

Plant Leaf Disease Identification

by

Mohammad Rakibul Hasan Mahin
20201220

Waheed Moonwar
20201219

Md. Shamsul Rayhan Chy
19201109

Md. Fahim Shahriar
19201046

Fahim Faisal Rafi
19201081

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
January 2023

© 2023. Brac University
All rights reserved.

Declaration

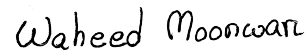
It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:



Mohammad Rakibul Hasan Mahin
20201220



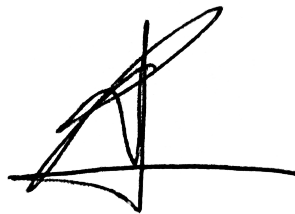
Waheed Moonwar
20201219



Md. Shamsul Rayhan Chy
19201109



Md. Fahim Shahriar
19201046



Fahim Faisal Rafi
19201081

Approval

The thesis/project titled “Plant Leaf Disease Identification” submitted by

1. Mohammad Rakibul Hasan Mahin (20201220)
2. Waheed Moonwar (20201219)
3. Md. Shamsul Rayhan Chy (19201109)
4. Md. Fahim Shahriar (19201046)
5. Fahim Faisal Rafi (19201081)

of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on January 19, 2023

Examining Committee:

Supervisor:
(Member)

**Annajiat
Alim
Rasel** Digitally signed by
Annajiat Alim Rasel
DN: cn=Annajiat Alim
Rasel, o=Brac University,
ou=CSE Department,
email=annajiat@bracu.ac.
bd, c=BD
Date: 2023.01.14 23:04:00
+06'00'

Annajiat Alim Rasel
Senior Lecturer
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)



Dewan Ziaul Karim
Lecturer
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi
Associate Professor and Chairperson
Department of Computer Science and Engineering
Brac University

Ethics Statement

This statement serves as a disclaimer, stating that the research paper in question did not break any laws protecting human welfare, rights, or dignity. The data was collected without bias or inequality, and the outcomes were not changed for any prejudicial purposes. The paper also includes emphasized citations from numerous dependable sources and incorporates the sincerity and integrity of the research participants. We can guarantee that we aspire for our work to reach our target audience from which they will be the ones to benefit the most.

Published, Accepted & Submitted Papers

Accepted & Presented Paper

1. **Interpretable Disease Classification in Plant Leaves using Deep Convolutional Neural Networks** – *2022 25th International Conference on Computer and Information Technology (ICCIT, 2022)* [Presented]

Submitted Paper

1. **Classifying Corn Leaf Diseases using Ensemble Learning with Dropout and Stochastic Depth Based Convolutional Networks.**

Abstract

Agriculture has consistently been an essential component of our day-to-day life over the centuries. Because of its contribution to our country's revenue, the importance of agriculture has been steadily growing over the course of the years. However, there are some counter factors that prevent us from reaping the full benefits that crops have to offer. The presence of a wide variety of natural diseases on plant leaves is one such factor. The most prominent causes of these problems are typically severe weather conditions and excessive use of pesticides, both of which put a strain on the economy of Bangladesh as a whole. To reduce the severity of the problem, we are going to design an image processing system that utilizes Deep Learning and Convolutional Neural Networks (CNN) to classify plant leaf diseases. Our primary demographic of interest consists of farmers and other people willing to tend to crops. We have concluded that the best way to go about this is by constructing a website and making it as simple and straightforward as possible. The user will select images of the diseased leaf, and our CNN model will predict and categorize the leaf's condition based on the chosen images.

After implementing CNN, we introduce another model, namely LIME, which is based on the concept of Explainable AI (XAI). An XAI is an artificial intelligence that mainly helps humans to understand the decisions or predictions made by an AI. In this scenario, after our CNN model classifies the diseased leaves, the XAI aids us in understanding the reason and cause behind the leaves mentioned above being classified as how they are by the CNN model.

Conclusively, following the completion of running our models, we managed to get a 99.54% accuracy rate in our testing phase.

Keywords: Neural Network. Convolutional Neural Network (CNN). Plant Leaf Disease Identification. Deep Learning. XAI. Image Processing.

Acknowledgement

First and foremost, thanks to the Almighty Allah, with whose help we were able to finish writing our thesis without too many setbacks. Second, we would like to thank our Supervisor and Co-Supervisor, Mr. Annajiat Alim Rasel, and Mr. Dewan Ziaul Karim for their thoughtful assistance and counsel. We appreciate everyone's efforts in the group to complete our work on time. And ultimately, without our parents' ongoing support, it might not be possible to reach where we are right now. We are currently preparing to graduate thanks to their kind prayers and support.

Table of Contents

Declaration	i
Approval	ii
Ethics Statement	iv
Published Submitted Papers	v
Abstract	vi
Acknowledgment	vii
Table of Contents	viii
List of Figures	xi
List of Tables	xiv
Nomenclature	xv
1 Introduction	1
1.1 Problem Statement	1
1.2 Background Information	1
2 Related Work	4
2.1 Machine Learning	4
2.1.1 Logistic Regression	4
2.1.2 Decision Tree	5
2.1.3 SVM Algorithm	5
2.1.4 Naive Bayes Algorithm	6
2.1.5 K-NN Algorithm	6
2.1.6 Random Forest Algorithm	6
2.2 Deep Learning Models	7
2.2.1 CNN	7
2.3 Explainable Artificial Intelligence	9
2.4 Transformers	10
3 Research Objectives	12
3.1 Workflow	12

4	Dataset	14
4.1	Data Analysis	14
4.2	Data Preprocessing	15
4.3	Detailed Dataset	16
4.4	Dataset Classes	16
4.4.1	Apple	16
4.4.2	Blueberry	20
4.4.3	Cherry	20
4.4.4	Corn	22
4.4.5	Grape	25
4.4.6	Orange	27
4.4.7	Peach	28
4.4.8	Pepper	30
4.4.9	Potato	31
4.4.10	Raspberry	33
4.4.11	Soybean	34
4.4.12	Squash	35
4.4.13	Strawberry	36
4.4.14	Tomato	37
5	Methodology	44
5.1	Convolutional Neural Network(CNN)	45
5.2	Proposed Methodology	45
5.3	Vision Transformer	48
5.4	Compact Convolutional Transformers	51
5.5	ConvMixer	52
6	Pre-trained CNN Models	54
6.1	InceptionResNet V2	54
6.2	ResNet 50	54
6.3	Inception V3	56
6.4	VGG 16	57
6.5	VGG19	59
6.6	DenseNet 201	61
6.7	XCEPTION	61
7	Experimental Results	64
7.1	Results	64
7.2	Confusion Matrix and Classification Report	67
7.2.1	Confusion Matrix	67
7.2.2	Classification Report	67
8	Web Implementation	70
9	Discussions and Further Improvements	73
9.1	Discussion	73
9.2	Further Improvement	73
10	Conclusion	75

List of Figures

3.1	Work Flow of our Proposed CNN Model	13
3.2	Proposed Approach	13
4.1	10 Random Images from Training set	14
4.2	10 Random Images from Validation set	14
4.3	10 Random Images from Testing set	15
4.4	Image Distribution of Training, Validation and Testing	15
4.5	Distribution of Training Images	15
4.6	Distribution of Validation Images	16
4.7	Distribution of Testing Images	16
4.8	Distribution of apple leaves	18
4.9	Apple Scab Samples	18
4.10	Apple Black Rot Samples	19
4.11	Apple Cedar Apple Rust Samples	19
4.12	Apple Healthy Samples	20
4.13	Distribution of blueberry leaves	20
4.14	Blueberry healthy Samples	21
4.15	Distribution of cherry leaves	21
4.16	Cherry Healthy Samples	22
4.17	Cherry Powdery Mildew Samples	22
4.18	Distribution of corn leave	23
4.19	Corn Cercospora leaf spot Gray Leaf Spot Samples	23
4.20	Corn Common Rust Samples	24
4.21	Corn Healthy Samples	24
4.22	Corn Northern Leaf Blight Samples	25
4.23	Distribution of grape leaves	25
4.24	Grape Black Rot Samples	26
4.25	Grape Esca (Black Measles) Samples	26
4.26	Grape Healthy Samples	27
4.27	Grape Leaf Blight (Isariopsis Leaf Spot) Samples	27
4.28	Distribution of orange leaves	28
4.29	Orange Huanglongbing (Citrus Greening) Samples	28
4.30	Distribution of peach leaves	29
4.31	Peach Bacterial Spot Samples	29
4.32	Peach Healthy Samples	30
4.33	Distribution of pepper leaves	30
4.34	Pepper Bell Bacterial Spot Samples	31
4.35	Pepper Bell Healthy Samples	31

4.36	Distribution of potato leaves	32
4.37	Potato Early Blight Samples	32
4.38	Potato Healthy Samples	33
4.39	Potato Late Blight Samples	33
4.40	Distribution of raspberry leaves	34
4.41	Raspberry Healthy Samples	34
4.42	Distribution of soybean leaves	35
4.43	Soybean Healthy Samples	35
4.44	Distribution of squash leaves	36
4.45	Squash Powdery Mildew Samples	36
4.46	Distribution of strawberry leaves	37
4.47	Strawberry Healthy Samples	37
4.48	Strawberry Leaf Scorch Samples	38
4.49	Distribution of tomato leaves	38
4.50	Tomato Bacterial Spot Samples	38
4.51	Tomato Early Blight Samples	39
4.52	Tomato Healthy Samples	39
4.53	Tomato Late Blight Samples	40
4.54	Tomato Leaf Mold Samples	40
4.55	Tomato Septoria Leaf Spot Samples	41
4.56	Tomato Spider Mites Samples	41
4.57	Tomato Target Spot Samples	42
4.58	Tomato Mosaic Virus Samples	42
4.59	Tomato Yellow Leaf Curl Virus Samples	43
5.1	Illustration of our Proposed CNN Architecture	44
5.2	Picture of different leaves in Different states	44
5.3	Feature Map	46
5.4	Image 1	47
5.5	Image 2	47
5.6	Image 3	47
5.7	Image 4	48
5.8	Image 5	48
5.9	Image 6	48
5.10	Shifted Patch Tokenization	50
5.11	Locality Self Attention	50
5.12	Patch Tokenization	51
5.13	ViT Accuracy Curve	51
5.14	ViT Loss Curve	51
5.15	CCT Architecture	52
5.16	CCT Accrucacy Curve	52
5.17	CCT Loss Curve	52
5.18	Architecture of ConvMixer	52
5.19	ConvMixer Accuracy	53
5.20	ConvMixer Loss	53
6.1	Architecture of InceptionResNet V2	54
6.2	Inception V2 Accuracy	55
6.3	Inception V2 loss	55

6.4	Architecture of ResNet 50	56
6.5	ResNet 50 Accuracy	56
6.6	ResNet 50 loss	56
6.7	Architecture of Inception V3	57
6.8	Inception V3 Accuracy	58
6.9	Inception V3 loss	58
6.10	Architecture of VGG16	58
6.11	VGG 16 Accuracy	59
6.12	VGG 16 loss	59
6.13	Architecture of VGG19	60
6.14	Architecture of DenseNet 201	61
6.15	DenseNet 201 Accuracy	61
6.16	DenseNet 201 Loss	61
6.17	XCEPTION Accuracy	63
6.18	XCEPTION Loss	63
7.1	Training, Validation and Testing Accuracy Comparison	66
7.2	Parameter Comparison	67
7.3	Confusion Matrix	68
7.4	Proposed Model Accuracy	68
7.5	Proposed Model loss	68
8.1	Representation of our Website Before Prediction	71
8.2	Representation of our website after prediction	72
8.3	Workflow of our Web	72

List of Tables

4.1	Number of Classes, Healthy and Diseased Image for each Plant	15
4.2	Dataset class details	17
5.1	Layers with Output Shape and Parameters for proposed model	49
6.1	InceptionResNet V2 layers with output shape and parameters	55
6.2	Layer Description of ResNet 50	57
6.3	Layer Description of Inception V3	58
6.4	Layers with output shape and parameters of VGG 16	59
6.5	Layer and Parameter of VGG 19	60
6.6	Layer Description of DenseNet 201	62
6.7	Layer and Parameter Description of Xception	63
7.1	Hyperparameters used for Training models	64
7.2	Training Accuracy and other Metrics	65
7.3	Testing Accuracy and other Metrics	66
7.4	Accuracy Comparison of Proposed model with Transformers	67
7.5	Classification Report	69

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ANN Artificial Neural Network

CCT Compact Convolutinal Transformers

CNN Convolutional Neural Networks

DL Deep Learning

FN False Negative

FP False Positive

KNN K-Nearest Neighbour

LIME Local Interpretable Model-agnostic Explanations

ML Machine Learning

SVM Support Vector Machine

TN True Negative

TP True Positive

ViT Vision Transformer

XAI Explainable Artificial Intelligence

Chapter 1

Introduction

1.1 Problem Statement

Agriculture is the process of improving the soil, growing crops, and caring for animals for human use. The marketplace is where processed products, which can include both plants and animals, are offered for sale. [79] The income of around 87 percent of farmers living in rural parts of Bangladesh is primarily dependent on the growing of crops in order to make a living. On the other hand, the component that was indicated before that has a tendency to do the most damage is one that involves plant diseases or viruses.[11] Viruses and diseases have a propensity to undergo genetic alteration over the course of time and adapt to their environments in order to survive harsh conditions. Because of this, identifying a virus can be a difficult and time-consuming task. [3]In a nation like Bangladesh, where research and studies are scarce, farmers have very limited access to financial resources. As a consequence of this, it ends up being an expensive process for people who put in a lot of effort to discover a reliable answer to their issues. [31]For example, late blight, a disease very commonly found in potatoes which in turn results in a loss of around 25%-57% in the yield of potatoes annually. Therefore, rather than the farmer having to go around seeking assistance from botanists and phytologists, we have devised a solution that will enable the farmer to determine the type of illness that their crops possess within the palm of their hand. It is difficult to keep track of the vast amounts of different combinations of symptoms that are present due to the fact that multiple diseases can share similar symptoms. This can lead to a possible misidentification of the condition if it is not treated with the assistance of a trained professional. Employing a specialist to diagnose the illness will incur expenditures, both monetarily and in terms of the amount of time it will take.

1.2 Background Information

The proliferation of agriculture paved the stage for the rise of civilizations at various points throughout history. Before agriculture became widely practiced, most people subsisted on the land by foraging, hunting, and gathering various wild plants and animals to add variety to their diets. [94] Around 12,000 years ago, our ancestors transitioned from a hunter-gatherer lifestyle to one more closely associated with agriculture by beginning the cultivation of grain and root crops. Even though agriculture is a branch of science, agriculture and science have had an equal im-

pact on each other's development. It has been of great assistance to the goals of a great many researchers and also has been the source of many important discoveries made. This connection is responsible for the transformation and modernization of agriculture; it has paved the way for the mass production of crops through the development of technologies such as genetic modification, pesticides, and fertilizers, amongst other advancements. Agriculture research has been crucial in advancing our understanding of various aspects of the natural world, such as the flora and fauna, weather, climate, soil, and seasonal shifts. We have the power to oversee and take care of the food we eat, thereby guaranteeing that it is produced in an ethical manner and does not include any harmful chemicals.

Agriculture has an extremely significant role in Bangladesh's economy. The modernization of irrigation techniques, the introduction of high-yielding genetically modified crops, the streamlining of market operations by eliminating the need for intermediaries, and the increased use of automation are all factors that have contributed to the expansion of the agricultural sector. Agriculture is vital to the lives of people living in rural areas of Bangladesh. [79] More than 87 percent of rural residents are dependent, at least somewhat, on income from agriculture. In this way, approximately two-thirds of rural families are able to provide for themselves. An increase in farm profits of 10% results in a 6% gain in non-farm incomes. This is one of how Bangladesh's nonfarm sector has been bolstered by the agricultural boom that helps the poor. As incomes from non-farming activities continue to climb, the government should concentrate on expanding the market in rural areas.

As a result, this is the stage in which our proposed Deep Learning (DL) model performs its duties. Image processing of plant leaves, which is accomplished by deploying Deep Learning models, is a tried-and-true method that is also up to date. This method enables accurate disease identification. [39] It is possible to get test accuracy rates of above 90% by utilizing Convolutional Neural Networks (CNNs) like VGG and ResNet. Now, thanks to developments in technologies for machine learning, it is possible to train lightweight models to be used through mobile applications with sufficient accuracy so that they can be practically deployed in field applications. This is made possible because these models can now be used through mobile applications. This would be beneficial to farmers since it is simple to obtain, it would save them time, and it would be cost-effective.

[13] One of the importance in 'precision agriculture' research fields is the plant disease diagnosis using visual evidence from leaves. The broadening of our horizons and the improvement of the precision with which we protect and nurture plants can be assisted by the development of artificial intelligence, graphical processing units, and image processing. [24] Visible symptoms can vary widely across the spectrum for most plant diseases; hence, models should have excellent observational skills so that they can detect the distinctive signs of every disease.

Currently, a variety of artificial intelligence methods are employed for the detection as well as classification of plant diseases. [35] (Deep CNN) Deep convolutional neural network, (SVM) Support vector machine, (KNN) K-nearest neighbors, logistic regression, and decision tree are among the most popular methodologies. In

order to improve feature extraction, these strategies are merged with various image preprocessing techniques. The KNN relies primarily on a similarity score to place data into categories which are mostly classified based on memory. This method was used to determine the identities of unlabeled items by comparing them to nearby identified items. The decision tree is a form of supervised learning in which the neurons stand for individual choice characteristics, the leaves for the many classes, and the branches for the various possible outcomes. However, data overfitting and overlapped nodes are significant drawbacks of this algorithm. As a subset of linear regression, logistic regression uses probability distributions to arrive at a categorical output value.[2] The SVM, an algorithm for machine learning in the presence of supervision, is built using a separating hyper-plane in which statistical learning techniques can be utilized for tasks like regression and classification. [26]In the last couple of years, SVMs have undergone widespread use in many different areas, particularly those involving the classification of images and texts.

A deep learning method, known as deep CNN, was employed in this investigation. Deep learning goes beyond traditional machine learning by incorporating hierarchical data representations and increasing computational complexity. [42]Deep convolutional neural networks have a wide variety of uses in image classification, recommendation systems, audio recognition, object detection, and natural language processing. In comparison to conventional pre-trained models, the Deep CNN model we propose is fairly lightweight, consuming a fair amount fewer computational resources. However, vast quantities of training data is needed to enhance Deep CNN's efficiency. To overcome this issue, image augmentation is applied that produces additional training images from the given images using a range of image processing techniques, such as rotations, image flip, and shifting, among others.

Chapter 2

Related Work

2.1 Machine Learning

In this review, a comparative study of related works in automated plant disease classification is conducted. Some commonly used ML (Machine Learning) methods are studied as they have reached excellent efficiency in achieving related objectives. The flexible framework provided by ML algorithms for making automated expert decisions makes it a powerful tool in feature extraction and classification, which the field of agriculture would also undoubtedly benefit from. But through observation, it is seen that these algorithms are popularly used as classifiers rather than feature extractors, as CNNs are generally better at it.

2.1.1 Logistic Regression

A discrete outcome's probability can be modeled using the logistic regression technique given an input variable. The most important factor in this approach is the input. This method is mostly used for probabilities that have only two outcomes. Therefore they can only have one of two potential values, such as yes or no, true or false, etc. Multinomial logistic regression may be used to describe scenarios with more than two discrete potential outcomes. It is a helpful analysis tool that is used for situations involving classification. [52]The authors of the paper employed Haralick's technique to extract features and use logistic regression to classify them. They achieved an accuracy of 67.3% using these two methods on a Kaggle dataset of 14000 photos. [69]The paper managed to achieve a 96.60% success rate in identifying tomato leaves that are diseased. Their dataset contained 15,989 (254x254 pixel) pictures of tomato leaves from plantvillage.com, which were then categorized into ten different classes. They created a new method for feature extraction by making use of attention-based dilated CNN. This allows them to extract the important features in the shortest amount of time. During their preprocessing, they employed bilateral filtering and Otsu image segmentation. For the purpose of handling unbalanced, noisy, or incorrectly labeled data and achieving accurate prediction results, a synthetic image has been created using the Conditional Generative Adversarial Network (CGAN). Then they are categorized using a logistic regression (LR) classifier that is both quick and easy to understand.

2.1.2 Decision Tree

A decision tree is a useful and widespread method of classification and prediction. It follows a binary tree data structure resembling a flowchart. Each leaf node (terminal node) of our decision tree has a class label, whereas the internal nodes examines an attribute, and each branch represents its potential outcome. [57]This paper proposed an accuracy of 95% in identifying four diseases across five different types of leaves by implementing a decision tree algorithm on a dataset of 1000 images (254 x 254 pixels) taken from kaggle.com. The authors based their proposal on a dataset of 1000 images taken from kaggle.com. After first smoothing the image with a refinement filter, the sheet image is subsequently trimmed to isolate an interesting portion of the picture for further processing. Increasing the contrast of an image is another aspect of image enhancement. The process of segmentation includes splitting an image into groups of pixels according to a number of different criteria. An image serves as the input for the segmentation algorithm, which then produces a curated selection of results. The size of a picture can be reduced by a process known as feature extraction, which displays the interesting aspects of an image as a condensed vector. [51]The authors of this paper introduced the Random Oversampling (RO) method for class branching and also implemented 3 different techniques for feature selection to enhance the algorithm. As a result, the accuracy increased to 98.10%. These techniques are the 1. Consistency filter (Cons), 2. Correlation-based Feature Selection filter(CFS), and 3. Random Forest Importance filter (RFI). The strategy that was presented was used to analyze the SBL dataset that was obtained through UCI Repository of ML. This dataset contains information regarding a wide variety of soybean illnesses that are influenced by a variety of meteorological elements as well as the global and local characteristics of plants. In total, the SBL dataset contains 683 occurrences that are missing certain values. These cases were dealt with using the random forest imputation approach.

2.1.3 SVM Algorithm

Support Vector Machines or SVM's goal is to draw the optimal decision line or boundary to partition or separate and classify clusters of data in any number of dimensions. This boundary may be either linear or nonlinear. This allows us to appropriately categorize any new data points in the future. [52]Authors of used Haralick's algorithm as a feature extractor and SVM as a classifier and got 87.6% accuracy on a Kaggle dataset of 14000 images. SVM was used for detecting vine leaf diseases and bacterial genome associations in [18][19] and detecting citrus greening of lemon trees in [7][33]. In [7][33], using multi-band imaging sensor inputs improves the accuracy up to 85% in [7] and 92.8% by fluorescence imaging in [33]. In papers [4][10][37], hyperspectral images were used with SVM [4][10]. Detecting Tomato yellow leaves and leaf curls with 90% accuracy was done through SVM in [15]. Authors of [21] obtained 88.89% accuracy by using Downy Mildew color features and Powdery Mildew texture as feature extractors, then passing them through SVM classifiers.

2.1.4 Naive Bayes Algorithm

A family of classification algorithms that assumes that every pair of characteristics being categorized are not dependent on each other. As these algorithms are based on Bayes' Theorem, they are known as Naive Bayes classifiers. [29] In plant disease diagnosis was made by processing the images through Naive-Bayes classification, which achieved 81% accuracy in field conditions. To robustly characterize the color characteristics that are present in an infected picture, they created a primary segmentation approach that makes use of a Naive Bayes classifier. During the first segmentation, just a few color channels, including Lab and HSL, are used. The picture is divided into several feature clusters based on these features, where a cluster denotes a collection of pixels with comparable visual or textural feature values. A Naive Bayesian segmentation model is fed by each picture cluster. The choice of the Bayesian model acceptance limit ensures that any blob in the picture that has a chance of displaying the desired illness gets labeled as a candidate.

2.1.5 K-NN Algorithm

K-Nearest Neighbor is a robust Supervised Learning machine learning algorithm that places new instances in classes that are comparable to the other available classes. It does this based on the similarity between the new instance and previous instances. This indicates that when fresh data comes, it will readily be able to be classified into a category that is best suited to it by utilizing the K- NN algorithm. The K-NN technique has applications in both Regression and Classification, although the classification problems are where it sees the greatest use. Regression is not its primary focus. Unlike SVM and other machine learning methods, It does not require any kind of prior training experience. It doesn't need to be retrained if new training patterns are integrated into the current batch of data. 96.76% disease classification accuracy was obtained in [47] on a dataset of 273 images by using K-NN Classifiers. The model classifies five types of diseases of various plant species. Considering color space conversion and color segmentation for feature extraction provides a useful way to read important color information of diseased leaves. In [50] K-NN classifiers are done to classify groundnut leaf diseases. In this study, the k-nearest neighbor classifier is removed from the process of segmenting colored images. A distance measure is also employed to determine similarity by comparing a pixel's characteristics to those of its closest neighbors. The Euclidean distance measure is applied in this study. Three sample areas are first taken from a leaf picture, converted to LAB space, and then divided into three groups using distinct colors. Throughout this research, K-NN classifiers having 3 neighbors were utilized for the segmentation. The segmented parts of the leaf, the disease, and the background are shown in green, red, and blue, respectively.

2.1.6 Random Forest Algorithm

RF (Random Forest) algorithm concatenates the results of many decision trees used on various sections of the input, combines them, and then utilizes the combined results to improve prediction accuracy. RF takes the predictions made by each tree and determines the outcome by taking into account which tree's predictions received the most votes, rather than relying on a single decision tree. Because there are so

many trees in the forest, it is impossible to overfit the data and achieve greater accuracy. [73] Researchers were able to achieve an accuracy rate of 96.1% in their study by putting enhanced photos into a CNN, which then used the Random Forest algorithm as a classifier. This analysis made use of a dataset from plantvillage.com that had 37,315 photos of five different plant species: apple, corn, potato, tomato, and rice.[52] On a Kaggle dataset consisting of 14000 photos, the authors of this paper employed Haralick's approach as a feature extractor and the Random Forest algorithm as a classifier. They achieved an accuracy of 70.05 percent.

2.2 Deep Learning Models

2.2.1 CNN

ANNs (Artificial Neural Networks) are computerized input processors that deal with complex data through mathematical models. They are highly utilized in this field, and their function is inspired by the natural input processors and neuron-neuron communication found in the animal brain.

The authors in [21][36] presented high parameter CNNs using varying datasets for detecting different plant leaf diseases. They also used various convolution and pooling layers for each plant. In [38], several research papers are studied based on DL (Deep Learning) technologies. They then tried to tackle several agricultural problems using the studied techniques. A staggering 98.77% was achieved by the authors in [41] by using techniques such as Stochastic Pooling, Batch Normalization, Dropout, and on a 14-layered CNN to detect brain Sclerosis. [34] In another study, a data augmented 13-layered CNN reached 94.94% final accuracy in classifying fruits using momentum-based stochastic gradient descent. [14][16] Deep CNNs were used by authors of these papers to identify multiple diseases of plants; it can be done for pest detection, too, as used in [27] for tomatoes. 85.54% accuracy was reached for identifying multi-temporal crops in [43].

[45] The paper handles a huge issue, which is the dataset or the number of images. However, we can use data augmentation, but that does not solve many real-life issues. In this paper, what they did was focus on individual lesions and diseased spots. Using this technique can also handle cases like if any leaves have more than one disease. They tested on Original images and background removed images, complete and reduced images. In the paper [48], the author focuses on a big dataset and various types of plant leaf diseases. The usage of CNN is clearly mentioned, a large dataset of images has been trained, and data augmentation has also been implemented. Furthermore, the Adam optimizer was incorporated into the methodology alongside image pre-processing. The paper reached a decent average of 96.5% accuracy with different results across 32 different types of disease identification.

Authors of [40] compare different types of CNN methods using a single dataset that contains a decent number of images. Here, several parameters have been considered, as well as the computing time for searching for the best possible architecture in a convolutional neural network. Additionally, the amount of model loss has also been counted in the paper. Finally, ResNet 101, 99.66%, was found with the highest test

accuracy. [61]The research paper uses Loss function optimization (Softmax, Center-loss) on their Data augmented CNN, which produces better results than AlexNet, VGG16, ResNet-50, and GoogleNet on their dataset, with a test set accuracy of 98.93% on classifying plant diseases.

Joyanta and Farhadul et al.[72] propose a lightweight CNN model which used serially decreasing dense layers in the fully connected classifying block. Their model has very less amount of parameters than other existing models and reached 98.18% accuracy.

In this paper [71], the authors have detected different plants by using transfer learning on three different CNNs: 1. EfficientNetV2L, 2. MobileNetV2, and 3. ResNet152V2. They showed EfficientNetV2L having the better accuracy among them, an accuracy of 99.63%. They also explained the prediction reasonings behind EfficientNetV2L by using LIME , an XAI framework. Their dataset was taken from Kaggle which contained healthy and diseased images of apple, orange, soybean, potato, corn etc. with a total of 14 plants and 38 diseases.

Another study [67], used 3 CNNs: InceptionV3, DenseNet201, and EfficientNetV2S to identify five diseases of rice leaf in Bangladesh and showed detailed comparison between the models. Among them, DenseNet201 achieved the highest accuracy of 92.05%.

This paper [66] proposes a more novel technique of plant disease identification. It handles the separation of crops and disease classification independently. They proposed a trilinear CNN model that used bilinear pooling operations and emphasized that real world environment images require a different strategy as they are not taken in perfect conditions as laboratory images. The crop identification accuracy is 84.11% and disease identification accuracy is 75.58% on the PlantVillage dataset.

Another paper [56] used GoogLeNet architecture on a Kaggle dataset to detect rice leaf diseases with 94% accuracy which they integrated into a website-based application.

The authors of the paper [58] tested leading CNNs such as ResNet-50, VGG-16, VGG-19 etc. to predict nitrogen deficiency of leaf. They replaced the models' last layers with SVM classifiers which increased the accuracy of the models. It is seen that ResNet-50 with SVM had the best result of 99.84% accuracy. Their limitation was that the images are in laboratory conditions and are trained over only 5790 images.

The paper [30] proposed a CNN which was inspired by LeNet-5 and AlexNet to identify 10 common rice diseases. It only has 3 convolution layers and it achieves an accuracy of 95.48%.

The authors of [44] suggested a CNN to identify leaf diseases focusing on the real domain of images from PlantDisease.com. These images are gathered from various settings, angles, and weather conditions and their method had a 93.67% test accu-

racy. The author wants to do additional research on identifying diseases in various stages and from various regions of the plant body.

A data augmented ResNet CNN was used to identify plant leaf disease in another paper [55]. 500 plant images from a tailored dataset were utilized in it and a detailed comparison of proposed models and handcrafted Non-deep ML approaches. In the end, the model obtained a high accuracy of 98.96%.

In [60], the author used CNNs put forth a method for recognizing the Cassava disease of plants. To simulate real world scenarios, they used low-resolution images. The model uses Chebyshev orthogonal functions for image color histograms. Pre-trained MobileNet-v2 was used in this experiment and managed 99.7% accuracy on the Cassava Disease 2019 dataset.

To identify plant diseases, the author of [54] highlights the use of Bayesian DL models. The PlantVillage dataset was utilized for this study. The author fine-tuned various CNN architectures like VGG16, SGD, SGLD etc. The improved VGG16 got the highest accuracy among the models which is 96%.8.

2.3 Explainable Artificial Intelligence

Artificial intelligence (AI) applications are proliferating as a result of machine learning's spectacular success. Despite their ubiquitous use, machine learning models are still largely unexplored. Future developments should result in automated self-learning systems that make decisions independently. However, their incapacity to justify their choices and behaviors to human users limits their usefulness. When determining whether to deploy a new model or how much faith to place in a prediction or when planning to act in response to a prediction, understanding the motivations behind predictions is crucial. A trustworthy model or prediction may be created using these model-specific insights that can be used to refine inaccurate predictions. New machine-learning algorithms will be able to justify their actions, identify their advantages and disadvantages, and provide a sense of how they will act in the future. The approach to accomplishing that objective is to create new or altered machine-learning approaches that will result in more comprehensible models. Aiming to help people understand, appropriately trust, and competently navigate the emerging era of artificial intelligence, the Explainable AI (XAI) initiative seeks to develop a suite of machine learning techniques that generate more comprehensible models while maintaining high-performance learning and precision of predictions.[20] Trust is crucial for productive engagement with machine learning systems, and the publication argues that this trust can be established through the clarification of individual predictions. In order to accurately and understandably describe the predictions of any model, they thus introduced LIME, a modular and expandable method. In addition, they came up with SP-LIME, a method for identifying relevant predictions that provides a complete picture to the model's end users. Their experiments with expert and non-expert users in the text and image domains revealed that explanations are beneficial for a wide range of models in tasks involving trust. These tasks include selecting a model, evaluating its trustworthiness, improving an unreliable model, and understanding its assumptions.

2.4 Transformers

Transformers are one of the newest most influential models in the field of computer science. Transformers are Neural Networks that look for relationships and patterns in sequential data. It is widely used for Natural Language Processing as it looks for the meaning of the input phrases in a certain context. To find this meaning, these models use a series of mathematical operations referred to as "self-attention". A methodology for classifying images called the Vision Transformer, or ViT, applies a Transformer-like design to selected areas of the image. Two main elements of the transformer architecture are feed-forward networks and self-attention. Self-attention changes n elements (or tokens) into n output tokens after receiving n input tokens. It is a sequence-to-sequence module that compares each input token to each token in the sequence. Also computed for each of these couples is an attention score. Then, it sets the current token to equal the weighted average of all input tokens, with the weights determined by the attention scores [32]. ViT comes in a variety of forms, including ConvMixers, Compact Vision Transformers (CVT), Compact Convolutional Transformers (CCT). Convolutional embeddings are used instead of patch embeddings in CCTs, which improves inductive bias and eliminates the need for positional embeddings. With smaller ViTs, CCT is more accurate than ViT-Lite and has more flexible input parameters. ConvMixers, however, are ViTs with additional convolutions. ConvMixer blocks are regularly repeated in place of the sequential convolution, pooling, and transformers of CNNs and ViTs. A depthwise separable convolution, a common component of contemporary CNN architectures, is slightly adjusted in a single ConvMixer block. By pooling or using strided convolutions, a standard CNN gradually reduces the feature size while increasing the number of channels. On the other hand, ConvMixer's intermediate features all have uniform size.

Authors of the paper [53] compared three of the original ViT models against ResNet on several datasets. The three models were ViTBase which has 12 layers and 86M parameters, ViTLarge which has 24 layers and 307M parameters and ViTHuge with 32 Layers and 632M parameters. These numbers are only possible due to ViT's efficiency. ViTLarge and ViTHuge surpassed ResNet's accuracy on all datasets tested, as ResNet reached 87.52%, ViTLarge 87.76% and ViTHuge 88.55% on ImageNet.

Since ViTs require an enormous size of dataset, it is difficult to implement them due to lack of locality inductive bias. But the paper [70] proposes a novel method for working with ViTs and small sized datasets by using the modules Shifted Patch Tokenization (SPT) and Locality Self-Attention (LSA). Just by adding them to different ViTs can result from upto 2.96% to 4.08% accuracy gains. ViTs tested are: ViT, PiT, T2T and CaiT.

This paper [62] extensively compares CNN performances against ViTs, CCTs and CVTs. They tested on CIFAR-10, CIFAR-100, MNIST, ImageNet etc. datasets and used different CNNs like ResNets, MobileNets and Proxyless-G. The results show that CCTs outperform CNNs and all other ViTs.

Another paper [68] tested ConvMixers and Transformers for object segmentation

on the Tabletop Object Dataset. The accuracies range around 83% to 85.55% with varying degrees of GFLOPs required. It is seen that ConvMixers outperform the tested transformers. They also tested the impact of CMCF modules, patch embedding and different input modes on the ConvMixer’s performance.

The paper [74] uses ConvMixer to recognize hypertension of blood using ballistocardiography spectrograms and compares it with other methods. Though the ConvMixer did better than other authors’ methods, using ResNets on the spectrograms yielded better results. And even though ConvMixer did better than other related works, they were not done on the same dataset.

Chapter 3

Research Objectives

We want to construct a Deep Learning Model utilizing a lightweight CNN that is capable of detecting plant leaf diseases in a diverse range of plants, such as apples, blueberries, cherries, and corn, in addition to citrus fruits like oranges as well as peaches. The fact that this model has a significantly smaller number of parameters than usual was the sole factor that influenced the decision to employ a lightweight CNN. As a direct consequence of this, a lower amount of resources will be consumed, and a minimum level of computational power will be necessary. Despite the fact that we have used a very limited number of parameters, one of our primary concerns is to guarantee that the accuracy of our results is never jeopardized. This model will be able to identify diseases of the plants from the image of the leaves. The user will be required to provide our system with an image as their input. When all of the necessary preprocessing steps have been completed, the image will then be run through our proposed CNN model, which will determine whether or not the image consists of a diseased plant and, if so, what disease the plant is suffering from.

At the end of this research, we tend to

1. Have a solid understanding of AI and how it will help in our Plant Leaf Disease identification.
2. Have a thorough understanding of the Deep Learning Model.
3. Develop a model that will detect whether a plant is diseased or not.
4. Evaluate our proposed CNN model and compare it with other models.
5. Implement XAI to explain the results our CNN model displays.
6. Create user Friendly website to detect plant leaf disease.
7. Talk about improvements and further work that can be done.

3.1 Workflow

To start, we will investigate the PlantVillage dataset that was published by spMohanty. It has around 60,343 images. In addition to that, it consists of images of all 38 classes shown in unfiltered, grayscale, and segmented forms. On the other hand, just the unfiltered photos were used. After that, we begin the preparation stage by applying the data augmentation methodology applied, such as rotating the image and flipping the image horizontally on random images. The images were resized from 256x256 to 100x100, and data was split into the train, test, and validation

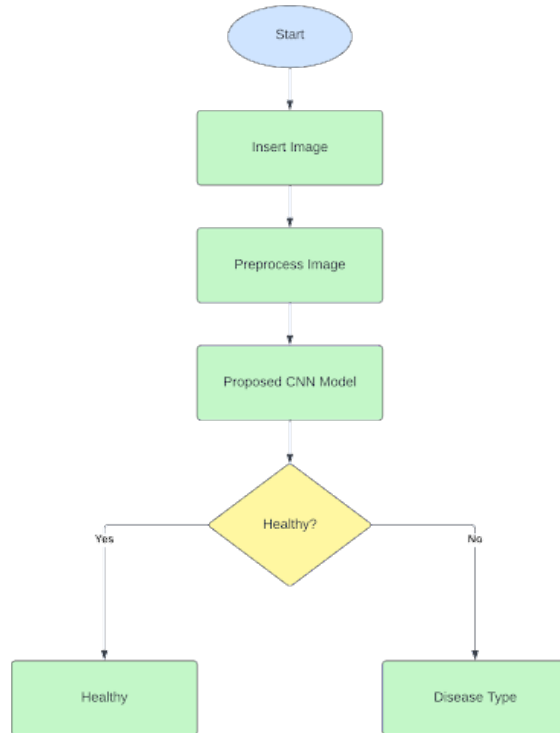


Figure 3.1: Work Flow of our Proposed CNN Model

sets. Furthermore, we then construct our CNN model and then train it with the help of some training images, and lastly, we will validate it using some validation images. In order to understand the CNN model's output or prediction, we first save our model in .h5 format and then upload the predicted image to LIME. This is done after confirming the accuracy of the outcomes and utilizing test images to check them. Finally we will create a simple user friendly website, where user can upload an image and know whether the leaf is diseased or not.

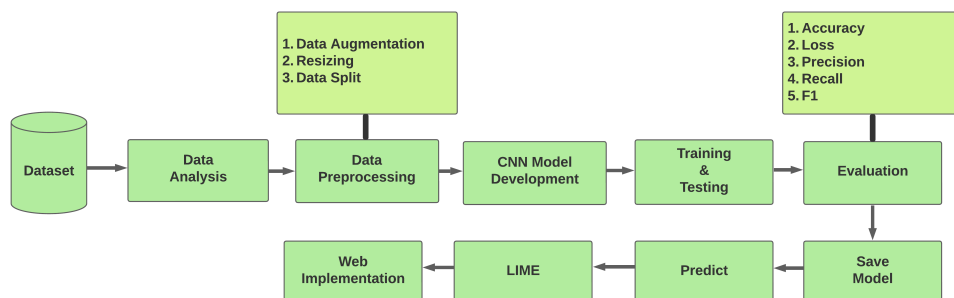


Figure 3.2: Proposed Approach

Chapter 4

Dataset

4.1 Data Analysis

The PlantVillage Dataset is the most important reference material that we make use of[12]. This collection comprises 38 unique plant leaf photo categories. A total of 60,343 pictures are included in this particular dataset that has been compiled. The majority of the images show leaves that are both affected and unaffected by the disease. These photographs have been organized into 38 distinct categories for your viewing pleasure. In addition, three components make up the dataset: training, validation, and testing, each of which has a different number of images: 48280, 6035, and 6028. This dataset includes at least one thousand pictures for each of its categories. In Table 4.1, we can see what plants and how many diseased and healthy images are present in our dataset. It also shows how many different classes are there for each plant. Figure 4.1, 4.2 and 4.3 contains 10 sample images from training, validation and testing set.



Figure 4.1: 10 Random Images from Training set

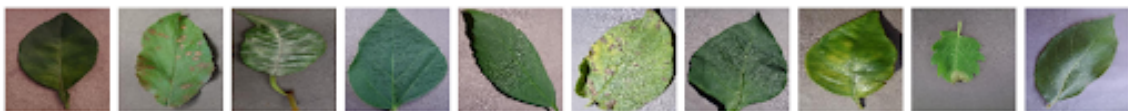


Figure 4.2: 10 Random Images from Validation set

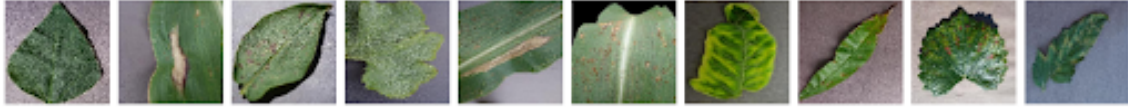


Figure 4.3: 10 Random Images from Testing set

Table 4.1: Number of Classes, Healthy and Diseased Image for each Plant

Type	No. of Classes	Healthy	Diseased
Apple	4	1645	3000
Blueberry	1	1502	0
Cherry	2	1000	1052
Corn	4	1162	3192
Grape	4	1000	3639
Orange	1	0	5507
Peach	2	1000	2297
Pepper	2	1478	1000
Potato	3	1000	2000
Raspberry	1	1000	0
Soybean	1	5090	0
Squash	1	0	1835
Strawberry	2	1000	1109
Tomato	10	1591	17244
Total	38	18468	41875

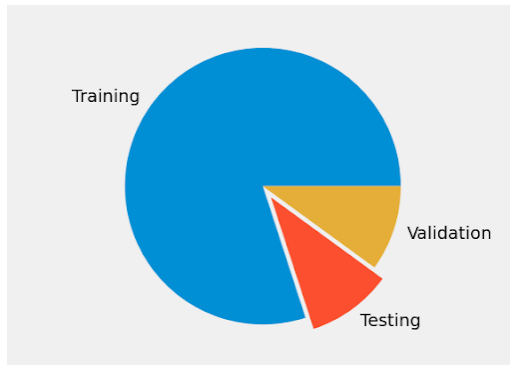


Figure 4.4: Image Distribution of Training, Validation and Testing

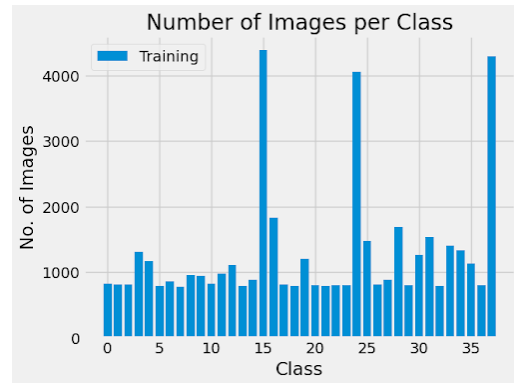


Figure 4.5: Distribution of Training Images

4.2 Data Preprocessing

Data pre-processing is a technique meant to eliminate extraneous variables that do not contribute to improving CNN model accuracy. This is accomplished by incorporating a technique known as "de-noising." [59] In addition, to improve the performance of the CNN models, raw data are modified, resulting in greater accuracy

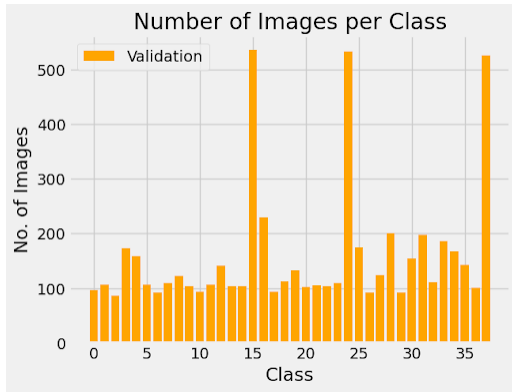


Figure 4.6: Distribution of Validation Images

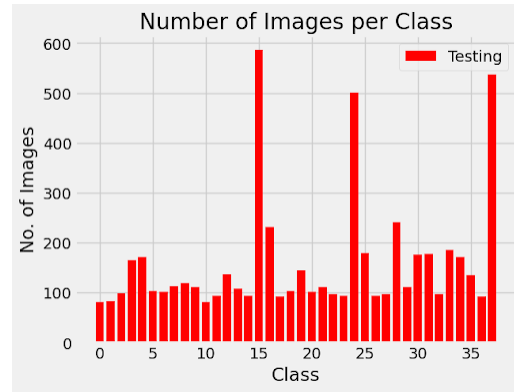


Figure 4.7: Distribution of Testing Images

and better results. The first phase in processing our data is referred to as "Image Resizing," and the second step is known as "Data Augmentation." Each image is 256 by 256 pixels. First, we will resize the image to 100 by 100, which will accelerate our training. The second step was data augmentation. A) rotation and B) horizontal flips have been performed to supplement the data. Then, we turned the photos into a matrix and normalized them by dividing them by 255, so conserving computer resources.

4.3 Detailed Dataset

Table 4.2 shows image distribution of each classes in training, testing and validation set.

4.4 Dataset Classes

4.4.1 Apple

In the Fig 4.8, we can see the image distribution of apple- healthy and its diseased leaf on Training, Validation and Testing set.

Apple Scab

On newly sprouting and immature leaves in the early spring, apple scab infections begin. Ten days later, early lesions show up as regions of paler green than the surrounding leaf tissue. The development of asexual spores causes lesions to enlarge, change color to an olive shade, and become velvety. On young leaves, scab sores can grow to be more than one centimeter in diameter. However, ontogenetic resilience in older leaves typically results in fewer lesions or no symptoms at all. Eventually, damaged tissues may develop distortions and puckers, and leaf lesions may develop cracks and tears. Leaves that are seriously ill fall from the trees. Trees may become weaker after two to three successive defoliation occurrences, making them more vulnerable to other pressures like freezing damage, insect harm, and other diseases.[76]. Sample of Apple Scab is shown at Fig 4.9.

Table 4.2: Dataset class details

NAME	Training	Testing	Validation	Total
Apple Scab	821	82	97	1000
Apple Black Rot	811	83	106	1000
Apple Cedar Apple Rust	815	99	86	1000
Apple Healthy	1306	166	173	1645
Blueberry Healthy	1172	172	158	1502
Cherry Healthy	789	104	107	1000
Cherry Powdery Mildew	858	102	92	1052
Corn Cercospora Leaf Spot Gray Leaf Spot	777	114	109	1000
Corn Common Rust	950	120	122	1192
Corn Healthy	946	112	104	1162
Corn Northern Leaf Blight	825	82	93	1000
Grape Black Rot	979	95	106	1180
Grape Esca (Black Measles)	1105	137	141	1383
Grape Healthy	789	108	103	1000
Grape Leaf Blight (Isariopsis Leaf Spot)	878	94	104	1076
Orange Haunglongbing (Citrus Greening)	4385	586	536	5507
Peach Bacterial Spot	1836	232	229	2297
Peach Healthy	814	92	94	1000
Pepper Bell Bacterial Spot	784	104	112	1000
Pepper Bell Healthy	1200	145	133	1478
Potato Early Blight	796	102	102	1000
Potato Healthy	783	112	105	1000
Potato Late Blight	799	97	104	1000
Raspberry Healthy	796	94	110	1000
Soybean Healthy	4056	501	533	5090
Squash Powdery Mildew	1481	179	175	1835
Strawberry Healthy	814	94	92	1000
Strawberry Leaf Scorch	887	98	124	1109
Tomato Bacterial Spot	1686	241	200	2127
Tomato Early Blight	796	112	92	1000
Tomato Healthy	1261	176	154	1591
Tomato Late Blight	1533	178	198	1909
Tomato Leaf Mold	792	97	111	1000
Tomato Septoria Leaf Spot	1399	186	186	1771
Tomato Spider Mites	1336	172	168	1676
Tomato Target Spot	1126	136	142	1404
Tomato Mosaic Virus	806	93	101	1000
Tomato Yellow Leaf Curl Virus	4293	538	526	5357
Total	48280	6035	6028	60343

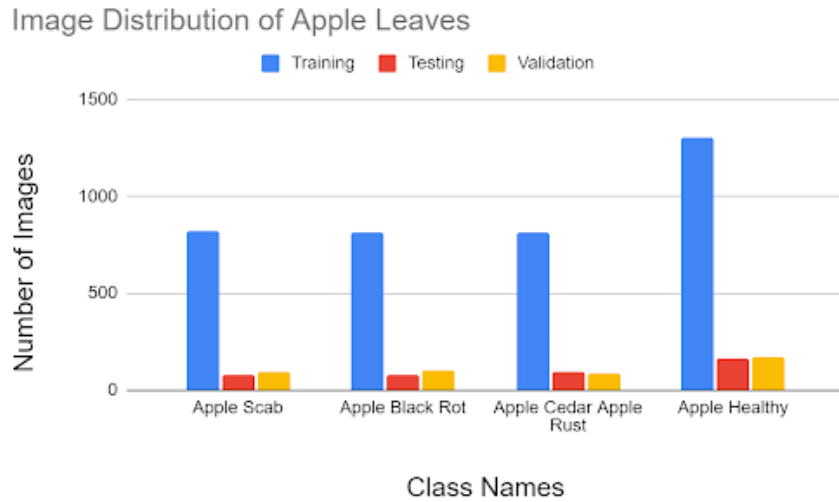


Figure 4.8: Distribution of apple leaves

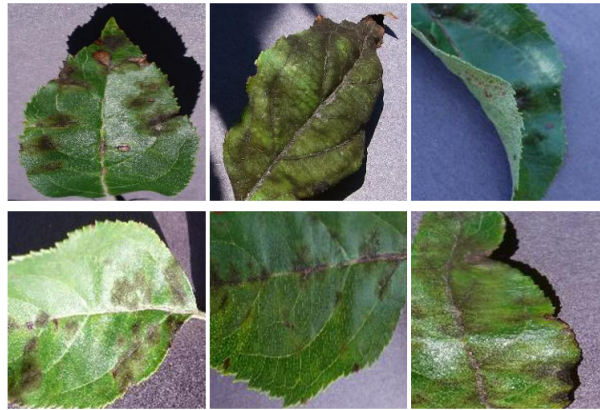


Figure 4.9: Apple Scab Samples

Apple Black Rot

One stage of the common and harmful apple disease is known as black rot. Black rot and frog-eye leaf spot are the terms used to describe the fruit rot stage. Losses from fruit spoilage prior to harvest and during storage, tree weakening from defoliation, and blighting and dieback of twigs and limbs brought on by girdling cankers are all possible effects of the disease. Small, low-quality fruit can be produced as a result of infected leaves falling off too early, which lowers crop yield the next year. Fruit rot appears to affect all apple cultivars in the same way. On affected leaves, little purple specks start to develop. These specks grow into spots that are about 1/8 and 1/4 inches in diameter. The spherical to irregularly lobed spots take on a frog-eye appearance due to their visibility with a light brown to the grayish core, one or more circumferential rings made of dark brown, and a purple edge [80]. In Fig 4.10, diseased leaf infected with Black Rot is shown.

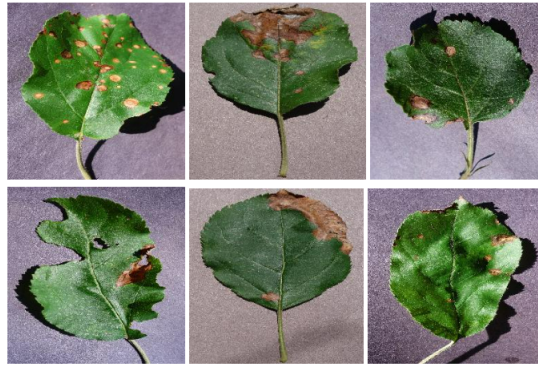


Figure 4.10: Apple Black Rot Samples

Apple Cedar Apple Rust

Gymnosporangium juniperi-virginianae is a fungus pathogen that causes the common plant disease cedar-apple rust. In the United States, cedar-apple rust is a frequent disease that affects apple farmers. On redcedar and juniper, cedar-apple rust is normally a non-lethal disease, but the symptoms can be severe and destructive on apples. Many efficient management techniques are in place now to reduce production losses from this disease, which has been researched since the early 1900s. On apple and crabapple plants, small yellow dots that later turn orange and form red rings around the edge are the first signs of a fungal infection. On these lesions on the upper surface of the leaf, small, sporulating pustules that are orange or dark in color appear. Aecia, or fungal fruiting bodies, start to develop as hairy, tube-like growths on the underside of the leaf a short time later.[81] Example of such condition is shown in Fig 4.11.

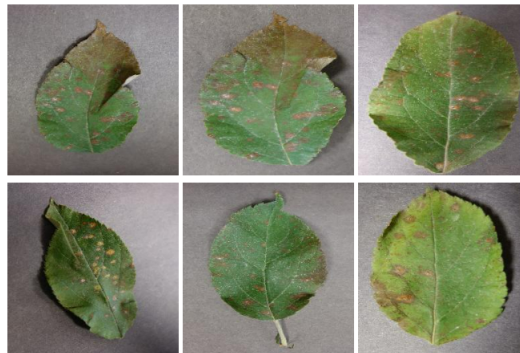


Figure 4.11: Apple Cedar Apple Rust Samples

Apple Healthy

A healthy apple leaf should be a dark green color and have a glossy appearance. The leaf should be free of spots, holes, or discoloration, and should be firmly attached to the tree. Additionally, the leaf should be a consistent shape and size and have smooth edges. If the leaf is wilted, yellowing, or appears to be diseased in any way, it may indicate that the tree is not healthy and may require attention from a professional. Fig 4.12 shows how healthy apple leaf might look like.

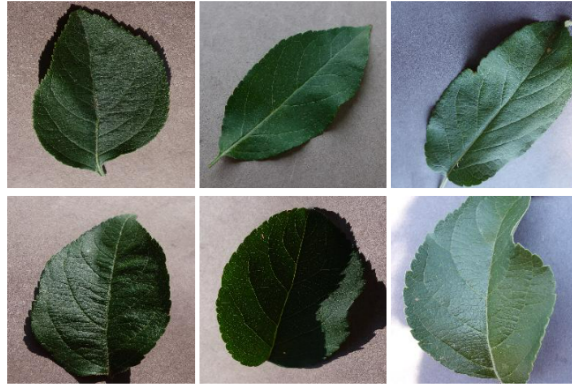


Figure 4.12: Apple Healthy Samples

4.4.2 Blueberry

In the Figure 4.13 we can see the image distribution of blueberry healthy and diseased leaf on Training, Validation and Testing set.

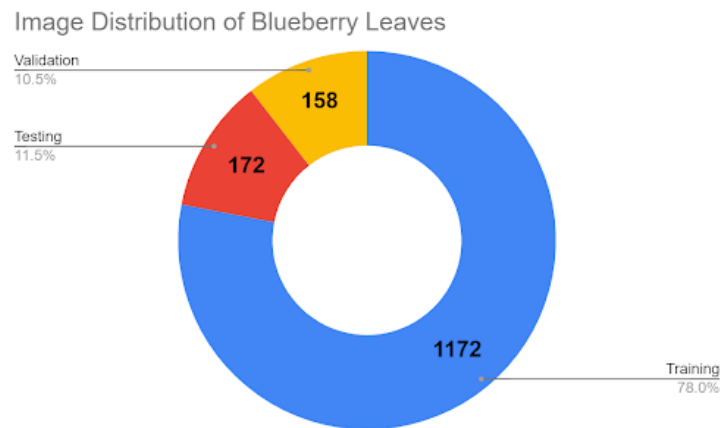


Figure 4.13: Distribution of blueberry leaves

Blueberry Healthy

Dark green in hue and glossy are characteristics of a healthy blueberry leaf. The leaf's edges should be smooth and not brittle, and it should be devoid of any blemishes or discoloration. The leaf should be strong and difficult to tear or damage. The blueberry leaves will remain healthy if they receive the proper care, which includes enough water, sunlight, and nutrients. Additionally, regular insect and disease inspections can aid in avoiding any problems that might damage the leaves. In general, a strong blueberry leaf indicates that the plant is flourishing and receiving proper care as shown in Fig 4.14.

4.4.3 Cherry

In the Figure 4.15 we can see the image distribution of cherry healthy and diseased leaf on Training, Validation and Testing set.



Figure 4.14: Blueberry healthy Samples

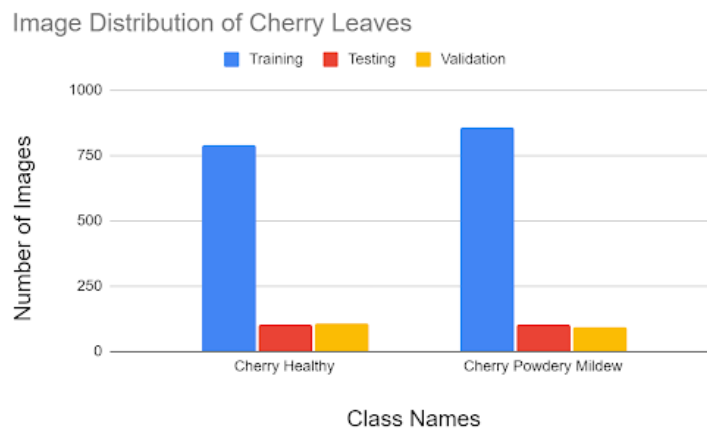


Figure 4.15: Distribution of cherry leaves

Cherry Healthy

A healthy cherry leaf should be a vibrant green color, with a smooth and glossy texture. The leaf should be firm to the touch and have an evenly shaped, symmetrical appearance. The edges of the leaf should be smooth, without any frayed or jagged edges. The leaf stem, or petiole, should be a green or reddish-green color and be firmly attached to the leaf, with no signs of wilting or discoloration. In general, a healthy cherry leaf should be free from any visible signs of disease or damage, such as spots, discolorations, or holes. Additionally, a healthy cherry tree should have a good amount of leaves for photosynthesis and a consistent leaf growth. It's also worth noting that during the spring cherry leaves can appear smaller and are hairier than leaves that grow during summer.

Cherry Powdery Mildew

Podosphaera clandestina is an obligatory biotrophic fungus that causes powdery mildew on cherries. Commonly, mid and late-season sweet cherry varieties are impacted, rendering them unmarketable due to a surface layer of white fungal growth. On young, vulnerable leaves, the first symptoms are typically light, approximately circular, powdery-looking patches that appear 7 to 10 days after the start of the first irrigation. Older leaves are inherently more resistant to infection than younger

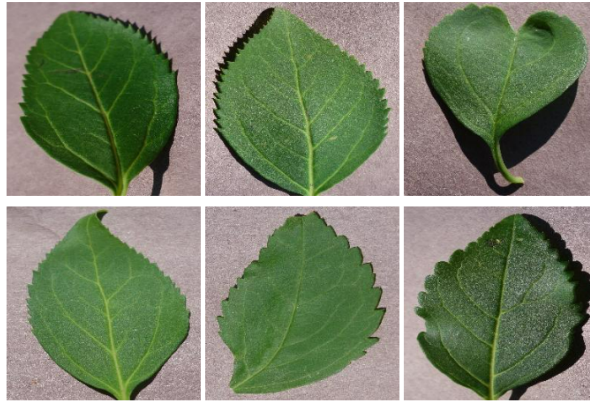


Figure 4.16: Cherry Healthy Samples

leaves and naturally resist powdery mildew with age.[82]



Figure 4.17: Cherry Powdery Mildew Samples

4.4.4 Corn

In the Figure 4.18 we can see the image distribution of corn healthy and diseased leaf on Training, Validation and Testing set.

Corn Cercospora leaf Spot Gray Leaf Spot

Practically every growing season, the fungus *Cercospora zae-maydis*, which causes gray leaf spots, manifests itself. Economic losses could happen if the development of a disease is encouraged. About two to three weeks prior to tasselling, lower leaves first show signs of the condition. The leaf lesions are rectangular, long, narrow, and light tan in color. The lesions may eventually turn gray. The leaf veins typically serve as their boundaries, although they can clump together and kill whole leaves. The disease is frequently more severe in corn planted after corn because the fungus persists in corn residue. Wind and splashing water can spread spores. Warm, humid conditions are more conducive to disease growth and infection of corn leaves. Environmental factors and hybrid susceptibility affect disease severity.[84] Fig 4.19 shows corn leaves with such conditions.

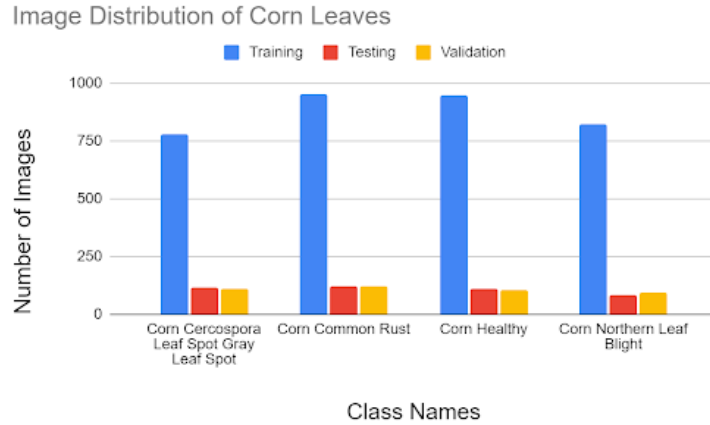


Figure 4.18: Distribution of corn leaf

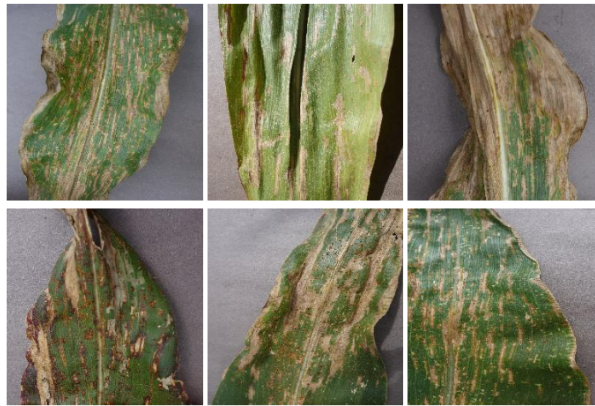


Figure 4.19: Corn Cercospora leaf spot Gray Leaf Spot Samples

Corn Common Rust

Mid to late summer is a common time for common rust to appear in the northern United States. Rarely does it get to the point where commercial hybrids lose yield. The worst times are when there are extended periods of chilly, rainy weather. When rust illnesses manifest as brown pustules, they are typically simple to spot. On both leaf surfaces, common rust causes elongated, rust-colored to dark-brown pustules. Rust spores with a cinnamon-brown hue are seen in the pustules. Pustules become darker with time. Sheaths and leaves can both contract an infection. Chlorosis and even the death of leaves can happen under extreme circumstances.[83] In fig 4.20, corn leaves suffering from common rust is shown.

Corn Healthy

A healthy corn leaf should have several key features. Firstly, it should be a vibrant green color, with a smooth and glossy surface. The leaf should be firmly attached to the stem and be of an appropriate size and shape for the corn plant. The leaf should be free from any signs of disease or pest damage, such as yellowing, wilting, spotting, or holes. Secondly, the leaf should have well-defined veins that are the same color as the leaf blade. Thirdly, the leaf should not have any mechanical damage or other



Figure 4.20: Corn Common Rust Samples

signs of injury, such as tears or frayed edges. Lastly, the leaf should be free from any signs of fungal or bacterial infections, such as powdery mildew or leaf spot. The leaves of the corn plant are important for photosynthesis, which helps the plant to convert light energy into chemical energy, this process is essential for the growth and development of the corn plant. A healthy leaf will help ensure the proper growth and development of the corn, leading to a bountiful harvest. Fig 4.21 illustrates how they might look like.



Figure 4.21: Corn Healthy Samples

Corn Northern Leaf Blight

Canoe-shaped lesions between an inch and six inches long are typical signs of northern maize leaf blight. The margins of the lesions are initially gray-green. They gradually turn tan and may have black fungal sporulation spots on them. When different corn hybrids react with different resistance genes, the length or extent of lesions can vary. Upper leaves then become affected by lesions that start on the lower leaves. When symptoms are severe, they can spread quickly and cause blighted leaves. The disease is more common when there are lengthy periods of mild temperatures and dampness. The disease often manifests at or after silking, but when infection develops sooner, the condition is typically more severe.[91] Fig 4.22 shows what they might look like.

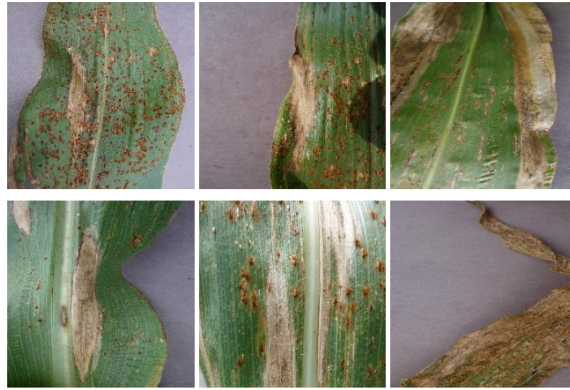


Figure 4.22: Corn Northern Leaf Blight Samples

4.4.5 Grape

In the Fig 4.23 we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

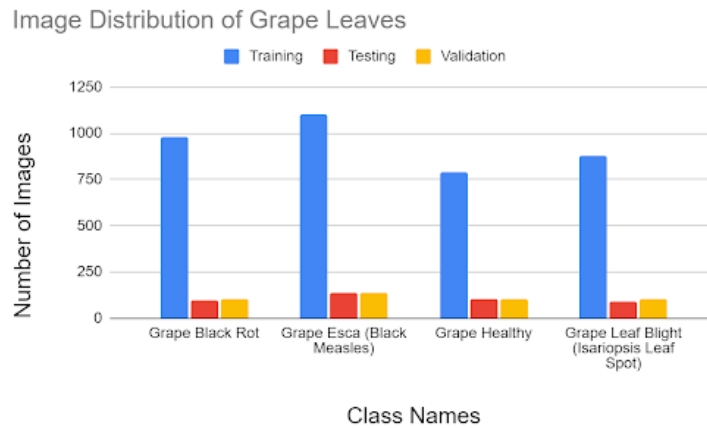


Figure 4.23: Distribution of grape leaves

Grape Black Rot

One of the most harmful grape diseases currently recognized is black rot. The fungus *Guignardia bidwellii* is responsible for the illness. The fungus can affect grape cluster stems, leaves, shoots, berries, tendrils, and rachises. Warm, humid weather is favorable for the development of disease. Black rot symptoms initially manifest as little yellow patches on leaves. Larger patches feature tan to dark brown cores and a border that is dark brownish-red. As shown in Fig 4.24, tiny black dots start to emerge in the lesion as the infection spreads; these dots are typically arranged in rings close to the lesion’s edge. These specks are fungus formations, which are filled with spores that can infect new tissue and number in the thousands.[87]

Grape Esca (Black Measles)

Many various diseases affect grape producers, including "trunk illnesses," which typically result in the unexpected death of grapevines. There are a number of

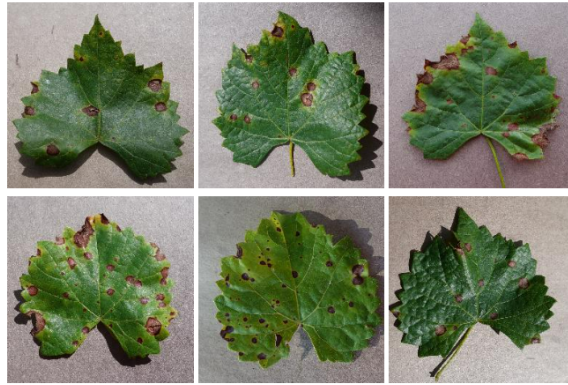


Figure 4.24: Grape Black Rot Samples

known grapevine trunk diseases, but recently, esca appears to be the most destructive. *Phaeoacremonium aleophilum*, *Phaeoaniella chlamydospora*, and *Fomitiporia mediterranea* are only a few of the fungi that produce esca. On mature grapevines in vineyards that are 5 to 7 or even 20 years old, esca symptoms become visible. Dark red or yellow bands on leaves, which gradually dry out and turn necrotic, are the first signs of esca.[85] Fig 4.25 shows few grape leaves with such condition.

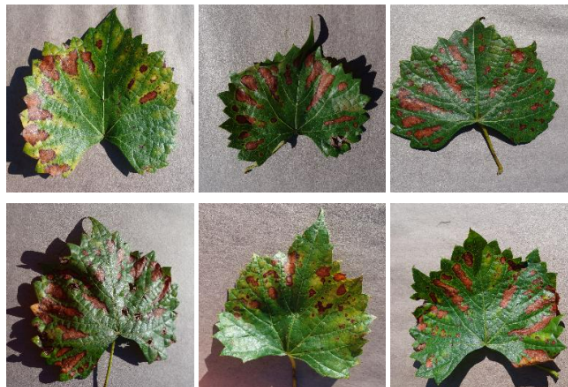


Figure 4.25: Grape Esca (Black Measles) Samples

Grape Healthy

As shown in Fig 4.26, a healthy grape leaf should be glossy and shiny with a bright green color. The leaf should be firmly fastened to the stem and fit the grape variety in terms of size and shape. In addition, the leaf should not exhibit any disease or pest damage symptoms, such as yellowing, wilting, spotting, or holes. A healthy grape leaf should also have clearly visible veins that match the color of the leaf blade.

Grape Leaf Blight (Isariopsis Leaf Spot)

Grapevine bacterial blight is a dangerous, persistent, and systemic disease that damages commercially significant cultivars. The bacterium *Xylophilus ampelinus*, which persists in the vascular tissues of infected plants, is the cause of it. A considerable decline in the health of the grapevine and significant harvest losses can result from

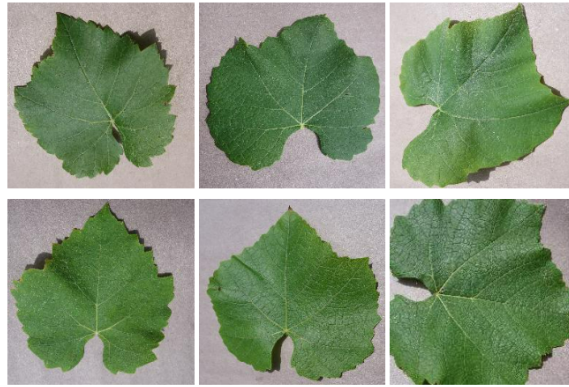


Figure 4.26: Grape Healthy Samples

severe infection of vulnerable varieties. Linear reddish-brown lines that start on the shoot and spread upward, darken, crack, and eventually turn into cankers are the main symptoms. Following this, the shoots begin to wilt, droop, and dry out. Young shoots may also start to acquire light yellowish-green patches on their lowest internodes. On very young shoots, discoloration is less frequent, but the entire shoot withers. Tissue browning is visible in the stem cross-section.[75] Few samples of the discussed condition is shown in Fig 4.27.



Figure 4.27: Grape Leaf Blight (Isariopsis Leaf Spot) Samples

4.4.6 Orange

Fig 4.28 illustrates the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Orange Huanglongbing (Citrus greening)

The worst orange disease, Huanglongbing, known famously as citrus greening, is currently destroying the global citrus industry. The health of the trees along with the fruit growth, ripening, as well as quality of the juice of citrus fruits are all impacted by the suspected causative bacterial pathogen *Candidatus Liberibacter* spp. It has been suggested that a significant buildup of starch in the upper portions of symptomatic trees is additionally brought on by reduced degradation and compromised transport, this results in a poorly distributed distribution of photoassimilates

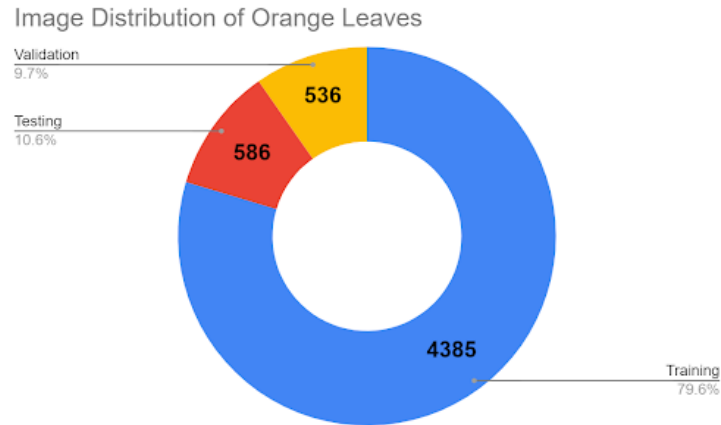


Figure 4.28: Distribution of orange leaves

among mature citrus leaves, roots, and young leaves. Accumulation of starch in leaves has also been correlated with reduced deterioration and poor transport. The sugar content of fruit would be impacted by this imbalance in sugar transport and accumulation. Because the starch does not break down even during the night cycles, it persists permanently in the aerial plant parts, starving the roots and causing severe health deterioration and eventual death of trees.[46] Some examples of annotated samples are exhibited in Fig. 4.29.



Figure 4.29: Orange Huanglongbing (Citrus Greening) Samples

4.4.7 Peach

In the Fig. 4.30, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Peach Bacterial Spot

On older peach trees and nectarines, bacterial leaf spot of peach is a prevalent disease. The bacteria *Xanthomonas campestris* pv. *pruni* is the root cause of this peach tree leaf spot disease. Bacterial spots on peach trees induce fruit loss and general tree malaise due to recurring defoliation. The jagged purple to purple brown patches on foliage that are a hallmark of peach tree leaf spot are followed by the

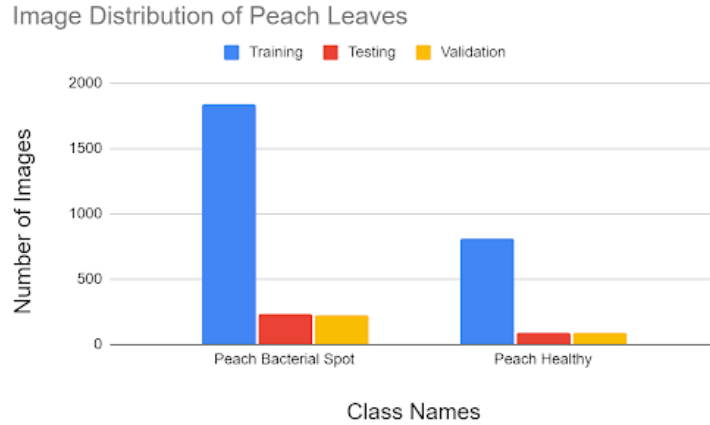


Figure 4.30: Distribution of peach leaves

core of the lesion dropping out, which results in a “shot hole” type look into the appearance of the leaves. The leaves quickly turn yellow and fall [77]. Fig. 4.31 shows what it might look like.



Figure 4.31: Peach Bacterial Spot Samples

Peach Healthy

A healthy peach leaf should have several key features. Firstly, it should be a vibrant green color, with a smooth and glossy surface. The leaf should be firmly attached to the stem and be of an appropriate size and shape for the peach tree. The leaf should be free from any signs of disease or pest damage, such as yellowing, wilting, spotting, or holes. Secondly, the leaf should have well-defined veins that are the same color as the leaf blade. Thirdly, the leaf should not have any mechanical damage or other signs of injury, such as tears or frayed edges. Lastly, the leaf should be free from any signs of fungal or bacterial infections, such as powdery mildew or leaf spot. Having all these features indicate the tree is in good health and able to photosynthesize well, which helps to support the growth and development of the peach tree. Fig. 4.32 illustrates few healthy peach leaf samples.



Figure 4.32: Peach Healthy Samples

4.4.8 Pepper

In the Fig. 4.33, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

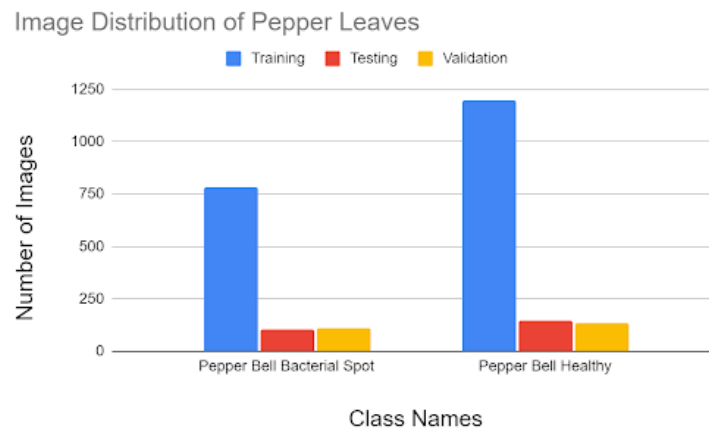


Figure 4.33: Distribution of pepper leaves

Pepper Bell Bacterial Spot

The most prevalent and harmful disease affecting peppers is bacterial leaf spot, which is brought on by *Xanthomonas campestris pv. vesicatoria*. Leaf spot, fruit spot, and stem canker are just a few of the signs that can show up on the plant's surface. Early symptoms, on the other hand, manifest as water-soaked lesions on leaves that can swiftly turn from green to dark brown and grow into patches that are up to 1/4 inch diameter with mildly higher borders. These patches may eventually become dry in less humid conditions, allowing the affected tissues to fall off and giving the affected leaves a tattered appearance.[86] Few samples of the discussed condition is shown in Fig. 4.34.

Pepper Bell Healthy

A healthy bell pepper leaf should have several key features. Firstly, it should be a vibrant green color, with a smooth and glossy surface. The leaf should be firmly



Figure 4.34: Pepper Bell Bacterial Spot Samples

attached to the stem and be of an appropriate size and shape for the bell pepper plant. The leaf should be free from any signs of disease or pest damage, such as yellowing, wilting, spotting, or holes. Secondly, the leaf should have well-defined veins that are the same color as the leaf blade. Thirdly, the leaf should not have any mechanical damage or other signs of injury, such as tears or frayed edges. Lastly, the leaf should be free from any signs of fungal or bacterial infections, such as powdery mildew or leaf spot. Bell pepper leaves are essential for photosynthesis process, this process is what the plant uses to produce chemical energy from light energy to support its growth and development; a healthy leaf will help ensure the proper growth of the plant and an abundant harvest of bell peppers. Fig. 4.35 shows what healthy bell pepper leaves should look like.



Figure 4.35: Pepper Bell Healthy Samples

4.4.9 Potato

In the Fig. 4.36. we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Potato Early Blight

According to J E van de Waals et al [1] early blight of potato is prevalent in the majority of potato-growing countries. *Alternaria solani* is responsible for causing the sickness. Typically, the condition may be identified when tiny, uneven, dark

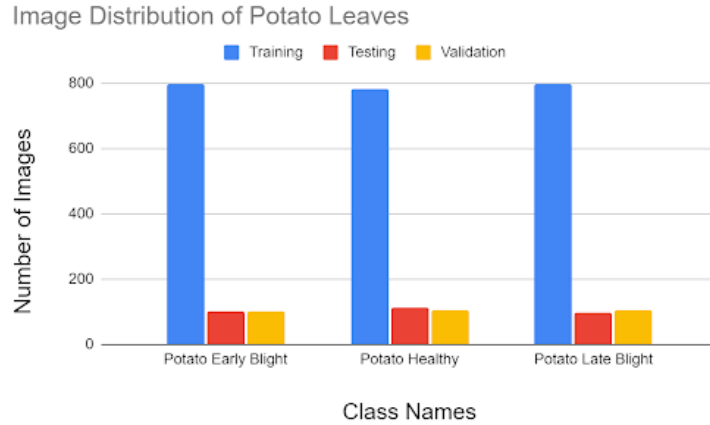


Figure 4.36: Distribution of potato leaves

brown patches appear. However, these spots might expand and cause the entire leaf to yellow. Due to the fact that temperature fluctuations, moisture, and radiation are the primary causes of this illness, it is uncommon in greenhouse-inoculated plants. Few samples of the discussed condition is shown in Fig. 4.37.



Figure 4.37: Potato Early Blight Samples

Potato Healthy

Potato leaves with a smooth texture and a bright green color are in good health. They should not show any signs of yellowing, wilting, or holes, and they should be firmly attached to the stem. Additionally, the leaves have to be devoid of any blemishes or discolorations that can represent the presence of a disease or pests. A healthy potato plant will have uniformly sized and shaped leaves that show no symptoms of curling or clawing. Additionally, the leaves will be slightly bent, which indicates that the plant is receiving the proper quantity of water and sunlight. It is also crucial to remember that a mature potato plant's leaves will begin to yellow and fall off naturally as the plant ages. Fig.4.38 shows some sample of healthy potato leaves.

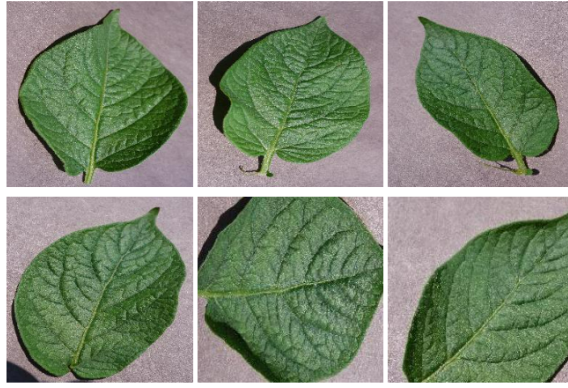


Figure 4.38: Potato Healthy Samples

Potato Late Blight

The main agent that is the cause for potato late blight is known as *Phytophthora infestans*. This disease is one of the biggest threats in potato production. To identify the disease, we should inspect the lower leaves' tips or edges, it will appear as water-soaked spots. This is the place where we find the spots because tips and edges of lower leaves usually tend to collect dews. So, from this we can conclude that on wet condition or weather this disease progresses rapidly and. However when the weather is warm or dry it does not stop but the progress slows marginally.[5] Few samples of the discussed condition is shown in Fig. 4.39.

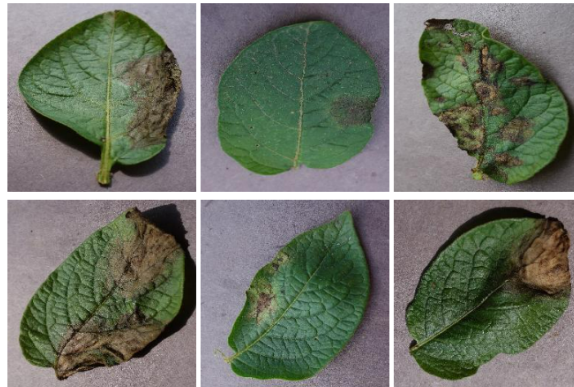


Figure 4.39: Potato Late Blight Samples

4.4.10 Raspberry

In the Fig. 4.40, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Raspberry Healthy

A healthy raspberry leaf should have several key features. Firstly, it should be a vibrant green color, with a smooth and glossy surface. The leaf should be firmly attached to the stem and be of an appropriate size and shape for the raspberry plant. The leaf should be free from any signs of disease or pest damage, such as yellowing,

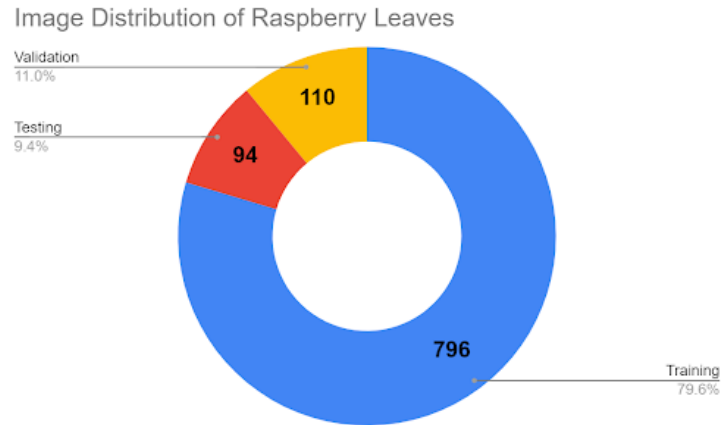


Figure 4.40: Distribution of raspberry leaves

wilting, spotting, or holes. Secondly, the leaf should have well-defined veins that are the same color as the leaf blade. Thirdly, the leaf should not have any mechanical damage or other signs of injury, such as tears or frayed edges. Lastly, the leaf should be free from any signs of fungal or bacterial infections, such as powdery mildew or leaf spot. Raspberry leaves are important for photosynthesis, which happens when light energy in plants is converted to chemical energy, which is crucial for the growth and development of the raspberry bush. A healthy leaf will help ensure the proper growth of the bush, leading to a bountiful harvest of raspberries. Few sample of healthy raspberry is shown in Fig. 4.41.

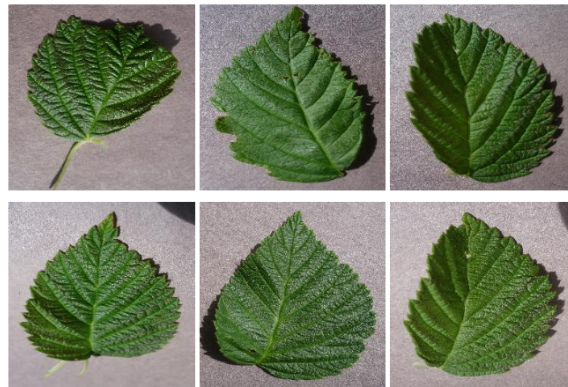


Figure 4.41: Raspberry Healthy Samples

4.4.11 Soybean

In the Fig. 4.42, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Soybean Healthy

A vibrant green soybean leaf with a glossy, smooth texture is indicative of health. The leaf should have a symmetrical shape, be firmly attached to the stem, and show no evident symptoms of injury or disease. Typically, soybean leaves are large

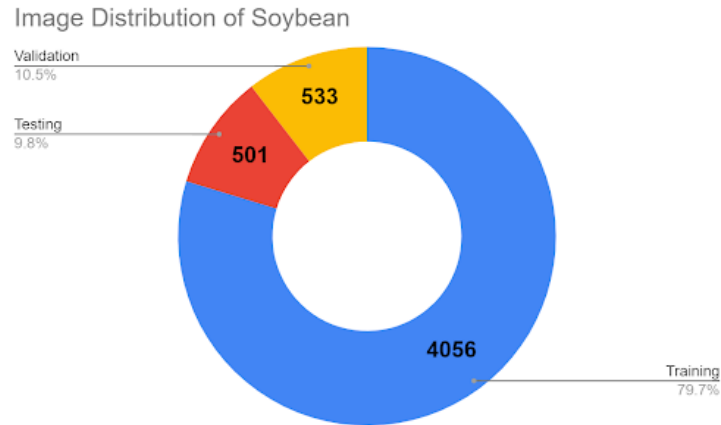


Figure 4.42: Distribution of soybean leaves

and trifoliate, which means they contain three leaflets. The leaves of a healthy soybean plant should be uniform in size and shape, spaced equally along the stem. For soybean leaves to stay healthy, they need a balanced diet, enough water, and sunlight. To avoid harming the leaves and the overall health of the plant, pests and illnesses should be watched for and treated as needed. Some sample of healthy soybean leaves is shown in Fig. 4.43.

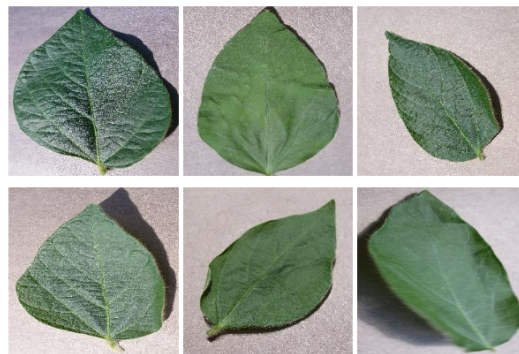


Figure 4.43: Soybean Healthy Samples

4.4.12 Squash

In the Fig. 4.44, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Squash Powdery Mildew

One of the most prevalent illnesses affecting squash is squash powdery mildew. It often results from a number of distinct fungal species. This fungus exclusively grows on the exterior of the leaves, not in the internal tissue, which is an intriguing detail about it. We may thus conclude that it can only grow on the leaf's surface. Because it is one of the most prevalent diseases affecting plant leaves, it is quite simple to recognize. The leaf appears to be covered with talcum powder. Generally, the hue ranges from white to gray. Because the powder dust may move from one leaf to

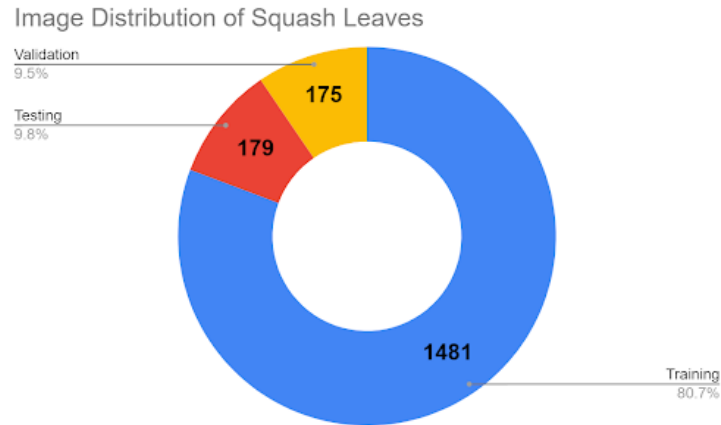


Figure 4.44: Distribution of squash leaves

another during strong winds, we can claim that this illness is particularly infectious. If left untreated, the powdery mildew will eventually change into small, tiny, round rings that become brown before dying and turning black. When it's warm and dry outside, this illness is particularly prevalent. This illness can occasionally be mistaken for organic leaf markings.[65] Few samples of the discussed condition is shown in Fig. 4.45.



Figure 4.45: Squash Powdery Mildew Samples

4.4.13 Strawberry

Fig. 4.46 illustrates the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.

Strawberry Healthy

As shown in Fig. 4.47, healthy strawberry leaves are typically a dark green color and have a glossy appearance. They should be firmly attached to the plant and have no signs of yellowing, wilting, or holes. The leaves should also be free of any spots or discoloration, which can indicate the presence of disease or pests. A well-grown strawberry plant will have leaves that are uniform in size and shape, with no signs of curling or clawing. Strawberry leaves can also have some slight curving, which is a sign that the plant is getting enough sunlight and the right amount of water.

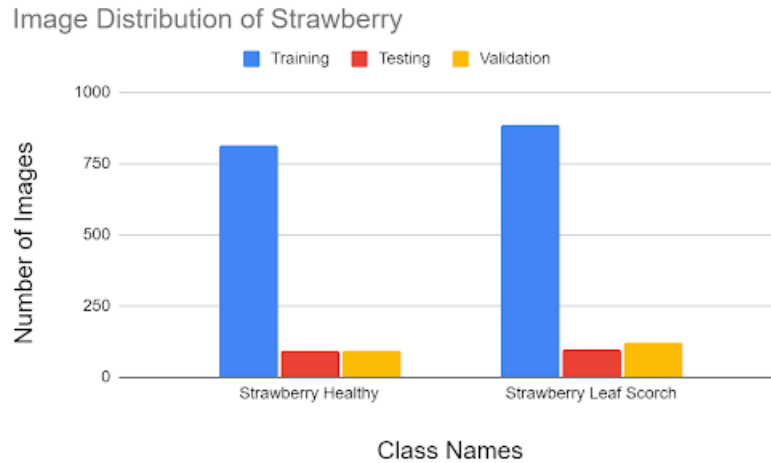


Figure 4.46: Distribution of strawberry leaves

However, in some varieties of strawberry leaves can be slightly hairy and might not be as glossy. It is also important to note that strawberry plant have leaves that are slightly different from the ground leaves, they are called Rosette leaves. They are usually smaller and more circular than the mature leaves, but they should also be free of any signs of disease or damage.

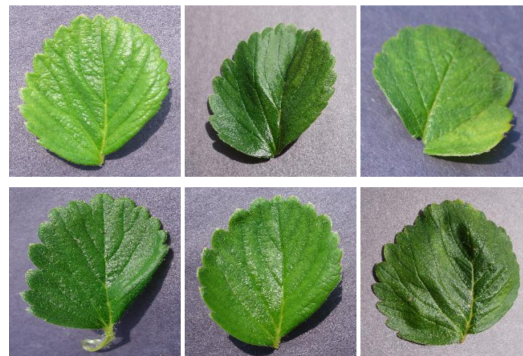


Figure 4.47: Strawberry Healthy Samples

Strawberry Leaf Scorch

It is another fungal infection which affects the foliage of strawberry plantings. The agent that is responsible for this disease is known as *Diplocarpon earliana*. The first sign of this disease is basically emergence of tiny, purple lesions on the foliage's underside [93]. Few samples of the discussed condition is shown in Fig. 4.48.

4.4.14 Tomato

In the Fig. 4.49, we can see the image distribution of healthy and each diseased leaf on Training, Validation and Testing set.



Figure 4.48: Strawberry Leaf Scorch Samples

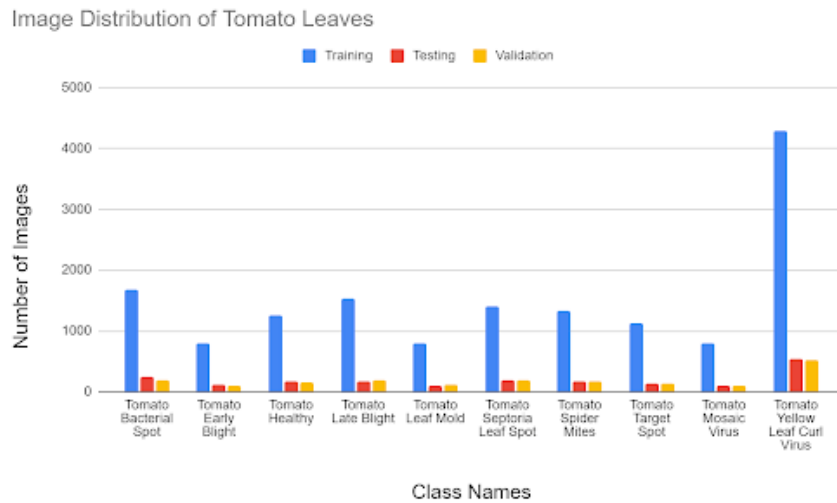


Figure 4.49: Distribution of tomato leaves

Tomato Bacterial Spot

This is one of the most prevalent tomato leaf diseases. This illness first seems quite similar to other tomato diseases. The illness can be identified by brown, round patches encircled by a yellow halo. In addition, there are gaps in the heart of the leaves, and the spots lack concentric rings. Occasionally, similar evidence can also be observed on the stem.[78] Few samples of the discussed condition is shown in Fig. 4.50.

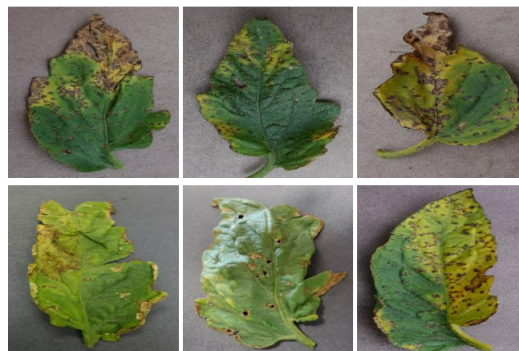


Figure 4.50: Tomato Bacterial Spot Samples

Tomato Early Blight

It is another frequent symptom similar to tomato bacterial spot. *Alternaria solani* is the principal causal agent of this illness. It is very infectious since it affects all sections of tomato plants, including leaf, stem, and fruit. This does not result in death, although productivity will be diminished. Typically, older plants are damaged by this disease. This disease may be caused by the soil or the seeds. However, it is a relatively frequent wintertime sickness. This condition might be mistaken for *Septoria* leaf spot. This disease causes leaf patches that eventually turn yellow and die. On elder plants, black patches with concentric rings may form, and the surrounding foliage may also turn yellow [88]. Few samples of the discussed condition is shown in Fig. 4.51.



Figure 4.51: Tomato Early Blight Samples

Tomato Healthy

As shown in Fig. 4.52, healthy tomato leaves must have a vibrant green color and have a smooth texture. They should be firmly attached to the stem and have no signs of yellowing, wilting, or holes. The leaves should also be free of any spots or discoloration, which can indicate the presence of disease or pests. Tomato leaves are also a good source of nutrients for the plant and play a vital role in photosynthesis, how the plant transforms solar energy into chemical energy. A well grown tomato plant should also have uniform leaves size, no curling or clawing, and some slight curving on the leaves. This is a good indication that the plant is getting enough sunlight and the right amount of water.

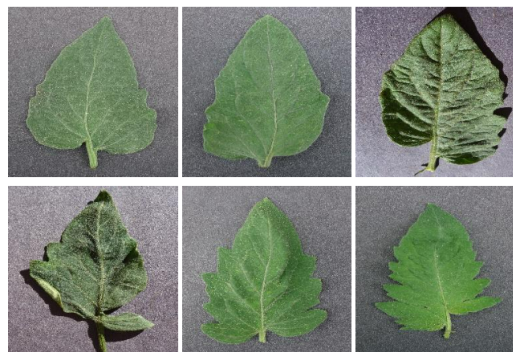


Figure 4.52: Tomato Healthy Samples

Tomato Late Blight

On areas saturated with water, this fungus initially appears small but quickly enlarges into purple-brown, oily-looking blotches. On the underside of the leaves, white mycelium and spore-forming structures may surround the spots. This can result in death, and it can spread to young leaves [89]. Few samples of the discussed condition is shown in Fig. 4.53.



Figure 4.53: Tomato Late Blight Samples

Tomato Leaf Mold

This illness is mostly caused by the pathogen *Passalora fulva*. This occurs frequently in humid environments, notably in plastic greenhouses. Some signs include pale green to yellowish spots on the leaf's top surface, which gradually turn brilliant yellow. This dispersed patches then converge. This can result in the plant's demise [95]. Few samples of the discussed condition is shown in Fig. 4.54.



Figure 4.54: Tomato Leaf Mold Samples

Tomato Septoria Leaf Spot

This is another sickness caused by the fungus *Septoria lycopersici*. It is regarded as one of the most damaging tomato diseases. The illness becomes more severe in moist, humid environments. Lower leaves should be studied to diagnose the illness. Upon investigation, we may observe round spots with dark brown edges, tan to gray centers, and little black fruiting structures. This dots will nearly cover the entire leaf [92]. Few samples of the discussed condition is shown in Fig. 4.55.

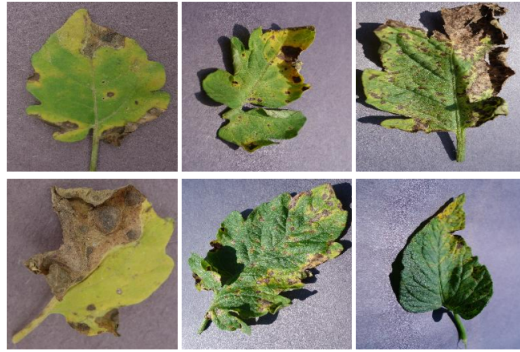


Figure 4.55: Tomato Septoria Leaf Spot Samples

Tomato Spider Mites

This is also sometimes known as Two-spotted spider mite. In the summer, this condition is prevalent. This is mostly due to heat, drought, water stress, and an abundance of weeds. To identify this sickness, we must search for a pair of black dots that are visible through the orange body often present under the skin; this is one of the reasons why freshly formed individuals may lack the spots [6]. Few samples of the discussed condition is shown in Fig. 4.56.



Figure 4.56: Tomato Spider Mites Samples

Tomato Target Spot

Compared to other tomato diseases, identifying the illness at an early stage is a bit more challenging. This disease begins on the elder leaves, then begins to spread upwards. The initial sign of this condition may be irregularly shaped patches with yellow margins that may grow to a diameter of 10 mm. This illness will eventually cause the leaf to turn yellow and die. These dots are also visible on the leaves' dewdrops. Additionally, this may cause tiny, light brown patches on the tomato itself. This is especially frequent in wind- and rain-affected regions. Very rapidly, plants will lose their leaves as the disease spreads [90]. Few samples of the discussed condition is shown in Fig. 4.57.

Tomato Mosaic Virus

This disease can be found globally and on other plant species as well. This illness is immediately distinguishable due to its rod-shaped nature. This comprises four



Figure 4.57: Tomato Target Spot Samples

distinct proteins that play a significant role in the replication and spread of the illness [100]. At a very early stage, they may seem yellow and generally stunted. The leaves may occasionally twist, deform, or diminish in size. This disease may accelerate fruit ripening, resulting in a decrease in tomato yield. This illness is prevalent in high-temperature regions [96]. Few samples of the discussed condition is shown in Fig. 4.58.



Figure 4.58: Tomato Mosaic Virus Samples

Tomato Yellow Leaf Curl Virus

This condition is commonly referred to as TYLC, and it is extremely rare. It is also one of the most destructive tomato pathogens. This illness occasionally affects the plant's growth. The leaves of infected plants develop little upward curls, as well as prominent crumpling, interveinal yellowing, and marginal yellowing. Typically, the leaves have a bushy appearance, similar to 'bonsai' or 'broccoli' [97]. Few samples of the discussed condition is shown in Fig. 4.59.



Figure 4.59: Tomato Yellow Leaf Curl Virus Samples

Chapter 5

Methodology

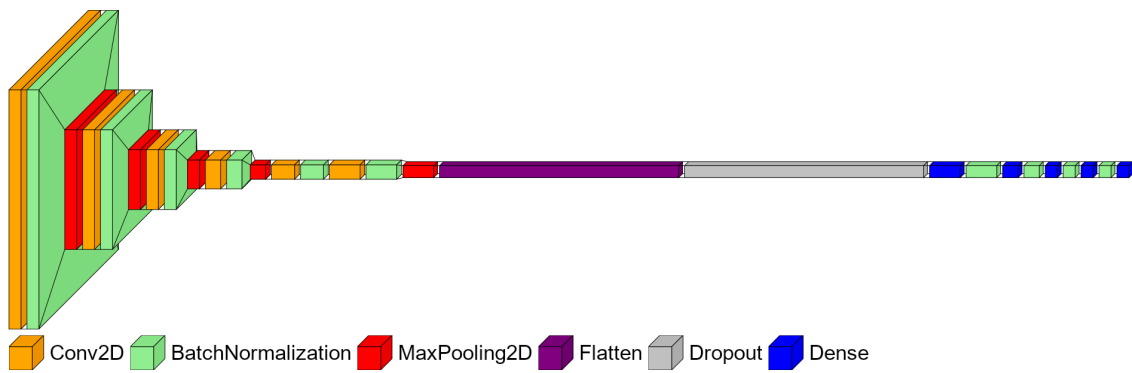


Figure 5.1: Illustration of our Proposed CNN Architecture



Figure 5.2: Picture of different leaves in Different states

In this part of the article, we will provide our idea for the design of a CNN that is capable of predicting plant leaf disease from a given image. To begin with, we create a variation of the conventional CNN architecture by modifying a number of important parameters and functions.

5.1 Convolutional Neural Network(CNN)

Convolution Neural Networks (CNN) are artificial neural networks created solely for the purpose of doing vector input analysis. CNNs are utilized in image processing as well as image recognition.

Neural networks are a special kind of computer architecture that emulates the way the brain's neurons communicate with one another. As image processing requires a more specialized neural network than the conventional ones, the input images have to be of a low resolution in order for them to function well. A CNN's "neurons" are more comparable to those found in the frontal lobe of the brain, which is the region of the brain in different animals as well as human beings that is accountable for visual input processing. Given that the neural layers are organized to cover the whole visual field, the problem of incomplete image processing, which is experienced by normal neural networks, has been eliminated.

A CNN makes use of a method that is analogous to a multilayer perceptron and is geared toward obtaining a better processing rate. It is composed of several pooling, convolutional, fully connected (FC/Dense), dropout, and normalizing layers, in addition to an input, output, and a hidden layer.

5.2 Proposed Methodology

We are aiming to reduce the size of our model as much as possible while preserving its accuracy by employing the minimum number of parameters possible. In addition, maintaining reasonable values for the parameters is another way to help slow down the computation. First, by utilizing the dataset, our CNN model creates an image of a leaf that has a resolution of 256×256 pixels and three separate channels. Right now, let's decrease the resolution so that it's only 100×100 pixels. In our model, there are a total of six Conv2D layers utilized and the kernel size of each was maintained at 3 by 3. Now, in order to stop the model from becoming overfit, we are going to apply a dropout layer that has a dropout rate of 35%. After applying the Flatten layer, the Dropout layer was the next one we used. On the other hand, we use a pooling layer of size 2×2 to reduce the computational power of our model. This layer is known as MaxPooling 2D layers. This layer contributes to a reduction in the amount of computing that is required for the subsequent levels.

We used RELU activations in all the layers except the last output layer because it is better compared to Tanh or Sigmoid functions. The activation function RELU helps us to speed up the stochastic gradient descent (SGD), which is missing in other activation functions. Then, we use a Flatten Layer to create a one-dimensional array, followed by our Dropout Layer as described above. The CNN model's fully connected layer is then used, starting with 512, 256, 128 and 64 nodes. Previously, as we said we use the RELU function on every layer except the last output layer. Here we use SoftMax activation for the last output layer. At the end because Softmax is used, in this last layer, we want to classify our nodes according to various classes, in this instance 38 different classes. A generalized binary variation of logistic regression is the softmax classifier. The enhanced hyper-parameters were used in the model's

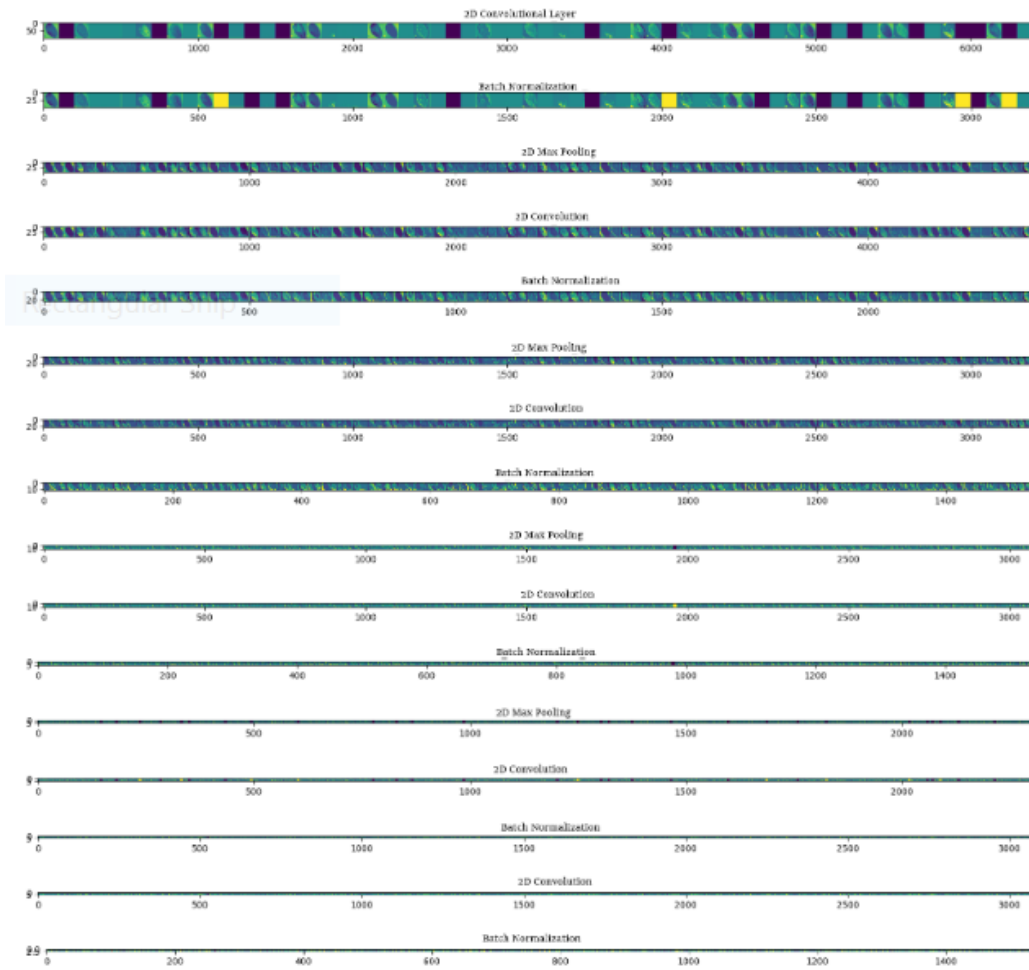


Figure 5.3: Feature Map

construction.

We utilized the Adam optimizer in order to achieve our goal of maximizing production efficiency while maintaining an initial learning rate of 0.002. In addition to this, we made use of the ReduceLROnPlateau technique, which modifies our learning rate whenever there is a local loss minima present. Every time a more efficient loss was found, we made sure to save our model as well. We continued to keep the batch size at 32, and we ran our model for a total of 90 epochs.

Feature Map is shown at figure 5.3.

In addition to using LIME as our Explainable AI model, we additionally utilized SHAP. Just after the prediction made by our CNN model has been finished, the image will be forwarded to LIME, where it will explain why our proposed model assigned this image to a given class.

Due to the fact that LIME is model-independent, it can be combined with any machine learning or deep learning model. LIME helps us to understand how the model works and makes predictions on the basis of what features from the image have been extracted. It actually changes the input image slightly and observes how that affects the previous prediction. It is possible for it to faithfully explain the predictions of any classifier or regressor when it has been localized with an interpretable model. In its most basic form, LIME investigates the connection that exists between input and output, as portrayed by the model, by postulating the existence of a machine learning black box model between the two. We used 10000 samples which means the image will be altered 10000 times, and each prediction will be observed. We have discussed how LIME helps us to understand the reason for categorizing the leaf to a certain class.

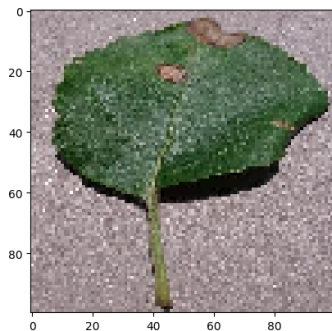


Figure 5.4: Image 1

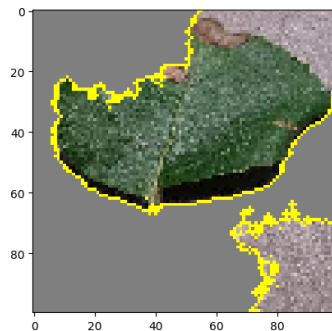


Figure 5.5: Image 2

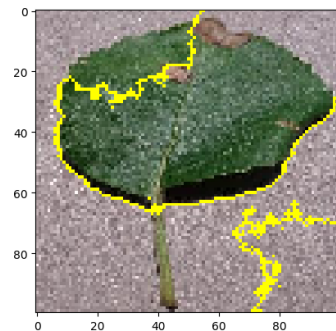


Figure 5.6: Image 3

Here Figure 5.4 is the picture that our model believes best conveys the appearance of Apple Black Rot. Now, we send this image together with the model to LIME, and it analyzes both of them. Then, it informs us what part of the image caused our CNN to arrive at the conclusion that this particular image portrays Apple Black Rot.

When Figure 5.4 and Figure 5.5 are examined side by side, it is obvious to observe that the vast majority of the pixelated gray zone has been filtered out of the image.

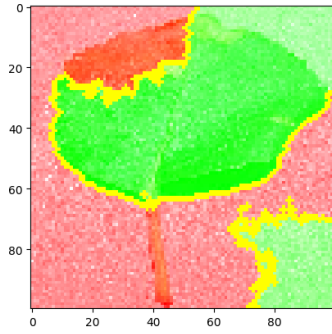


Figure 5.7: Image 4

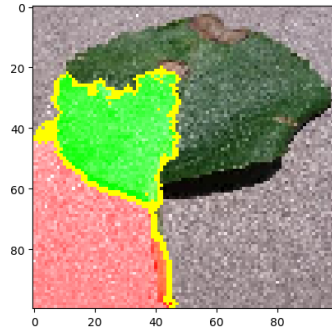


Figure 5.8: Image 5

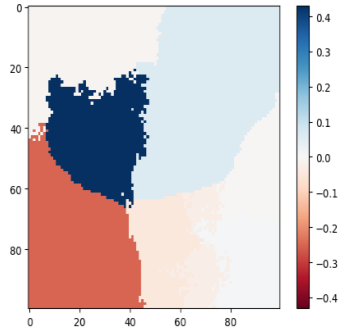


Figure 5.9: Image 6

This is mostly as a result of the fact that our CNN model has arrived at the conclusion that the excluded area is not a part of the prediction that we are utilizing to detect diseases, and this has been confirmed by LIME.

Figure 5.6 is a subset of Figure 5.4 and Figure 5.5 that highlights the regions covered by the CNN prediction model while disregarding all other external factors when decisions are made based on the original image.

Figure 5.7 is the result of running LIME, which illustrates different aspects of the CNN model's prediction by making use of a variety of color codes. In this instance, areas that are colored red indicate that they were not helpful when it came to formulating predictions. On the other hand, the green parts suggest that these areas have been used for the purpose of prediction and decision-making.

Figure 5.8 is a composite of Images 1 and 4, highlighting the segment of the image from which the vast majority of the information is obtained. Figure 5.8 is shown above. The areas that are colored light green denote those from which the most quantity of data was acquired for the prediction, whilst the remaining section red color denotes those from which very little or no data was collected.

Figure 5.9 shows a heatmap highlighting the leaf regions in a chart-like format. Here, from the majority of the information part in Figure 5.8, we classify the data as either valid or invalid. The blue part shows legitimate data that was considered for the given prediction, whereas the red section indicates data ignored by our proposed CNN model.

5.3 Vision Transformer

Vision Transformers or ViT, according to the authors of [53], requires a significant quantity of data. Therefore, they recommended that the model be trained on a big dataset and then fine-tuned on a medium-sized dataset. If this is adhered to, then the model can outperform the most advanced Convolutional Neural Network models.

However, our dataset is far smaller than the one required for Vision Transformers. In lieu of this, we will train our ViT model on a minimal data set. According to the authors of [64], two strategies, Shifted Patch Tokenization and Locality Self Atten-

Table 5.1: Layers with Output Shape and Parameters for proposed model

Layers	Output Shape	Parameters
2D Convolutional Layer	(None, 100, 100, 64)	1792
Batch Normalization	(None, 100, 100, 64)	256
2D Max Pooling	(None, 50, 50, 64)	0
2D Convolutional Layer	(None, 50, 50, 96)	55392
Batch Normalization	(None, 50, 50, 96)	384
2D Max Pooling	(None, 25, 25, 96)	0
2D Convolutional Layer	(None, 25, 25, 128)	110720
Batch Normalization	(None, 25, 25, 128)	512
2D Max Pooling	(None, 12, 12, 128)	0
2D Convolutional Layer	(None, 12, 12, 256)	295168
Batch Normalization	(None, 12, 12, 256)	1024
2D Max Pooling	(None, 6, 6, 256)	0
2D Convolutional Layer	(None, 6, 6, 384)	885120
Batch Normalization	(None, 6, 6, 384)	1536
2D Convolutional Layer	(None, 6, 6, 512)	1769984
Batch Normalization	(None, 6, 6, 512)	2048
2D Max Pooling	(None, 3, 3, 512)	0
Flatten	(None, 4608)	0
Dropout	(None, 4608)	0
Dense	(None, 512)	2359808
Batch Normalization	(None, 512)	2048
Dense	(None, 256)	131328
Batch Normalization	(None, 256)	1024
Dense	(None, 128)	32896
Batch Normalization	(None, 128)	512
Dense	(None, 64)	8256
Batch Normalization	(None, 64)	256
Dense	(None, 38)	2470
Total parameters: 5,662,534		
Trainable parameters: 5,657,734		
Non-trainable parameters: 4,800		

tion, can be used to accomplish this.

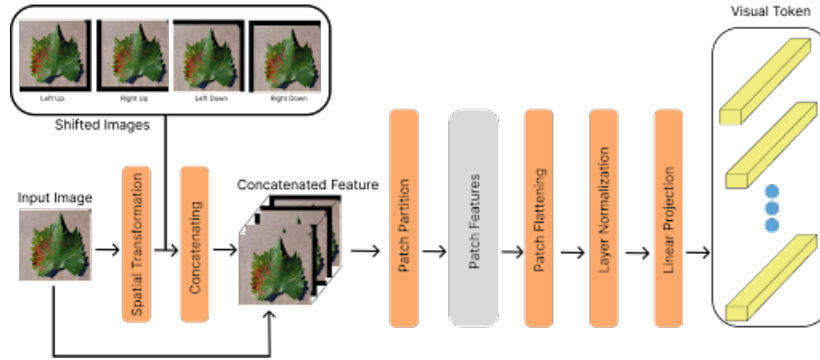


Figure 5.10: Shifted Patch Tokenization

In Moved Patch Tokenization, a picture is first captured and then shifted diagonally, as seen in Fig. 5.12. Then, we concatenate the original picture with images with diagonal shifts. Then, we extract the concatenated image patches. The patches are then flattened into three-dimensional space. The image is then subjected to Layer Normalization before being projected. Figure 5.10 provides a graphic illustration of this Patch Tokenization method.

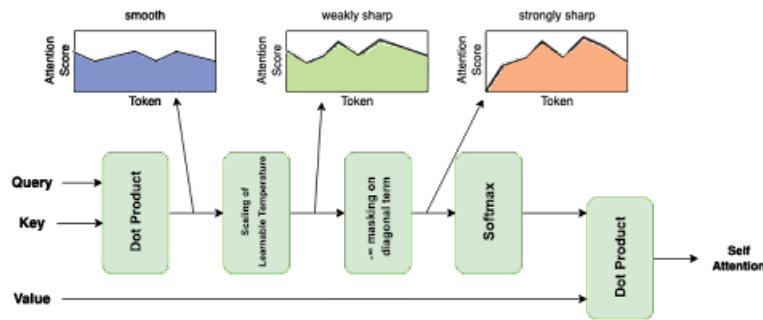


Figure 5.11: Locality Self Attention

Locality Self Attention is another strategy that accepts a query, key, and value from the same input. The last step is to use dot product to examine the similarity between our query and the key. This dot product will produce big self-token relations rather than inter-token interactions. Before using the softmax function, we scale our dot product of query and key by the square root of the key's dimension to minimize the possibility of an excessively tiny gradient. After scaling the dot product, the softmax function is used. In addition, this softmax may provide a greater likelihood of self-token links than inter-token interactions. To mitigate this issue, the authors of [98] propose a masking approach in which the diagonal of our dot product is concealed. The value is modified using the attention weights as the final step. This operating concept is illustrated in Figure 5.11.

The Formula for the whole Locality self technique is,

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (5.1)$$

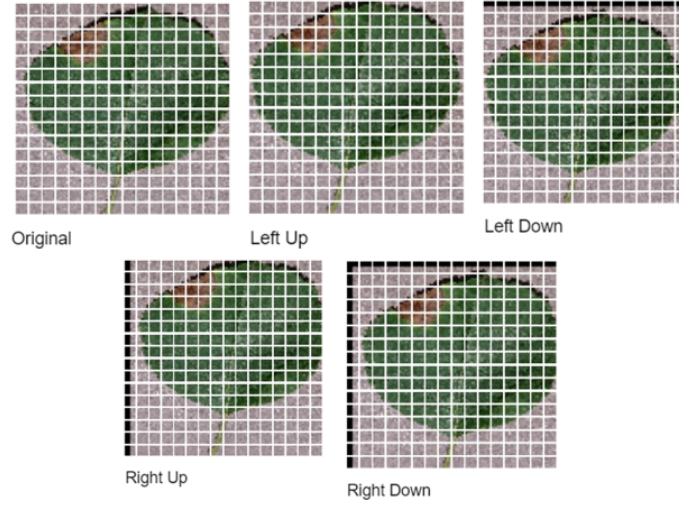


Figure 5.12: Patch Tokenization

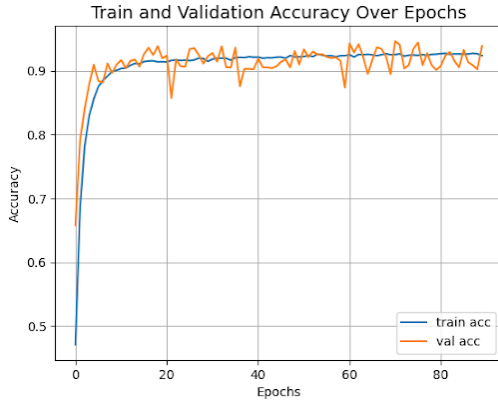


Figure 5.13: ViT Accuracy Curve

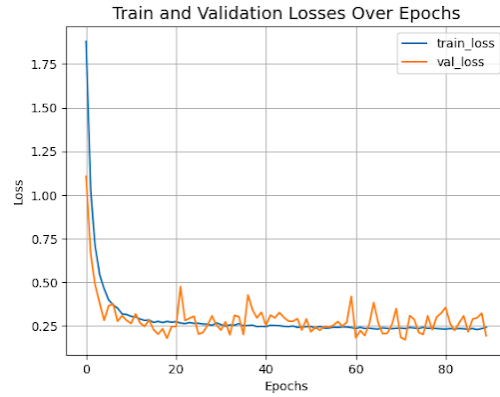


Figure 5.14: ViT Loss Curve

5.4 Compact Convolutional Transformers

CCT, also called Compact Convolutional Transformer, is a strategy for training transformers with little data suggested in a study [63].

This approach is identical to Tokenization in ViT. Here, stochastic depth is employed for regularization; it is similar to a Dropout layer, but instead of turning off a single node, the entire block of nodes in a layer is disabled. Utilized mostly before the residual blocks of a Transformer Encoder. First the input passes Convolutional Tokenization before it reaches the transformer. There are convolutional layer, pooling layer and reshape. After passing through the convolutional tokenization it enters the Transformer with Sequence Pooling. We have applied the same data augmentation strategies as previous models. We have integrated attention pooling, also known as

sequence pooling, in our CCT model. In ViT, just the feature maps matching to the tokens are used for categorization. In CCT, however, the Transformer Encoder output is weighted before transmission to the classification layer. Fig 5.15 shows the working principle of Compact Convolutional Transformers.

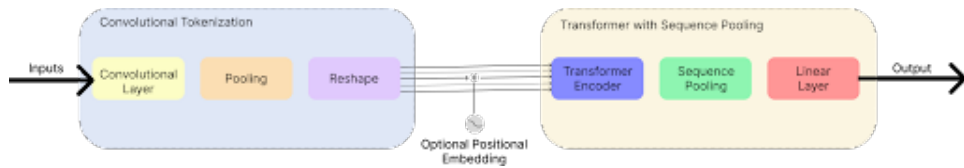


Figure 5.15: CCT Architecture

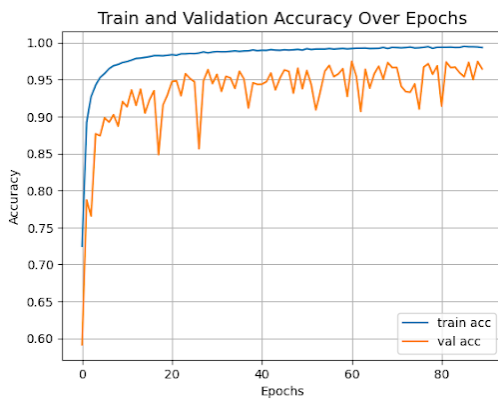


Figure 5.16: CCT Accrucacy Curve

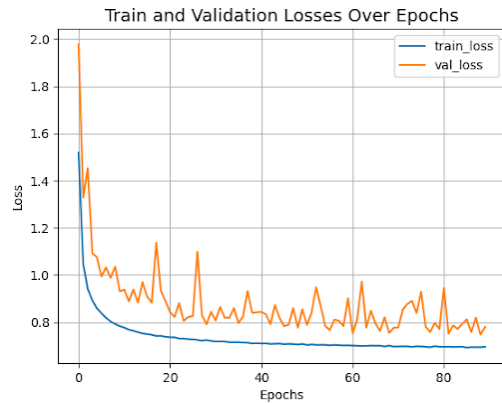


Figure 5.17: CCT Loss Curve

5.5 ConvMixer

Similarly to previous models, we have employed the same data augmentation approaches here. This is quite similar to MLP-mixer, but utilizes conventional convolution layers instead of completely coupled layers. In addition, BatchNormalization is used instead of LayerNormalization. If spatial placement of pictures is desired, Depthwise Convolution layers are utilized. In contrast, pointwise convolution is utilized for channel-wise information mixing across patches. Figure 5.18 displays the architecture of ConvMixer.

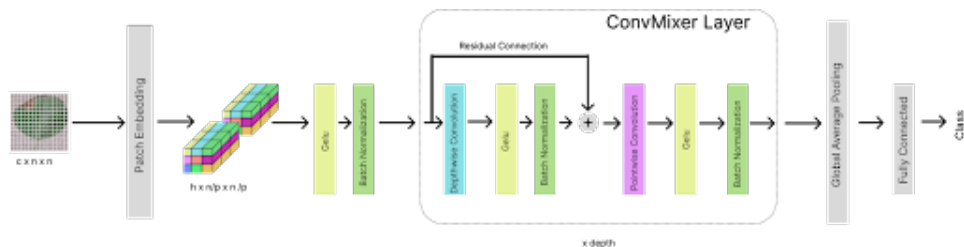


Figure 5.18: Architecture of ConvMixer

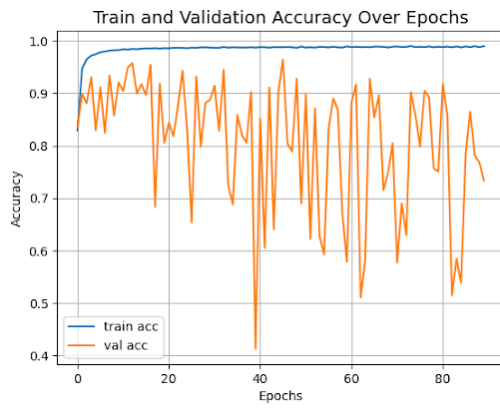


Figure 5.19: ConvMixer Accuracy

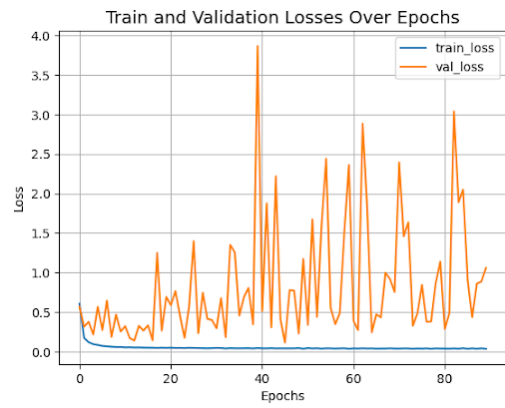


Figure 5.20: ConvMixer Loss

Chapter 6

Pre-trained CNN Models

6.1 InceptionResNet V2

InceptionResNet V2 is a CNN architecture like the other pre-trained models discussed above. It is just an advanced form of the Inception family that we have seen before [22]. However, there is an exception compared to the traditional Inception family which is, it contains residual connections instead of filter connections as traditional inception family. It contains cheaper Inception blocks. We know that reduction is induced by the inception blocks now to compensate the inception blocks are followed by a filter-expansion layer which is just a 1 x 1 convolution layer without any activation function. This helps to increase the dimensionality of the filter. Another small difference between traditional Inception and InceptionResNet is that the use of BatchNormalization can be seen on traditional layers but not on summation layers. Here the residual connections are mainly used as a shortcut in the model and which gives an opportunity to show better results compared to traditional Inception models. Figure 6.1 illustrates the architecture of InceptionResNet V2.

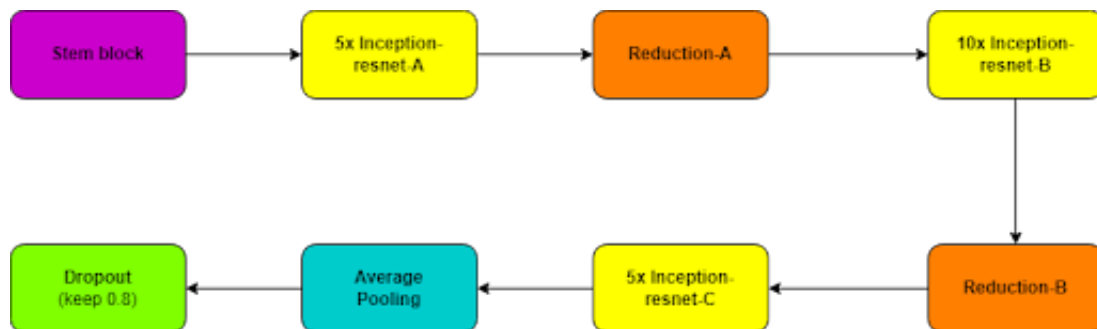


Figure 6.1: Architecture of InceptionResNet V2

6.2 ResNet 50

This model is referred to in its full form as Residual Networks. This paradigm can be implemented in a wide variety of various ways. ResNet 50 establishes that it can function with as many as 50 layers of neural networks. [17] This ResNet 50 model is

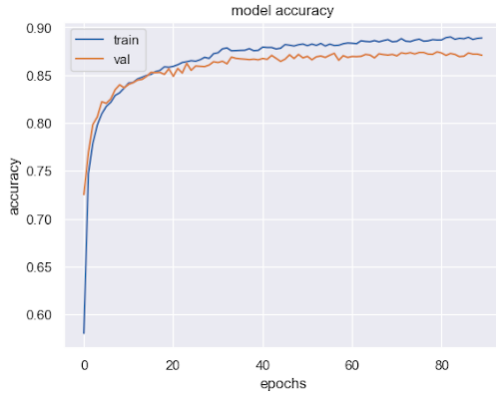


Figure 6.2: Inception V2 Accuracy

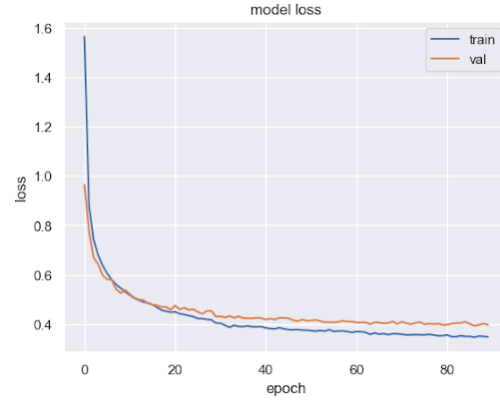


Figure 6.3: Inception V2 loss

Table 6.1: InceptionResNet V2 layers with output shape and parameters

Layer (type)	Output Shape	Parameters
input_1 (InputLayer)	(None, 150, 150, 3)	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0
block2_conv1 (Conv2D)	(None, 75, 75, 128)	73856
block2_conv2 (Conv2D)	(None, 75, 75, 128)	147584
block2_pool (MaxPooling2D)	(None, 37, 37, 128)	0
block3_conv1 (Conv2D)	(None, 37, 37, 256)	295168
block3_conv2 (Conv2D)	(None, 37, 37, 256)	590080
block3_conv3 (Conv2D)	(None, 37, 37, 256)	590080
block3_pool (MaxPooling2D)	(None, 18, 18, 256)	0
block4_conv1 (Conv2D)	(None, 18, 18, 512)	1180160
block4_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block4_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block4_pool (MaxPooling2D)	(None, 9, 9, 512)	0
block5_conv1 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv2 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv3 (Conv2D)	(None, 9, 9, 512)	2359808
block5_pool (MaxPooling2D)	(None, 4, 4, 512)	0
Total params : 14,714,688		
Trainable params : 0		
Non-trainable params : 14,714,688		

capable of providing a solution for the vanishing gradient problem.

Our initial convolutional layer is made up of 64 unique kernels, each measuring 7 x 7 and having a stride of 2. Then, with a stride value of 2, we observe a MaxPooling layer that is 3 x 3. After that, we have three 1 x 1 and three 3 x 3 instances of the 64 kernel, as well as three 1 x 1 instances of the 256 kernel. We are bringing the total number of layers to 9 at this stage. Following that, we will get the opportunity to view 1 x 1 and 3 x 3 with 128 kernels and 1 x 1 with 512 kernels for a total of 4

times, bringing the total number of layers to 12. Following that, we will see 1×1 and 3×3 with a 256 kernel, as well as 1×1 with a 1024 kernel, which will result in 18 layers. Finally, in the very last convolutional layer, we get to witness 1×1 and 3×3 with 512 kernels, as well as 1×1 with 2048 kernels, bringing the total number of layers to 9. With this final layout, we are able to see a total of 50 layers, which is how the network was given the name ResNet 50. The final layer is the Average pool, which is followed by the fully connected layer. The softmax function is used to calculate the number of neurons in this layer based on the number of classes in our dataset. Figure 6.4 illustrates the architecture of ResNet 50 and Table 6.1 shows details of layers in ResNet 50.

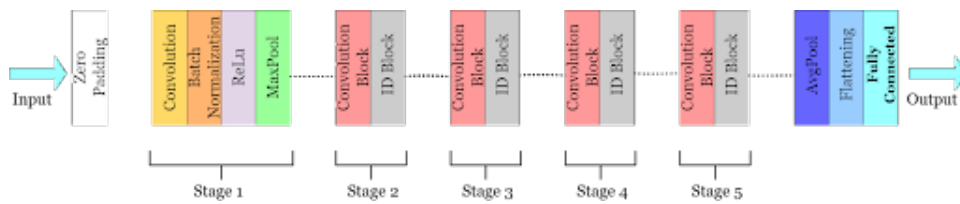


Figure 6.4: Architecture of ResNet 50

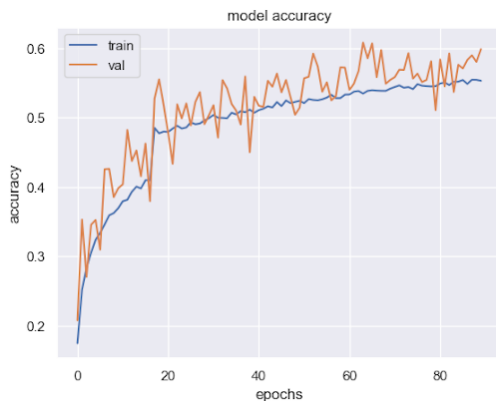


Figure 6.5: ResNet 50 Accuracy

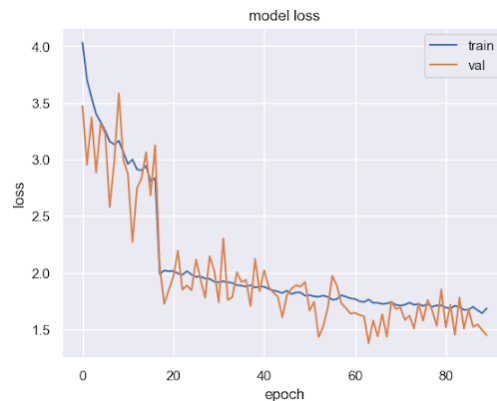


Figure 6.6: ResNet 50 loss

6.3 Inception V3

[23]A major point of emphasis in Inception version 3 is to reduce the number of computing resources needed to run the program by implementing various changes to the earlier Inception architectures. In this part of the process, we did not make many changes to the layers, but to accommodate for the number of classes in our dataset, we did establish an output layer with 38 nodes. Factorized convolutions are useful for keeping things in check on the efficiency of a network and lowering the computing efficiency required to do so because it lowers the total parameter count involved. Convolutions that are larger are being phased out in favor of convolutions that are smaller, which will result in faster training. If we have a fully linked layer and a 3×3 convolution layer before that, then the weights of the 3×3 layer can be shared among themselves, which results in a reduction in the amount of processing power

Table 6.2: Layer Description of ResNet 50

Layers	ResNet 50	Number of Layers
2D Convolutional Layer	7 x 7, 64, stride 2	1
2D Convolutional Layer	3 x 3 max_pool, stride 2 [1 x 1, 64] x 3 [3 x 3, 64] x 3 [1 x 1, 256] x 3	9
2D Convolutional Layer	[1 x 1, 128] x 4 [3 x 3, 128] x 4 [1 x 1, 512] x 4	12
2D Convolutional Layer	[1 x 1, 256] x 6 [3 x 3, 256] x 6 [1 x 1, 1024] x 6	18
2D Convolutional Layer	[1 x 1, 512] x 3 [3 x 3, 512] x 3 [1 x 1, 2048] x 3	9
	Average Pool, 38	1

required. Instead of using a 3x3 convolutional layer, the asymmetric convolutions method uses a 1x3 convolutional layer, succeeded by a 3x1 convolutional layer. This is done in an effort to reduce the total parameter count required for the algorithm. In addition, during the training process, minor CNN layers are created in between the layers, and the loss from these layers is added to the loss from the main network. In conclusion, the help of the pooling layers is taken in order to reduce grid size. Figure 6.7 shows the architecture of Inception V3.

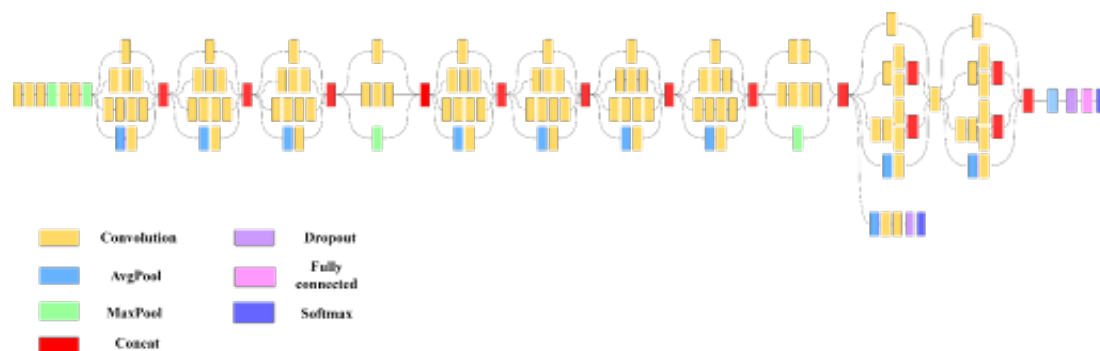


Figure 6.7: Architecture of Inception V3

6.4 VGG 16

[9]Simonyan et al. (2014) suggested the VGG16 CNN model. VGG16 obtained a testing accuracy of 92.7% on ImageNet, a dataset that is comprised of around 14 million images which are then further categorized into a thousand classes. Thirteen convolutional layers are comprised in this model; the model starts with an input layer followed by three convolutional layers and 2D max-pooling that were trained over several weeks on NVIDIA Titan Black GPUs. There are 16 layers and their

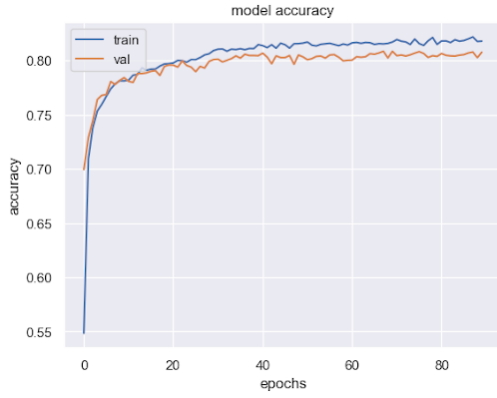


Figure 6.8: Inception V3 Accuracy

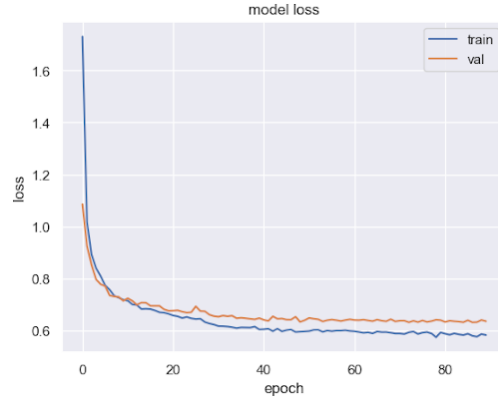


Figure 6.9: Inception V3 loss

Table 6.3: Layer Description of Inception V3

Type	Kernel Size/stride	Input size
Convolution	3 x 3/2	299 x 299 x 3
Convolution	3 x 3/1	149 x 149 x 32
Convolution	3 x 3/1	147 x 147 x 32
Pooling	3 x 3/2	147 x 147 x 64
Convolution	3 x 3/1	73 x 73 x 64
Convolution	3 x 3/2	71 x 71 x 80
Convolution	3 x 3/1	35 x 35 x 192
Inception Module	Three Modules	35 x 35 x 288
Inception Module	Five Modules	17 x 17 x 768
Inception Module	Two Modules	8 x 8 x 1,280
Pooling	8x8	8 x 8 x 2,048
Linear	Logits	1 x 1 x 2,048
Softmax	Output	1 x 1 x 1,000

parameters were adjustable, Thirteen convolutional layers and 3 completely linked layers as a result of the AlexNet ReLU tradition. [9]The model was given the designation VGG16. [8]It contains approximately 500 megabytes of storage space and 138 million parameters. Fig 6.2 demonstrates the VGG16 model summary. It is the traditional model that contains 2D convolutional layers and 2D max-pooling layers. It starts with 64 filters, and the number is multiplied by 2 in every subsequent block until it reaches 512. In this case, it does not have any hidden layer but just one output layer with 38 nodes capable of determining the number of classes present in the dataset. Illustration in Figure 6.10 explains architecture of VGG16.

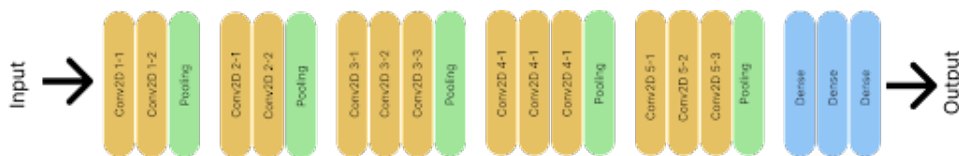


Figure 6.10: Architecture of VGG16

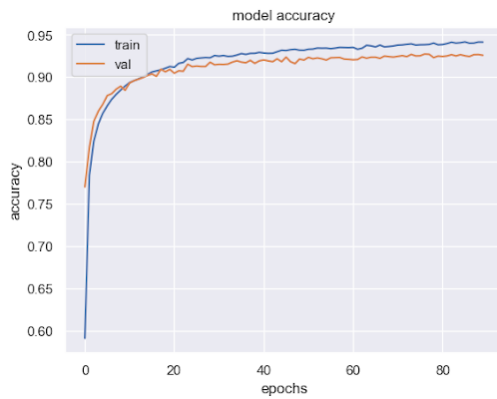


Figure 6.11: VGG 16 Accuracy

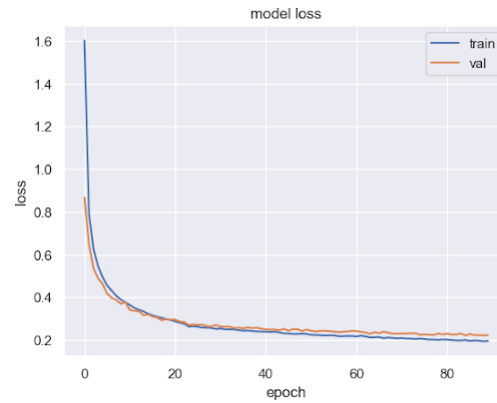


Figure 6.12: VGG 16 loss

Table 6.4: Layers with output shape and parameters of VGG 16

Layers	Output Shape	Parameters
Input Layer	(None, 100, 100, 3)	0
2D Convolutional Layer	(None, 100, 100, 64)	1792
2D Convolutional Layer	(None, 100, 100, 64)	36928
2D Max Pooling	(None, 50, 50, 64)	0
2D Convolutional Layer	(None, 50, 50, 128)	73856
2D Convolutional Layer	(None, 50, 50, 128)	147584
2D Max Pooling	(None, 25, 25, 128)	0
2D Convolutional Layer	(None, 25, 25, 256)	295168
2D Convolutional Layer	(None, 25, 25, 256)	590080
2D Convolutional Layer	(None, 25, 25, 256)	590080
2D Max Pooling	(None, 12, 12, 256)	0
2D Convolutional Layer	(None, 12, 12, 512)	1180160
2D Convolutional Layer	(None, 12, 12, 512)	2359808
2D Convolutional Layer	(None, 12, 12, 512)	2359808
2D Max Pooling	(None, 6, 6, 512)	0
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Max Pooling	(None, 3, 3, 512)	0
Flatten	(None, 4608)	0
Dense	(None, 38)	175142
Total parameters: 14,889,830		
Trainable parameters: 175,142		
Non-trainable parameters: 14,714,688		

6.5 VGG19

VGG-19 is a deeper version of VGG-16. As the name suggests, it has a total of 19 layers among which 16 are convolution layers, 3 are Fully connected layers. Aside from that it has 5 MaxPool layers and a SoftMax layer. The only difference between it and VGG-16 is that it has 3 more convolution layers and 13 more ReLU layers.

The three extra convolution layers are added to input shapes 25, 12 and 6. Figure 6.13 illustrates the architecture of VGG19.

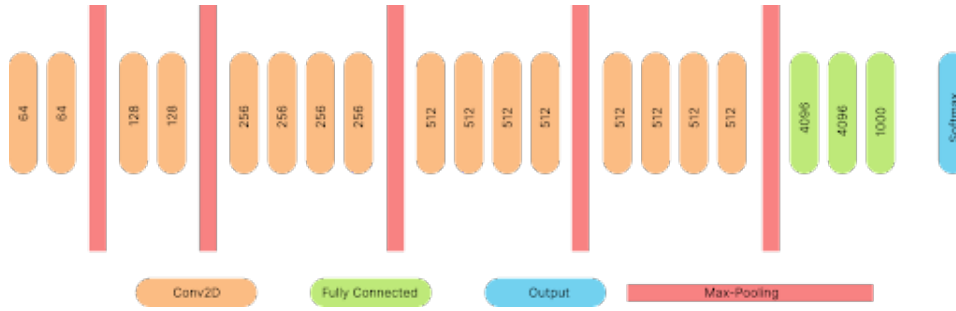


Figure 6.13: Architecture of VGG19

Table 6.5: Layer and Parameter of VGG 19

Layers	Output Shape	Parameters
Input Layer	(None, 100, 100, 3)	0
2D Convolutional Layer	(None, 100, 100, 64)	1792
2D Convolutional Layer	(None, 100, 100, 64)	36928
2D Max Pooling	(None, 50, 50, 64)	0
2D Convolutional Layer	(None, 50, 50, 128)	73856
2D Convolutional Layer	(None, 50, 50, 128)	147584
2D Max Pooling	(None, 25, 25, 128)	0
2D Convolutional Layer	(None, 25, 25, 256)	295168
2D Convolutional Layer	(None, 25, 25, 256)	590080
2D Convolutional Layer	(None, 25, 25, 256)	590080
2D Convolutional Layer	(None, 25, 25, 256)	590080
2D Max Pooling	(None, 12, 12, 256)	0
2D Convolutional Layer	(None, 12, 12, 512)	1180160
2D Convolutional Layer	(None, 12, 12, 512)	2359808
2D Convolutional Layer	(None, 12, 12, 512)	2359808
2D Convolutional Layer	(None, 12, 12, 512)	2359808
2D Max Pooling	(None, 6, 6, 512)	0
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Convolutional Layer	(None, 6, 6, 512)	2359808
2D Max Pooling	(None, 3, 3, 512)	0
Flatten	(None, 4608)	0
Dense	(None, 38)	175142
Total Parameters: 20,199,526		
Trainable Parameters: 175,142		
Non-trainable Paramters: 20,024,384		

6.6 DenseNet 201

It was seen that if the transitions of layers close to input layer and output layer are shortened, it can result in increased depth, accuracy, and efficiency. In standard CNNs, images are processed by a sequence of convolution layers to get feature information. Mapping identity in ResNets promotes gradient propagation since ResNet modules are used in place of convolutions. Whereas every layer in DenseNet is connected to every other layer next to it, resulting in feeding forward additional inputs from all previous modules. Then those modules pass their own inputs to all the next layers. As information is getting stacked up in each layer from the previous layers, the information is getting concatenated [49]. These layers are renamed as Dense Blocks. Illustration in Figure 5.14 showcases the architecture of DenseNet 201.

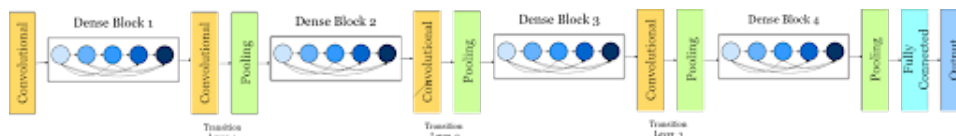


Figure 6.14: Architecture of DenseNet 201

The CNN becomes compact as the quantity of channels is reduced due to this concatenation. This operation loses its viability if feature-maps vary in size, which is an essential part of CNNs as it is required to down-sample layers. To work around this, the network is separated into multiple connected dense blocks. DenseNet uses a sequence of batch normalization, ReLU activations and 3 dimensional convolutions (Conv) as a composite function. The layers between the dense blocks (transition layers) used in DenseNets consist of a batch normalization layer and a 1-dimensional convolution layer and a 2-dimensional MaxPool layer. DenseNets have several advantages as they have a very low number of parameters. They also address the vanishing-gradient problem, increasing efficiency by reusing features [28].

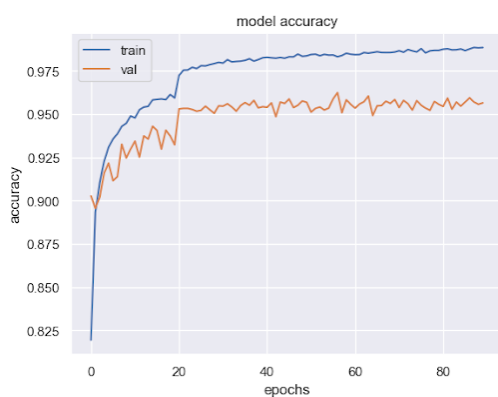


Figure 6.15: DenseNet 201 Accuracy

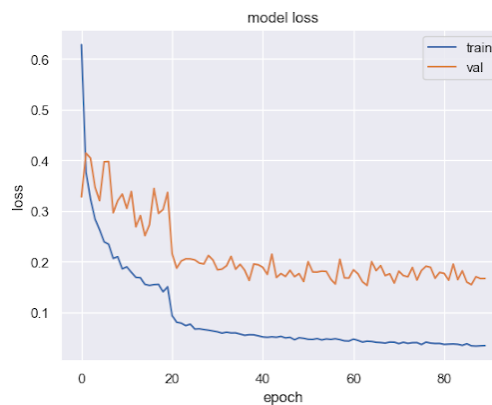


Figure 6.16: DenseNet 201 Loss

6.7 XCEPTION

Xception is a CNN developed by the creator of Keras, François Chollet of Google in 2016. It achieved 94.5% accuracy on imagenet. It is an improved version of

Table 6.6: Layer Description of DenseNet 201

Layers	Output Size	DenseNet-201
Convolution	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Dense Block (1)	56×56	$[1 \times 1$ conv, 3×3 conv] $\times 6$
Transition Layer (1)	56×56	1×1 conv
	28×28	2×2 average pool, stride 2
Dense Block (2)	28×28	$[1 \times 1$ conv, 3×3 conv] $\times 12$
Transition Layer (2)	28×28	1×1 conv
	14×14	2×2 average pool, stride 2
Dense Block (3)	14×14	$[1 \times 1$ conv, 3×3 conv] $\times 48$
Transition Layer (3)	14×14	1×1 conv
	7×7	2×2 average pool, stride 2
Dense Block (4)	7×7	$[1 \times 1$ conv, 3×3 conv] $\times 32$
Classification Layer	1×1	7×7 global average pool
		1000D fully-connected, softmax

Inception modules in InceptionV3 CNNs as it mimics depthwise separable convolution operation instead of the regular one. Their hypothesis is that the mapping of cross-channel and spatial correlations in CNN feature maps can be separated. Inception modules are resource intensive because of regular convolutions, which occurs spatially and depthwise. For every additional filter, it performs convolutions over the input depth just to calculate one output map, which becomes a huge liability in the CNN. Authors behind the inception module hypothesized that the depth can be reduced by doing 1-dimensional convolutions across the depth, which looks across multiple channel's spatio-information, while compressing the dimensions. On the other hand, Xception does the opposite, as it applies the filters first then compresses the input space using 1-dimensional convolutions. Because of this being a better and extreme variant of the theory underlying Inception, the architecture was named Xception. It is 71 layers deep and has thirty six convolutional layers which are structured into fourteen Xception modules. All the modules have skip-over connections around them, excluding the first and last ones. Since the CNN is very deep with 71 layers, it is structured in three parts: 1. entry, middle and exit. The data flows through these sequentially but repeats itself 8 times in the middle. In short, the Xception CNN is a sequence of depthwise separable convolutions with linear connection skips. It has the same parameter count as Inception V3, but performs better since it uses those parameters efficiently[25].

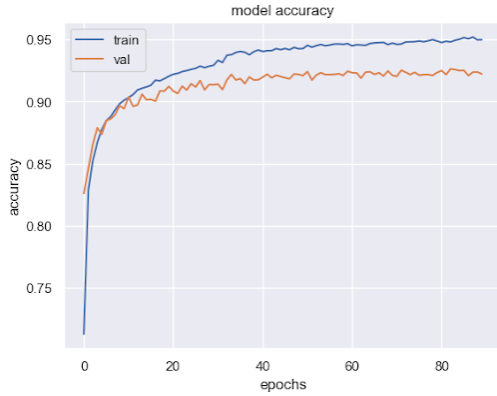


Figure 6.17: XCEPTION Accuracy

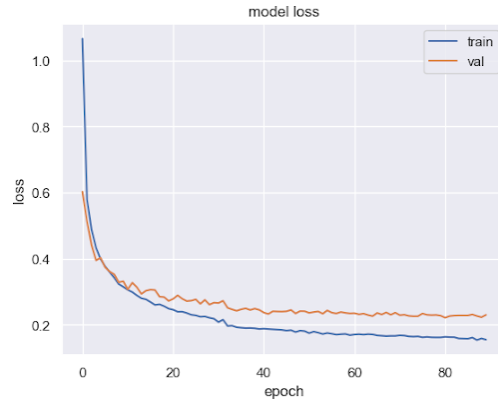


Figure 6.18: XCEPTION Loss

Table 6.7: Layer and Parameter Description of Xception

Layers	Input	Parameters
seperable_conv2d_29 (SeperableC)	(None, 19, 19, 728)	536536
batch_normalization_ 34 (BatchNo)	(None, 19, 19, 728)	2912
re_lu_38 (ReLU)	(None, 19, 19, 728)	0
add_10 (Add)	(None, 19, 19, 728)	0
re_lu_39 (ReLU)	(None, 19, 19, 728)	0
seperable_conv2d_30 (SeperableC)	(None, 19, 19, 728)	536536
batch_normalization_ 35 (BatchNo)	(None, 19, 19, 728)	2912
_lu_40 (ReLU)	(None, 19, 19, 728)	0
seperable_conv2d_31 (SeperableC)	(None, 19, 19, 1024)	752024
conv2d_5 (Conv2D)	(None, 10, 10, 1024)	745472
batch_normalization_ 36 (BatchNo)	(None, 19, 19, 1024)	4096
batch_normalization_ 37 (BatchNo)	(None, 10, 10, 1024)	4096
max_pooling2d_3 (MaxPooling2D)	(None, 10, 10, 1024)	0
add_11 (Add)	(None, 10, 10, 1024)	0
seperable_conv2d_32 (SeperableC)	(None, 10, 10, 1536)	1582080
batch_normalization_ 38 (BatchNo)	(None, 10, 10, 1536)	6144
re_lu_41 (ReLU)	(None, 10, 10, 1536)	0
seperable_conv2d_33 (SeperableC)	(None, 10, 10, 2048)	3159552
batch_normalization_ 39 (BatchNo)	(None, 10, 10, 2048)	8192
global_average_ pooling2d (Global)	(None, 2048)	0
dense (Dense)	(None, 1000)	2049000
Total Params : 22,910,480		
Trainable params : 22,855,952		
Non-trainable params : 54,528		

Chapter 7

Experimental Results

7.1 Results

In order to make a fair comparison between our suggested model and other models, we trained and tested each model in the same environment. All models are trained and tested using 24 GB RAM and an Intel Core i5 7500 processor. (2400 MHz), and an RTX 3060 graphics card. All models utilized the same data augmentation strategy. All of our models are trained for 90 epochs. We also kept the image size, learning rate and others the same throughout the experiment. We can see the Training hyper parameter on Table 7.1.

Table 7.1: Hyperparameters used for Training models

Training Details	
Optimizer	Adam
Learning Rate	0.002
Batch Size	32
Epochs	90
Image Size	100 X 100
Callbacks	ReduceLRonPlateau, ModelCheckPoint

We compared our results to those of seven other pretrained models: Inception V3, ResNet 50, Xception, Densenet 201, VGG 16, VGG19, and InceptionResNet V2. The parameter that belongs to VGG 16 and has the value of around 14.8 Million is the one with the lowest value among all of the tested pretrained models. The model that we have provided has high assessment metric values as well as a parameter that is well-balanced. This sets it apart from other models that are already in use. Accuracy, precision, recall, F1, and loss are some of the measures that are included in this study's measurements.

In addition, we have trained and tested our dataset on many different Transformer architectures, all of which are explained in the methodology. Vision Transformer (ViT), Compact Convolutional Transformer (CCT), and ConvMixer are the three different types of transformer architectures that we have implemented.

The proportion of correct predictions a model makes compared to all possible forecasts can be conceived of as the model’s accuracy.

In order to determine whether a percentage of identifications are accurate, the level of precision is used. To determine accuracy, take the sum of all predicted positive outcomes (TP) and divide it by the sum of all predicted and actual positive outcomes (TP + FP). In order to calculate it, we use the formula:

$$Precision = \frac{TP}{TP + FP} \quad (7.1)$$

The recall rate is calculated to determine the accuracy of detection. Important in determining accuracy is the ratio of true positives (TP) to total data (TP + FN). The formula for determining recall is as follows:

$$Recall = \frac{TP}{TP + FN} \quad (7.2)$$

To that end, the F1 Score is widely used to evaluate the performance of machine learning programs. Accuracy and recall are averaged out to get this number. The F1 Score is calculated using the following formula:

$$F1 = \frac{2 * Recall * Precision}{Recall + Precision} \quad (7.3)$$

When compared to previous approaches and earlier studies, the testing accuracy of our model along with the other metrics, exhibit quite a bit of promise after making use of the Plant Village dataset. The other models were either more expensive in terms of the number of parameters they required or more complicated in terms of the amount of processing power they required. Our approach required fewer parameters. The comparison between training and testing accuracy can be found in tables 7.2 and 7.3.

Table 7.2: Training Accuracy and other Metrics

Model Name	Accuracy	Epochs	Recall	Precision	Loss	F1 Score
Inception V3	81.78%	90	0.7615	0.8856	0.5817	0.8189
ResNet50	55.31%	90	0.4539	0.7074	1.6845	0.5530
Xception	94.99%	90	0.9346	0.9645	0.1551	0.9493
DenseNet201	98.86%	90	0.9877	0.9894	0.0346	0.9885
VGG16	94.14%	90	0.9216	0.9614	0.1954	0.9411
VGG19	92.58%	90	0.8997	0.9521	0.2431	0.9252
InceptionResNet V2	88.88%	90	0.8541	0.926	0.3482	0.8886
Proposed Model	99.87%	90	0.9985	0.9988	0.0047	0.9986

In order to make a fair comparison with our own model, we ensured that the number of epochs, picture size, and learning rate were all held consistent across all

Table 7.3: Testing Accuracy and other Metrics

Model Name	Accuracy	Epochs	Recall	Precision	Loss	F1 Score
Inception V3	81.99%	90	0.7672	0.8792	0.6081	0.8194
ResNet50	60.73%	90	0.5026	0.7397	1.4074	0.5985
Xception	92.69%	90	0.9132	0.9453	0.2418	0.9290
DenseNet201	96.35%	90	0.9627	0.966	0.1526	0.9643
VGG16	93.27%	90	0.913	0.951	0.2155	0.9316
VGG19	91.80%	90	0.8959	0.9434	0.2693	0.9190
InceptionResNet V2	86.76%	90	0.8363	0.9028	0.4119	0.8683
Proposed Model	99.54%	90	0.9954	0.9957	0.0137	0.9955

pre-trained models.

Fig 7.1. shows a visual representation of accuracy comparison with other Pre-Trained models.

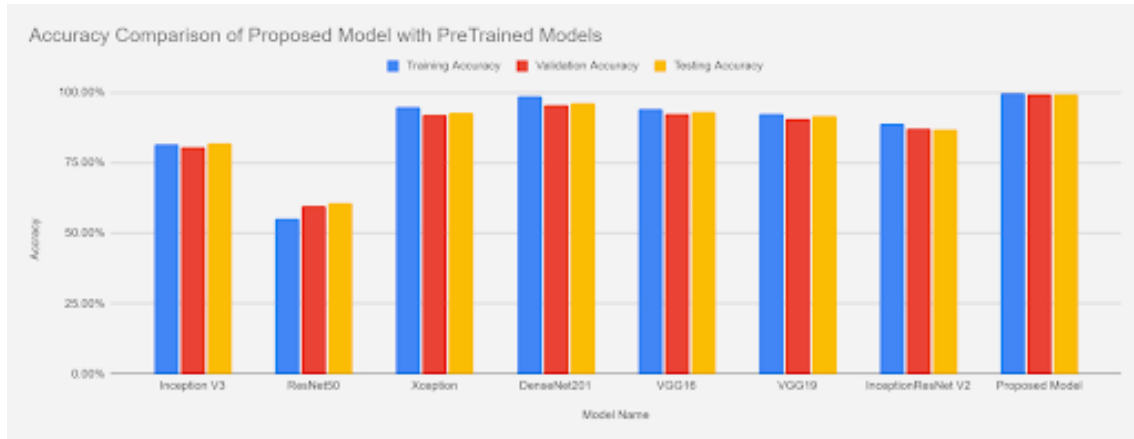


Figure 7.1: Training, Validation and Testing Accuracy Comparison

If we compare the parameter of our model with other models parameters we will see that our model requires the lowest number of parameters which is only 5,662,534. With this we can say that the computational power required for our model is very less compared to other Pre-Trained models. The comparison of parameters is illustrated in Fig 7.2.

As stated earlier, in addition to comparing our model to Pre-Trained models, we also compared it to Transformer architectures. In the Methodology section, all of the transformer designs utilized for comparison have been detailed. All Transformer models did quite well relative to the size of our data set, as a Transformer typically demands a bigger data set. In order for the transformers to perform effectively on our dataset, we applied these strategies in our experiment. Table 7.4 compares our model to a number of well-known Transformer models.

Table 7.4: Accuracy Comparison of Proposed model with Transformers

Accuracy Comparison with Transformers	
Architecture Name	Testing Accuracy (%)
Compact Convolutional Transformers (CCT)	97.22%
ConvMixer	96.09%
Vision Transformer (ViT)	93.99%
Vision Transformer with Shifted Patch Tokenization and Locality Self Attention	94,53%
Proposed Model	99.54%

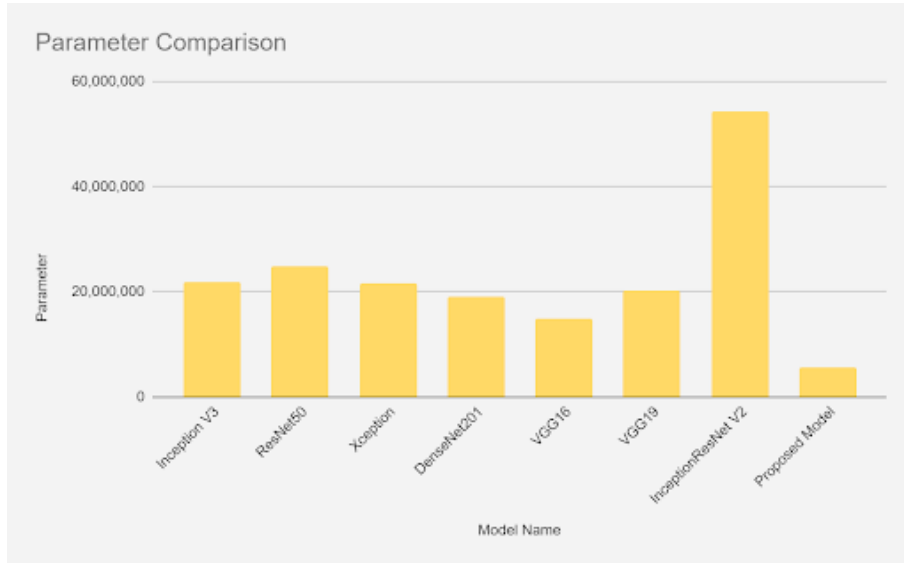


Figure 7.2: Parameter Comparison

7.2 Confusion Matrix and Classification Report

7.2.1 Confusion Matrix

The confusion matrix for the model we suggest is displayed below. Predictions are on the horizontal axis, and the correct response is on the vertical. From the fig below, we observe that our model was more or less correct in its predictions. Here are the graphs of model accuracy and model loss of our proposed model.

7.2.2 Classification Report

Our model metrics are quite high, as shown in the classification report. The model with the lowest score is Corn Cercospora leaf spot, Gray leaf spot, with an accuracy of 0.95, a recall of 0.98, as well as an f1-score of 0.97 and support of 114.

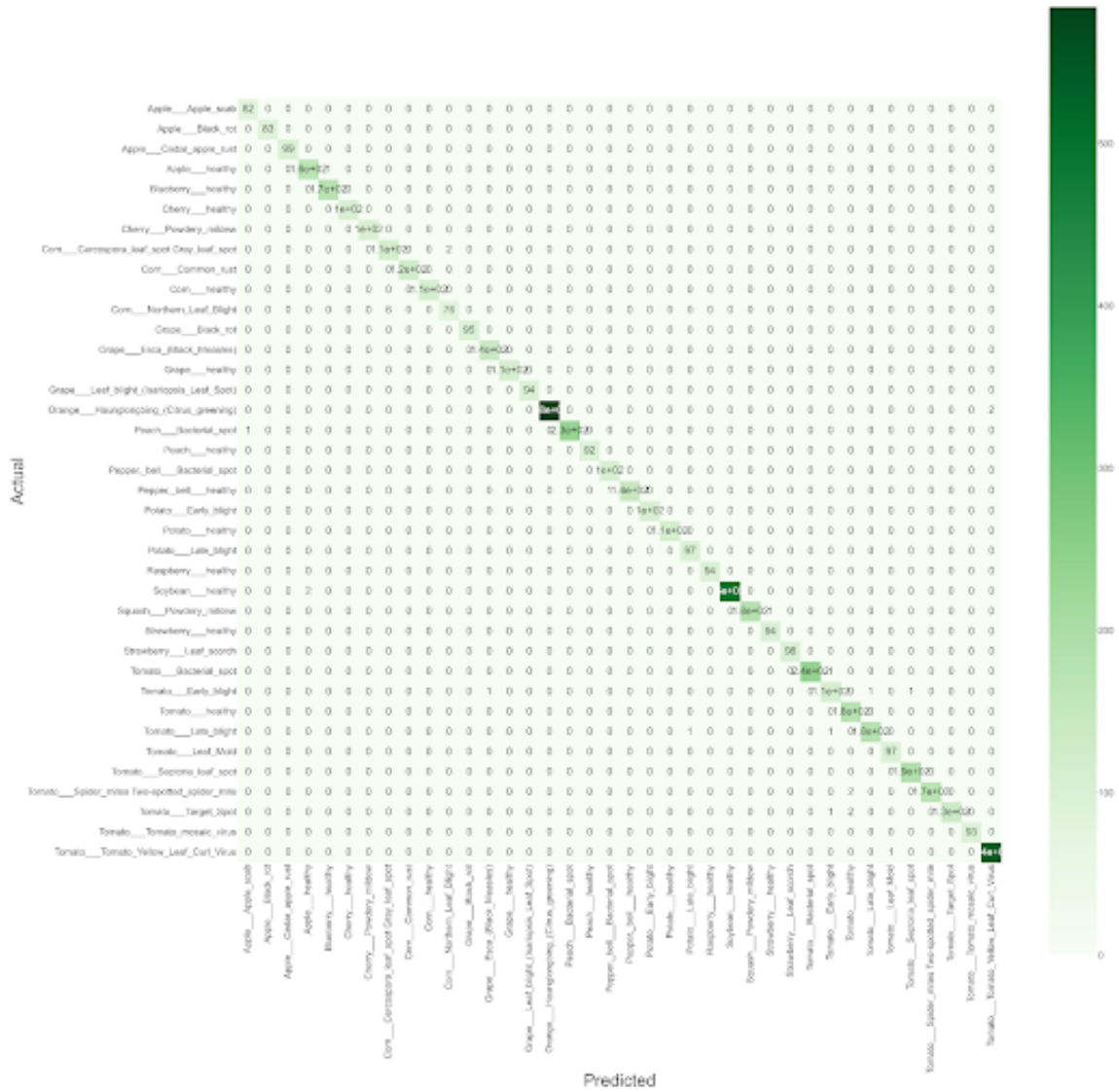


Figure 7.3: Confusion Matrix

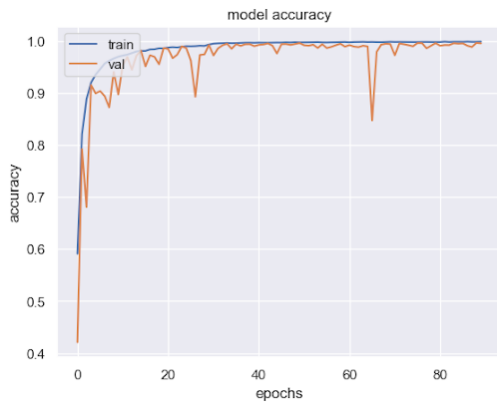


Figure 7.4: Proposed Model Accuracy

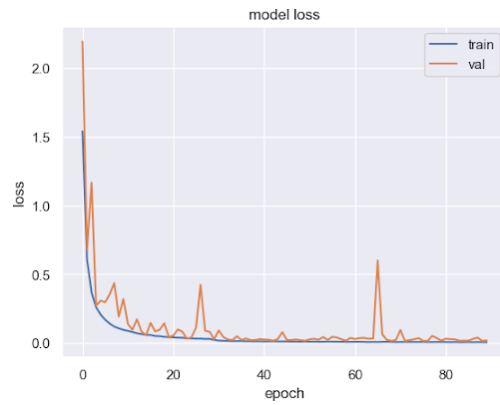


Figure 7.5: Proposed Model loss

Table 7.5: Classification Report

Classification Report	Precision	Recall	F1 Score	Support
Apple Apple scab	0.99	1.00	0.99	82
Apple Black rot	1.00	1.00	1.00	83
Apple Cedar apple rust	1.00	1.00	1.00	99
Apple healthy	0.99	0.99	0.99	166
Blueberry healthy	0.99	1.00	1.00	172
Cherry healthy	1.00	1.00	1.00	104
Cherry Powdery mildew	1.00	1.00	1.00	102
Corn Cercospora leaf spot Gray leaf spot	0.95	0.98	0.97	114
Corn Common rust	1.00	1.00	1.00	120
Corn healthy	1.00	1.00	1.00	112
Corn Northern Leaf Blight	0.97	0.93	0.95	82
Grap Black rot	1.00	1.00	1.00	95
Grape Esca (Black Measles)	0.99	1.00	1.00	137
Grape healthy	1.00	1.00	1.00	108
Grape Leaf blight (Isariopsis Leaf Spot)	1.00	1.00	1.00	94
Orange Haunglongbing (Citrus greening)	1.00	1.00	1.00	586
Peach Bacterial spot	1.00	1.00	1.00	232
Peach healthy	1.00	1.00	1.00	92
Pepper, bell Bacterial spot	0.99	1.00	1.00	104
Pepper, bell healthy	1.00	0.99	1.00	145
Potato Early blight	1.00	1.00	1.00	102
Potato healthy	1.00	1.00	1.00	112
Potato Late blight	0.99	1.00	0.99	97
Raspberry healthy	1.00	1.00	1.00	94
Soybean healthy	1.00	1.00	1.00	501
Squash Powdery mildew	1.00	0.99	1.00	179
Strawberry healthy	0.99	1.00	0.99	94
Strawberry Leaf scorch	1.00	1.00	1.00	98
Tomato Bacterial spot	1.00	1.00	1.00	241
Tomato Early blight	0.97	0.97	0.97	112
Tomato healthy	0.98	1.00	0.99	176
Tomato Late blight	0.99	0.99	0.99	178
Tomato Leaf Mold	0.99	1.00	0.99	97
Tomato Septorialeaf spot	0.99	1.00	1.00	186
Tomato Spider mites Two-spotted spider mite	1.00	0.99	0.99	172
Tomato Target Spot	1.00	0.98	0.99	136
Tomato Tomato mosaic virus	1.00	1.00	1.00	93
Tomato Tomato Yellow Leaf Curl Virus	1.00	1.00	1.00	538
Accuracy			1.00	6035
Macro average	0.99	0.99	0.99	6035
Weighted average	1.00	1.00	1.00	6035

Chapter 8

Web Implementation

For the web implementation, we utilized the well-known Python web framework Flask. It is a Python module that enables us to construct web apps quickly and with minimal effort. It has routing and a template engine that makes it more user-friendly. Werkzeug WSGI toolkit and Jinja2 template engine underpin Flask [99].

Every time a URL is entered into our search engine, a request is sent to the backend using routes in Flask or any other web framework. Werkzeug WSGI administers this procedure.

We always convey data from our backend to our frontend, and a template engine is necessary to build HTML logic. For this purpose, Flask uses the famous Python template engine Jinja2. Here, we employ curly brackets and percentage marks to enclose Python code. However, if we wish to present any data received from the backend on the frontend, we utilize two curly brackets and write the variable's name in between them.

For the website's frontend, we utilized HTML with jinja2 to create the framework and custom CSS and Bootstrap for style. We utilized Flask for the backend and TensorFlow for our trained model.

We attempted to make the design of the website's frontend as basic as possible so that anybody may use it without any prior experience. It begins with a title bar with the phrase "Plant Leaf Disease Identification." The next part provides a brief textual instruction on how to utilize this website, so that everyone may read and comprehend it. In the following part, there is an image upload option and a predict button. The user will first submit the photograph before clicking "Predict Disease." After the prediction is complete, a new section with the projected illness type and a picture will display.

As previously indicated, we utilized Flask for the backend. In addition, we utilized the flask "render_template" and "request" modules. We used render_template to display our HTML pages and the request module to retrieve information from forms, such as retrieving the uploaded picture in this example.

We developed a read and transform img function that accepts the image's url as

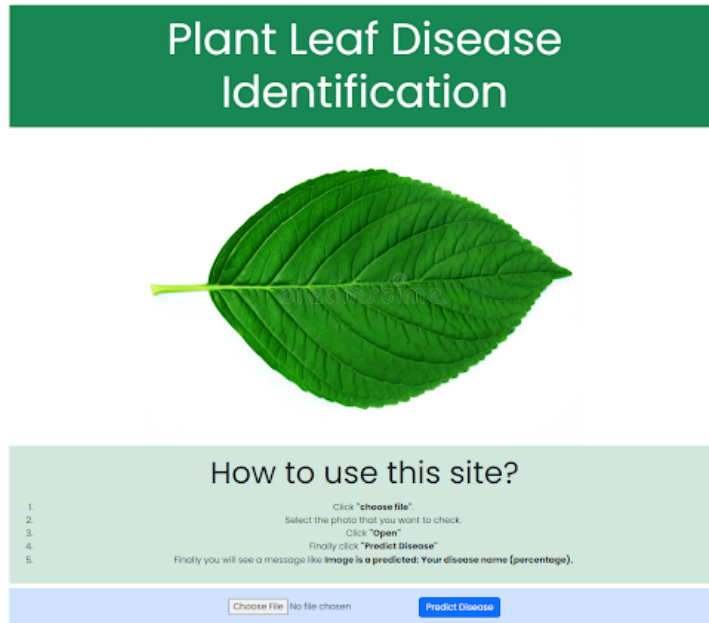


Figure 8.1: Representation of our Website Before Prediction

an input and resizes it to the desired size, which in our instance is 100 by 100. The picture is then transformed to an array using `img to array` from the `keras.utils` package, and the array is returned.

Next, we load our model using a `load_model` from `keras.models` and generate a Python list containing the names of all 38 classes.

As indicated earlier, we wanted to make our website as basic as possible, therefore we opted for a Single Page Application (SPA). The page has only two routes, which are identical. The only distinction between the two techniques is that one is a GET request and the other is a POST request. Whenever a visitor views the website or makes a GET request, the `index.html` file is rendered as seen in Figures 8.1 and 8.2. However, when a user uploads a picture and clicks on Predict Disease, a POST request is sent to our second route. Here, the picture is initially captured using the `request` module, and a path is then generated. As this picture is a static file, all submitted images are kept under a static folder within the image folder. Then, we turn our picture to an array using the `read` and `transform img` method. Then, we apply our model to forecast the picture, and we obtain the class index using `argmax` from `numpy`. In addition, to determine the percentage, we multiplied `max` from `numpy` by 100 and rounded the result to two decimal places. During rendering, we then transmit the class name, percentage, and image link to `index.html`. Fig 8.3. shows the workflow of our website.

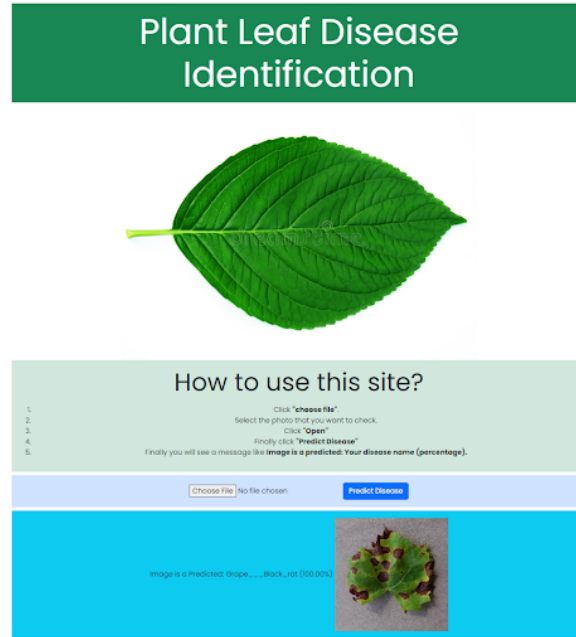


Figure 8.2: Representation of our website after prediction

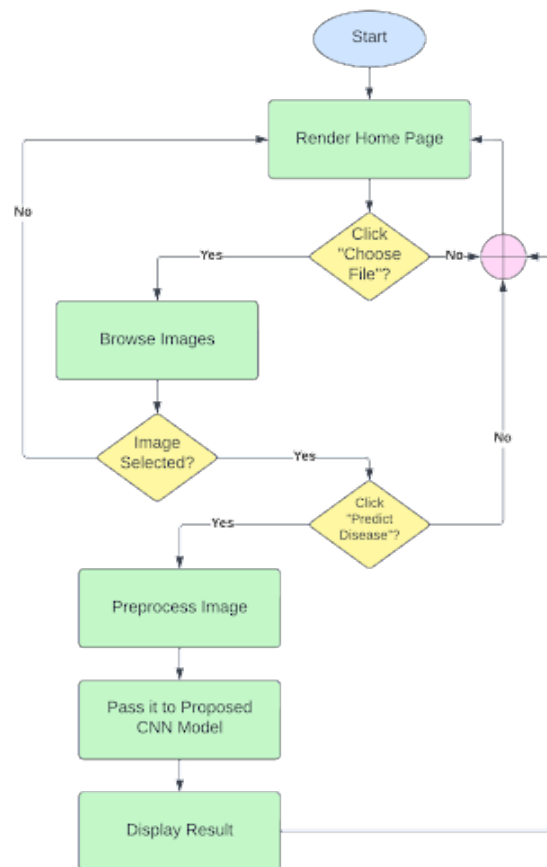


Figure 8.3: Workflow of our Web

Chapter 9

Discussions and Further Improvements

9.1 Discussion

Our model benefits from the increased stability afforded by the presence of numerous thick layers inside the completely connected layers. By incorporating different techniques like dropouts and batch normalization, our model can circumvent the common drawback caused by overfitting.

The evaluation metrics reveal, through the utilization of images with a lower resolution based base, that our proposed model has an excellent balance in terms of the outputs along with the parameters, as can be shown in Table 7.2 and Figure 7.2. As a direct consequence of this, users do not require a high-speed internet connection or an expensive mobile phone in order to upload photographs of low resolution. In addition, implementing our model to a web-based system will make life of many farmers very easy. As they no longer need to hire a botanist to identify the disease. They can take a photo of the diseased leaf and pass it to our model with the website and with lightning speed they can find out whether the leaf is diseased or not. As our model is very light weight due to the low number of parameters, farmers don't need to have a high end device.

Despite the model's exceptional ability to detect leaf illness, there are a few flaws with the research that needs to be resolved. Only data from a single region provided by PlantVillage have been used to assess the validity of our suggested model. As a result, additional research must be conducted on a variety of datasets pertaining to various areas.

9.2 Further Improvement

We aim to compare the proposed model in the future to others that have already been trained. To expand the breadth of the study and improve the quantity of information we have access to, we intend to continue collecting new images of plants from a variety of continents and countries, as well as information regarding leaf development patterns, growing circumstances, image quality, and mode. We will be

able to undertake substantially more detailed research as a result. In the not-too-distant future, our strategy will be useful for combating additional types of plant leaf diseases. In the future, we intend to reduce the amount of time and space-based complexity. In a future study, we plan to propose the development of an artificial intelligence (AI) system that is cloud-hosted, employs deep learning techniques, and includes additional data variants. Moreover, in order to diagnose plant illnesses, the focus of our current and future research will shift from the plant's leaves to other plant parts, such as the flowers and the fruits. This is important in order to identify plant-infecting pathogens.

Our model is quite lightweight, which enables us to quickly upload it to a website or develop an application that can be used by farmers on their computing devices that have limited power. Using this strategy, farmers will have an easier time identifying the malady that has affected the plant.

Chapter 10

Conclusion

Agriculture, one of the earliest discoveries made by mankind, was essential to the advancement of civilization to its current degree of prosperity. Therefore, modern farming practices need to be modified to meet the information and communication technology era (ICT). We intend to contribute to the modernization of agriculture by inventing a highly accurate and resource-efficient technique for the automated diagnosis of plant diseases. This would save time and money for farmers. Using cutting-edge technology that is both adaptable and cutting-edge, such as deep learning, it is possible for us to attain our objective. With the assistance of our suggested CNN model, we anticipate reaching a greater degree of disease detection accuracy in comparison to the currently pre-trained models.

As a result, within the scope of our research, we utilize leaf photos and the limited computing capabilities available to us to anticipate the development of plant diseases. In order to accomplish this, a lightweight version of the Deep CNN architecture has been built. 38 separate types of healthy and diseased leaf images can be categorized by this architecture with an accuracy comparable to that of competing systems. The model we have presented has fewer parameters than previous models based on deep learning along with machine learning. This is done to illustrate that the model we have provided can perform admirably while retaining its correctness.

Bibliography

- [1] J. E. van der Waals, L. Korsten, and T. A. S. Aveling, “A review of early blight of potato,” *African plant protection*, vol. 7, pp. 91–102, 2001.
- [2] C.-C. Yang, S. Prasher, P. Enright, *et al.*, “Application of decision tree technology for image classification using remote sensing data,” *Agricultural Systems*, vol. 76, pp. 1101–1117, Jun. 2003. DOI: 10.1016/S0308-521X(02)00051-3.
- [3] M. H. Mondal, “Crop agriculture of bangladesh: Challenges and opportunities,” *Bangladesh Journal of Agricultural Research*, vol. 35, pp. 235–245, 2010.
- [4] T. Rumpf, A.-K. Mahlein, U. Steiner, E.-C. Oerke, H.-W. Dehne, and L. Plümer, “Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance,” *Computers and Electronics in Agriculture*, vol. 74, pp. 91–99, Oct. 2010. DOI: 10.1016/j.compag.2010.06.009.
- [5] L. R. Cooke, H. T. A. M. Schepers, A. Hermansen, *et al.*, “Epidemiology and Integrated Control of Potato Late Blight in Europe,” en, *Potato Research*, vol. 54, no. 2, pp. 183–222, Jun. 2011, ISSN: 1871-4528. DOI: 10.1007/s11540-011-9187-0. [Online]. Available: <https://doi.org/10.1007/s11540-011-9187-0> (visited on 01/16/2023).
- [6] D. A. Majumdar, *Managing Mites On Tomato Crops*, en-US, Aug. 2012. [Online]. Available: <https://www.growingproduce.com/crop-protection/insect-control/managing-mites-on-tomato-crops/> (visited on 01/16/2023).
- [7] F. Garcia-Ruiz, S. Sankaran, J. M. Maja, W. S. Lee, J. Rasmussen, and R. Ehsani, “Comparison of two aerial imaging platforms for identification of huanglongbing-infected citrus trees,” *Computers and Electronics in Agriculture*, vol. 91, pp. 106–115, Feb. 2013. DOI: 10.1016/j.compag.2012.12.002.
- [8] M. Lin, Q. Chen, and S. Yan, *Network in network*, 2013. DOI: 10.48550/ARXIV.1312.4400. [Online]. Available: <https://arxiv.org/abs/1312.4400>.
- [9] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv 1409.1556*, Sep. 2014.
- [10] R. Calderón Madrid, J. Navas Cortés, and P. Zarco-Tejada, “Early detection and quantification of verticillium wilt in olive using hyperspectral and thermal imagery over large areas,” *Remote Sensing*, vol. 7, pp. 5584–5610, May 2015. DOI: 10.3390/rs70505584.

- [11] S. Haruta and N. Kanno, “Survivability of microbes in natural environments and their ecological impacts,” *Microbes and environments / JSME*, vol. 30, pp. 123–125, Jun. 2015. DOI: 10.1264/jsme2.ME3002rh.
- [12] D. Hughes and M. Salathe, “An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing,” Nov. 2015.
- [13] S. D. Khirade and A. Patil, “Plant disease detection using image processing,” in *2015 International Conference on Computing Communication Control and Automation*, 2015, pp. 768–771. DOI: 10.1109/ICCUBEA.2015.153.
- [14] S. H. Lee, C. S. Chan, P. Wilkin, and P. Remagnino, “Deep-plant: Plant identification with convolutional neural networks,” in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 452–456. DOI: 10.1109/ICIP.2015.7350839.
- [15] U. Mokhtar, M. A. S. Ali, A. E. Hassanien, and H. Hefny, “Identifying two of tomatoes leaf viruses using support vector machine,” in *Information Systems Design and Intelligent Applications*, J. K. Mandal, S. C. Satapathy, M. Kumar Sanyal, P. P. Sarkar, and A. Mukhopadhyay, Eds., New Delhi: Springer India, 2015, pp. 771–782, ISBN: 978-81-322-2250-7.
- [16] G. Grinblat, L. Uzal, M. Larese, and P. Granitto, “Deep learning for plant identification using vein morphological patterns,” *Computers and Electronics in Agriculture*, vol. 127, pp. 418–424, Sep. 2016. DOI: 10.1016/j.compag.2016.07.003.
- [17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Jun. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.
- [18] P. Martínez-García, E. López-Solanilla, C. Ramos, and P. Rodriguez, “Prediction of bacterial associations with plants using a supervised machine-learning approach: Prediction of bacterial associations with plants,” *Environmental Microbiology*, vol. 18, May 2016. DOI: 10.1111/1462-2920.13389.
- [19] X. Pantazi, D. Moshou, A. A. Tamouridou, and S. Kasderidis, “Leaf disease recognition in vine plants based on local binary patterns and one class support vector machines,” vol. 475, Sep. 2016, pp. 319–327, ISBN: 978-3-319-44943-2. DOI: 10.1007/978-3-319-44944-9_27.
- [20] M. T. Ribeiro, S. Singh, and C. Guestrin, ““why should i trust you?”: Explaining the predictions of any classifier,” in *NAACL 2016*, 2016.
- [21] S. Sladojevic, M. Arsenovic, A. Anderla, and D. Stefanović, “Deep neural networks based recognition of plant diseases by leaf image classification,” *Computational Intelligence and Neuroscience*, vol. 2016, pp. 1–11, Jun. 2016. DOI: 10.1155/2016/3289801.
- [22] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” *AAAI Conference on Artificial Intelligence*, vol. 31, Feb. 2016. DOI: 10.1609/aaai.v31i1.11231.
- [23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308.

- [24] A. A. Bharate and M. S. Shirdhonkar, “A review on plant disease detection using image processing,” in *2017 International Conference on Intelligent Sustainable Systems (ICISS)*, 2017, pp. 103–109. DOI: 10.1109/ISS1.2017.8389326.
- [25] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” Jul. 2017, pp. 1800–1807. DOI: 10.1109/CVPR.2017.195.
- [26] M. Ebrahimi, M.-H. Khoshtaghaza, S. Minaee, and B. Jamshidi, “Vision-based pest detection based on svm classification method,” *Computers and Electronics in Agriculture*, vol. 137, pp. 52–58, May 2017. DOI: 10.1016/j.compag.2017.03.016.
- [27] A. Fuentes, S. Yoon, S. Kim, and D. Park, “A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition,” *Sensors*, vol. 17, p. 2022, Sep. 2017. DOI: 10.3390/s17092022.
- [28] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger, “Densely connected convolutional networks,” Jul. 2017. DOI: 10.1109/CVPR.2017.243.
- [29] A. Johannes, A. Picon, A. Alvarez-Gila, *et al.*, “Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case,” *Computers and Electronics in Agriculture*, vol. 138, pp. 200–209, Jun. 2017. DOI: 10.1016/j.compag.2017.04.013.
- [30] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, “Identification of rice diseases using deep convolutional neural networks,” *Neurocomputing*, vol. 267, pp. 378–384, 2017, ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2017.06.023>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231217311384>.
- [31] D. y. h. a. q. a. t. p. A. o. e. d. i. H. C. f. +. S. more, *GEOPOTATO - Control fungal disease in potato in Bangladesh*, en-us, Oct. 2017. [Online]. Available: <https://www.wur.nl/en/project/geopotato-control-fungal-disease-in-potato-in-bangladesh.htm> (visited on 01/16/2023).
- [32] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” Jun. 2017.
- [33] C. Wetterich, R. Neves, J. Belasque, R. Ehsani, and L. Marcassa, “Detection of huanglongbing in florida using fluorescence imaging spectroscopy and machine-learning methods,” *Applied Optics*, vol. 56, pp. 15–23, Jan. 2017. DOI: 10.1364/AO.56.000015.
- [34] Y. Zhang, Z. Dong, X. Chen, *et al.*, “Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation,” *Multimedia Tools and Applications*, vol. 78, pp. 3613–3632, 2017.
- [35] E. El Houby, “A survey on applying machine learning techniques for management of diseases,” *Journal of Applied Biomedicine*, vol. 16, Aug. 2018. DOI: 10.1016/j.jab.2018.01.002.
- [36] K. Ferentinos, “Deep learning models for plant disease detection and diagnosis,” *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, Feb. 2018. DOI: 10.1016/j.compag.2018.01.009.

- [37] Y. Guo, S. Han, Y. Li, C. Zhang, and Y. Bai, “K-nearest neighbor combined with guided filter for hyperspectral image classification,” *Procedia Computer Science*, vol. 129, pp. 159–165, Jan. 2018. DOI: 10.1016/j.procs.2018.03.066.
- [38] A. Kamilaris and F. X. Prenafeta-Boldú, “Deep learning in agriculture: A survey,” *Comput. Electron. Agric.*, vol. 147, pp. 70–90, 2018.
- [39] E. Too, Y. Li, S. Njuki, and L. Yingchun, “A comparative study of fine-tuning deep learning models for plant disease identification,” *Computers and Electronics in Agriculture*, vol. 161, Mar. 2018. DOI: 10.1016/j.compag.2018.03.032.
- [40] E. Too, Y. Li, S. Njuki, and L. Yingchun, “A comparative study of fine-tuning deep learning models for plant disease identification,” *Computers and Electronics in Agriculture*, vol. 161, Mar. 2018. DOI: 10.1016/j.compag.2018.03.032.
- [41] S. Wang, C. Tang, J. Sun, *et al.*, “Multiple sclerosis identification by 14-layer convolutional neural network with batch normalization, dropout, and stochastic pooling,” *Frontiers in Neuroscience*, vol. 12, 2018.
- [42] R. Wason, “Deep learning: Evolution and expansion,” *Cognitive Systems Research*, vol. 52, Aug. 2018. DOI: 10.1016/j.cogsys.2018.08.023.
- [43] L. Zhong, L. Hu, and H. Zhou, “Deep learning based multi-temporal crop classification,” *Remote Sensing of Environment*, vol. 221, pp. 430–443, Dec. 2018. DOI: 10.1016/j.rse.2018.11.032.
- [44] M. Arsenovic, M. Karanovic, S. Sladojevic, A. Anderla, and D. Stefanović, “Solving current limitations of deep learning based approaches for plant disease detection,” *Symmetry*, vol. 11, p. 21, Jul. 2019. DOI: 10.3390/sym11070939.
- [45] J. Barbedo, “Plant disease identification from individual lesions and spots using deep learning,” *Biosystems Engineering*, vol. 180, pp. 96–107, Apr. 2019. DOI: 10.1016/j.biosystemseng.2019.02.002.
- [46] B. M. Dala-Paula, A. Plotto, J. Bai, *et al.*, “Effect of Huanglongbing or Greening Disease on Orange Juice Quality, a Review,” *Frontiers in Plant Science*, vol. 9, 2019, ISSN: 1664-462X. [Online]. Available: <https://www.frontiersin.org/articles/10.3389/fpls.2018.01976> (visited on 01/16/2023).
- [47] E. Hossain, M. F. Hossain, and M. A. Rahaman, “A color and texture based approach for the detection and classification of plant leaf disease using knn classifier,” in *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, 2019, pp. 1–6. DOI: 10.1109/ECACE.2019.8679247.
- [48] S. V. Militante, B. D. Gerardo, and N. V. Dionisio, “Plant leaf detection and disease recognition using deep learning,” in *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*, 2019, pp. 579–582. DOI: 10.1109/ECICE47484.2019.8942686.
- [49] S.-H. Tsang, *Review: DenseNet — Dense Convolutional Network (Image Classification)*, en, Mar. 2019. [Online]. Available: <https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803> (visited on 01/16/2023).

- [50] M. Vaishnnave, K. S. Devi, P. Srinivasan, and G. A. P. Jothi, “Detection and classification of groundnut leaf diseases using knn classifier,” in *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, 2019, pp. 1–5. DOI: 10.1109/ICSCAN.2019.8878733.
- [51] A. Bhatia, A. Chug, and A. Singh, “Plant disease detection for high dimensional imbalanced dataset using an enhanced decision tree approach,” *International Journal of Future Generation Communication and Networking*, vol. 13, pp. 71–78, Dec. 2020. DOI: 10.33832/ijfgcn.2020.13.4.07.
- [52] D. Das, M. Singh, S. Mohanty, and S. Chakravarty, “Leaf disease detection using support vector machine,” Jul. 2020, pp. 1036–1040. DOI: 10.1109/ICCSP48568.2020.9182128.
- [53] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, *An image is worth 16x16 words: Transformers for image recognition at scale*, Oct. 2020.
- [54] S. Hernández and J. L. López, “Uncertainty quantification for plant disease detection using bayesian deep learning,” *Applied Soft Computing*, vol. 96, p. 106 597, 2020, ISSN: 1568-4946. DOI: <https://doi.org/10.1016/j.asoc.2020.106597>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1568494620305354>.
- [55] F. Marzougui, M. Elleuch, and M. Kherallah, “A deep cnn approach for plant disease detection,” in *2020 21st International Arab Conference on Information Technology (ACIT)*, 2020, pp. 1–6. DOI: 10.1109/ACIT50332.2020.9300072.
- [56] H. D. Prasetyo, H. Triatmoko, Nurdiansyah, and I. N. Isnainiyah, “The implementation of cnn on website-based rice plant disease detection,” in *2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*, 2020, pp. 75–80. DOI: 10.1109/ICIMCIS51567.2020.9354329.
- [57] B. Rajesh, M. V. Sai Vardhan, and L. Sujihelen, “Leaf disease detection and classification by decision tree,” in *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, 2020, pp. 705–708. DOI: 10.1109/ICOEI48184.2020.9142988.
- [58] P. Sethy, N. Barpanda, A. Rath, and S. Behera, “Nitrogen deficiency prediction of rice crop based on convolutional neural network,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, Nov. 2020. DOI: 10.1007/s12652-020-01938-8.
- [59] S. Tang, S. Yuan, and Y. Zhu, “Data preprocessing techniques in convolutional neural network based on fault diagnosis towards rotating machinery,” *IEEE Access*, vol. 8, pp. 149 487–149 496, 2020. DOI: 10.1109/ACCESS.2020.3012182.
- [60] O. Abayomi-Alli, R. Damaševičius, S. Misra, and R. Maskeliunas, “Cassava disease recognition from low-quality images using enhanced data augmentation model and deep learning,” *Expert Systems*, vol. 38, Nov. 2021. DOI: 10.1111/exsy.12746.

- [61] G. Baskar and G. Devi, “Identifying and classifying plant disease using resilient lf-cnn,” *Ecological Informatics*, vol. 63, p. 101 283, Mar. 2021. DOI: 10.1016/j.ecoinf.2021.101283.
- [62] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the big data paradigm with compact transformers,” *ArXiv*, vol. abs/2104.05704, 2021.
- [63] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the big data paradigm with compact transformers,” *ArXiv*, vol. abs/2104.05704, 2021.
- [64] S. H. Lee, S. Lee, and B. C. Song, “Vision transformer for small-size datasets,” *ArXiv*, vol. abs/2112.13492, 2021.
- [65] J. Walliser, *Powdery Mildew on Squash: What is It and How Do You Get Rid of It?* en-US, Sep. 2021. [Online]. Available: <https://savvygardening.com/powdery-mildew-on-squash/> (visited on 01/16/2023).
- [66] D. Wang, J. Wang, W. Li, and P. Guan, “T-cnn: Trilinear convolutional neural networks model for visual detection of plant diseases,” *Computers and Electronics in Agriculture*, vol. 190, p. 106 468, 2021, ISSN: 0168-1699. DOI: <https://doi.org/10.1016/j.compag.2021.106468>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0168169921004853>.
- [67] A. S. Hosain, M. H. K. Mehedi, T. Jerin, *et al.*, “Rice leaf disease detection with transfer learning approach,” Sep. 2022, pp. 1–6. DOI: 10.1109/IICAJET55139.2022.9936780.
- [68] X. Huang, R. Azzam, S. Javed, *et al.*, “Cm-unet: Convmixer unet for segmentation of unknown objects in cluttered scenes,” *IEEE Access*, vol. 10, pp. 123 622–123 633, 2022. DOI: 10.1109/ACCESS.2022.3224588.
- [69] M. S. Islam, S. Sultana, F. A. Farid, *et al.*, “Multimodal hybrid deep learning approach to detect tomato leaf disease using attention based dilated convolution feature extractor with logistic regression classification,” *Sensors*, vol. 22, no. 16, 2022, ISSN: 1424-8220. DOI: 10.3390/s22166079. [Online]. Available: <https://www.mdpi.com/1424-8220/22/16/6079>.
- [70] S. Lee, S. Lee, and B. C. Song, “Improving vision transformers to learn small-size dataset from scratch,” *IEEE Access*, vol. 10, pp. 123 212–123 224, 2022. DOI: 10.1109/ACCESS.2022.3224044.
- [71] M. H. K. Mehedi, A. S. Hosain, S. Ahmed, *et al.*, *Plant leaf disease detection using transfer learning and explainable ai*, Oct. 2022.
- [72] J. J. Mondal, M. F. Islam, S. Zabeen, A. B. M. A. A. Islam, and J. Noor, “Note: Plant leaf disease network (plead-net): Identifying plant leaf diseases through leveraging limited-resource deep convolutional neural network,” in *ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies (COMPASS)*, ser. COMPASS ’22, Seattle, WA, USA: Association for Computing Machinery, 2022, pp. 668–673, ISBN: 9781450393478. DOI: 10.1145/3530190.3534844. [Online]. Available: <https://doi.org/10.1145/3530190.3534844>.

- [73] A. Singh, S. Sreenivasu, U. Mahalaxmi, H. Sharma, D. Patil, and E. Asenso, “Hybrid feature-based disease detection in plant leaf using convolutional neural network, bayesian optimized svm, and random forest classifier,” *Journal of Food Quality*, vol. 2022, p. 16, Feb. 2022. DOI: 10.1155/2022/2845320.
- [74] S. T. A. Ozcelik, H. Uyanik, E. Deniz, and A. Sengur, “Automated hypertension detection using convmixer and spectrogram techniques with ballistocardiograph signals,” *Diagnostics*, vol. 13, no. 2, 2023, ISSN: 2075-4418. DOI: 10.3390/diagnostics13020182. [Online]. Available: <https://www.mdpi.com/2075-4418/13/2/182>.
- [75] <https://www.planthealthaustralia.com.au/wp-content/uploads/2013/11/Bacterial-blight-of-grapevine-FS.pdf>.
- [76] *Apple scab*, en-US. [Online]. Available: <https://www.apsnet.org/edcenter/disandpath/fungalasco/pdlessons/Pages/AppleScab.aspx> (visited on 01/16/2023).
- [77] *Bacterial Leaf Spot Of Peach - Tips On Controlling Leaf Spot On Peaches*, en. [Online]. Available: <https://www.gardeningknowhow.com/edible/fruits/peach/bacterial-spot-on-peach-trees.htm> (visited on 01/16/2023).
- [78] *Bacterial spot of tomato and pepper*, en. [Online]. Available: <https://extension.umn.edu/disease-management/bacterial-spot-tomato-and-pepper> (visited on 01/16/2023).
- [79] *Bangladesh: Growing the Economy through Advances in Agriculture*, en, Text/HTML. DOI: 10/07/bangladesh-growing-economy-through-advances-in-agriculture. [Online]. Available: <https://www.worldbank.org/en/results/2016/10/07/bangladesh-growing-economy-through-advances-in-agriculture> (visited on 01/16/2023).
- [80] *Black Rot and Frogeye Leaf Spot of Apple*, en. [Online]. Available: <https://ohioline.osu.edu/factsheet/plpath-fru-07> (visited on 01/16/2023).
- [81] *Cedar-Apple Rust*, en. [Online]. Available: <https://ohioline.osu.edu/factsheet/plpath-tree-10> (visited on 01/16/2023).
- [82] *Cherry Powdery Mildew — WSU Tree Fruit — Washington State University*, en-US. [Online]. Available: <http://treefruit.wsu.edu/crop-protection/disease-management/cherry-powdery-mildew/> (visited on 01/16/2023).
- [83] *Common rust on corn*, en. [Online]. Available: <https://extension.umn.edu/corn-pest-management/common-rust-corn> (visited on 01/16/2023).
- [84] *Crop Protection Network*. [Online]. Available: <https://cropprotectionnetwork.org/encyclopedia/gray-leaf-spot-of-corn> (visited on 01/16/2023).
- [85] *Esca: Devastating “trunk disease” of grapevines – eVineyard blog*, en-US. [Online]. Available: <https://www.evineyardapp.com/blog/2016/06/06/esca-devastating-trunk-disease-of-grapevines/> (visited on 01/16/2023).
- [86] *Extension — Bacterial Leaf Spot of Pepper*, en. [Online]. Available: <https://extension.wvu.edu/lawn-gardening-pests/plant-disease/fruit-vegetable-diseases/bacterial-leaf-spot-of-pepper> (visited on 01/16/2023).
- [87] *Grape Black Rot*, en. [Online]. Available: <https://ohioline.osu.edu/factsheet/plpath-fru-24> (visited on 01/16/2023).

- [88] *How to Identify and Control Early Blight on Tomatoes*, en. [Online]. Available: <https://www.thespruce.com/early-blight-on-tomato-plants-1402973> (visited on 01/16/2023).
- [89] *Late Blight / Tomato / Agriculture: Pest Management Guidelines / UC Statewide IPM Program (UC IPM)*. [Online]. Available: <https://ipm.ucanr.edu/agriculture/tomato/late-blight/> (visited on 01/16/2023).
- [90] *New Disease Report - Target Spot of Tomato*, en. [Online]. Available: <https://vegcropshotline.org/article/new-disease-report-target-spot-of-tomato/> (visited on 01/16/2023).
- [91] *Northern corn leaf blight*, en. [Online]. Available: <https://extension.umn.edu/corn-pest-management/northern-corn-leaf-blight> (visited on 01/16/2023).
- [92] *Septoria Leaf Spot of Tomato*. [Online]. Available: <https://www.missouribotanicalgarden.org/gardens-gardening/your-garden/help-for-the-home-gardener/advice-tips-resources/pests-and-problems/diseases/fungal-spots/septoria-leaf-spot-of-tomato> (visited on 01/16/2023).
- [93] *Strawberry Leaf Scorch Control: How To Treat Leaf Scorch On Strawberry Plants*, en. [Online]. Available: <https://www.gardeningknowhow.com/edible/fruits/strawberry/strawberries-with-leaf-scorch.htm> (visited on 01/16/2023).
- [94] *The Development of Agriculture — National Geographic Society*. [Online]. Available: <https://education.nationalgeographic.org/resource/development-agriculture> (visited on 01/16/2023).
- [95] *Tomato Leaf Mold Treatment: How To Treat Leaf Mold Of Tomato Plants*, en. [Online]. Available: <https://www.gardeningknowhow.com/edible/vegetables/tomato/managing-tomato-leaf-mold.htm> (visited on 01/16/2023).
- [96] *Tomato viruses*, en. [Online]. Available: <https://extension.umn.edu/disease-management/tomato-viruses> (visited on 01/16/2023).
- [97] *Tomato Yellow Leaf Curl / Tomato / Agriculture: Pest Management Guidelines / UC Statewide IPM Program (UC IPM)*. [Online]. Available: <https://ipm.ucanr.edu/agriculture/tomato/tomato-yellow-leaf-curl/> (visited on 01/16/2023).
- [98] *Transformers from scratch — peterbloem.nl*. [Online]. Available: <https://peterbloem.nl/blog/transformers> (visited on 01/16/2023).
- [99] *What is Flask Python - Python Tutorial*. [Online]. Available: <https://pythonbasics.org/what-is-flask-python/> (visited on 01/16/2023).
- [100] *What is Tomato Mosaic Virus?* en-US. [Online]. Available: <https://www.creative-diagnostics.com/blog/index.php/what-is-tomato-mosaic-virus/> (visited on 01/16/2023).