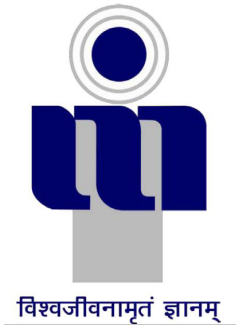
Crop Disease Detection using Deep Learning

A project report submitted in fulfillment of the requirements for B.Tech. Project

B.Tech.

by

Mohana Karthik Ianala (2019IMG-037)



ABV INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT GWALIOR-474 015

2022

CANDIDATES DECLARATION

I hereby certify that the work, which is being presented in the report, entitled **Crop Disease Detection using Deep Learning**, in fulfillment of the requirement for the award of the Degree of **Bachelor of Technology** and submitted to the institution is an authentic record of my own work carried out during the period *June 2022* to *September 2022* under the supervision of **Prof. Mahua Battacharya**. I have also cited the references about the text(s)/figure(s)/table(s) from where they have been taken.

Date: Signatures of the Candidates

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: Signatures of the Research Supervisor

ABSTRACT

Crop diseases pose a serious threat to food security, however, it might be difficult to quickly recognise them in many regions of the world. since critical evidence is frequently ignored and crucial foundations aren't there. With the development of precise algorithms, impressive achievements in the field of leaf-based image categorization were achieved. This study utilizes a CNN to detect diseases in cotton plant leaves. It offers more effective ways to spot bacterial infections and environmental effects. Our proposal covers the following implementation phases: dataset construction, feature extraction, classifier training, and classification. To categorise the images of sick and healthy leaves, the sick and healthy leaf datasets are integrated and trained using a CNN model. Utilizing EfficientNetV2-B2, we extract features from the images and train the model. Overall, by using machine learning to train the extensive publicly available data sets, we have a significant probability of identifying the illness that is present in plants on a large scale. Deep learning using CNNs has proven very successful in classifying a variety of plant illnesses that damage the leaves. It provides a novel technique to disease diagnosis that is efficient. In this project's experiments, the maximum test accuracy was 99.28%, which is higher than in earlier studies.

*Keywords:*Cotton Plant Diseases, Keras, CNN, DenseNet, EfficientNetV2.

ACKNOWLEDGEMENTS

I am highly indebted to **Prof. Mahua Battacharya**, and am obliged for giving me the autonomy of functioning and experimenting with ideas. I express my wholehearted gratitude to her for her academic guidance, interest in my project, and constant support. The support of the present work are mainly due to her valuable advice and suggestions. The mentor always answered countless doubts with smiling kindness and immense patience, never making me feel like a novice by constantly listening to opinions, appreciating and improving them, and giving the project a free hand. The present work has attained its stage because of her overwhelming interest and helpful attitude.
Finally, I am grateful to my family and friends, whose constant encouragement renewed my spirit, refocused my attention and energy, and helped me carry out this work.

(Mohana Karthik Ianala)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

CNN	Convolutional Neural Network.
ANN	Artificial Neural Network.
NAS	Neural Architecture Search.
DenseNet	Densely Connected Convolutional Networks.
HTML	HyperText Markup Language.
CSS	Cascading Style Sheets.

NOTATIONS

l	l^{th} layer.
ϕ	Rectified Linear Unit(ReLU) activation function.
$Y_i^{(l)}$	Output of l^{th} layer.
$B_i^{(l)}$	Bias Matrix.
$K_{i,j}^{(l)}$	Filter of$2w_k + 1 \times 2h_k + 1$.
f_i	Output for global average pooling layer

CHAPTER 1

INTRODUCTION

The emergence of plant diseases distresses agrarian production. If vegetation disorders are not diagnosed in time, food scarcity will intensify. Plant diseases, pests, and weeds threaten production and quality farming, resulting in crop loss and economic loss. That means about 15-25% of food production in India. Various other factors degrade the eminence and quantum of agricultural products, such as climate change and modern cultivation techniques with large amounts of chemical fertilizers. Infected plants often show apparent signs or sores on plant leaves, trunks, flowers, or fruits. In general, each disease or insect environment produces a single visual archetype that can be used to interpret anomalies. Generally, plant leaves are a significant source of plant disease, and most prophetic significance of the disease may initiate to emerge on the plant leaves.

In habitual, agricultural, and plant pathology experts visit the farmland or farmers to identify plant disorders and pests based on acquaintances. This approach is not only humble, but also ambitious and ineffective. Agriculturists with less knowledge may misjudge and use pesticides or insecticides indiscriminately during the screening process. This has resulted in indispensable economic losses. To address these challenges, image processing using an automatic plant leaf disease detection approach is essential. Timely perception is the baseline for effective interdiction and supervision of plant leaf diseases, and they play an essential role in the supervision and decision-making of agrarian products.

In a recent study, computer vision and machine learning based techniques were developed for plant leaf disease detection. Real-time plant disease detection has some significant challenges, such as complex background and severity of the disease due to the images being captured in real-time scenarios from the farm field.

1.1 Context

According to the Food and Agricultural Organization(FAO) of the United Nations, the global population is expected to reach 9.2 billion people by 2050. A higher level of efficiency in the current farming practises is required to be able to feed the world's steadily expanding population. Thus, one of the key goals of agricultural operations is to enhance crop output. As per estimates, one fifth of the world's annual crop production is lost to herbivorous insects. To make the process of learning from empirical data more straightforward, numerous strategies were employed to comprehend the laws and relationships from various data sets. The major objective of these strategies is to make sense of real-world data, and they work well on synthetic test data sets. When the problem is too complex to build a solution with guarantees, machine learning (ML) offers an alternative to the conventional engineering method.

A subcategory of machine learning called "deep learning" is entirely dependent on neural networks[3]. It's popular now because we previously had a plenty of data but insufficient processing power. Neurons are a formal definition of deep learning. Deep learning, a particular kind of machine learning, gains tremendous strength and flexibility by learning to represent the world as a layered hierarchy of concepts.

A particular type of neural network called a convolutional neural network is utilised for image recognition. Because convolutional networks can recognise a certain pattern in an image anywhere after learning it, they are far more effective than artificial neural networks (ANNs)[4]. CNNs also benefit from the local spatial coherence of images.

1.2 Motivation

The leaves of a plant are sensitive, and crop classification evaluation is fluid. The texture and colour of the leaves serve as the main visual element. Therefore, it is essential to classify leaf diseases in order to assess agricultural goods, raise their market worth, and maintain high standards of quality. Identifying the infections and taking additional measures to prevent their spread is also beneficial. When identification and classification are carried out physically, the procedure will be too extremely slow because they call for error-prone and inconvenient expert assistance. The classification of labour according to factors like colour, size, etc. will result in speedier and error-free work if these quality methods are recorded into an automatic system using the right computer program.

Since farmers are the main users of our module, we wanted to design it in a way that

even a person with no programming knowledge could utilize it to learn about plant diseases and understand how to treat them. It suggested an approach to forecast leaf diseases. For each ailment, a distinct amount of images are acquired and categorized into input images and database images.

1.3 Convolution Neural network

A convolutional neural network (CNN) is a type of artificial neural network that is designed specifically to process pixel data and is used in image recognition and processing.

An input layer, an output layer, and a hidden layer comprising numerous convolutional layers, pooling layers, fully connected layers, and normalising layers comprise the layers of a CNN[5]. The elimination of constraints and improvement in efficiency for image processing results in a system that is substantially more effective and easy to train for image processing and natural language processing.

1.3.1 Convolution Layer

The foundation of a CNN is a convolutional layer. It has a number of filters (or kernels), the settings of which must be learned during training. Typically, the filters are smaller in size than the original image. After converging with the picture, each filter generates an activation map.

In each convolutional layer, indexed by l, a convolution operation and an additive bias will be applied to the input, for a feature map indexed by $f \in \{1, \ldots, f(l)\}$. So the output, $Y_i^{(l)}$, of the l^{th} layer for the i^{th} feature map, is derived from the output of the previous layer, $Y_i^{(l-1)}$, by:

$$Y_i^{(l)} = \phi\left(B_i^{(l)} + \sum_{j=1}^{f^{(l-1)}} K_{i,j}^{(l)} * Y_j^{(l-1)}\right), \tag{1.1}$$

where ϕ is Rectified Linear Unit (ReLU) activation function, $B_i^{(l)}$ is a bias matrix, $K_{i,j}^{(l)}$ is the filter of size $2w_k + 1 \times 2h_k + 1$.

1.3.2 Pooling Layer

The pooling layer is crucial to the pre-processing of an image. When the images are too big, the pooling layer minimises the number of parameters. The image obtained from the prior layers is "downscaled" in the pooling process.

1.3.3 Fully Connected Layer

The input from the other layers will be flattened into a vector and transmitted to the Fully connected layer. The output will be changed by the network into the desired number of classes.

Instead of using flattening on the input of Fully connected layers, Global average pooling layer is preferred. Global average pooling (GAP) layers tries minimize overfitting by reducing the total number of parameters in the model. GAP layers reduce each feature map to a single number by simply taking the average of all values in the feature map. So, for every i^{th} feature of layer l, $Y_i^{(l)}$ we get an output for global average pooling layer which is denoted using f_i:

$$f_i = GP\left(Y_i^{(l)}\right) = average\left(Y_i^{(l)}\right) \tag{1.2}$$

1.3.4 Output Layer

The output layer is the final layer in the neural network where desired predictions are obtained. There is one output layer in a neural network that produces the desired final prediction. It has its own set of weights and biases that are applied before the final output is derived. The activation function for the output layer may be different than the hidden layers based on the problem. For example, Softmax activation is used to derive the final classes in a classification problem[6].

1.4 Thesis Organization

In this paper, Chapter 2 consists of background and related research regarding the problem.The predominent goal of Chapter 3 is to have a run down through the step involved and explain about the model architecture design. Chapter lays emphasis on various approaches, the results of these approaches and various predictions made through the web-app. Chapter 5 discusses about the conclusion drawn, advantages of the approach and possible future scope elements.

CHAPTER 2

LITERATURE SURVEY

2.1 Background Study

2.1.1 Plant leaf disease detection in agriculture

Plant diseases is one kind of natural disasters that affect the normal growth of plants and even cause plant death during the whole growth process of plants from seed development to seedling and to seedling growth.

Plant diseases detection is a very important research content in the field of machine vision. It is a technology that uses machine vision equipment to acquire images to judge whether there are diseases in the collected plant images. At present, machine vision-based plant diseases detection equipment has been initially applied in agriculture and has replaced the traditional naked eye identification to some extent.

For traditional machine vision-based plant diseases detection method, conventional image processing algorithms or manual design of features plus classifiers are often used. This kind of method usually makes use of the different properties of plant diseases to design the imaging scheme and chooses appropriate light source and shooting angle, which is helpful to obtain images with uniform illumination. Although carefully constructed imaging schemes can greatly reduce the difficulty of classical algorithm design, but also increase the application cost. At the same time, under natural environment, it is often unrealistic to expect the classical algorithms designed to completely eliminate the impact of scene changes on the recognition results.

In real complex natural environment, plant diseases detection is faced with many challenges, such as small difference between the lesion area and the background, low contrast, large variations in the scale of the lesion area and various types, and a lot of noise in the lesion image. Also, there are a lot of disturbances when collecting plant

diseases images under natural light conditions. At this time, the traditional classical methods often appear helpless, and it is difficult to achieve better detection results.

In recent years, with the successful application of deep learning model represented by convolutional neural network (CNN) in Plant leaf disease detection is being prominently used in agricultural as well as domestic use.

2.1.2 DenseNet

DenseNet, a convolutional neural network variant, has dense connections between layers[7], via Dense Blocks to directly link all layers (with corresponding feature-map sizes) as shown in Figure 2.1. Each layer takes extra inputs from all previous layers and broadcasts its own feature-maps to all following layers to keep the system feed-forward.

2.1.2.1 DenseNet Architecture

When the composite function operation is used, an output from the first layer becomes an input for the second. The convolution layer, pooling layer, batch normalisation, and non-linear activation layer comprise this composite procedure. As a result of these interconnections, the network has L(L+1)/2 direct connections. L represents the number of architectural layers.

The DenseNet is divided into DenseBlocks, each of which has the same dimensions but differs in a number of filters. It is a crucial step in CNN that Transition Layer applies batch normalisation via downsampling.Global average pooling is performed in the Classification Layer, and a softmax classifier is added after the last dense block.

After applying a composite of operations, traditional feed-forward neural networks connect the output of the layer to the next layer. DenseNets concatenate the layer's output feature maps with the incoming feature maps rather than summing them.

The equation for this is,

$$x_l = H_l([x_0, x_1, \ldots, x_{l-1}]) \tag{2.1}$$

Because we are concatenating feature maps, the channel dimension grows with each layer. If we programme H_l to generate k feature maps every time, we can generalise to the $l - th$ layer:

$$k_l = k_0 + k * (l - 1) \tag{2.2}$$

The growth rate is represented by the hyperparameter k. The growth rate governs how

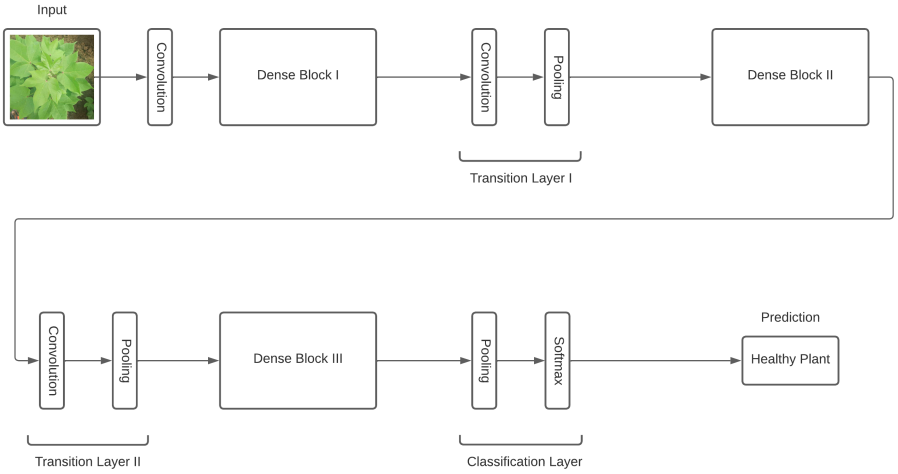


Figure 2.1: DenseNet Architecture with three DenseBlocks[2]

much data is added to the network at each layer.

2.1.2.1.1 Dense Block In convolutional neural networks, a Dense Block is a module that connects all layers directly(with corresponding feature-map sizes). It was first suggested as a component of the DenseNet design.
Dense Block consist of,

- conv 1x1
- conv 3x3

2.1.2.1.2 Transition Layer DenseNet concatenates all the feature maps.Concatenating different-sized feature maps is not practical (although some resizing may work). Each layer's feature maps are thus the same size in each dense block.

However, CNN relies heavily on downsampling. Between two dense blocks, transition layers ensure this function.
Transition Layer consist of,

- Batch Normalization
- RelU
- conv 1x1
- pool 2x2

2.1.2.1.3 Classification Layer It is the last layer in DenseNet architecture through which the prediction is made.
Classification Layer consist of,

- 7×7 global average pool
- 1000D fully-connected Softmax Classifier

2.1.3 EfficientNetV2

Accuracy, parameters, FLOPs, and inference time are a few bottlenecks that an efficient CNN architecture must perfectly balance. Numerous architectures, depending on their demands, concentrate on various bottlenecks. Inference speed is optimized through RegNet and ResNet but these methods often come with expensive overhead on large parameter size. EfficientNet[8], for instance, aims to increase accuracy with lesser parameters.

EfficientNetV2 [9] architecture trains CNN efficiently with respect to parameters and increases training speed by combining training-aware neural architecture search (NAS) and scaling to improve both training speed and parameter efficiency. In the Table 2.1, EfficientNetV2-S, a base variant of EffiecientNetV2, is used for comparison.

EfficientNetV2 extensively uses both MBConv and the newly added fused-MBConv in the early layers unlike EfficienNet which only uses MBConv layers. It prefers smaller 3x3 kernel sizes, but it adds more layers to compensate the reduced receptive field resulted from the smaller kernel size. Lastly, EfficientNetV2 completely removes the last stride-1 stage in the original EfficientNet, perhaps due to its large parameter size and memory access overhead.

EfficientNetV2 is built on the MBConv module with depthwise convolution and the Fused-MBConv module without depthwise convolution, as shown in Figure 2.2. Despite the fact that there are fewer parameters, depthwise convolution is slower because it does not use modern accelerators.

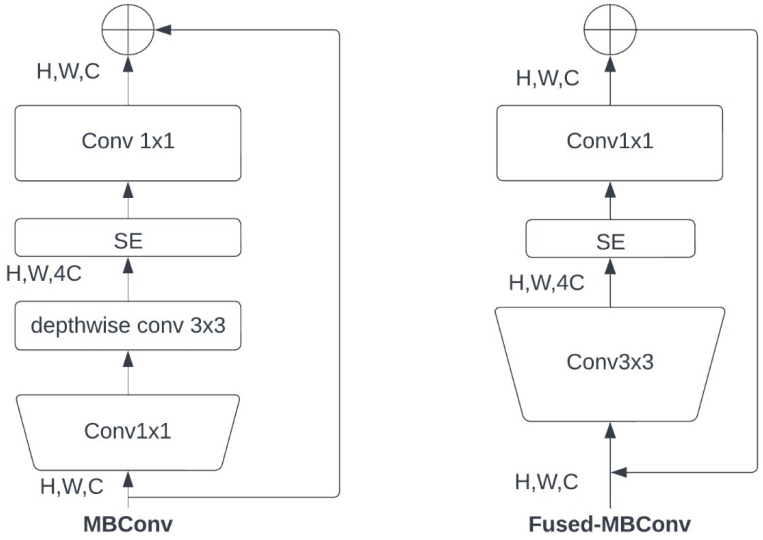


Figure 2.2: Architecture of MBConv and Fused MBConv blocks

Table 2.1: Comparison of accuracy, parameters, FLOPs, Infer-time and train time of EfficientNet-B4,B5 and ResNet-101 against EfficientNetV2-S

	EfficientNet-B4	EfficientNet-B5	ResNet-101	EfficientNetV2-S
Top-1 Accuracy	82.9%	83.7%	83.0%	83.9%
Top-5 Accuracy	96.4%	96.7%	92.8%	96.7%
Parameters	19.5M	30M	48M	22M
Infer-time(ms)	30	60	31	24
FLOPs	4.2B	10B	13B	8.8B

2.2 Key Related Research

Table 2.2: Related Work

	Author	Title	Year	Publisher	Work
1	Sunil C.K. Jaidhar C.D.[10]	Cardamom Plant Disease Detection Approach Using EfficientNetV2	2022	IEEE	Cardamom plant disease detection using U^2-Net and EfficientNetV2
2	Lili Li, Shujuan Zhang, Bin Wang[4]	Plant Disease Detection and Classification by Deep Learning	2021	IEEE	Conducted symptom-wise identification of four cucumber diseases using a deep CNN.
3	Wajeeha Shakeel, Mudassar Ahmad, Nasir Mahmood[11]	Early Detection of Cercospora Cotton Plant Disease by Using Machine Learning Technique	2020	IEEE	Detecting Cercospora cotton plant disease at early stage by using hybrid feature extraction.
4	Pratiti Saha Dr. Nachappa MN [12]	Cotton Plant Disease Prediction Using Deep Learning	2022	IJRASET	The CNN model is used to predict whether or not the plant is infected.
5	Mr.V Suresh, D Gopinath, M Hemavarthini, K Jayanthan, Mohana Krishnan[13]	Plant Disease Detection using Image Processing	2020	IJERT	Plant Disease Detection using ANN
6	Dr. S. Ramacharan [14]	A 3-Stage Method for Disease Detection of Cotton Plant Leaf using Deep Learning CNN Algorithm	2021	IJRASET	Convolutional Neural Network to identify cotton plant leaf diseases.
7	Chit Su Hlaing, SaiMaung MaungZaw[15]	Plant Diseases Recognition for Smart Farming	2017	arxiv.org	The benefits of the GP distribution model and its use in plant disease categorization
8	Shubham Bavaskar, Vaibhav Ghodak, Gayatri Deshmuk, Pranav Chillawa, Atul Kathole[16]	Image Classification Using Deep Learning Algorithms for Cotton Crop Disease Detection	2022	IEEE	Using CNN for disease detection purposes using image classification.
9	Ch. Usha Kumari, S. Jeevan Prasad, G. Mounika[6]	Leaf DiseaseDetection: Feature Extraction with K-means clustering and Classification with ANN	2019	IEEE	The K-means clustering approach is utilised to distinguish between the stained and healthy leaf regions.

Sunil C.K. , Jaidhar C.D.[10] built a model for Cardamom Disease Detection using EfficientNetV2 and using U^2-Net algorithm to remove the unwanted background in the image. The accuracy produced was 98.26.%

Lili Li, Shujuan Zhang, Bin Wang[4] have introduced the basic knowledge of deep learning and presented a comprehensive review of recent research work done in plant leaf disease recognition using deep learning.

Wajeeha Shakeel, Mudassar Ahmad, Nasir Mahmood[11] proposed effective and efficient technique for detecting Cercospora cotton plant disease at early stage by using hybrid feature extraction. The proposed technique emphasized on detecting and extracting the manipulated regions with higher accuracy and robustness.

Pratiti Saha, Dr. Nachappa MN [12] tend to gift a review of the utilization of neural network models within the field of plant disease detection.

V Suresh, D Gopinath, M Hemavarthini, K Jayanthan, Mohana Krishnan[13] used a CNN model to train the dataset and produced an accuracy of 90.34%.

Dr. S. Ramacharan[14] built a model for Disease Detection of Cotton Plant Leaf using ResNet-50 has produced an accuracy of 96.6%.

Chit Su Hlaing, SaiMaung MaungZaw[15] shown the advantages of GP distribution model for SIFT descriptor and successfully applied in plant disease classification.

Shubham Bavaskar, Vaibhav Ghodak, Gayatri Deshmuk, Pranav Chillawa, Atul Kathole[16] used four distinct CNN models to create a plant disease detection models based on Deep Learning and Convolutional Neural Network. The four models were:CNN, Inception V3, ResNet50, ResNet152 V2. The accuracies are 95.35%, 93.31%, 65.12%, 98.26% respectively.

Ch. Usha Kumari, S. Jeevan Prasad, G. Mounika[6], the leaf disease detection is done using neural network classifier.

2.3 Research Gaps

Research[10] on cardamom disease detection, EfficientNetV2 Model was trained on the dataset. Before training, U^2-Net algorithm was used to remove the unnecessary background of the images which in-turn results in loss of some features when this processed image is used for training. The accuracy of the proposed model is 98.26%.

Research[14] on disease detection of cotton plant leaf, ResNet-50 was used and it produced an accuracy of 96.6%. The CNN model used, ResNet-50 is an old CNN architecture and parametrically inefficient. The reseach also does not have any emphasis on vanishing gradient problem which is prominent in ResNets'.

Research[16] on plant disease detection using deep learning, multiple CNN architectures have been proposed. ResNet152 V2 was standing out of then with an accuracy of 98.26%. The reseach also does not have any emphasis on vanishing gradient problem which is prominent in ResNets'.

Research[13] on plant disease detection using CNN, a CNN architecture was developed and is trained on the dataset. The developed CNN architecture is unable to explore the deep layers of the images.

Research[15] used 10-Fold cross validation with SVM classifiers are applied to show that our experiment has no data bias and exclude theoretically derived values which is not experimental data.

Research[6] used very less samples of each disease so the accuracy was only 92.5%.

2.4 Objectives

- Classification of the cotton plant disease is the project's main goal.
- Using different deep learning models and training them on the leaves and plant dataset to distinguish the photographs of diseased and healthy plants.
- Comparing the trained models and using the best among them.
- Creating an ease of use web-app for prediction using the best model.

CHAPTER 3

METHODOLOGY

3.1 System Architecture

The proposed system architecture predicts if a cotton plant or leaf is infected or not by using a dataset of cotton plants and leaves and preprocessing algorithms to prepare the data for further steps. The model is then trained using CNN once the features are retrieved. The website then uses the trained model to predict the input photos of cotton plants and leaves.
The system's structure is depicted in the Figure 3.1.

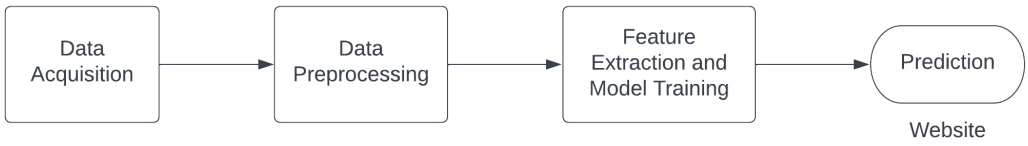


Figure 3.1: System Architecture of Module

3.1.1 Data Acquisition

The cotton plant and leaf images used in this project were obtained from the Kaggle dataset named Cotton Disease Dataset. Cotton plant and Leaf images were from the aforementioned Kaggle dataset and were downloaded in JPEG format.

3.1.2 Data Preprocessing

These image paths are stored in a dataset pathimages in a column and labels of the image and then we iterate a loop over the dataset and classify these images into their respective directories.

3.1.2.1 BGR-RGB

When we're using CNN, it generates an RGB image, which measure 224*224. Because the majority of the photos are BGR, we use the cv2 module[17] to convert them to RGB.

3.1.2.2 Scaling

Each pixel's value range from 0 to 255. So, by dividing each pixel value by 255, it can be scaled down from 0 to 1. For EfficientNetV2, by default input preprocessing is included as a part of the model (as a Rescaling layer)[11].

3.1.2.3 Splitting of Data into Train, Test, validation

Total data in the datasets are divided into Test, Train data accordingly as shown in Table 3.1.

- Train Data: 80%
- Validation Data: 10%
- Test Data: 10%

The dataset is a collection of burned cotton leaf, diseased cotton plant, healthy cotton leaf, healthy cotton plants as shown in Figure 3.2 respectively[18].

Table 3.1: Train and Test Dataset

Subjects	Train	Test
Burned cotton leaf	288	3
Diseased cotton plant	815	5
Healthy cotton leaf	427	5
Healthy cotton plant	421	5

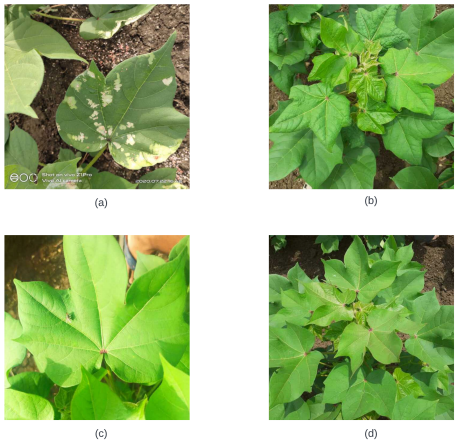


Figure 3.2: (a) Burned Cotton Leaf, (b) Diseased Cotton Plant, (c) Fresh Cotton Leaf, (d) Fresh cotton plant.

3.1.3 Feature Extraction and Model Training

In this stage, features are extracted by applying a pre-trained CNN to the train data and a model is created.

3.1.4 Prediction

In this stage, a web-app is built and loaded with the trained model. When the web-app receives the input image, the model is utilised to predict and is shown alongside all other relevant information.
The web-app is built with HTML, CSS, Javascript, and Flask.
Flask is a web framework[19]; it is a Python package that allows you to easily build web applications. Its core is small and easy to expand. It is a microframework that lacks an object relationship manager and other similar features.

3.2 Model Structures

3.2.1 CNN model

A CNN model is developed using different layers to and is trianed on the dataset. The layers used are convolution layer, max pooling and fully connected layers. Flatten is used before fully connected layers. Softmax activation function is used in the last dense layer. The architecture of the developed CNN model is shown in the Table 3.2.

Table 3.2: Developed CNN Architecture.

Stage	Operator	Channels
1	Conv5x5	32
2	Max Pooling 2x2	-
3	Conv3x3	64
4	Max Pooling 2x2	-
5	Conv3x3	96
6	Max Pooling 2x2	-
7	Conv3x3	96
8	Max Pooling 2x2	-
9	Flatten	-
10	Fully connected, softmax	Dense-512 Dense-7

3.2.2 InceptionV3

The InceptionV3[1] is a deep learning model based on Convolutional Neural Networks, which is used for image classification. The InceptionV3 is a superior version of the basic model InceptionV1. The InceptionV3 model is made up of 42 layers which is a bit higher than the previous inception V1 and V2 models. But the efficiency of this model is really impressive. InceptionV3 architecture is shown in the Table 3.3 along with Figure 3.3.

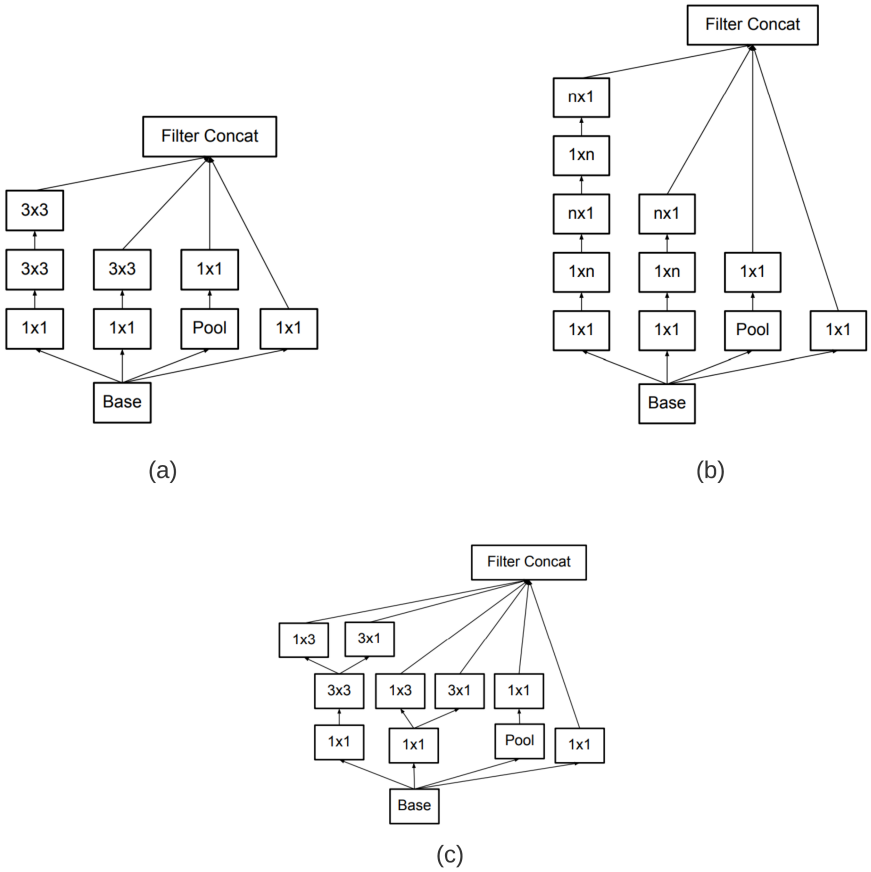


Figure 3.3: (a) Module-1, (b) Module-2 and (c) Module-3[1].

Table 3.3: Inception V3 Architecture[1].

Operator	**Stride**	**Input Size**
Conv3x3	2	299×299×3
Conv3x3	1	149×149×32
Conv padded 3x3	1	147×147×32
Pool 3x3	2	147×147×64
Conv3x3	1	73×73×64
Conv3x3	2	71×71×80
Conv3x3	1	35×35×192
3 × Inception	Module-1	35×35×288
5 × Inception	Module-2	17×17×768
2 × Inception	Module-3	8×8×1280
Pool 8x8	-	8 × 8 × 2048
Linear	Logits	1 × 1 × 2048
Softmax	Classifier	1 × 1 × 1000

3.2.3 DenseNet201

A convolutional neural network with 201 layers is called DenseNet-201 as shown in Table 3.4[17]. It was developed primarily to overcome the decreased accuracy brought on by high-level neural networks' vanishing gradient. In plainer terms, the information disappears before getting there because of the lengthier journey between the input layer and the output layer. Each layer in DenseNet receives extra inputs from all levels that came before it and transmits its own feature-maps[20] to all layers that came after it. You utilise concatenation. Each layer receives "collective knowledge" from the levels that came before it.

Each layer receives feature maps from all previous layers, resulting in a more compact and thin network with fewer channels. The growth rate k is the additional number of channels for each layer.

Consequently, it has improved memory and processing efficiency.

Table 3.4: DenseNet201 architecture for ImageNet[2].

Layers	Output Size	DenseNet-201
Convolution	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Dense Block I	56×56	$\begin{bmatrix} 1 \times 1 \text{ con } v \\ 3 \times 3 \text{ con } v \end{bmatrix} \times 6$
Transition Layer I	56×56	1×1 conv
	28×28	2×2 average pool, stride 2
Dense Block II	28×28	$\begin{bmatrix} 1 \times 1 \text{ con } v \\ 3 \times 3 \text{ con } v \end{bmatrix} \times 12$
Transition Layer II	28×28	1×1 conv
	14×14	2×2 average pool, stride 2
Dense Block III	14×14	$\begin{bmatrix} 1 \times 1 \text{ con } v \\ 3 \times 3 \text{ con } v \end{bmatrix} \times 64$
Transition Layer III	14×14	1×1 conv
	7×7	2×2 average pool, stride 2
Dense Block IV	7×7	$\begin{bmatrix} 1 \times 1 \text{ con } v \\ 3 \times 3 \text{ con } v \end{bmatrix} \times 48$
Classification Layer	1×1	7×7 global average pool
		1000D fully-connected, softmax

3.2.4 EfficientNetV2-B2

EfficientNetV2-B2 a varient of EfficientNetV2 is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth,width and resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients.

The architecture of model EfficientNetV2-B2 which are shown in Table 3.5 can be developed by downscaling the architecture of EfficientNetV2-S [9].

Table 3.5: EfficientNetV2-B2 architecture.

Stage	Operator	Stride	Channels	Layers
1	Conv3x3	2	32	1
2	Fused-MBConv	1	16	2
3		2	32	3
4		2	56	3
5	MBConv	2	104	4
6		1	120	6
7		2	208	9
8	Conv1x1	1	1408	1
9	Average Pooling Layer	-	1408	1

3.3 Summary

In this chapter, the implementation of proposed system is explained which done through several steps namely data acquisition in which dataset is obtained from external source, then this data is pre-processed to make it optimal for training of the model.Various pre-processing techniques such as BGR-RGB conversion, scaling are explained. Next, feature extraction and training EfficientNetV2-B2 model on the dataset available, followed by loading of trained model into the developed web-app for prediction of the input images is explained. Then the model architecture of EfficientNetV2-B2 is explained.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Experiment Design

After deciding on a project, I conducted some online research. Considering that I am familiar with keras and tensor flow applications. Therefore, I used the CNN model.fitgenrator[14] to train the dataset.The train data set is used to train the suggested model. While training, various parameters are tuned, including learning rate, number of epochs. The accuracy and the loss are the performance indicators used to assess the model. Our key goal was to improve accuracy while also reducing loss. TensorFlow Framework was used. The dataset was obtained from Kaggle.

Note: These results are for 432 fresh cotton leaf, 426 fresh cotton plant, 291 burned cotton leaf, 820 diseased cotton plant so, these metrics may change if the no.of input images change.

4.1.1 Experiment 1

CNN model is developed based on the architecture showed in Table 3.2 and trained on the dataset available.

Parameters:
No. of epochs=100
Leaning rate = 0.01

The accuracy and loss of the CNN developed is 96.78% and 0.0974 respectively. The loss and accuracy metrics are shown in the Figure 4.1.

The CNN developed seems to give us a good result but not the best and the model is slightly over-fitting.

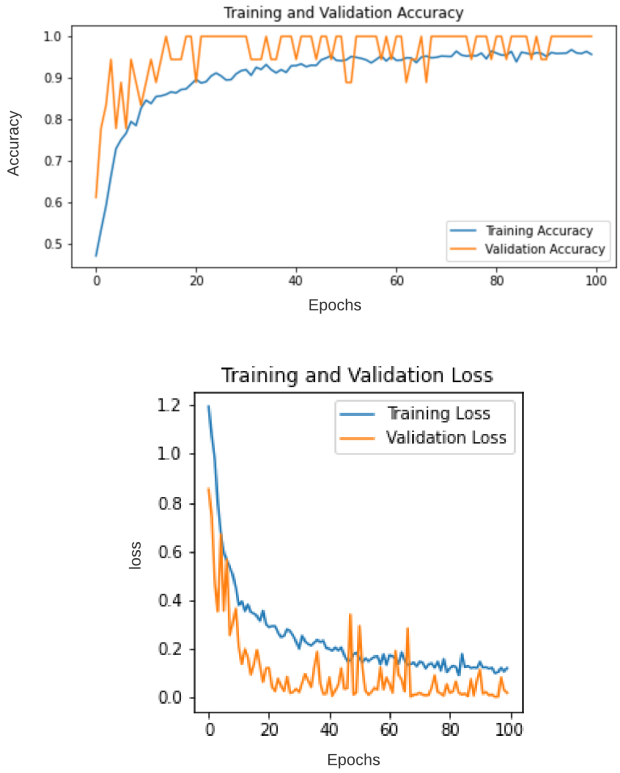


Figure 4.1: CNN accuracy and loss graphs

4.1.2 Experiment 2

A pre-trained CNN, InceptionV3 Model is used to train the dataset.

Parameters:
No. of epochs=100
Leaning rate = 0.01

The accuracy and loss of the InceptionV3 Model used is 99.08% and 0.2028 respectively. The loss and accuracy metrics are shown in the Figure 4.2.

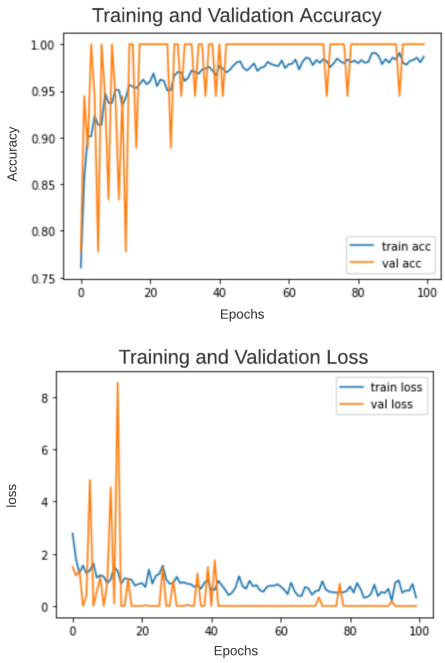


Figure 4.2: InceptionV3 accuracy and loss graphs

The CNN model used, InceptionV3 has great accuracy but the loss is comparatively more and the model is highly over-fitting.

4.1.3 Experiment 3

A pre-trained CNN, DenseNet-201 Model is used to train the dataset.

Parameters:
No. of epochs=30
Leaning rate = 0.01

The accuracy and loss of the InceptionV3 Model used is 98.87% and 0.0539 respectively. The loss and accuracy metrics are shown in the Figure 4.3.

The CNN model used, DenseNet-201 has great accuracy and suited very well but there is scope for improvement is model training time and model size by using newer CNN models.

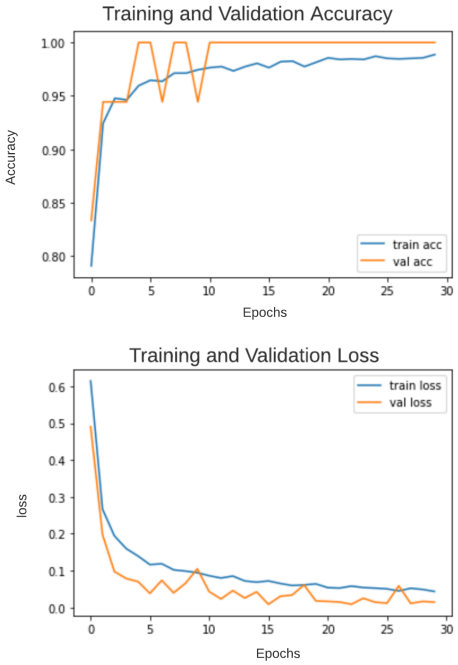


Figure 4.3: DenseNet-201 accuracy and loss graphs

4.1.4 Experiment 4

A pre-trained CNN, EfficientNetV2-B2 Model is used to train the dataset.

Parameters:
No. of epochs=30
Leaning rate = 0.01

The accuracy and loss of the EfficientNetV2-B2 Model used is 99.28% and 0.0315 respectively. The loss and accuracy metrics are shown in the Figure 4.4.

The CNN model used, EfficientNetV2-B2 has best accuracy among all the models used and suited very well.

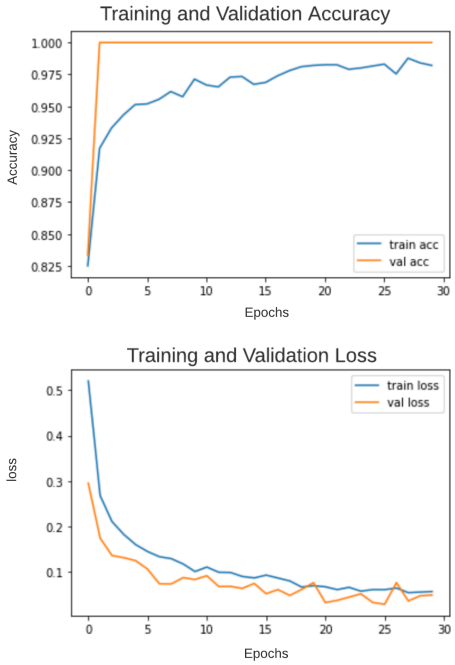


Figure 4.4: EfficientNetV2-B2 accuracy and loss graphs

4.1.5 Experimental Conclusion

In the experiments performed above by training different CNN models, developed CNN and InceptionV3 are over-fitting models and DenseNet201 and EfficientNetV2-B2 are optimal models. Out of these two EfficientNetV2-B2 provides us with better accuracy. The comparison of the models is done in the Table 4.1. .

Table 4.1: Comparision of Different Models.

Architecture	Accuracy	Loss
CNN	96.78%	0.0974
InceptionV3	99.08%	0.2028
DenseNet-201	98.87%	0.0539
EfficientNetV2-B2	99.28%	0.0315

4.2 Web Application Design

4.2.1 Flask app Prediction Structure

Using Flask app for prediction:

- Save the training model in 'model.h5' with keras and pass it into the flask app.
- Predict the model using if-else constraints as shown in the Table 4.2.

Table 4.2: Prediction of Dataset

Subjects	Perd
Burned cotton leaf	0
Diseased cotton plant	1
Healthy cotton leaf	2
Healthy cotton plant	3

- Using app route it will create web application(through html) and deploy the image of plant it will give you the result of plant or leaf whether it has disease or not.

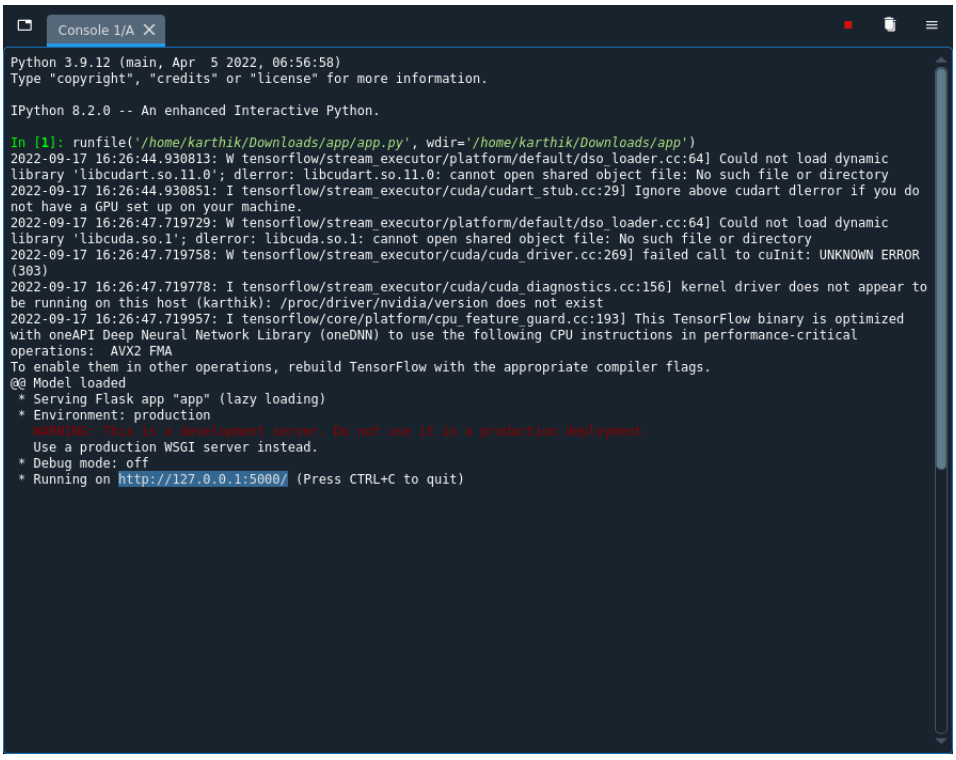


Figure 4.5: Compiling Flask App

- Copy this local url as shown in Figure 4.5 and paste it in the browser to launch the web-app.

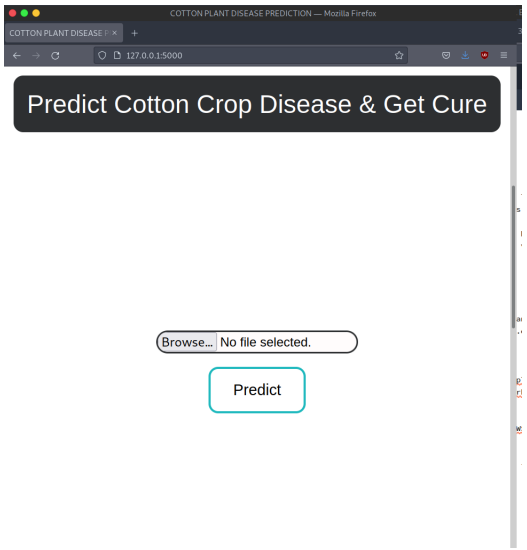


Figure 4.6: Home Page

- Now upload the image in the given dialoguebox as shown in Figure 4.6 to make the prediction.

4.2.2 Results

When images of plant or leaf is given to the flask app through input dialogue box, the model is utilized to predict whether the plant is diseased or not.

4.2.2.1 Healthy Plant Prections

These are the healthy plant predictions done through the web-app. Healthy plant predictions are shown in the Figure 4.7.

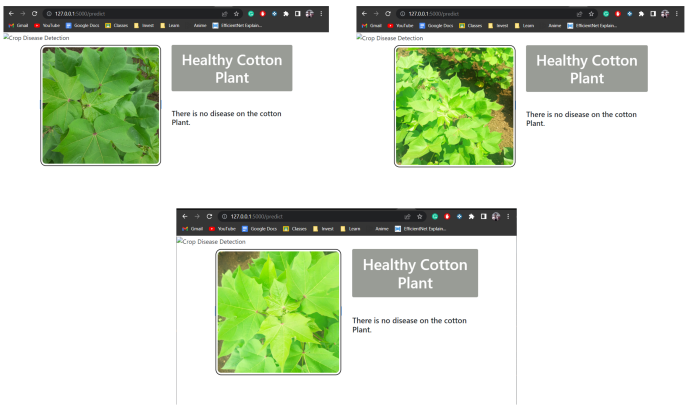


Figure 4.7: Healthy Cotton Plant Prediction

4.2.2.2 Healthy Leaf Prections

These are the healthy leaf predictions done through the web-app. Healthy leaf predictions are shown in the Figure 4.8.

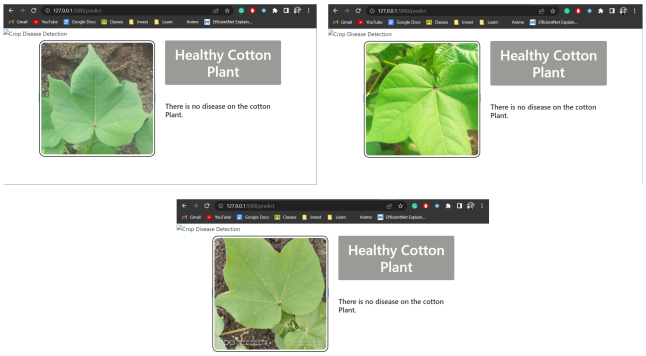


Figure 4.8: Healthy Cotton Leaf Prediction

4.2.2.3 Burned Leaf Prections

These are the burned leaf predictions done through the web-app. Burned leaf predictions are shown in the Figure 4.9.

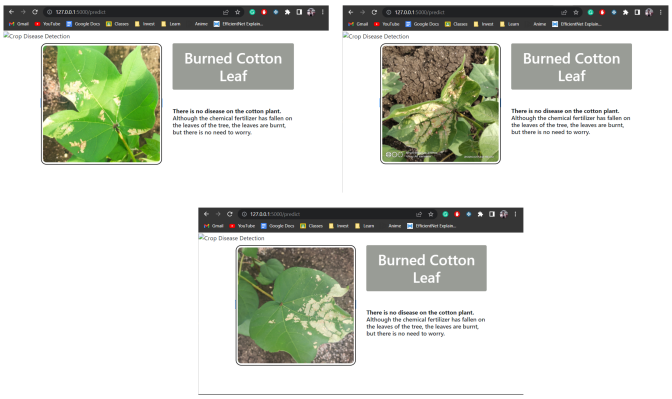


Figure 4.9: Burned Cotton Leaf Prediction

4.2.2.4 Diseased Plant Prections

These are the Diseased plant predictions done through the web-app and it also provides with additional infomation on the disease and it cure. Diseased plant predictions are shown in the Figure 4.10.

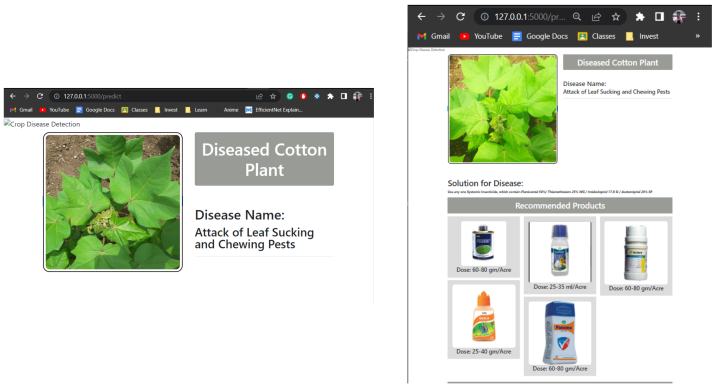


Figure 4.10: Diseased Cotton Plant Prediction

4.3 Summary

In this chapter, various CNN models are experimented to find the optimal one. Among developed CNN, InceptionV3,DenseNet-201 and EfficientNetV2-B2, EfficientNetV2-B2 produced better accuracy then the other models used. The accuracy of EfficientNetV2-B2 is 99.28%. Predictions of plant and leaf images is also discussed.

CHAPTER 5

CONCLUSION

This report provides a critical analysis for CNN-Architectures, proposed originally for image analysis. These CNN-Architectures are used to differentiate disease vs healthy based on plant and leaf images. The CNN architectures used are a developed CNN, InceptionV3, DenseNet-201 and EfficientNetV2-B2.

EfficientNetV2-B2 which gave us best accuracy by giving epochs 30 among all other proposed architectures as shown in the Table 4.1 is finalized and loaded in .h5 file format and passed it to the flask app where the prediction is done. With the help from these techniques we implemented neural network based on plant leaf disease detection and it takes less amount of time to for detection

Deep learning system with more layers for the most accurate and acceptable results. In comparison to manual testing, automated testing takes less time and costs less. Web-app can run on low-end devices and doesn't need internet connectivity.It can also be applied to classify various plants and leaves.

5.1 Future Scope

- One website app for several plant species.
- Removing other unwanted noise from the image, such as text markings on plant and leaf images, to improve clarity.
- The accuracy of the model will be enhanced by the collection of more images of plants and leaves.

REFERENCES

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.

[2] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] L. Li, S. Zhang, and B. Wang, "Plant disease detection and classification by deep learning—a review," *IEEE Access*, vol. 9, pp. 56683–56698, 2021.

[5] M. Türkoğlu and D. Hanbay, "Plant disease and pest detection using deep learning-based features," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 27, no. 3, pp. 1636–1651, 2019.

[6] C. U. Kumari, S. Jeevan Prasad, and G. Mounika, "Leaf disease detection: Feature extraction with k-means clustering and classification with ann," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 1095–1098, 2019.

[7] A. Mortensen, M. Dyrmann, H. Karstoft, R. Jørgensen, and R. Gislum, "others.(2016). semantic segmentation of mixed crops using deep convolutional neural network," in *CIGR-AgEng Conference, 26-29 June 2016, Aarhus, Denmark. Abstracts and Full papers*, pp. 1–6.

[8] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

[9] Tan and Q. Le, "Efficientnetv2: Smaller models and faster training," in *International Conference on Machine Learning*, pp. 10096–10106, PMLR, 2021.

[10] S. C. K., J. C. D., and N. Patil, "Cardamom plant disease detection approach using efficientnetv2," *IEEE Access*, vol. 10, pp. 789–804, 2022.

[11] W. Shakeel, M. Ahmad, and N. Mahmood, "Early detection of cercospora cotton plant disease by using machine learning technique," in *2020 30th International Conference on Computer Theory and Applications (ICCTA)*, pp. 44–48, 2020.

[12] P. Saha and M. Nachappa, "Cotton plant disease prediction using deep learning,"

[13] V. Suresh, M. Krishnan, M. Hemavarthini, and D. Jayanthan, "Plant disease detection using image processing," *International Journal of Engineering Research and*, vol. V9, 03 2020.

[14] S. Ramacharan, "A 3-stage method for disease detection of cotton plant leaf using deep learning cnn algorithm," 07 2021.

[15] C. S. Hlaing and S. M. M. Zaw, "Plant diseases recognition for smart farming using model-based statistical features," in *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, pp. 1–4, 2017.

[16] S. Bavaskar, V. Ghodake, G. Deshmukh, P. Chillawar, and A. Kathole, "Image classification using deep learning algorithms for cotton crop disease detection," in *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, pp. 1–8, IEEE, 2022.

[17] K. Yamini, K. S. Swetha, P. L. Prasanna, M. R. V. Swathi, and V. R. Maddumala, "Image colorization with deep convolutional open cv," *Journal of Engineering Science*, vol. 11, no. 4, pp. 533–543, 2020.

[18] N. Patil, S. Kelkar, M. Ranawat, and M. Vijayalakshmi, "Krushi sahyog: Plant disease identification and crop recommendation using artificial intelligence," in *2021 2nd International Conference for Emerging Technology (INCET)*, pp. 1–6, 2021.

[19] K. Relan, "Beginning with flask," in *Building REST APIs with Flask*, pp. 1–26, Springer, 2019.

[20] S. Barburiceanu, S. Meza, B. Orza, R. Malutan, and R. Terebes, "Convolutional neural networks for texture feature extraction. applications to leaf disease classification in precision agriculture," *IEEE Access*, vol. 9, pp. 160085–160103, 2021.