers and used models in different times. From gathered information we choose CNN as appropriate model. CNNs are a type of deep learning algorithm specifically designed for image analysis and pattern recognition. They automatically learn and extract meaningful features from images. CNNs have shown remarkable success in various computer vision tasks, including the detection and classification of diseases in tea leaves.

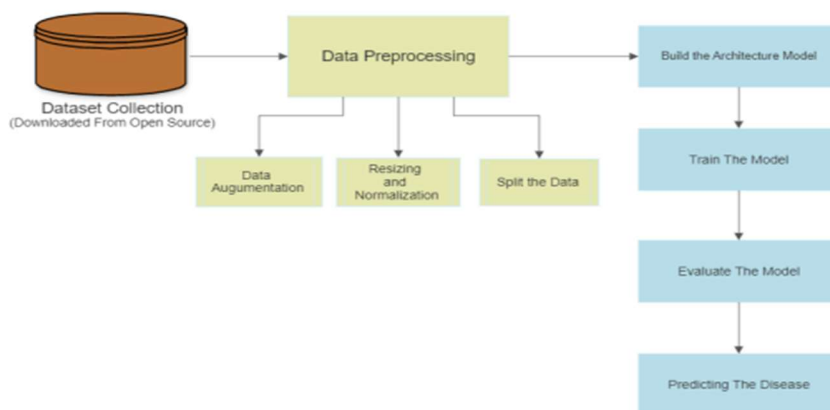


Fig 5.1: flow diagram

5.1 Dataset Collection

The database is collected from Kaggle to perform our experiment. The size of the dataset is 1GB. It has total 5867 images. This dataset contains tea leaves showing 6 common diseases of tea:

- **Red spot:** 1000 images
- **Algal spot:** 1000 images
- **Brown blight:** 1000 images
- **Gray blight:** 867 images
- **Healthy:** 1000 images
- **Heliopolis:** 1000 images

We have used 80% of the dataset for Training, 20% for validation and for Testing.

5.2 Data Preprocessing

To improve the quality and get the tea leaf photos ready for training, preprocess them. Resizing the photos to a uniform size, leveling the pixel values, and enhancing the dataset using methods like rotation, flipping, and zooming to boost its variety are typical preprocessing operations.

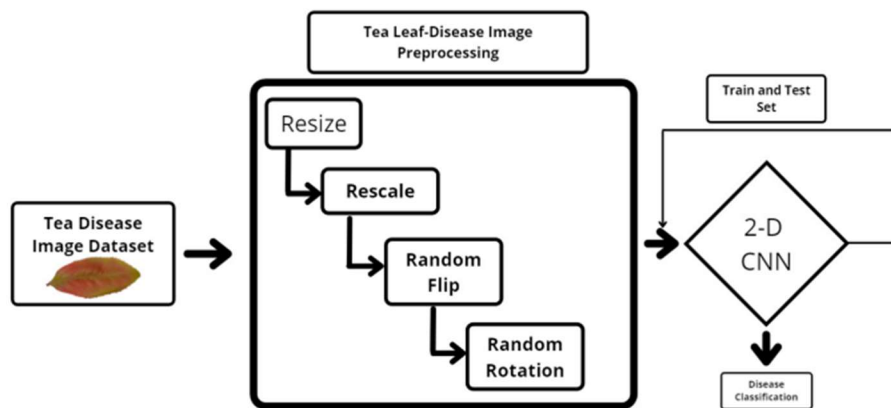
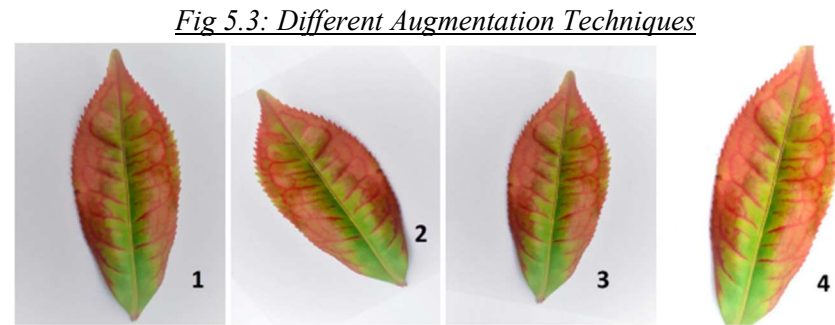


Fig 5.2: Augmentation method

- **Rescaling:** The Rescale option is used as 1./255. It guarantees that the pixel values are within appropriate range to train the models.

- **Data Loading and splitting:** The dataset is loaded using `flow_from_directory` function. Set Class mode: "Categorical". Set the batch size. Splitting data into training and validation subset into 80:20.
- **Data Augmentation:** Data Augmentation is applied to generate additional images artificially to increase the size of the dataset.



(a) Original,(b)Skew and Random Left Rotation;(c)Random Right Rotation;(d)Scaling and Rotation with Change in contrast

- **Data Augmentation Techniques:** Rotation, Flipping, cropping, zooming.

5.3 Model Selection

Selected the deep learning models to be applied in the illness detection process. The models selected in this instance are VGG16, ResNet, and Sequential. These models have shown to be highly effective in tasks involving picture categorization.

5.4 Train Model

Divide the previously processed dataset into sets for testing, validation, and training. Utilizing the training set, train the VGG16, ResNet, and Sequential models, then use the validation set to assess how well they performed. Modify hyperparameters to maximize the models' performance, such as batch size and learning rate.

5.5 Validate the Model

After training, the model is evaluated on the validation set using the `evaluate` method. This is done to assess the model's performance on data it has not seen during training. The evaluation results, including validation loss and accuracy, are printed to the console.

5.6 Test the Model

Testing a model involves evaluating its performance on a set of data that it has not seen during training or validation. To test a model in the context of machine learning, you typically use a separate dataset called the test set.

5.7 Evaluate the Model

Evaluate the trained models to assess their accuracy, precision, recall, F1 score, support and confusion matrix. There are various metrics to evaluate different machine learning methodologies performance. The most common seven metrics, *accuracy*, *precision*, *recall*, *specificity*, *F1 score*, *loss function*, and *confusion matrix*, are used to evaluate the proposed method's performance.

The recognition accuracy of the framework is determined by *Mean Average Precision*. It is the basic measurement used to perceive objects for every class. Mean Average Precision is calculated by dividing the number of correct detections for every one of the classes over the aggregate of several correctly detected and the number of incorrectly detected images. Mean average precision is observed for different types of parameters. These parameters include minimum batch size, the picture scale that is additionally the short edge of the picture, and the scaled input picture's maximum pixel size. Mean average precision is calculated for each class/object detected in the image. Average precision calculates the average precision over 0 to 1 esteem for recall value using the following formula:

$$P = \frac{\text{No of True detection}}{\text{No of True detections} + \text{No of False detections}}$$

A confusion matrix is a table used in machine learning to assess a classification algorithm's performance. It may also be applied to multi-class situations; however, it is most helpful for binary classification problems. The True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) measurements make up the confusion matrix. Numerous more performance measures, including recall, accuracy, precision, and F1 score, may be computed using these measurements. Here's the basic structure of a confusion matrix:

		Predicted class	
		<i>P</i>	<i>N</i>
Actual Class	<i>P</i>	True Positives (TP)	False Negatives (FN)
	<i>N</i>	False Positives (FP)	True Negatives (TN)

Fig 5.4: Basic Structure of Confusion Matrix

Table 5.1: Metric Equations and explanation.

Metric	Equation	Measure
Accuracy	$\frac{T_P + T_N}{T_P + T_N + F_P}$	A measure of the ratio of all correct classifications to the total number of the classifications
Precision	$\frac{T_P}{T_P + F_P}$	The ratio of the true positive cases over the total classified positive cases
Recall	$\frac{T_P}{T_P + F_N}$	(Sensitivity) The measure of the proportion of the actual positive cases that were classified correctly
Specificity	$\frac{T_N}{T_N + F_P}$	The measure of the proportion of the actual negative cases that were classified correctly
F1-Score	$\frac{2T_P}{2T_P + F_P + F_N}$	The harmonic mean of the precision and recall

5.8 Convolutional Neural Network(CNN)

1. A convolutional layer.
2. A pooling layer
3. A fully connected layer
4. Dropout layer.

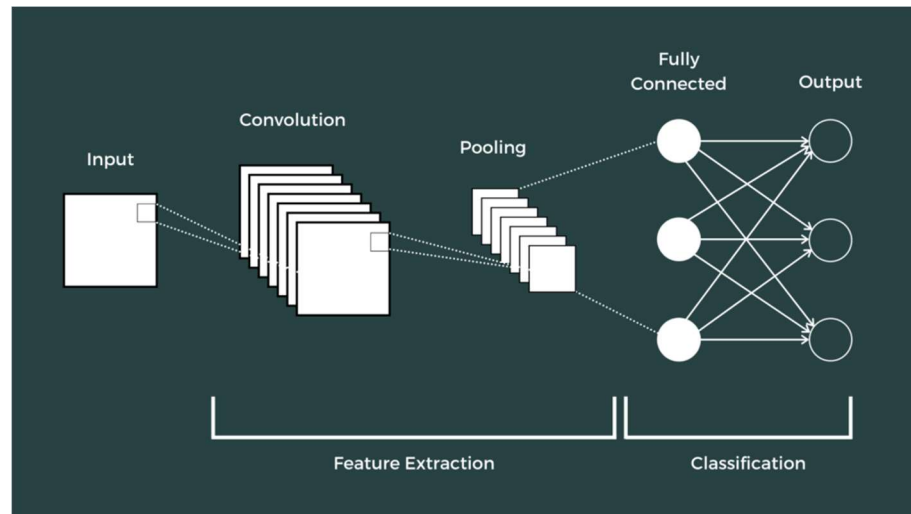


Fig 5.5: Basic CNN Architecture

5.8.1 Convolutional Layer

In convolutional neural networks (CNNs), a kind of deep learning architecture intended for processing structured grid input, like photographs, a convolutional layer is a fundamental building piece. By swiping a tiny filter, sometimes referred to as a kernel, across the input data and calculating the element-wise multiplication and summing of the filter and local input regions, convolutional layers apply a convolution operation to the input.

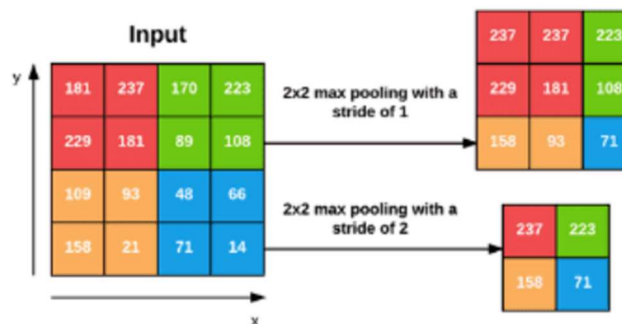


Fig 5.6: Convolution Layer

5.8.2 Pooling Layer

Convolutional neural networks (CNNs) frequently include pooling layers to lower the computational complexity of the network by downsampling the spatial dimensions of the input feature maps. There are two common kinds of pooling layers:

1. **Maximum Pooling:** When using max pooling, the maximum value from a specific area of the input feature map is chosen. Using max pooling to highlight dominating features works well.

2. **Average Pooling:** The process of average pooling entails calculating the mean value within a certain area of the input feature map. Its goal is to minimize the spatial dimensions and smooth the input data while maintaining an overall sense of the patterns.

5.8.3 Fully Connected Layer

A Fully Connected Layer (also known as a dense or fully connected layer) is a type of neural network layer where each neuron or node is connected to every neuron in the previous layer.

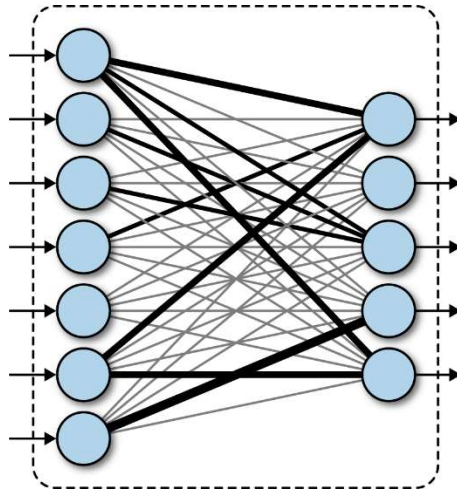


Fig 5.7: Fully Connected Layer

5.8.4 Dropout Layer:

The Dropout layer randomly sets a fraction of input units to zero during training. This helps prevent overfitting by reducing reliance on specific neurons and encourages the network to learn more robust features.

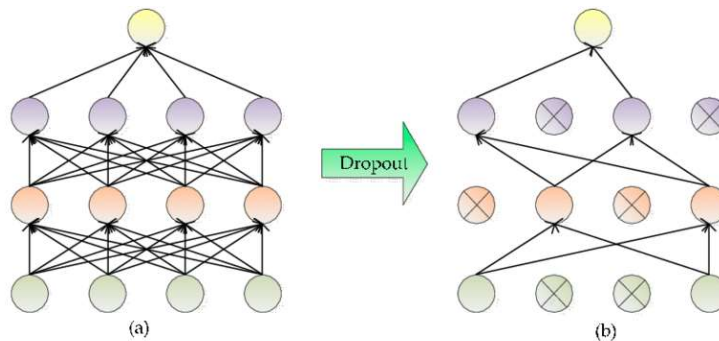


Fig 5.8: Dropout Layer

5.8.5 ReLU Activation Function:

ReLU is an activation function that outputs the input for positive values and zero for negative values. ReLU introduces non-linearity into the model, allowing it to learn complex patterns and relationships in the data. It is a popular choice for activation functions in hidden layers.

Mathematically, $\text{ReLU}(x) = \max(0, x)$.

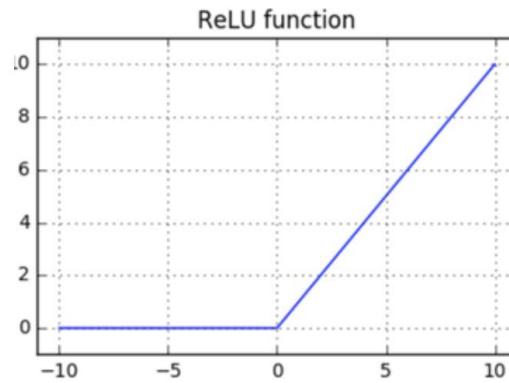


Fig 5.9: ReLu

Chapter 6

EXPERIMENTAL RESULT

7.1 MODEL ANALYSIS

In our tea leaf disease detection project, we employed various deep learning models to identify and classify diseases in tea leaves. Among the models trained, three of them achieved exceptional accuracy, showcasing their effectiveness in the task which are as follows:

1. RESNET Model

- ResNet (Residual Network) model, implemented using the Sequential API of TensorFlow, demonstrated remarkable performance in tea leaf disease detection.
- The training process involved the use of data augmentation, transfer learning, and fine-tuning to enhance the model's ability to generalize to unseen data.
- The ResNet model achieved perfect accuracy during the evaluation phase, accurately classifying tea leaf diseases with high precision.

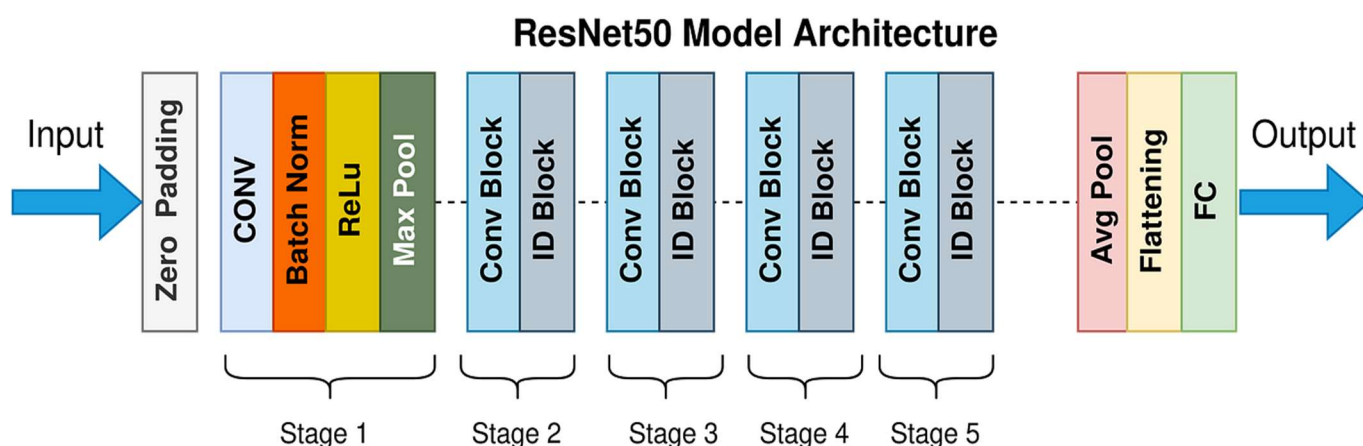


Fig 7.1.a: ResNet Model Architecture

A. Model Training

The model was trained using the following configuration:

- **Optimizer:** Adam
- **Learning Rate:** Adam optimizer
- **Loss Function:** Sparse Categorical Crossentropy
- **Batch Size:** 32
- **Epochs:** 20

Epoch 1/20
147/147 [=====] - 4s 16ms/step - loss: 0.5331 - accuracy: 0.8138 - val_loss: 0.2471 - val_accuracy: 0.9165
Epoch 2/20
147/147 [=====] - 2s 12ms/step - loss: 0.2458 - accuracy: 0.9141 - val_loss: 0.1933 - val_accuracy: 0.9293
Epoch 3/20
147/147 [=====] - 2s 12ms/step - loss: 0.1816 - accuracy: 0.9318 - val_loss: 0.1810 - val_accuracy: 0.9344
Epoch 4/20
147/147 [=====] - 2s 12ms/step - loss: 0.1503 - accuracy: 0.9465 - val_loss: 0.1274 - val_accuracy: 0.9497
Epoch 5/20
147/147 [=====] - 2s 12ms/step - loss: 0.1187 - accuracy: 0.9576 - val_loss: 0.1188 - val_accuracy: 0.9634
Epoch 6/20
147/147 [=====] - 2s 13ms/step - loss: 0.0995 - accuracy: 0.9665 - val_loss: 0.0943 - val_accuracy: 0.9651
Epoch 7/20
147/147 [=====] - 2s 12ms/step - loss: 0.0916 - accuracy: 0.9708 - val_loss: 0.1063 - val_accuracy: 0.9608
Epoch 8/20
147/147 [=====] - 2s 13ms/step - loss: 0.0737 - accuracy: 0.9738 - val_loss: 0.1039 - val_accuracy: 0.9574
Epoch 9/20
147/147 [=====] - 2s 12ms/step - loss: 0.0653 - accuracy: 0.9774 - val_loss: 0.0819 - val_accuracy: 0.9727
Epoch 10/20
147/147 [=====] - 2s 12ms/step - loss: 0.0470 - accuracy: 0.9840 - val_loss: 0.0937 - val_accuracy: 0.9676
Epoch 11/20
147/147 [=====] - 2s 12ms/step - loss: 0.0642 - accuracy: 0.9772 - val_loss: 0.0853 - val_accuracy: 0.9744
Epoch 12/20
147/147 [=====] - 2s 13ms/step - loss: 0.0476 - accuracy: 0.9832 - val_loss: 0.0869 - val_accuracy: 0.9753
Epoch 13/20
147/147 [=====] - 2s 13ms/step - loss: 0.0448 - accuracy: 0.9838 - val_loss: 0.0944 - val_accuracy: 0.9668
Epoch 14/20
147/147 [=====] - 2s 13ms/step - loss: 0.0504 - accuracy: 0.9798 - val_loss: 0.0852 - val_accuracy: 0.9719
Epoch 15/20
147/147 [=====] - 2s 12ms/step - loss: 0.0478 - accuracy: 0.9821 - val_loss: 0.0913 - val_accuracy: 0.9693
Epoch 16/20
147/147 [=====] - 2s 13ms/step - loss: 0.0353 - accuracy: 0.9876 - val_loss: 0.0553 - val_accuracy: 0.9804
Epoch 17/20
147/147 [=====] - 2s 13ms/step - loss: 0.0293 - accuracy: 0.9898 - val_loss: 0.0760 - val_accuracy: 0.9736
Epoch 18/20
147/147 [=====] - 2s 12ms/step - loss: 0.0316 - accuracy: 0.9896 - val_loss: 0.0715 - val_accuracy: 0.9779


```
Epoch 19/20
147/147 [=====] - 2s 12ms/step - loss: 0.0276 - accuracy: 0.9900 - val_loss:
0.0907 - val_accuracy: 0.9676
Epoch 20/20
147/147 [=====] - 2s 13ms/step - loss: 0.0336 - accuracy: 0.9885 - val_loss:
0.0707 - val_accuracy: 0.9770
```

B. Training Progress

The following plots illustrate the training and validation accuracy and loss over the epochs:

1. Accuracy
2. Model

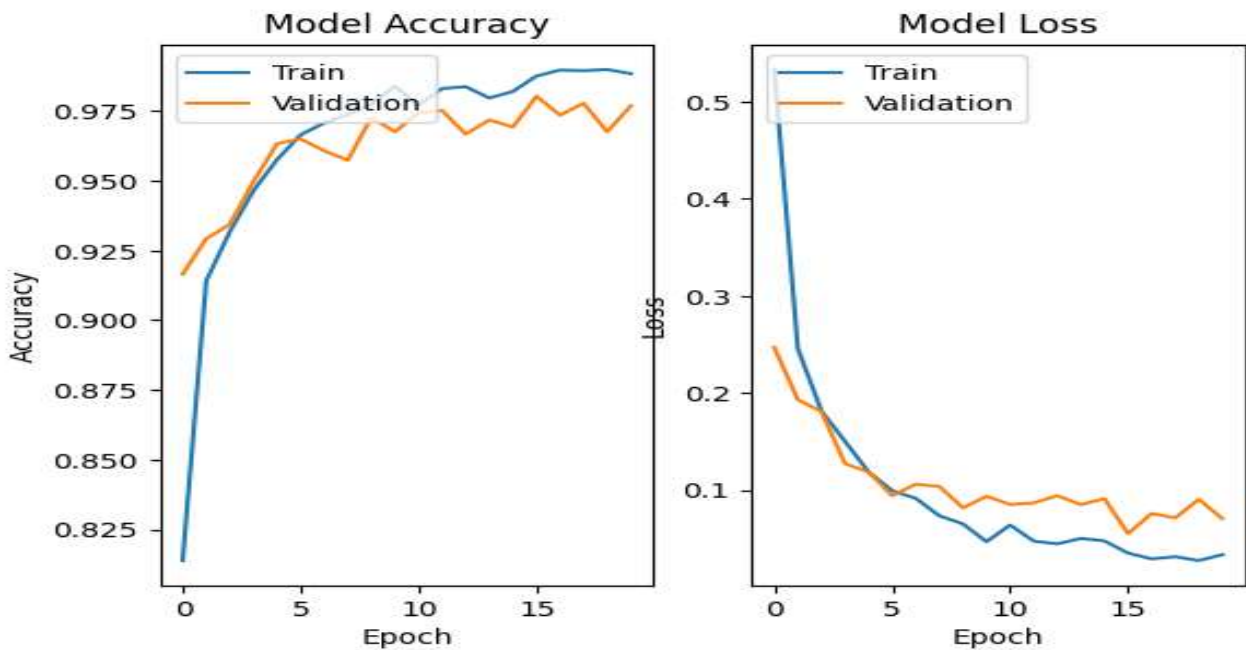


Fig 7.1.b: Accuracy and Model

C. Evaluation Metrics

The model was evaluated on the validation set, and the following metrics were obtained:

- **Validation Loss:** 0.07068120688199997
- **Validation Accuracy:** 0.9770017266273499

D. Confusion Matrix

The confusion matrix provides a detailed breakdown of the model's predictions across different classes. Each row represents the true class, while each column represents the predicted class.

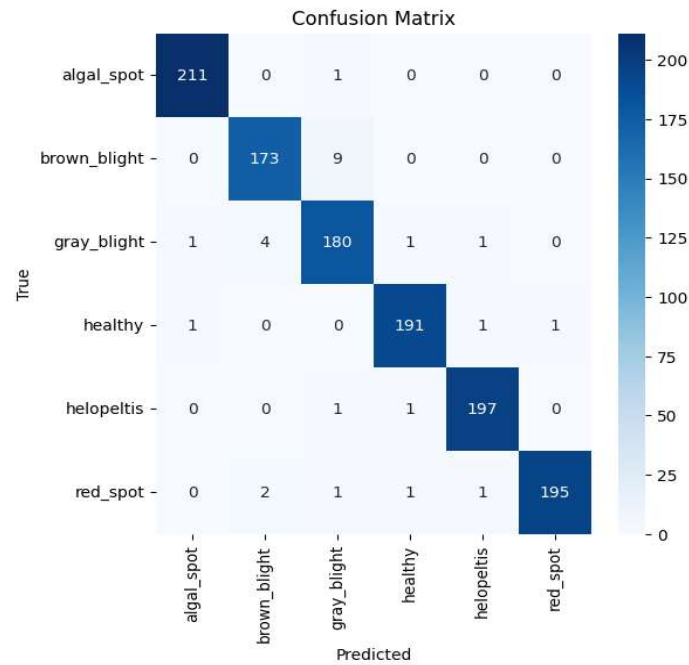


Fig 7.1.c. : Confusion Matrix

2. SEQUENTIAL Model

- The Sequential model, also implemented using TensorFlow's Sequential API, presented another successful approach to tea leaf disease detection.
- Data augmentation was applied during training to improve the model's robustness, and the use of transfer learning from a pre-trained VGG16 base contributed to its excellent performance.
- Similar to the ResNet model, the Sequential model achieved impeccable accuracy in identifying and classifying tea leaf diseases.

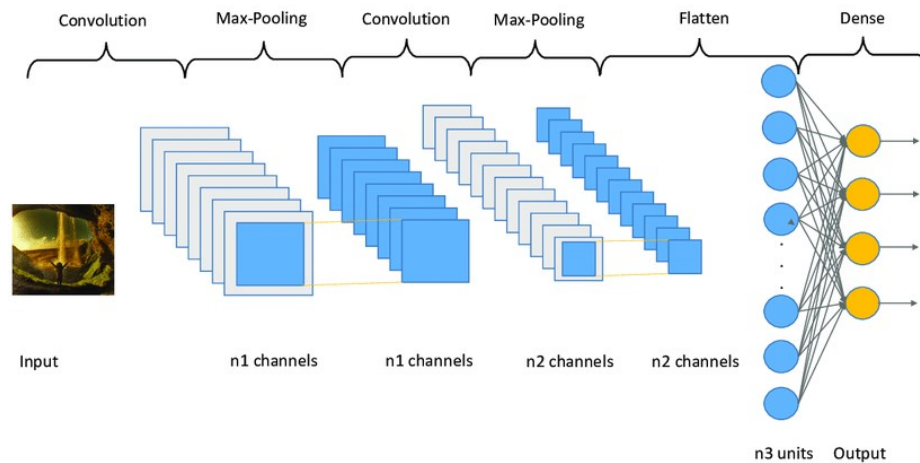


Fig 7.2.a: Architecture

A. Model Training

The model was trained using the following configuration:

- **Optimizer:** Adam
- **Learning Rate:** Adam optimizer
- **Loss Function:** Sparse Categorical Crossentropy
- **Batch Size:** 32
- **Epochs:** 20

```
Epoch 1/20
147/147 [=====] - 370s 2s/step - loss: 4.9641 - accuracy: 0.3458 -
val_loss: 1.3927 - val_accuracy: 0.4390
Epoch 2/20
147/147 [=====] - 360s 2s/step - loss: 1.2965 - accuracy: 0.4533 -
val_loss: 1.0829 - val_accuracy: 0.5550
Epoch 3/20
147/147 [=====] - 381s 3s/step - loss: 1.1522 - accuracy: 0.5175 -
val_loss: 0.9030 - val_accuracy: 0.6010
Epoch 4/20
147/147 [=====] - 347s 2s/step - loss: 1.0085 - accuracy: 0.5961 -
val_loss: 0.8505 - val_accuracy: 0.6368
Epoch 5/20
147/147 [=====] - 378s 3s/step - loss: 0.8429 - accuracy: 0.6785 -
val_loss: 0.6424 - val_accuracy: 0.7366
Epoch 6/20
147/147 [=====] - 365s 2s/step - loss: 0.7315 - accuracy: 0.7179 -
val_loss: 0.7124 - val_accuracy: 0.7212
Epoch 7/20
147/147 [=====] - 360s 2s/step - loss: 0.6385 - accuracy: 0.7486 -
val_loss: 0.5922 - val_accuracy: 0.7587
Epoch 8/20
```

```

147/147 [=====] - 340s 2s/step - loss: 0.5696 - accuracy: 0.7806 -
val_loss: 0.5278 - val_accuracy: 0.8116
Epoch 9/20
147/147 [=====] - 315s 2s/step - loss: 0.5072 - accuracy: 0.8134 -
val_loss: 0.6133 - val_accuracy: 0.7613
Epoch 10/20
147/147 [=====] - 312s 2s/step - loss: 0.4894 - accuracy: 0.8181 -
val_loss: 0.3876 - val_accuracy: 0.8602
Epoch 11/20
147/147 [=====] - 303s 2s/step - loss: 0.3855 - accuracy: 0.8637 -
val_loss: 0.3042 - val_accuracy: 0.9011
Epoch 12/20

147/147 [=====] - 332s 2s/step - loss: 0.2938 - accuracy: 0.8999 -
val_loss: 0.2864 - val_accuracy: 0.9062
Epoch 13/20
147/147 [=====] - 342s 2s/step - loss: 0.2827 - accuracy: 0.9009 -
val_loss: 0.3474 - val_accuracy: 0.8781
Epoch 14/20
147/147 [=====] - 290s 2s/step - loss: 0.2470 - accuracy: 0.9124 -
val_loss: 0.4393 - val_accuracy: 0.8542
Epoch 15/20
147/147 [=====] - 290s 2s/step - loss: 0.2483 - accuracy: 0.9178 -
val_loss: 0.4083 - val_accuracy: 0.8636
Epoch 16/20
147/147 [=====] - 300s 2s/step - loss: 0.2810 - accuracy: 0.9014 -
val_loss: 0.2866 - val_accuracy: 0.9096
Epoch 17/20
147/147 [=====] - 284s 2s/step - loss: 0.2373 - accuracy: 0.9239 -
val_loss: 0.2736 - val_accuracy: 0.9096
Epoch 18/20
147/147 [=====] - 272s 2s/step - loss: 0.2126 - accuracy: 0.9299 -
val_loss: 0.2758 - val_accuracy: 0.9199
Epoch 19/20
147/147 [=====] - 455s 3s/step - loss: 0.2074 - accuracy: 0.9295 -
val_loss: 0.1963 - val_accuracy: 0.9275
Epoch 20/20
147/147 [=====] - 346s 2s/step - loss: 0.1714 - accuracy: 0.9403 -
val_loss: 0.2821 - val_accuracy: 0.9079

```

B. Training Progress

The following plots illustrate the training and validation accuracy and loss over the epochs:

1. Accuracy
2. Model

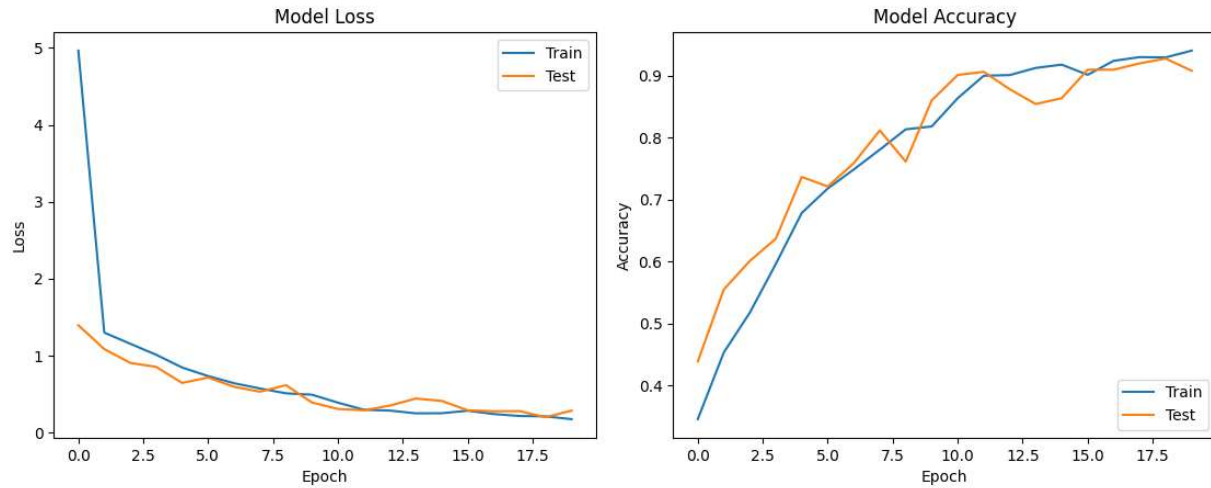


Fig 7.2.b: Graph for Loss and Accuracy

C. Evaluation Metrics

The model was evaluated on the validation set, and the following metrics were obtained:

- **Validation Loss:** 0.19274094700813293
- **Validation Accuracy:** 0.9369317889213562

D. Confusion Matrix

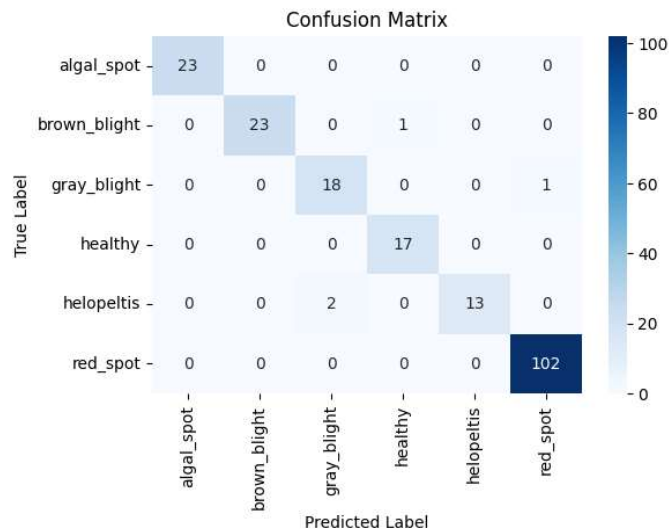


Fig 7.2.c: Confusion Matrix

3. VGG16 Model

- Our VGG16 (Visual Geometry Group 16) model, utilizing the VGG16 architecture pre-trained on ImageNet, proved to be a powerful tool for tea leaf disease detection.
- The model's features were extracted using the VGG16 base, followed by additional dense layers for classification.
- Training involved the use of data augmentation to diversify the dataset and improve model generalization.
- The VGG16 model achieved outstanding accuracy during the evaluation, demonstrating its capability to effectively recognize and categorize tea leaf diseases.



Fig 7.3.a: VGG16 Model Architecture

1. Model Training

The model was trained using the following configuration:

- **Optimizer:** Adam (0.001)
- **Learning Rate:** Adam optimizer
- **Loss Function:** Sparse Categorical Crossentropy
- **Batch Size:** 32
- **Epochs:** 20

```
146/146 [=====] - 139s 932ms/step - loss: 0.7581 - accuracy: 0.7313 - val_loss: 34.5313 - val_accuracy: 0.9037
Epoch 2/20
146/146 [=====] - 131s 890ms/step - loss: 0.2843 - accuracy: 0.9111 - val_loss: 23.7292 - val_accuracy: 0.9267
Epoch 3/20
146/146 [=====] - 133s 906ms/step - loss: 0.1899 - accuracy: 0.9376 - val_loss: 27.2304 - val_accuracy: 0.9327
Epoch 4/20
146/146 [=====] - 143s 974ms/step - loss: 0.1172 - accuracy: 0.9612 - val_loss: 26.9094 - val_accuracy: 0.9489
Epoch 5/20
146/146 [=====] - 142s 969ms/step - loss: 0.0909 - accuracy: 0.9712 - val_loss: 20.1928 - val_accuracy: 0.9489
Epoch 6/20
```

```

146/146 [=====] - 124s 845ms/step - loss: 0.0669 - accuracy: 0.9787 - val_loss:
18.7778 - val_accuracy: 0.9608
Epoch 7/20
146/146 [=====] - 122s 830ms/step - loss: 0.0588 - accuracy: 0.9795 - val_loss:
24.8544 - val_accuracy: 0.9506
Epoch 8/20
146/146 [=====] - 138s 943ms/step - loss: 0.0397 - accuracy: 0.9889 - val_loss:
24.5670 - val_accuracy: 0.9557
Epoch 9/20
146/146 [=====] - 128s 874ms/step - loss: 0.0392 - accuracy: 0.9857 - val_loss:
22.1381 - val_accuracy: 0.9566
Epoch 10/20
146/146 [=====] - 131s 891ms/step - loss: 0.0396 - accuracy: 0.9861 - val_loss:
30.1444 - val_accuracy: 0.9557
Epoch 11/20
146/146 [=====] - 126s 857ms/step - loss: 0.0309 - accuracy: 0.9902 - val_loss:
28.4189 - val_accuracy: 0.9523
Epoch 12/20
146/146 [=====] - 123s 839ms/step - loss: 0.0245 - accuracy: 0.9925 - val_loss:
34.2842 - val_accuracy: 0.9514
Epoch 13/20
146/146 [=====] - 122s 829ms/step - loss: 0.0229 - accuracy: 0.9934 - val_loss:
28.7438 - val_accuracy: 0.9540
Epoch 14/20
146/146 [=====] - 135s 921ms/step - loss: 0.0286 - accuracy: 0.9891 - val_loss:
42.8156 - val_accuracy: 0.9514
Epoch 15/20
146/146 [=====] - 130s 887ms/step - loss: 0.0275 - accuracy: 0.9915 - val_loss:
32.7781 - val_accuracy: 0.9574
Epoch 16/20
146/146 [=====] - 142s 971ms/step - loss: 0.0254 - accuracy: 0.9934 - val_loss:
28.3932 - val_accuracy: 0.9583
Epoch 17/20
146/146 [=====] - 167s 1s/step - loss: 0.0225 - accuracy: 0.9932 - val_loss:
29.0515 - val_accuracy: 0.9600
Epoch 18/20
146/146 [=====] - 182s 1s/step - loss: 0.0228 - accuracy: 0.9913 - val_loss:
25.8981 - val_accuracy: 0.9591
Epoch 19/20
146/146 [=====] - 194s 1s/step - loss: 0.0171 - accuracy: 0.9947 - val_loss:
36.3927 - val_accuracy: 0.9574
Epoch 20/20
146/146 [=====] - 190s 1s/step - loss: 0.0210 - accuracy: 0.9951 - val_loss:
35.2723 - val_accuracy: 0.9574

```

2. Training Progress

The following plots illustrate the training and validation accuracy and loss over the epochs:

1. Accuracy
2. Loss

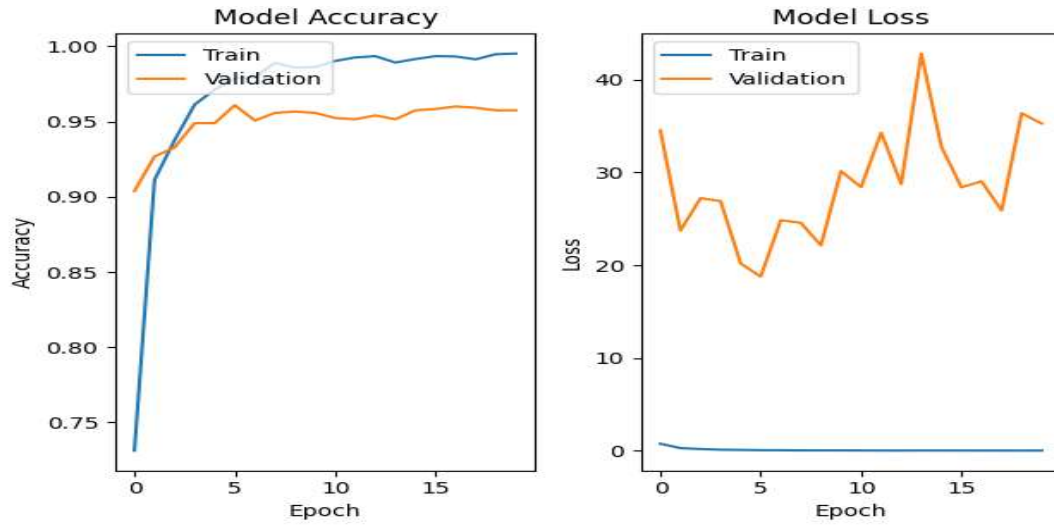


Fig 7.3.b: Comparison of Accuracy and Loss

3. Evaluation Metrics

The model was evaluated on the validation set, and the following metrics were obtained:

- **Validation Loss:** 35.27229309082031
- **Validation Accuracy:** 0.9574105739593506
-

4. Confusion Matrix

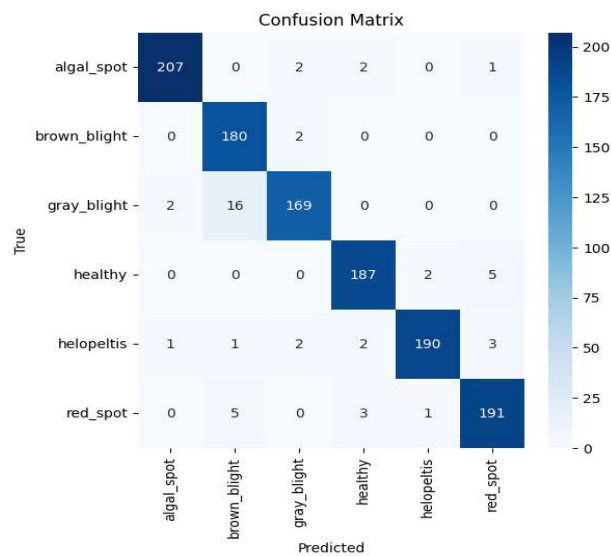

















Fig 7.3.c: Confusion Matrix

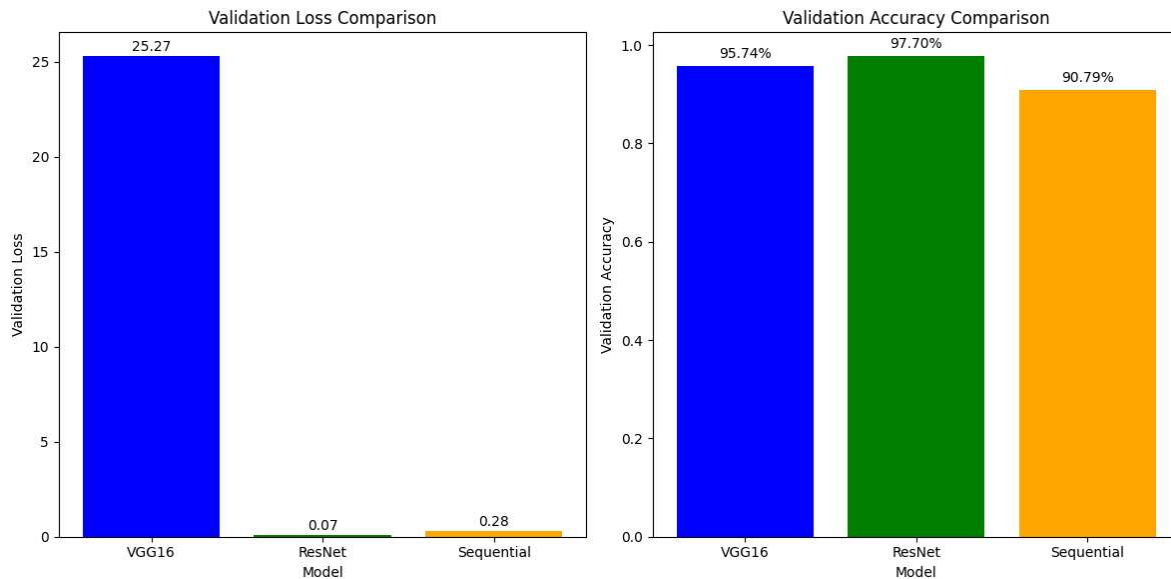
7.2 Precision, Recall and F1-Score values of the detection and identification results of different networks:

<u>Model</u>	<u>Classes</u>	<u>Precision</u>	<u>Recall</u>	<u>F1-Score</u>
RESNET	Algal Spot	0.99	1.00	0.99
	Brown blight	0.97	0.95	0.96
	Gray Blight	0.94	0.96	0.95
	Healthy	0.98	0.98	0.98
	Helopeltis	0.98	0.99	0.99
	Red Spot	0.99	0.97	0.98
Sequential	Algal Spot	1.00	1.00	1.00
	Brown blight	1.00	0.96	0.98
	Gray Blight	0.90	0.95	0.92
	Healthy	0.94	1.00	0.97
	Helopeltis	1.00	0.87	0.93
	Red Spot	0.99	1.00	1.00
VGG16	Algal Spot	0.99	0.98	0.98
	Brown blight	0.89	0.99	0.94
	Gray Blight	0.97	0.90	0.93
	Healthy	0.96	0.96	0.96
	Helopeltis	0.98	0.95	0.97
	Red Spot	0.95	0.95	0.95

7.3 Images with their true labels

<u>S.</u> <u>N</u> <u>O.</u>	<u>Model</u> <u>Name</u>	<u>Extracted Images Sample</u>				
1.	RESNET	True: brown_blight 	True: healthy 	True: healthy 	True: helopeltis 	True: gray_blight 
2	Sequental	Predicted: algal True: algal 	Predicted: redleaf True: redleaf 	Predicted: whitespot True: whitespot 	Predicted: graylight True: graylight 	Predicted: whitespot True: algal 
3.	VGG16	True: healthy 	True: algal_spot 	True: brown_blight 	True: red_spot 	True: helopeltis 

7.4 Comparison:



Based on above graph following conclusion:

1. VGG16 has a relatively high validation loss, suggesting that it may be struggling with generalization to unseen data. The validation accuracy is still reasonable but not as high as the ResNet model.
2. ResNet performs exceptionally well with an extremely low validation loss and the highest validation accuracy among the three models. This indicates that ResNet is likely better at capturing complex patterns in the data and generalizing well to new samples.
3. The Sequential model falls between VGG16 and ResNet in terms of both validation loss and accuracy. While it has a better validation loss than VGG16, it doesn't match the high accuracy achieved by ResNet.

Overall Conclusion:

ResNet emerges as the most promising model, boasting the lowest validation loss (**0.07 %**) and the highest accuracy (**97.70%**). VGG16, with its higher validation loss, faces challenges in generalization, while the Sequential model, while decent, falls short compared to **ResNet**.

Chapter 8

Future Scope and Conclusion

In image classification, Convolutional Neural Networks have emerged as a reliable technique that is increasingly utilized. The complexity needed for NN analysis is much reduced as compared to other methods, and the computation precision is also significantly raised. Furthermore, the high fault tolerance of CNNs makes it possible to employ blurry or unreadable background images, which greatly increases image recognition accuracy. With all these benefits, CNN turns out to be more efficient than other DL methods. In this study, a model for the classification of Tea Leaf illness has been built by considering a two-dimensional convolutional neural network with RGB photos as input. The suggested approach is capable of automatically identifying healthy and sick leaves and classifying tea leaf illnesses into four categories. As can be observed, recall and precision are 95% and 95%, respectively, while the classification's total accuracy is 97%. The findings support the notion that the suggested model is not over fitted. It will be possible to significantly increase the precision of identifying the illness in a leaf in the future. To identify other tea leaf illnesses, the suggested model might be improved even further. The approach may be implemented in real-world applications and coupled with Internet of Things devices. Moreover, the model may be expanded to include additional crop diseases.

Chapter 9

References

- [1] Shelar Nishant, Shinde Suraj , Sawant Subham , Dhumal Shreyash , Fakir Kausar.(2022). Plant Disease Detection Using CNN. ITM Web of Conferences 44, 03049 (2022).
<https://doi.org/10.1051/itmconf/20224403049>
- [2] Harte, Emma. (2020). Plant Disease Detection using CNN. School of Computing, National College of Ireland, Mayor Street, IFSC, Dublin 1 Dublin, Ireland.
10.13140/RG.2.2.36485.99048
- [3]M. Shobana; Vaishnavi S; Gokul Prasad C; Pranava Kailash SP; Madhumitha K P; Nitheesh C; Kumaresan N.(2022). 2022 International Conference on Computer Communication and Informatics (ICCCI). 10.1109/ICCCI54379.2022.9740975
- [4] Nilesh N. Thorat, Mayuresh Gulame, Aarti P. Pimplkar, Nilesh P. Sable, Pramod B. Dhamdhere, Nilesh Kulal.(2023). Plant Disease detection using CNN.
<https://doi.org/10.17762/ijritcc.v11i10.8492>
- [5] Liu Jun, Wong, Xvewel.(2021).Plant Diseases and Pest detection based on deep learning.
- [6] Kumar Sumit, Chaudhary Veerendra, Khaitan Chandra Supriya.(2021). Dept. of Computer Science and Engineering, Glagotias University, India, uttar Pradesh.Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021.
- [7] Hassan Mahmud SK.(2023). Revolusonizing Plant disease identification using convolutional neural network and deep learning. 10.1007/978-981-99-7240-1_33
- [8] Kolli Jahnavi, Vamsi Mohana Dhara, V. M. Manikandan.(2022). Plant Dieses detection using convolutional nural network. 2021 IEEE Bombay Section Signature Conference (IBSSC). 10.1109/IBSSC53889.2021.9673493
- [9] Muhammad Hammad Saleem, Johan Potgieter, and Khalid Mahmood Arif.(2019). Plant Disease Detection and Classification by Deep Learning. 10.3390/plants8110468