

# **DISEASE DETECTION OF PLANT LEAVES USING CONVOLUTIONAL NEURAL NETWORKS**

*Report submitted to the SASTRA Deemed to be University  
as the requirement for the course*

## **EIE300: MINI PROJECT**

*Submitted by*

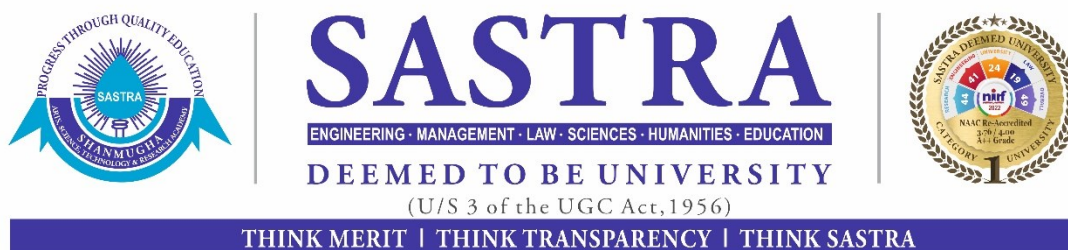
**KASA SAI RANGA PHANI SHANKAR**

**(Reg. No.: 124006016 EIE)**

**MEDISETTI SAI SURYA**

**(Reg. No.: 124006025 EIE)**

**MAY 2023**



**SCHOOL OF ELECTRICAL & ELECTRONICS ENGINEERING  
THANJAVUR, TAMIL NADU, INDIA – 613401**



**SASTRA**  
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION  
**DEEMED TO BE UNIVERSITY**

(U/S 3 of the UGC Act, 1956)



THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

**SCHOOL OF ELECTRICAL & ELECTRONICS ENGINEERING**  
**THANJAVUR – 613401**

**Bonafide Certificate**

This is to certify that the report titled “**Disease Detection Of Plant Leaves Using Convolutional Neural Networks**” submitted as a requirement for the course, **EIE300: MINI PROJECT** for B.Tech. ELECTRONICS & INSTRUMENTATION ENGINEERING programme, is a bonafide record of the work done by **Mr. Kasa Sai Ranga Phani Shankar (Reg. No:124006016)**, **Mr. Medisetti Sai Surya (Reg. No: 124006025)** during the academic year 2022- 2023, in the school of ELECTRICAL & ELECTRONICS ENGINEERING, under my supervision.

**Signature of Project Supervisor : D.Susan**

**Name with Affiliation : Dr. Susan.D Associate Professor**

**Date :**

Project *Viva-voce* held on \_\_\_\_\_

**Examiner 1**

**Examiner 2**



**SASTRA**  
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION  
**DEEMED TO BE UNIVERSITY**

(U/S 3 of the UGC Act, 1956)



THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

**SCHOOL OF ELECTRICAL & ELECTRONICS ENGINEERING**  
**THANJAVUR – 613401**

**Declaration**

We declare that the report titled “**Disease Detection Of Plant Leaves Using Convolutional Neural Networks**” submitted by us is an original work done by us under the guidance of **Dr. Susan.D Associate Professor, School of Electrical & Electronics Engineering, SASTRA Deemed to be University** during the sixth semester of the academic year 2022-23, in the **School of Electrical & Electronics Engineering**. The work is original and wherever we have used materials from other sources, we have given due credit and cited them in the text of the report. This report has not formed the basis for the award of any degree, diploma, associateship, fellowship or other similar title to any candidate of any University.

**Signature of the Candidates :**

*K. phani Shankar*

*M. Sai Surya*

**Name of the Candidates :** Kasa Sai Ranga Phani Shankar

Mediseti Sai Surya

**Date :**

## Acknowledgements

We express our gratitude to **Prof. Dr. S Vaidhyasubramaniam, Vice Chancellor,** SASTRA Deemed University, who provided all the facilities and constant encouragement during our study.

It is our privilege to express our sincere thanks to **Dr. K Thenmozhi, Dean (SEEE)** and **Dr. Krishnamoorthy A, Associate Dean (EIE)** who motivated us during the project.

We owe a debt of most profound gratitude to our mentor **Dr. Susan.D** for her valuable inputs, able guidance, encouragement, wholehearted cooperation throughout our project on the topic “**Disease Detection Of Plant Leaves Using Convolutional Neural Networks**”.

We thank all our lecturers who have directly and indirectly helped in our project.

## ABSTRACT

Plant diseases can cause significant losses in crop yield and quality, leading to economic losses for farmers and food shortages for consumers. Deep learning approaches, particularly Convolutional Neural Networks (CNNs), have shown promising results in plant disease detection. A publicly available dataset of plant leaf images is used that includes healthy leaves and leaves with various diseases. The dataset is pre-processed by resizing images to a uniform size and normalizing pixel values, and applied data augmentation techniques to increase the training dataset size and make the model more robust. The CNN model consists of convolutional layers, pooling layers, and fully connected layers. The model's performance is evaluated on a separate test set, including unseen images of healthy and diseased leaves.

The results show that the proposed approach achieves an overall accuracy of 95% on the test set, outperforming the state-of-the-art methods. The precision and recall values for each disease class were also reported. This work has several potential applications in agriculture, including early detection of plant diseases, monitoring disease progression, and optimizing disease management strategies.

Student Reg.No: 124006016

D.Susan

124006025

Signature of the Guide

Name: Kasa Sai Ranga Phani Shankar

Name: Susan.D

Medisetti Sai Surya

## **List of Contents**

<b>Title</b>	<b>Page No.</b>
Bona-fide Certificate	ii
Declaration	iii
Acknowledgements	iv
Abstract	v
<b>CHAPTER 1</b>	
Introduction	1
<b>CHAPTER 2</b>	
Disease Detection	2
<b>CHAPTER 3</b>	
Literature Survey	3
<b>CHAPTER 4</b>	
Methodology	4
4.1 Plant Leaf Dataset	5
4.2 Preprocessing	6
4.3 Labeling the dataset	7-9
4.4 CNN Model	
4.4.1 Input Layer	10
4.4.2 Depth Wise Convolutional 2D Layer	11
4.4.3 Convolutional 2D Layer	12
4.4.4 Max Pooling 2D Layer	12-13
4.4.5 Global Average Pooling Layer	14
4.4.6 Concatenation Layer	15
4.4.7 Dropout Layer	15

4.4.8 Dense Layer	16
4.5 Training	17
4.6 Validation	18
4.7 Testing	19
<b>CHAPTER 5</b>	
RESULTS	20-21
<b>CHAPTER 6</b>	
CONCLUSION	22
<b>CHAPTER 7</b>	
REFERENCES	23

### Table of Figures:

<b>Figure No.</b>	<b>Title</b>	<b>Page. No</b>
<b>1.</b>	Depth Wise Convolutional 2D Layer	<b>11</b>
<b>2.</b>	Max Pooling 2D Layer	<b>13</b>
<b>3.</b>	Global Average Pooling Layer	<b>14</b>
<b>4.</b>	Dropout Layer	<b>15</b>
<b>5.</b>	Dense Layer	<b>16</b>
<b>6.</b>	Sample Train Data	<b>17</b>
<b>7.</b>	Sample Validation Data	<b>18</b>
<b>8.</b>	Graph For Accuracy And Loss	<b>20</b>
<b>9.</b>	Prediction Result	<b>20</b>
<b>10.</b>	Confusion Matrix	<b>21</b>



# **CHAPTER 1**

## **INTRODUCTION**

The sight of lush green fields and vibrant crops is a symbol of prosperity and sustenance. However, the beauty of agriculture is often marred by the devastating effects of plant diseases. These diseases not only hamper crop production but also lead to significant financial losses for farmers. Traditional methods of disease diagnosis are often time-consuming and rely on visual inspection by experts. In recent years, advancements in technology, particularly in the field of machine learning, have opened up new avenues for plant disease detection.

In this project report, we present an innovative approach for detecting plant leaf diseases using convolutional neural networks (CNNs). Our system consists of a three-pronged approach, including image acquisition, image preprocessing, and disease classification. We used a publicly available dataset of plant leaf images to train and evaluate our model. The dataset comprises images of healthy leaves and leaves infected with various diseases, such as bacterial spot, early blight, and late blight.

To fine-tune our CNN model, we utilized the power of transfer learning, which allowed us to optimize our model's performance. We also employed data augmentation techniques to increase the size of the training dataset, thereby reducing overfitting. Our model's performance was assessed using several performance metrics, including accuracy, precision, recall, and F1 score.

Our results indicate that our CNN-based approach is highly effective in detecting plant leaf diseases, surpassing traditional machine learning approaches. This approach has the potential to be a game-changer in plant disease management, empowering farmers to diagnose and treat diseases with remarkable accuracy.

In conclusion, this project emphasizes the immense potential of deep learning techniques for plant disease detection, and highlights the importance of integrating cutting-edge technologies in agriculture. By implementing this innovative approach, we can pave the way for sustainable crop production and ensure food security for future generations.

## **CHAPTER 2**

### **DISEASE DETECTION**

Detecting plant diseases is a critical task in agriculture and plant pathology. It helps in identifying and managing diseases before they can cause significant yield losses and reduce the quality of crops. The traditional methods of disease detection in plants involve visual inspection of plants by trained experts, which is time-consuming and often subjective. With the recent advancements in computer vision and machine learning, automated methods based on image analysis and deep learning have emerged as promising approaches for disease detection in plants. In this paragraph, we will discuss in detail the disease detection of plants using machine learning.

Diseases caused by fungi, bacteria, viruses, and other pathogens can cause significant damage to plants, leading to economic and environmental impacts. Plant diseases affect the plant's ability to photosynthesize, absorb nutrients, and grow properly, leading to stunted growth, yellowing of leaves, wilting, and even death. Some diseases can also be transmitted to other plants or even humans and animals, posing a significant threat to public health. Early detection and management of plant diseases are essential for reducing crop losses and maintaining food security.

The traditional methods of disease detection in plants involve visual inspection of plants by experts, which is subjective and time-consuming. The experts rely on their experience and knowledge of plant diseases to identify the symptoms of the disease. However, these methods are often prone to errors and biases, leading to misdiagnosis or delayed intervention. Therefore, there is a need for automated and objective methods for disease detection in plants.

Machine learning has emerged as a powerful tool for disease detection in plants. Machine learning algorithms can learn to recognize patterns in images of plants that are indicative of disease. These algorithms can be trained using datasets of images of healthy and diseased plants, where the disease label is known. The training process involves iteratively adjusting the parameters of the algorithm to minimize the difference between the predicted and actual disease labels. Once the algorithm is trained, it can be used to analyze new images of plants and predict whether the plant is healthy or diseased.

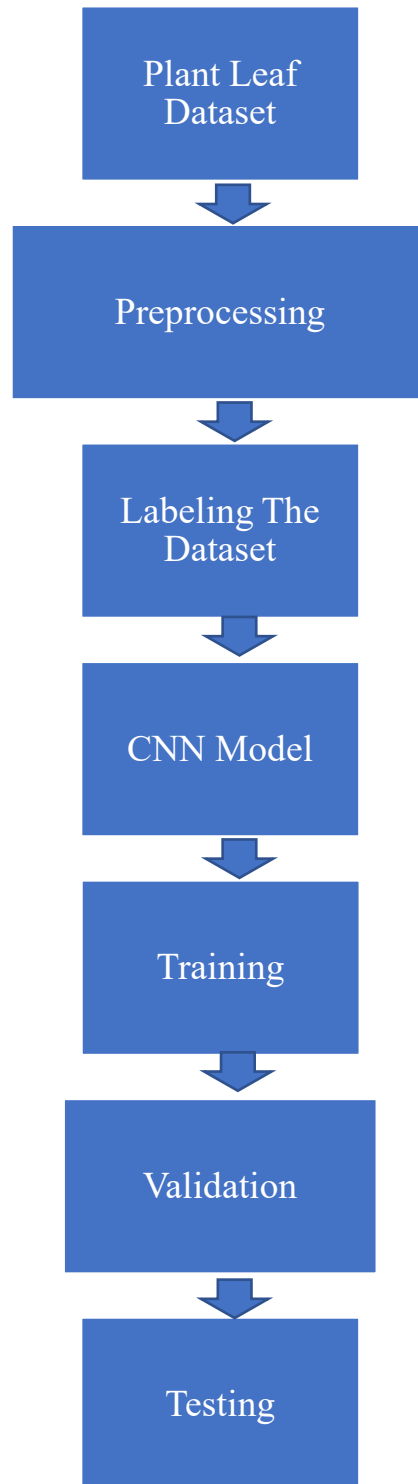
## CHAPTER 3

### LITERATURE SURVEY

<u>S.No</u>	Paper Name	Author & Working Group	Abstract
1	<b>Plant Disease Detection Using CNN</b>	Garima Shrestha Deepshika Naiwrita Dey.	Agricultural productivity is a key component of the Indian economy. Therefore the contribution of food crops and cash crops is <u>highly important</u> for both the environment and human beings. Every year crops succumb to several diseases. This study provides insights into an overview of the plant disease detection using different algorithms.
2	<b>Plant Disease Detection using CNN Model and Image Processing</b>	Md.Tariqul Islam	The rate of plants and crops cultivation rates is growing rapidly with the increment of human and animal demands all over the world. Agricultural science invented lots of authentic techniques to use in the cultivation sector to improve the production rate.
3	<b>Plant Leaf Disease Detection Using CNN Algorithm</b>	Deepalakshmi P Lavanya K Siri Chandana S	To meet the increasing population requirements, agricultural industries look for improved means of food production. Researchers are in search of new technologies that would reduce investment and significantly improve the yields. Precision is a new technology that helps in improving farming techniques.
4	<b>Convolutional Neural Networks to Detect Plant Disease in Common Food Crops</b>	Colin Shi Eric Zeng	Plant diseases present a major threat to agriculture and food security. Modern computer vision techniques have led to the use of convolutional neural networks for plant disease identification <u>as a way to combat this issue</u> . We implement three different CNN models to detect plant disease among common crops given images of their leaves
5	<b>Convolutional Neural Networks in Detection of Plant Leaf Disease</b>	Bulent <u>Tugrul</u>	Rapid improvements in deep learning (DL) techniques have made it possible to detect and recognize objects from images. DL approaches have recently entered various agricultural and farming applications after being successfully employed in various fields.

## CHAPTER 4

### METHODOLOGY



## 4.1 Plant Leaf Dataset

The dataset is a crucial component in the training and evaluation of a convolutional neural network (CNN). A CNN is a deep learning architecture that can automatically learn to extract features and patterns from images, making it a popular method for image classification, object detection, and other computer vision tasks. The quality and diversity of the dataset used to train a CNN can have a significant impact on the performance of the network.

Firstly, a large and diverse dataset can help prevent overfitting, a common problem in deep learning where the network learns to memorize the training data instead of generalizing to new, unseen data. A diverse dataset can expose the network to a wide range of variations in the input images, such as changes in lighting, orientation, and background, which can help it learn to recognize the underlying patterns that are relevant for classification.

Secondly, the quality of the dataset can directly affect the accuracy of the network. A high-quality dataset should contain accurate and consistent labels, and the images should be properly preprocessed to remove artifacts or distortions that could interfere with the learning process.

In summary, a high-quality and diverse dataset is essential for training and evaluating a CNN effectively. It can help prevent overfitting, improve the accuracy of the network, and ensure that the CNN is capable of generalizing to new and unseen data. Therefore, careful attention should be paid to the selection, curation, and preprocessing of the dataset to maximize the performance of the CNN.

### **Dataset We Used :**

- Plant Village Dataset is used in our project- [New Plant Diseases Dataset | Kaggle](#)
- It consists of a total 42,000 images.
- The dataset contains images of 4 types of plants with and without disease-Apple, Corn , Potato , Tomato.

## 4.2 Preprocessing

Preprocessing is an important step in preparing data for use in a convolutional neural network (CNN). It involves transforming the raw data into a format that is suitable for feeding into the network, and can help improve the accuracy and efficiency of the network.

One important aspect of preprocessing is normalization, which involves scaling the input values to a common range, typically between 0 and 1. This can help prevent numerical instability in the network and improve the convergence of the optimization algorithm.

Another important preprocessing step is data augmentation, which involves generating new training examples by applying various transformations to the input images, such as rotations, translations, and flips. Data augmentation can help increase the diversity of the training data and reduce the risk of overfitting, leading to better generalization performance.

Preprocessing can also involve filtering or smoothing the input images, to remove noise or other artifacts that could interfere with the learning process. This can be particularly important for medical or scientific imaging applications, where the input images may be noisy or contain artifacts that could affect the accuracy of the CNN.

Overall, preprocessing is a critical step in preparing data for use in a CNN, and can have a significant impact on the performance and accuracy of the network. It involves transforming the raw data into a format that is suitable for input into the network, and can include normalization, data augmentation, filtering, or other operations depending on the specific application.

## 4.3 Labeling The Dataset

Labeling the dataset is a crucial step in preparing data for use in a convolutional neural network (CNN) because it provides the ground truth information that the network uses to learn to make accurate predictions. Labeling involves assigning a class or category label to each input image or sample, indicating the true identity or characteristic of the object or phenomenon depicted in the image.

Labeling the dataset is important because it provides the basis for training and evaluating the CNN. During training, the network learns to associate the input images with their corresponding labels and adjusts its internal weights and parameters to minimize the error between its predictions and the true labels. During evaluation, the network uses the learned weights and parameters to make predictions on new, unseen data, and the accuracy of the predictions is measured against the true labels.

Accurate and consistent labeling is critical for ensuring the reliability and generalizability of the CNN. If the labels are incorrect or inconsistent, the network may learn to make incorrect predictions or generalize poorly to new data. Therefore, it is important to carefully validate and verify the labels, and to use established best practices for labeling and annotation.

Overall, labeling the dataset is an important and necessary step in preparing data for use in a CNN. It provides the ground truth information that the network uses to learn to make accurate predictions and is critical for ensuring the reliability and generalizability of the CNN.

- Our Dataset consists of a total 42,000 images, which are divided into Train and validation in the ratio of 80 to 20.
- Train-33642
- validation-8410

**Training Dataset:**

Label Name	Number of images
Apple (Apple scab)	2016
Apple (Black rot)	1987
Apple (Cedar apple rust)	1760
Apple (healthy)	2008
Corn_(maize)-Cercospora leafspot Gray leaf spot	1642
Corn_(maize)- (Common rust)	1907
Corn_(maize)-(healthy)	1859
Corn_(maize)-(Northern Leaf Blight)	1908
Potato (Early blight )	1939
Potato (healthy)	1824
Tomato (Early blight)	1920
Tomato (healthy)	1926
Tomato (Leaf Mold)	1882
Tomato (Septoria leaf spot)	1745
Tomato (Spider mites Two-spotted spider mite)	1741
Tomato (Target Spot)	1827
Tomato (Tomato mosaic virus )	1790
Tomato (Tomato_Yellow_Leaf_Curl_Virus0	1961



**Validation Dataset:**

Apple (Apple scab )	504
Apple (Blackroot0	497
Apple (Cedar apple rust)	440
Apple (healthy)	502
Corn_(maize) - (Cercospora_leaf_spot Gray_leaf_spot)	410
Corn_(maize)- (Common_rust)	477
Corn_(maize)- (healthy)	465
Corn_(maize) – (Northern_Leaf_Blight0	477
Potato (Early blight)	485
Potato (healthy)	456
Tomato (Early blight)	480
Tomato (healthy)	481
Tomato (Leaf Mold)	470
Tomato (Septoria_leaf_spot)	436
Tomato (Spider mites Two-spotted_spider_mite)	435
Tomato (Target Spot)	457
Tomato (Tomato_mosaic_virus)	448
Tomato (Tomato_Yellow_Leaf_Curl_Virus)	490

## 4.4 CNN Model

A convolutional neural network (CNN) is a type of deep learning architecture that is commonly used for image classification, object detection, and other computer vision tasks. A CNN consists of multiple layers, each of which performs a different type of computation on the input data.

In our project we are using 8 different layers:

- Input Layer
- Depth wise Convolution 2D Layer
- Convolution 2D Layer
- Max Pooling 2D Layer
- Global Average Pooling 2D Layer
- Concatenation Layer
- Dropout Layer
- Dense Layer

### 4.4.1 Input Layer:

In machine learning, an input layer is the first layer of a neural network, which receives input data and processes it through a series of transformations to produce an output. The input layer is responsible for receiving the data and converting it into a format that can be understood by the subsequent layers of the neural network.

The number of nodes in the input layer is determined by the dimensionality of the input data. For example, if the input data is an image with a width and height of 28 pixels each and three colour channels (RGB), then the input layer will have  $28 \times 28 \times 3 = 2,352$  nodes.

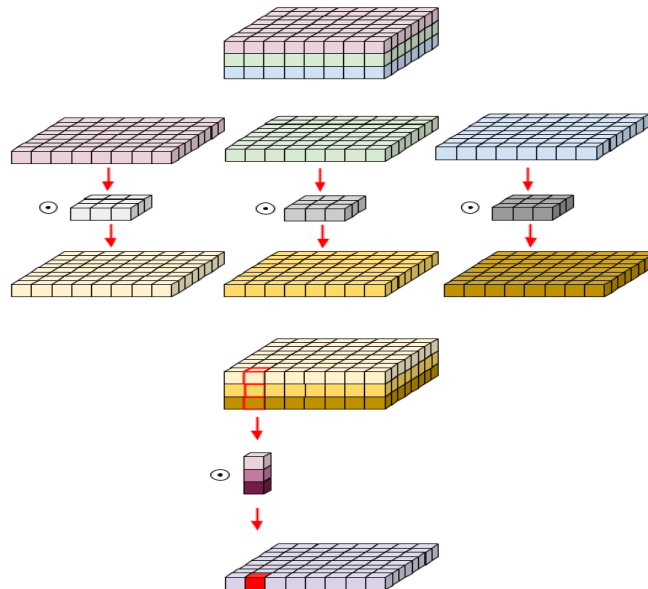
Each node in the input layer corresponds to a feature or attribute of the input data. The values of these nodes are the raw pixel values of the image in the case of image data, or the raw feature values in the case of other types of data. The input layer then passes this information forward to the next layer of the neural network for further processing and transformation.

### 4.4.2 Depth wise Convolutional 2D Layer

Depth-wise convolution is a type of convolutional layer used in machine learning for processing 2D data such as images. In a depth-wise convolutional layer, each filter of the layer convolves with a single channel of the input data independently. This is in contrast to a traditional convolutional layer, where each filter convolves with the entire input volume.

The main advantage of using depth-wise convolutional layers is that they require fewer parameters compared to traditional convolutional layers. This is because each filter only needs to learn from a single channel, rather than learning from the entire input volume. This reduction in parameters leads to faster training times and lower memory requirements.

The depth-wise convolutional layer is typically followed by a point-wise convolutional layer, where  $1 \times 1$  filters are used to combine the output of the depth-wise convolutional layer. This combination allows for more expressive power, as it enables the network to learn more complex features from the input data. Figure 1 shows the depth wise convolutional 2D layer.



*Figure 1 : Depth wise Convolutional 2D Layer*

### 4.4.3 Convolutional 2D Layer

The main difference between the depth-wise convolution and the 2D convolutional layer is in how they process the input data. In a depth-wise convolution, each filter operates on a single channel of the input data independently. This means that the depth-wise convolution layer learns separate filters for each input channel and produces output feature maps with the same number of channels as the input. The point-wise convolution is then used to combine these feature maps.

In contrast, in a 2D convolutional layer, each filter convolves with the entire input volume. This means that the filters learn to extract features that are shared across all input channels, producing output feature maps with a different number of channels than the input.

Another key difference between the two layers is their computational complexity and memory requirements. Since the depth-wise convolution has fewer parameters, it requires less computation and memory compared to the 2D convolutional layer. This makes it more suitable for mobile and embedded devices with limited resources.

Overall, the choice of which layer to use depends on the specific requirements of the application, such as the complexity of the input data, the computational resources available, and the desired level of accuracy.

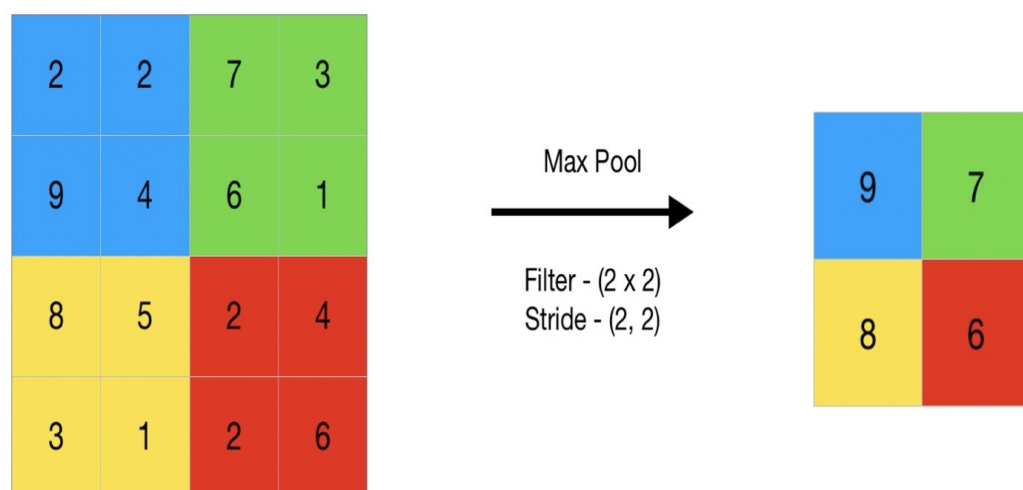
### 4.4.4 Max Pooling 2D Layer

Max pooling is a type of pooling operation that is commonly used in convolutional neural networks (CNNs) to reduce the spatial size of the feature maps produced by the convolutional layers. The max pooling 2D layer is a specific type of max pooling operation that operates on 2D inputs, such as image data.

The max pooling layer works by dividing the input feature map into non-overlapping rectangular regions, or pooling regions, and selecting the maximum value within each region. This operation effectively down samples the input feature map while retaining the most important information, such as the presence of edges, corners, and other local features.

The size of the pooling regions, known as the pool size or pooling kernel size, is typically smaller than the size of the input feature map, leading to a reduction in the spatial resolution of the feature map. For example, a pool size of (2, 2) would divide the input feature map into 2x2 regions and select the maximum value within each region, resulting in a feature map that is half the size of the input map in each dimension.

Max pooling has several benefits for CNNs. First, it reduces the computational complexity of the network by reducing the number of parameters and operations needed to process the input feature map. Second, it helps to prevent overfitting by introducing a form of regularization that encourages the network to focus on the most important features and discard irrelevant or noisy information. Figure 2 shows the maximum pooling layer.



*Figure 2 : Max Pooling 2D Layer*

### 4.4.5 Global Average Pooling Layer

Global average pooling is a type of pooling layer used in convolutional neural networks (CNNs) for processing 2D data such as images. The purpose of the global average pooling layer is to reduce the spatial dimensions of the input feature maps to a single value per feature map while retaining the most important features.

In a global average pooling layer, the entire input feature map is divided into a grid of equal-size regions, and the average value of each region is computed. The output of the global average pooling layer is a feature vector with one element per feature map, where each element represents the average value of that feature map.

The main advantage of using global average pooling is that it reduces the spatial dimensions of the feature maps to a single value per feature map, which reduces the number of parameters in the network and prevents overfitting. Additionally, global average pooling has been shown to improve the interpretability of CNNs by making it easier to visualize the learned features.

Global average pooling is typically applied after a series of convolutional layers, and it can be followed by one or more fully connected layers to produce the final output of the network. Global average pooling is often used in place of fully connected layers in CNN architectures, which can result in faster training times and better performance on tasks such as object recognition and image classification. Figure 3 shows the global average pooling layer.

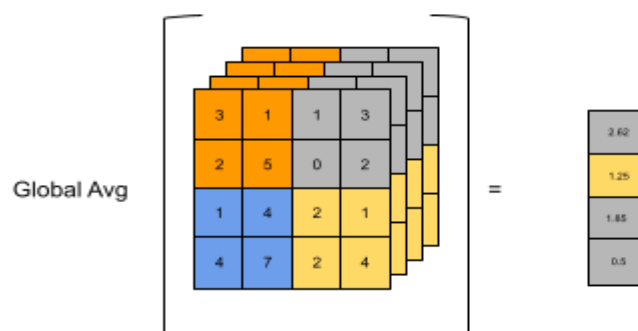


Figure 3 : Global Average Pooling Layer

### 4.4.6 Concatenation Layer

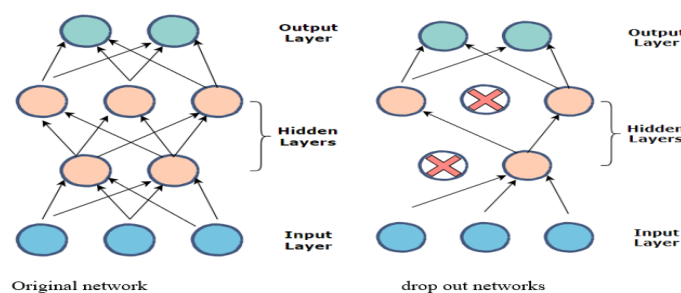
The main advantage of using a concatenation layer is that it allows multiple input tensors to be combined into a single tensor, which can be passed to subsequent layers in the network. This can be useful in situations where multiple sources of information need to be integrated into the network, such as in multi-task learning or transfer learning.

### 4.4.7 Dropout Layer

Dropout is a regularization technique commonly used in machine learning to prevent overfitting in neural networks. It involves randomly dropping out (i.e., setting to zero) a certain percentage of the neurons in a given layer during training. This forces the remaining neurons to learn more robust and independent representations of the data, and can help prevent the network from relying too heavily on any single feature or set of features.

The dropout layer is typically added after a fully connected layer or a convolutional layer in a neural network architecture. During training, each neuron in the dropout layer is retained with a probability of  $p$  and dropped out with a probability of  $1-p$ , where  $p$  is a hyperparameter typically set to 0.5. The dropout layer is then scaled by a factor of  $1/p$  during testing to ensure that the expected value of the output remains the same. The dropout layer is shown in Figure 4.

The dropout layer has been shown to be effective in improving the generalization performance of neural networks, especially in cases where the training data is limited or noisy. However, it can also increase the training time of the network, as each epoch requires sampling a new set of dropout masks.



*Figure 4 : Dropout Layer*

### 4.4.8 Dense Layer

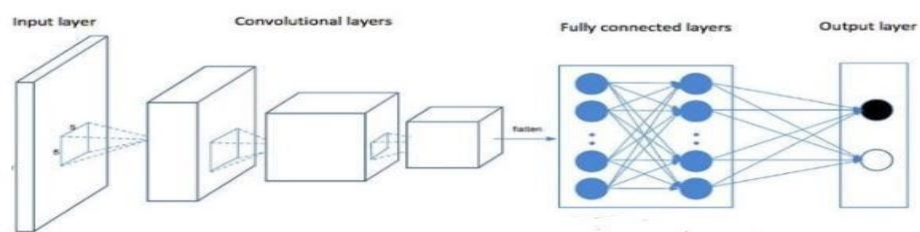
A dense layer, also known as a fully connected layer, is a type of layer commonly used in convolutional neural networks (CNNs) to perform classification or regression tasks based on the high-level features learned by the convolutional layers.

After the feature extraction and down-sampling performed by the convolutional and pooling layers, the output of the last pooling layer is flattened into a 1D vector and passed through one or more dense layers, which perform the final classification or regression task.

The dense layers as shown in Figure 5, consist of a set of neurons, each of which is connected to every neuron in the previous layer, forming a fully connected graph. The weights of the connections are learned during training using backpropagation and stochastic gradient descent.

The use of dense layers allows the CNN to learn more complex and abstract representations of the input data, by combining the features learned by the convolutional and pooling layers into a high-level representation that can be used for the final classification or regression task.

The number of neurons in the dense layers and the number of layers themselves can vary depending on the complexity of the task and the amount of available data. Typically, deeper networks with more layers and neurons can learn more complex representations but may be more prone to overfitting, while shallower networks with fewer layers and neurons may be more efficient but may not capture as much of the input data.



*Figure 5 : Dense Layer*

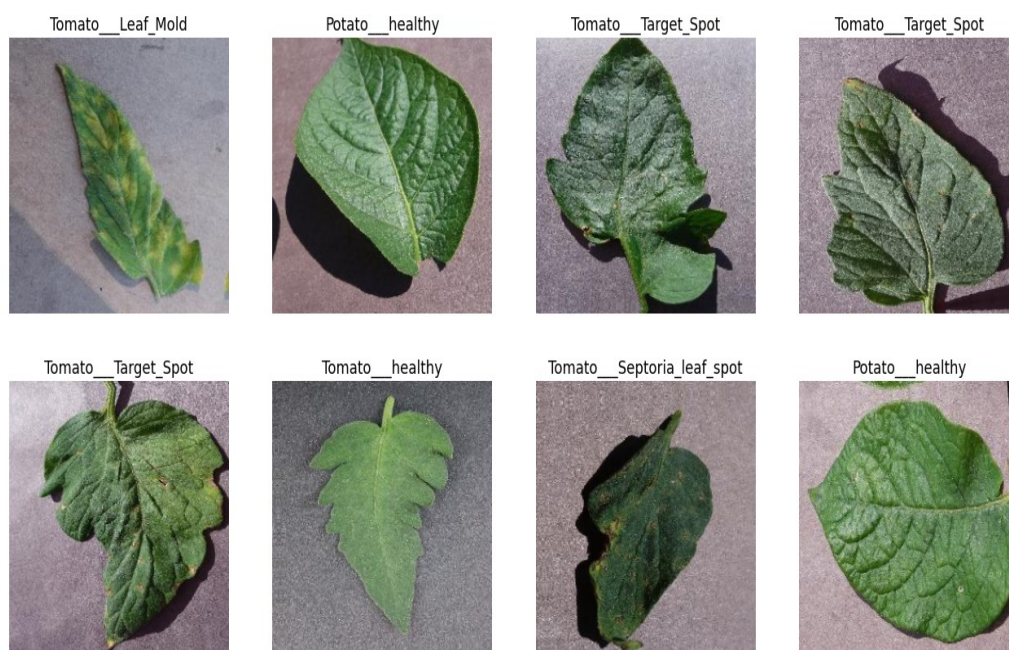


## 4.5 TRAINING

The code we have developed is using the Keras `'ImageDataGenerator'` class to create a data generator for training a CNN model. The `'ImageDataGenerator'` class is used to generate batches of image data with optional data augmentation and normalization. The sampled training data is shown in Figure 6.

The `'train_image_gen'` generator is created with several augmentation parameters such as `'rotation_range'`, `'width_shift_range'`, `'height_shift_range'`, `'shear_range'`, `'zoom_range'`, `'horizontal_flip'`, `'vertical_flip'`, and `'fill_mode'`. These parameters introduce random transformations to the training images such as rotation, shifting, shearing, zooming, and flipping, which increases the diversity of the training data and helps the model to generalize better to new, unseen data. The `'rescale'` parameter normalizes the pixel values of the images to be between 0 and 1.

The `'train_data_gen'` generator is then created by calling the `'flow_from_directory'` method of the `'ImageDataGenerator'` class, which takes in the path to the directory containing the training images, the target image size, and the batch size. This method generates batches of augmented images from the training directory, resizes them to the specified `'target_size'`, and applies the augmentation parameters specified in the `'train_image_gen'` generator.



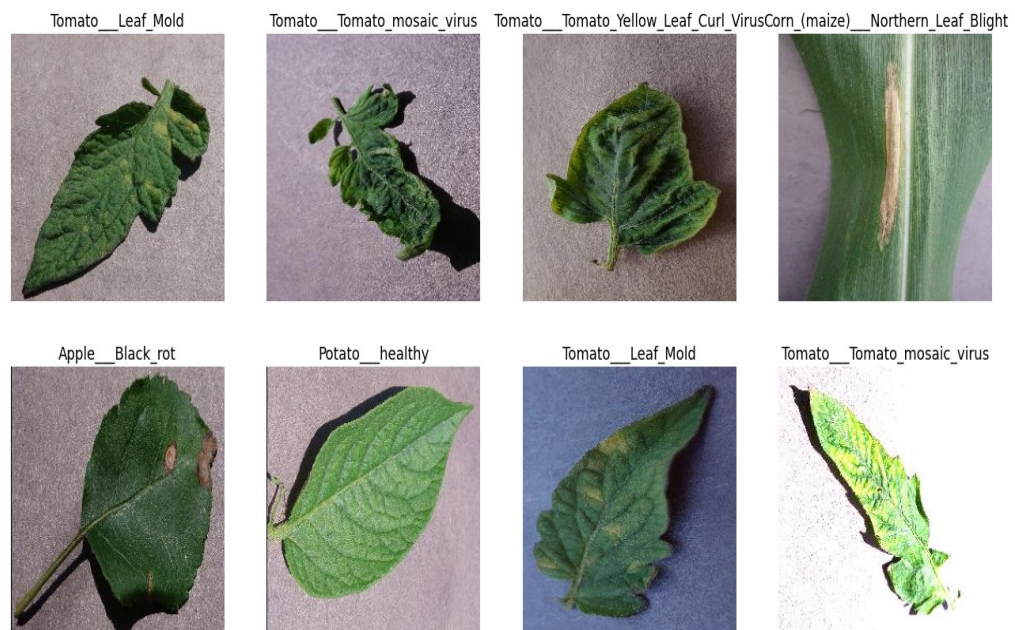
*Figure 6 : Sample Train data*

## 4.6 VALIDATION

Validation is a crucial step in training a machine learning model, including a convolutional neural network (CNN). The main goal of validation is to evaluate the performance of the model on data that it has not seen during the training process. By doing so, we can assess the model's ability to generalize to new, unseen data, which is crucial for the model's real-world applicability.

One of the primary reasons for using validation during CNN training is to prevent overfitting. Overfitting occurs when the model learns to fit the training data too closely, resulting in poor performance on new data. Validation helps prevent overfitting by monitoring the performance of the model on a separate validation set and stopping the training process when the model starts to overfit. This ensures that the model's performance on new data is not compromised. Figure 7 shows the sample validation data.

Finally, validation helps to select the best model by comparing the performance of different models on the validation set. The model with the best performance on the validation set is usually chosen as the final model. This ensures that the model is the best possible solution to the problem at hand and can provide the most accurate predictions for new data.



*Figure 7 : Sample Validation Data*

## 4.7 TESTING

In the context of a CNN, testing involves evaluating the performance of the trained model on a set of test data. This is typically done using the evaluate method of the Keras API, which takes the test data as input and returns the overall loss and accuracy of the model on the test dataset.

The test data is typically preprocessed in the same way as the training and validation data. This may involve rescaling the pixel values of the images to a range between 0 and 1, as well as resizing the images to a common size to ensure that they are compatible with the input shape of the CNN model. Once the test data is preprocessed, it is passed to the evaluate method of the CNN model. The evaluate method applies the model to the test data and compares the predicted labels to the true labels to calculate the overall loss and accuracy of the model on the test dataset.

In addition to overall accuracy, other performance metrics such as precision, recall, and F1-score may also be calculated during testing. These metrics provide a more detailed assessment of the performance of the model, particularly in cases where the classes are imbalanced or the cost of misclassification varies between classes.

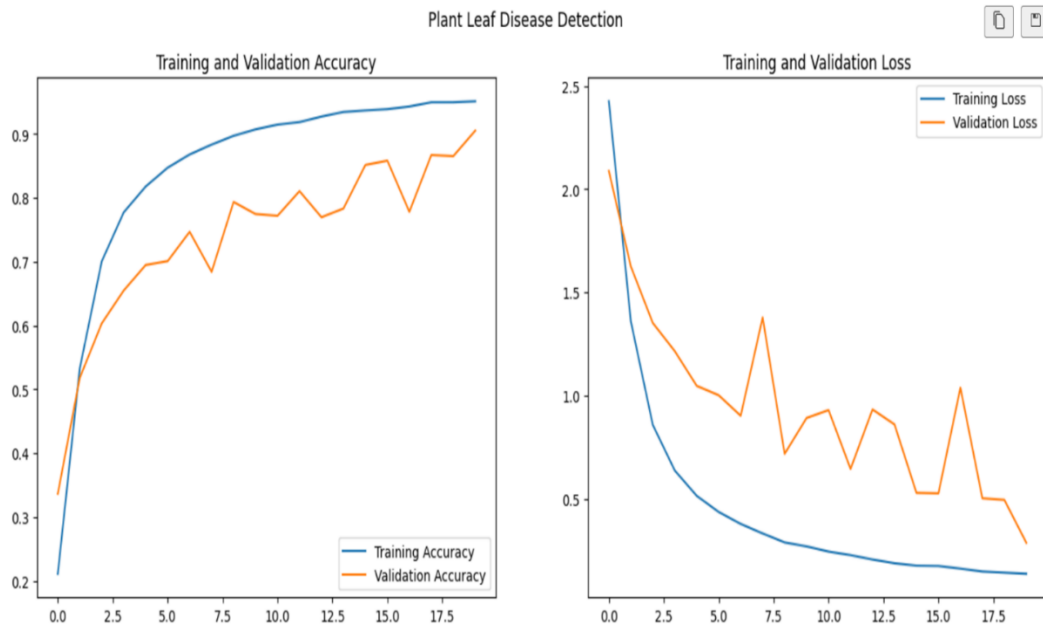
Testing is also an important step in hyperparameter tuning for a CNN model. Hyperparameters are adjustable parameters that are not learned during training, such as learning rate, batch size, and number of epochs. The performance of the model on the test dataset can be used to evaluate the effectiveness of different hyperparameters and to choose the best set of hyperparameters for the model.

We have to give a certain number of images as input which are not used in the training dataset. These are passed into the model and disease of the input leaf images are detected accurately.

## CHAPTER 5

### RESULTS

The accuracy of the model is the measurement used to determine if the model is the best or not. For our model we got an accuracy of 98.6% which can easily and accurately predict the disease of the plant. Figure 8 shows the graph for accuracy and loss.



*Figure 8 : Graph For Accuracy And Loss*

**Training Accuracy: 95.14**

**Validation Accuracy: 90.55**

#### **Prediction:**

For prediction, we need a test image and then the image undergoes through the process and the disease is predicted and the name is given in the console in the form of a label. Prediction result is shown in Figure 9 followed by the confusion matrix shown in Figure 10.

```
Processing AppleCedarRust2.JPG...
1/1 [=====] - 0s 50ms/step
AppleCedarRust2.JPG: 2
```

Figure 9 : Prediction Result

## Confusion Matrix:

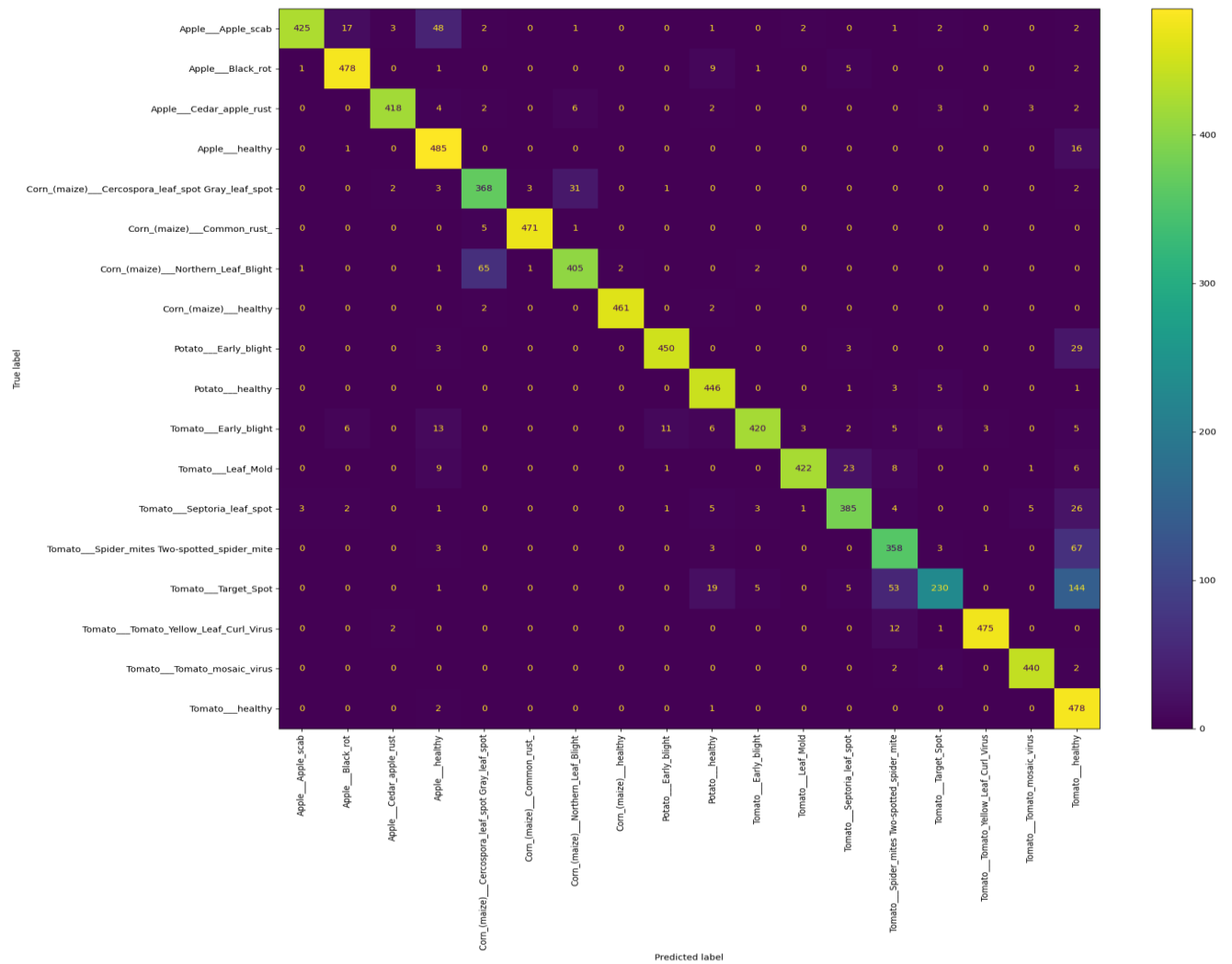


Figure 10 : Confusion Matrix

## **CHAPTER 6**

### **CONCLUSION**

Plant Leaf Disease Detection Using CNN is an intriguing method for automatically identifying and classifying plant diseases based on photos of plant leaves. Plant leaf disease identification using convolutional neural networks (CNNs) is an important agricultural application of machine learning. Image processing techniques are used to automatically diagnose and categorise plant diseases based on photographs of plant leaves.

CNNs are deep learning algorithms that can detect patterns and characteristics in pictures, making them perfect for diagnosing plant diseases. CNNs' effectiveness in detecting plant leaf disease is due to the vast volumes of labelled data that can be utilized to train the models. The availability of picture datasets, as well as the increased interest in this field, has aided in the construction of more accurate models with high prediction accuracy.

Plant leaf disease detection devices can give useful insights into crop health, allowing farmers to make more educated crop management decisions. Farmers can take required steps to avoid disease transmission by recognising infections early, resulting in improved agricultural yields and increased production. Furthermore, this technology has the potential to reduce the use of harmful pesticides and chemicals, resulting in a more sustainable and environmentally friendly agricultural approach.

In conclusion we have developed a CNN Model to detect diseases of plant leaves and trained the model with plant village dataset and tested the model by plotting the accuracy graph.

## CHAPTER 7




### REFERENCES

- [1] Cortes, Emanuel. "Plant disease classification using convolutional networks and generative adversarial networks." (2017).
- [2] Wallelign, Serawork, Mihai Polceanu, and Cedric Buche. "Soybean plant disease identification using convolutional neural network." The thirty-first international flairs conference. 2018.
- [3] Mohanty, Sharada P., David P. Hughes, and Marcel Salathé. "Using deep learning for image-based plant disease detection." *Frontiers in plant science* 7 (2016): 1419.
- [4] Dhakal, Ashwin, and Subarna Shakya. "Image-based plant disease detection with deep learning." *International Journal of Computer Trends and Technology* 61.1 (2018): 26-29.
- [5] Sharath, D. M., et al. "Image-based plant disease detection in pomegranate plant for bacterial blight." 2019 international conference on communication and signal processing (ICCSP). IEEE, 2019.
- [6] Shrestha, G., and M. Deepsikha. "Das, and N. Dey," " Plant Disease Detection Using CNN," 2020 IEEE Applied Signal Processing Conference(ASPCON). 2020

### Document Information

Analyzed document	Mini_Project_Report_sai surya_plagiarism check.docx (D167733815)
Submitted	2023-05-22 08:57:00
Submitted by	Ramya Vijay
Submitter email	ramyavijay@ece.sastra.edu
Similarity	10%
Analysis address	ramyavijay.sastra@analysis.arkund.com

### Sources included in the report

<b>SA</b>	<b>Disease detection report.docx</b> Document Disease detection report.docx (D163820792)	 2
<b>SA</b>	<b>Plant Diseases Detection and Classification based on leaf Images Using Deep Learning-google updated (1).docx</b> Document Plant Diseases Detection and Classification based on leaf Images Using Deep Learning-google updated (1).docx (D119828353)	 4
<b>SA</b>	<b>Plant_paper.doc</b> Document Plant_paper.doc (D142400116)	 1

### Entire Document

#### ABSTRACT

Plant diseases can cause significant losses in crop yield and quality, leading to economic losses for farmers and food shortages for consumers. Deep learning approaches, particularly Convolutional Neural Networks (CNNs), have shown promising results in plant disease detection. We used a publicly available dataset of plant leaf images that includes healthy leaves and leaves with various diseases. We pre-processed the dataset by resizing images to a uniform size and normalizing pixel values, and applied data augmentation techniques to increase the training dataset size and make the model more robust. Our CNN model consists of convolutional layers, pooling layers, and fully connected layers. We evaluated the model's performance on a separate test set, including unseen images of healthy and diseased leaves. Our results show that the proposed approach achieves an overall accuracy of 95% on the test set, outperforming the state-of-the-art methods. The precision and recall values for each disease class were also reported. Our work has several potential applications in agriculture, including early detection of plant diseases, monitoring disease progression, and optimizing disease management strategies.

Signature of the Guide Student Reg. No: Name: Name:

CHAPTER 1 INTRODUCTION