PLANT DISEASE DETECTION USING VGG-16 AND FLASK FRAMEWORK

Project report in partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology

In

COMPUTER SCIENCE & INFORMATION TECHNOLOGY

Submitted By

Biki Paul

Under the guidance of

Prof. (Dr.) Rajendrani Mukherjee

&

Prof. Arnab Chatterjee

Department of Computer Science & Engineering

UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA

University Area, Plot No. III – B/5, New Town, Action Area – III, Kolkata – 700160.

TABLE OF CONTENTS

ABSTRACT

Agriculture and modern farming is one of the fields where IoT and automation can have a great impact. Maintaining healthy plants and monitoring their environment in order to identify or detect diseases is essential in order to maintain a maximum crop yield. The implementation of current high rocketing technologies including artificial intelligence (AI), machine learning, and deep learning has proved to be extremely important in modern agriculture as a method of advanced image analysis domain. Artificial intelligence adds time efficiency and the possibility of identifying plant diseases, in addition to monitoring and controlling the environmental conditions in farms. Several studies showed that machine learning and deep learning technologies can detect plant diseases upon analyzing plant leaves with great accuracy and sensitivity. In this study, considering the worth of machine learning for disease detection, we present a convolutional neural network VGG-16 model to detect plant diseases, to allow farmers to make timely actions with respect to treatment without further delay. To carry this out, 15 different classes of plants diseases were chosen, where 54,000 plant leaf images (both diseased and healthy leaves) were acquired from the Plant Village dataset for training and testing. Based on the experimental results, the proposed model is able to achieve an accuracy of about 88.67% with the testing loss being only 0.4477. The proposed model provides a clear direction toward a deep learning-based plant disease detection to apply on a large scale in future.

Agriculture is the backbone of Indian government. Every human being has a requirement of a lot of production of crops to fulfill the needs of Indian government. Because of some diseases that we observe in this day to day life, a large amount of crop production is being decreased. There are various types of diseases on plant leaves and as well as for the crop, that causes problems in development of crops. Human eyes don't have the capacity to identify so strongly with our naked eye. It is too difficult to identify the plant diseases on leaves. The automatic disease detection system is used to automatically detect and identify the diseased part of the leaf images and it classifies plant leaf disease using image processing techniques. By gathering some of the leaves and training those leaves. We use this training data to train our data and then output will be predicted with optimum accuracy. For this we use the Flask framework. We upload the image into the website we have developed. Now the patterns of the uploaded image are compared with patterns available in the dataset, which is almost accurate, resulting in identification of the plant disease. At the starting stage, the disease can be easily identified. Proposed model helps to reduce efforts or hard work of farmers for monitoring big farms and related diseases to farms and crops.

1: INTRODUCTION

Agriculture has always been a basic human need ever since humans' existence as plants were a primary source of food. Even nowadays, agriculture is still considered an essential food resource and is the center of several aspects in humans' lives. As a matter of fact, agriculture serves as the pillar of economy in many countries regardless of their developmental stages. The various domains that show the importance of agriculture include the fact that agriculture is a main source of livelihood where approximately 70% of the population depends on plants and their cultivation for livelihood. This great percentage reflects on agriculture being the most important resource that can actually stand a chance in the face of the rapidly increasing population. One of the most critical challenges that face agriculture and affects it trade is plant diseases and how to timely detect them and deal with them to improve the health of crops. By definition, plant disease in a type of natural problems that occur in plants affecting their overall growth and might lead to plant death in extreme cases. Plant diseases can occur throughout the different stages of plant development including seed development, seedling, and seedling growth. When diseased, plants go through different mechanical, morphological, and biochemical changes. Truthfully, there are two main types of plant stress classified as biotic stress represented by living creatures that interact with plants in a way that negatively affects their growth such as bacteria, viruses, or fungi, or abiotic stress represented by the collection of non-living factors or the environmental factors. Fig. illustrates the collection of factors that contribute to plant diseases. Typically, the commonly used approach for farmers, scientists, and even breeders, to detect and identify plant disease was the manual inspection of plants. Of course, this process requires expertise and knowledge for the proper detection. With time, manual inspection became tiresome and time consuming and not as quite efficient especially when large amounts of plants needed to be inspected. Another factor that proves the inefficiency of manual inspection is the similar conditions that might be caused by different pathogens that might look alike in their effect on the plant.

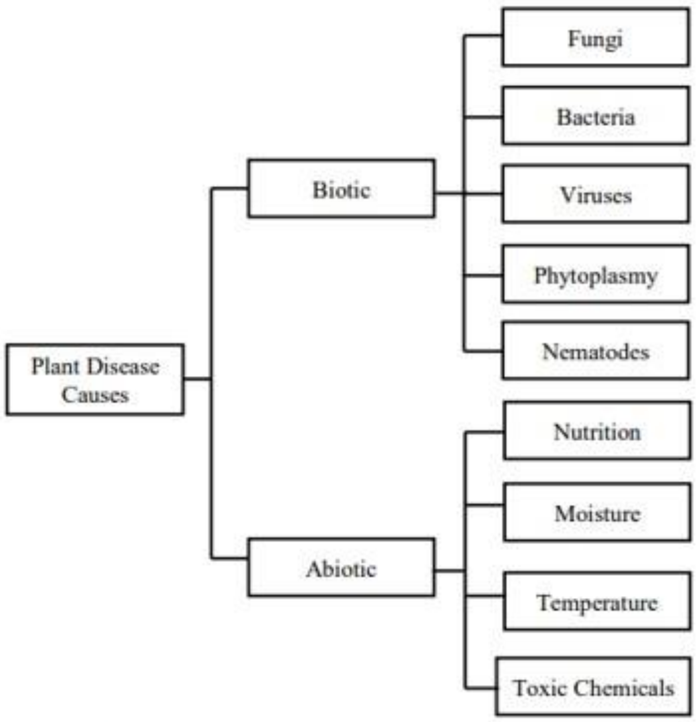


Fig: Plant Disease Causes as Biotic and Abiotic Factors

For this reason, humans needed a better suited technique that can deliver effective plant detection results in less time.

Due to technological advancement, this manual inspection can be replaced by automated systems using artificial intelligence and machine learning. These fields try to mimic human activities and embed human intelligence in machines. Artificial intelligence and machine learning have provided solution to many pattern recognition problems. License plate detection, optical character recognition, health monitoring systems, biometric systems, natural language processing, fingerprint recognition, face recognition, signature verification etc , all these systems are developed using artificial intelligence and machine learning. Deep learning, a subset field of machine learning is an improvement over previously existing machine learning algorithm. A tremendous amount of improvement in recognition results in every field is achieved using deep learning techniques. Deep learning is end-to-end learning where features are extracted automatically.

In traditional techniques, feature extraction is done manually, in contrary, deep learning automatically extract features using kernels. The most common deep learning architecture is convolutional neural network. This network uses convolved filters for feature extraction of different levels at different layers. Lower layers extract low level features such as gradients, color, points which are transformed into higher level features such as edges, corners etc. in higher layers of network. Convolutional neural networks take images as input and produces classes as their output in image classification tasks. Convolutional layers are mainly responsible for extracting features using convolved filters of different size. It also has pooling layers which help in dimensionality reduction. Pooling can be average pooling or max pooling depending upon the requirement. Softmax activation function is used in classification layer.

The most common CNN architectures are Alexnet, GoogleNet, VGG16, VGG19, Inception, ResNet etc. This work uses pretrained VGG16 network for disease detetction. Thus, VGG 16 is used for classification purpose. Tomato and potato images from plantvillage dataset are used in this work. The architecture of VGG 16 is explained in next section.

2: LITERATURE SURVEY

Artificial intelligence, computer vision and machine learning utilizations can greatly enhance the process of plant disease detection, and is already applied in multiple research papers. Such technologies are capable of not only detecting the presence of a disease, but it is also possible to determine its severity, and to classify exactly which kind of disease is present in a given plant sample.

Based on their depth, the plant disease detection methods can be divided into shallow architectures and deep architectures. Basic machine learning methods like Random Forest (RF), Support Vector Machine (SVM), Naïve Bayes (NB), and K-Nearest Neighbor (KNN) rely on specific design intended for features such that good features and patterns must be recognized. These specific features include hue saturation value (HSV), Histogram of Oriented gradient (HOG), linear binary pattern (LBP), and red-green-blue RGB color features. In machine learning, according to the complexity of the classifier, the more data is required for its training in order to achieve satisfactory results. A specific dataset is then created for the model, where the input images can be pre-processed before feature extraction can take place. Machine learning algorithms are capable of recognizing the changes in features upon comparison, and thus determining the output as diseased or healthy.

On the other hand, deep architectures like CNN (Convolutional Neural Networks) have also been heavily used in studies that are concerned with plant disease detection. These deep architectures differ from the shallow ones by not requiring hand-designed features since deep learning algorithms are able to learn the features themselves. Thus, deep learning approaches undergo three basic stages in detecting plant diseases classification, detection, and segmentation. After SVM machine learning approach was the most commonly used one for so long, approximately after the year 2015 CNN replaced SVM as the most popular ML technique for detection of diseases. CNN is considered state-of-the-art model that has been used in plant disease detection nowadays, especially since this task requires dealing with image data applications. CNN can execute tasks such as classification of images, segmentation, object detection, and recognition. In their structure, CNNs are made up of artificial neural networks where tens and even hundreds of layers are used. CNNs is made up of an input layer, several convolutional layers, along with pooling layers in between them, and finally full connection layer in addition to activation function layers, and output layer. There exist several forms of CNN architectures like VGG-16, Inception-V3, ResNet50, and AlexNet. However, CNN architectures need large data numbers which is often considered as a challenge. Since agriculture is essential there's a need to provide methods that enhance the agricultural methods in terms of planting, monitoring crop environment, detecting plant disease, and even harvesting. These important details led to significant research to be conducted and several papers to be published with the purpose of providing solutions to these agricultural challenges. This study proposes a model based on CNN, namely VGG-16 architecture in order to detect and classify a total of 19 plant conditions (several crop types and diseases) with the best accuracy possible. Our contributions in this study can be summed up as follows:

1) Updating a large dataset based on Plant Village. The dataset comprises 15 thousand images of plant leaves which are captured on the field, which means that they are photographed within their surroundings, and thus it is efficient in terms of not needing to isolate the plant for disease detection.

2) Implementing the proposed VGG-16 model which is an effective convolutional neural network architecture, and it achieves a great accuracy. Our proposed model is capable of scanning through thousands of leave images in order to identify if a plant has a certain type of disease based on its leaf image. The proposed model achieves a great accuracy of detection among 19 different disease classes in a short period of time, and it doesn't require a long time in training either.The arrangement of the current paper is as follows: section two is a description of some of the published similar studies about ML and DL in plant disease detection. Section three describes the proposed methodology including the dataset and the proposed model. Results are provided and compared theoretically with some of existing techniques in section four, while section five concludes the article by sharing the future research intentions.

2.1 Related work:

Eftekhar Hossain et al., [1] proposed a system for recognizing the plant leaf diseases with the appropriate classifier K-nearest neighbor (KNN). The features that were extracted through the images of diseased image were used to execute the classification. In the paper, the system KNN classifier classified the diseases commonly found in plants like bacterial blight, early blight, bacterial spot, leaf spot of various plant species. This method exhibited an accuracy of 96.76%.

Sammy et al., [2] proposed a CNN for classifying the disease types and in this paper the author used 9 different varieties of leaf diseases of tomato, grape, corn, apple and sugarcane. In this paper the training is conducted on the system for nearly about 50 epochs and they used 22 sizes of batch. In this model with the help of categorical cross entropy, Adam optimizer is conducted. Accuracy obtained is 96.5%.

Ch Usha Kumari et al., [3] developed a system that deploys the methods of K- Means clustering and Artificial Neural Network and performs computation of various features like Contrast, Correlation, Energy, Mean, Standard Deviation and Variance were performed. The major limitation was that accuracy of four different diseases was analyzed and the average accuracy is comparatively low.

Merecelin et al., [4] put forward a detailed study of identification of disease in plant (appleand tomato leaf) using the concepts of CNN. The model was trained on leaf image dataset containing 3663 images of apple and tomato plant leaf achieving an accuracy of 87%.

Jiayue et al., [5] performed the recognition of tomato fruits with disease, the technique called YOLOv2 CNN was used. YOLOv2 is based on regression model and uses a target detection algorithm, which exhibits fast detection speed and good accuracy. The MAP (mean Average Precision) was estimated to be around 97%. The major limitationof the paper was the need to perform different tuning if the images.

Robert G et al., [6] proposed a system using CNN to detect the type of tomato leaf diseases. This paper reported that the F-RCNN trained model obtained 80% confidence score, while accuracy of 95.75% was obtained by the Transfer Learning model. The automated image seizing method registered 91.67 % accuracy.

Halil et al., [7] proposed a deep learning model was deployed with two different deep learning network architectures, Alex Net and then Squeeze Net. The training and validation of these deep learning networks were performed on the Nvidia Jetson TX1. The Alex Net achieved an accuracy of 95.6% and on the other hand Squeeze Net model achieved an accuracy of 94.3%.

Sabrol et al., [8], the authors used an easy and uncomplicated mechanism is utilized for doing the process of classification of the different kind of diseases that occur in tomato leaves namely Early blight, Yellow curl virus, late blight, Mosaic virus, Bacterial spot and Healthy. The dataset contained 400 images clicked using a digital camera. Supervised learning method have been used for classification, where in the accuracy achieved was high, but decision tree has certain disadvantages – if instance of noisy data overfitting happens.

2.2 Technologies:

A. Deep Learning:

a category of machine learning algorithms which uses various layers to do the extraction of higher level from the raw input. Deep learning is a machine learning method that instruct a computer to do filtration of inputs across the layers Deep learning illustrates the way human brain does the filtration of information. Many deep learning techniques utilizes the neural network architectures. The term "deep" cite to the various hidden layers present inside neural network. In contrast to this conventional neural network that consists of 2-3 hidden layers, the deep neural networks can have as much as one hundred and fifty.

B. Convolutional Neural Network:

One variant of deep neural networks is called as convolutional neural networks (CNN). A CNN combines well-read features with input data, and then it uses 2D convolutional layers, and hence makes this architecture more suitable for processing 2D data, like images. CNNs abolish the demand for manual feature removal and extraction for the classification of the images. The CNN model of its own extracts features straight from images. The features that are extracted aren't pre-trained; they are well-read while the network is trained on few groups of images. The Convolutional Neural Network (CNN) model has numerous of layers which execute the processing of image in convolutional layers include- Input layer, Output Layer, Convo Layer, Fully, Soft-max layer, Connected layer, Pooling Layer.

C. Transfer Learning:

For transfer learning purposes, we used one of the state-of-the-art models i.e INCEPTIONv3 and used the weights of the model when it is trained on the IMAGENET dataset. We made the top layers of the model untrainable and added a few layers to the INCEPTIONv3 net according to the need of the dataset. The layers we added for our purpose is the flatten layer to flatten the output layer of the INCEPTIONv3, then two dense layers are added with different neurons but with the same activation function 'relu'. The last layer is same as that we used for our custom CNN with the same parameter and activation function. This model has only 21,802,784 parameters and the target size of our model is (150,150).

D. Visual Transformers:

In this approach, the image is divided into grids of 16*16. Each component is fed into a feedforward network to get its embedding and added with the embedding for its positions. These embeddings are used as tokens and then fed to another feedforward layer to get 3 tokens for query, key and value. These tokens are used to calculate attention. The output is then fed to a feedforward layer which could then be used as tokens for the next transformer layer. These repeated layers could capture semantic information without the need of a lot of parameters. Each feedforward and attention layer is preceded by a normalization layer and every repeated block of attention followed by feedforward has a residual connection with the previous one.

2 different sizes for transformer models were used:

Small Transformer Network (STN):

Model with token embedding representations having 256 dimensions, and feedforward layers also outputting 256 dimensions following each attention block. There are 8 attention blocks and input for each block is fed with 8 heads. This model has 3,499,046 parameters which is very less compared to the other models.

Large Transformer Network (LTN):

This model has token embedding representation of 128 dimensions, with feedforward layers following the attention blocks outputting 128 dimensions. It has 4 attention blocks and each block is fed 4 heads. This model has only 549,926 parameters which is the least of any model that we have tested in this study

E. VGG 16 Model:

VGG 16 Model is a CNN model used for Large-Scale Image. There are two tasks to be performed for best recognition of plant diseases. The first is to detect objects within an image coming from several classes, which is called object localization. The second is to classify images, each labelled with one of several categories, which is called image classification. The CNN model has seven different layers. Each layer has certain information processed in them. Those seven layers are as follows: Input layer, Output Layer, Convolutional Layer, Fully, Soft-max layer, connected layer, Pooling Layer.

Input layer: It contains data in the form of image. The parameters include height, width, depth and color information of the image (RGB). Input size is fixed to 224 X 224 RGB image.

Convo layer: Convolutional layer is also called as feature extraction layer. This layer extracts the prominent features from the given collection of images using dot products of the image dimensions.

Pooling Layer: The pooling layer helps to reduce the computational power in order to process the data by decreasing (or) reducing the dimensions of the featured matrix obtained by using the dot products.

Fully connected layer: It comprises of loads, neurons and biases. It connects neurons from one convolutional layer to another.

Softmax Layer/ Logistic Layer: Softmax executes multi-classification. Logistic layer executes the binary classification. It determines the probability of the presence of a given object in the image. If the object is present in the image, then the probability is ‘1’otherwise it is ‘0’.

Activation Function- ReLU: It transforms the total weighted input through the node and puts it into the operation, activates the node. Rectified Linear Unit (ReLU) is an activation function used in the neural networks for convolutional operations.

2.3 Dataset:

• The dataset used in this study is an augmented version of the PlantVillage dataset containing 87.9k images derived from the 54k images of the original dataset. This dataset contains 15 classes of plant-disease pairs and is divided into 80 percent for training and 20 percent for validation. This dataset contains the images of the healthy plant leaves with their respective diseased leaves also. The resolution of each image is 256*256 pixels.

3: PROBLEM STATEMENT

Agriculture is a significant part of the Indian economy. The Indian agriculture sector employs about half of the country's workers. India is known as the world's largest democracy. Pulses, rice, wheat, spices, and spice products are produced in significant quantities. Farmer's economic development depends on the quality of the items they produce, which is dependent on plant growth and development. As a result, in the sphere of agriculture, disease detection in plants plays an important role. a supporting role Plants are particularly susceptible to illnesses that disrupt their growth, which can lead to death. This, in turn, has an impact on the farmer's ecology. Use this method to detect a plant disease at an early stage. It is advantageous to use an automatic illness detection technique. Plant diseases manifest themselves in various areas of the plant, such as the leaves. It takes a long time to manually detect plant illness using leaf photos. As a result, computer approaches must be developed to automate the process of disease identification and categorization using leaf photos.

Plant diseases are a major challenge for farmers, leading to significant crop losses and economic damage. Early detection and identification of plant diseases can help prevent their spread and reduce losses. In recent years, deep learning models have shown great promise in detecting and diagnosing plant diseases.

Plant disease detection is still a work in progress research topic, despite the challenges described in the problem statement. Over the years, various ways have been offered. Using pathogen vectors, a strategy of detecting and distinguishing plant diseases can be achieved in traditional systems. Algorithms for machines This approach has been used to diagnose sugar beet illnesses, with classification accuracy ranging from 65 percent to 90 percent depending on the kind and stage of the disease. Plant disease classification was performed with K-means as a clustering algorithm, again employing a leaf-based approach and using ANN as an automatic detection tool. ANN consists of ten hidden layers. With the example of a healthy leaf, the number of outings is 6, which is the number of classes represented by five disorders.

The VGG-16 model is a popular convolutional neural network architecture that has shown excellent performance in image classification tasks. This project aims to use the VGG-16 model to develop an accurate and reliable plant disease detection system.

The problem statement for this project is to develop a deep learning model that can accurately classify images of plants into different disease categories. The model will be trained on a large dataset of images of healthy and diseased plants, and will be tested on a separate dataset to evaluate its performance.

The specific objectives of this project are:

1. Collect and pre-process a large dataset of images of healthy and diseased plants.
2. Train a VGG-16 deep learning model on the dataset to classify plant images into different disease categories.
3. Evaluate the performance of the model on a separate test dataset, using metrics such as accuracy, precision, and recall.
4. Compare the performance of the VGG-16 model with other state-of-the-art deep learning models for plant disease detection. The ultimate goal of this project is to develop an accurate and reliable plant disease detection system that can help farmers detect and diagnose plant diseases early, and take appropriate action to prevent their spread and minimize crop losses.

4: PROPOSED SOLUTION

We have used plant village dataset to build this project. We have used only 4000 images to train the model.

VGG-16 is one of the most commonly used CNN architecture, especially since it works well with the ImageNet, which is large project utilized for visual object recognition procedures and it is considered one of the best models to be proposed so far due to its extreme usefulness in the image classification's field in the deep learning domain. Initially, this model was created by Karen Simonyan and Andrew Zisserman in 2014, where they developed in during their work in Oxford University titled "Very Deep Convolutional Networks for Large-Scale Image Recognition". In fact, "V" means Visual," G" Geometry while "G" stands for research group who contributed in the development of this Convolutional Neural Network model, whereas the number 16 refers to the neural network layer's number This architecture is one of the top 5 models in terms of performance achievement in the ImageNet dataset, where its accuracy reached 88.67%. As an approach for the AlexNet enhancement, this architecture was submitted to ImageNet. Large Scale Visual Recognition Challenge (ILSVRC), where this model has replaced the large kernel-sized filters of numbers 11 and 5 in both first and second convolutional layer, respectively by a multiple three × three kernel-sized filters consecutively.

4.1 VGG-16 Architecture and Training Procedure:

The training Procedure is made up of three consecutive steps as shown:

- Preparing the images.
- Classifying the photos.
- Printing the decision. Image Processing:

The input of the convents is 224 × 224 RGB image with a fixed size where the value of each pixel is subtracted from the RGB mean value of the training image.

Classifying the data: The proposed model is made up of thirteen convolution layers, two batch normalization layers, along with five max-pooling layers and three full connection layers. The processed image passes through several convolutional layers that contains filters that are characterized by a receptive field of size 3 × 3 for capturing the notions of left and right, up and down along with the center. Despite its small size of the mentioned filter, this filter is accompanied by the same efficiency as that of a receptive field of size 7 × 7 due to its deep characteristics such as including more nonlinearities and lesser parameters. In addition to that, a 1x1 convolution filter was used as an input channel's linear transformation in a certain configuration. On the other hand, both spatial padding and the convolution stride are fixed to 1 pixel for 3 × 3 convolutional layers, in which the spatial resolution's preservation becomes easy to occur. Also, spatial pooling is easier in case of a five max-pooling layers' addition after some of the convolutional layers and the Max-pooling layer takes place over a 2×2-pixel window, with stride 2.

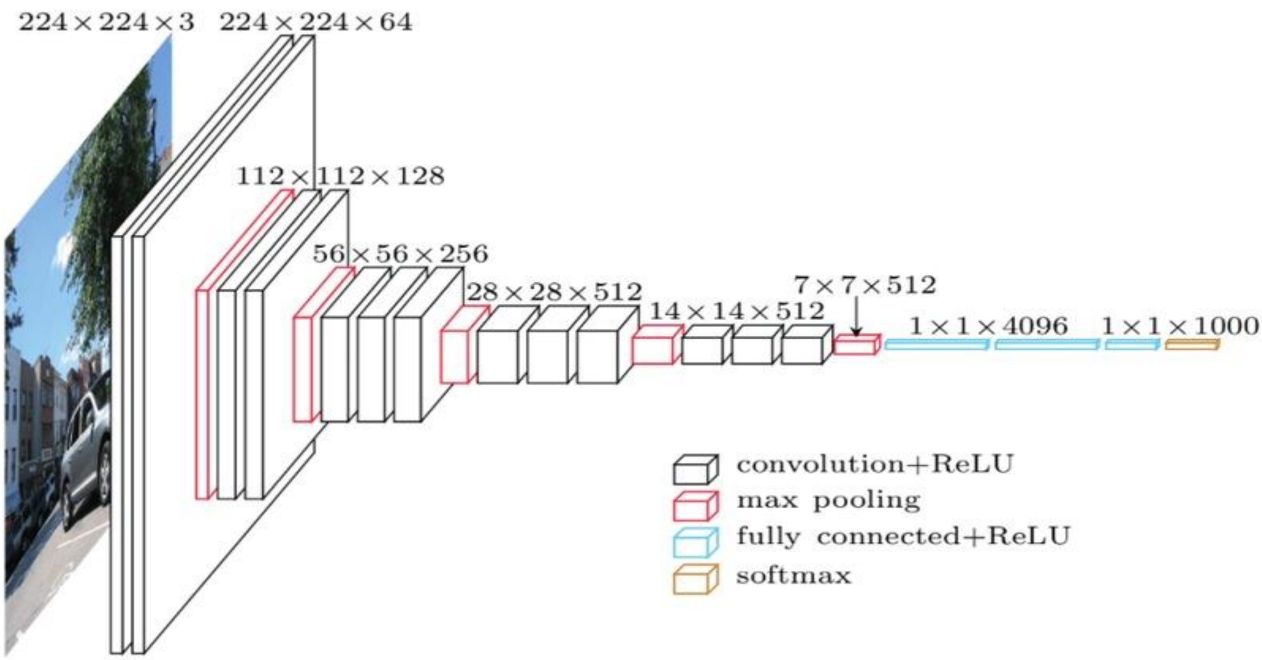


Fig: VGG16 Architecture

The 16 in VGG16 refers to 16 layers that have weights. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers but it has only sixteen weight layers i.e., learnable parameters layer.

- The 16 in VGG16 refers to 16 layers that have weights. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers but it has only sixteen weight layers i.e., learnable parameters layer.
- VGG16 takes input tensor size as 224, 244 with 3 RGB channel
- Most unique thing about VGG16 is that instead of having a large number of hyper-parameters they focused on having convolution layers of 3x3 filter with stride 1 and always used the same padding and maxpool layer of 2x2 filter of stride 2.
- The convolution and max pool layers are consistently arranged throughout the whole architecture
- Conv-1 Layer has 64 number of filters, Conv-2 has 128 filters, Conv-3 has 256 filters, Conv 4 and Conv 5 has 512 filters.
- Three Fully-Connected (FC) layers follow a stack of convolutional layers: the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer.

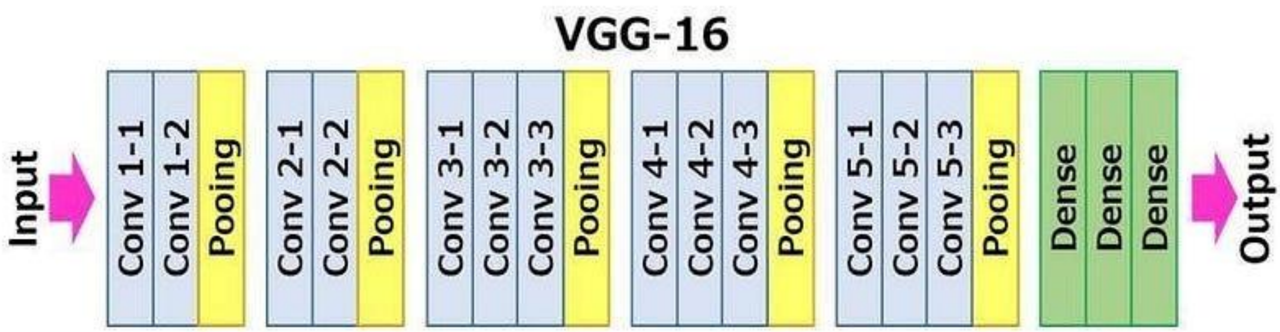
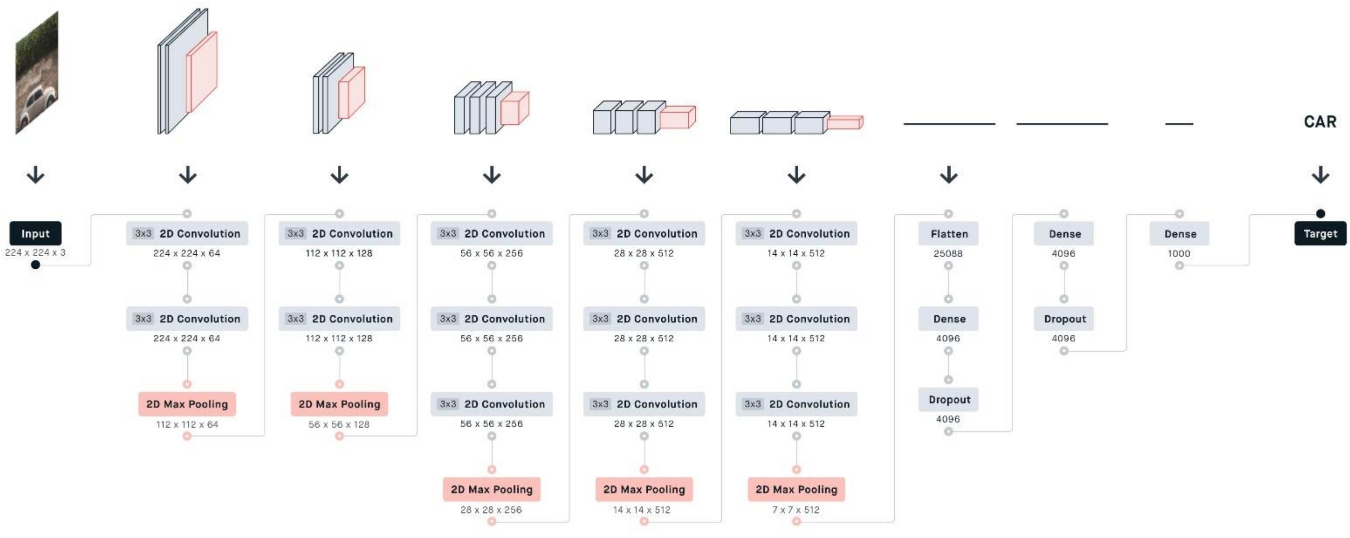


Fig: VGG-16 architecture Map

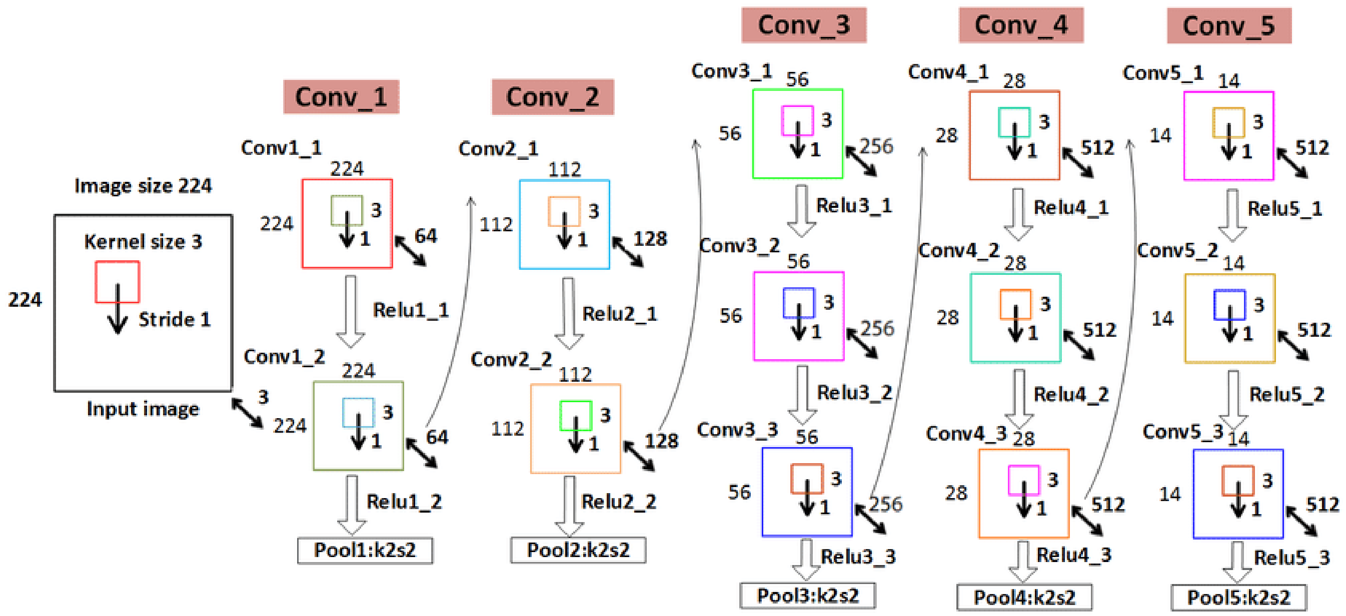
- VGG16 contains 16 layers and VGG19 contains 19 layers. A series of VGGs are exactly the same in the last three fully connected layers. The overall structure includes 5 sets of convolutional layers, followed by a MaxPool. The difference is that more and more cascaded convolutional layers are included in the five sets of convolutional layers.

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

- Each convolutional layer in AlexNet contains only one convolution, and the size of the convolution kernel is 7, 7. In VGGNet, each convolution layer contains 2 to 4 convolution operations. The size of the convolution kernel is 3 3, the convolution step size is 1, the pooling kernel is 2 * 2, and the step size is 2. The most obvious improvement of VGGNet is to reduce the size of the convolution kernel and increase the number of convolution layers.



- Using multiple convolution layers with smaller convolution kernels instead of a larger convolution layer with convolution kernels can reduce parameters on the one hand, and the author believes that it is equivalent to more non-linear mapping, which increases the Fit expression ability.



- Two consecutive 3 X 3 convolutions are equivalent to a 5 X 5 receptive field, and three are equivalent to 7 X 7. The advantages of using three 3 X 3 convolutions instead of one 7 X 7 convolution are twofold: one, including three ReLu layers instead of one, makes the decision function more discriminative; and two, reducing parameters. For example, the input and output are all C channels. 3 convolutional layers using 3 3 require 3 (3 X 3 C C) = 27 C C, and 1 convolutional layer using 7 X 7 requires 7 X 7 C C = 49C C. This can be seen as applying a kind of regularization to the 7 X 7 convolution, so that it is decomposed into three 3 X 3 convolutions.

- The 1 1 convolution layer is mainly to increase the non-linearity of the decision function without affecting the receptive field of the convolution layer. Although the 1 X 1 convolution operation is linear, ReLu adds non-linearity.

In addition to that, a total of three varying FC (Fully Connected) layers in depths are fixed behind a group of convolutional layers, where the first two FC layers is made up of 4096 channels per FC layer, and the third performs 1000- way ILSVRC classification and is made up of 1000 channels for each class. Finally, the final layer is the soft-max layer, it's important to say the Fully Connected Layer's configuration does not vary among different networks.

Activation Function Used Two activation Functions were used for our model training where the Softmax activation and the ReLU function.

The ReLU function was used at the fully connected layers, where the ReLU or "Rectified Linear Unit" is one of the popular activation functions used in Neural Networks and specifically in Convolutional Neural Networks and is defined as in (1):

$$\boldsymbol{y = max(0, x)} \qquad \textbf{(1)}$$

Moreover, the Softmax activation function is used for the output layers and this activation function is a type of logistic regression that is able of normalizing the inputted vector to a new vector where its probability distribution is equal to 1 and it is defined as in (2):

$$\boldsymbol{\sigma(\vec{\alpha})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_i}}} \qquad \textbf{(2)}$$

Loss Functions used In the machine learning domain, the cost functions tend to optimize the model in the training procedure and the aim of the training procedure is to minimize the loss function and the model obtained is better as much as we tend to minimize this loss function. Therefore, one of the most important loss functions is the Cross Entropy Loss Function where it is used for Classification model's optimization and the complete understanding of this loss function depends on the Softmax activation function understanding. Moreover, in our project, the Sparse Categorical Cross Entropy is used for training our model where it has the same loss function as that of the cross entropy as in (3):

$$\boldsymbol{Loss} = -\sum_{i=1}^{output\ size} y_i \log \hat{y_l} \qquad \textbf{(3)}$$

However, the truth labelling procedure is what differs between the two loss functions, where in the case of a one hot encoded true labels ([1,0,0], [0,1,0] and [0,0,1] in 3 classification problem) the categorical cross entropy is used, while the cross entropy is used in the case of an integer truth labels coding ([1],[2],[3]).

There are five configurations of the VGG network, from A to E. The configuration's depth increases from A to B, each with more added layers. The following table describes all possible network architectures.
The table below listed different VGG architectures. We can see that there are 2 versions of VGG-16 (C and D). There is not much difference between them except for one that except for some convolution layers, (3, 3) filter size convolution is used instead of (1, 1). These two contain 134 million and 138 million parameters respectively.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Source for this and the following images: Simonyan and Zisserman, Arxiv.org

Every configuration follows a common architectural pattern, differing only in depth. Network A has 11 weight layers (8 convolutional layers and 3 fully connected layers), while network E has 19 weight layers (16 convolutional layers and 3 fully connected layers).

There are few convolutional layer channels—the number ranges from 64 channels in the first layer to 512 in the last layer (it increases by a factor of two for every max-pooling layer). The figure below shows the total number of parameters in millions.

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

The process of training VGG is similar to that of AlexNet (Krizhevsky et al.). They both involve optimizing a multinomial logistic regression function to achieve backpropagation. VGG uses mini-batches to avoid a vanishing gradient that arises due to the depth of the network.

During training, the batch size was set to be 256, while the momentum was set to be 0.9. The VGG model introduced dropout regularization in two of the fully connected layers, with the dropout ratio set to 0.5. The network's initial learning rate was 0.001. When the accuracy of the validation set stopped improving, the learning rate decreased by a factor of 10. The learning rate dropped three times, and training ended after 74 epochs (370,000 iterations).

It took 2-3 weeks to train a VGG network using 4 NVIDIA Titan Black GPUs on the ILSVRC dataset on 1.3 million training images.

The VGG16 network far outperformed previous models in the ILSVRC-2012 and 2013 competitions in an image classification task. The VGG16 architecture achieved the best results in terms of single net performance (7.0% test error). The table below shows the error rates.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3

The two main drawbacks of using VGG are its long training time and large model size of 500MB. Modern architectures use skip connections and inceptions to reduce the number of trainable parameters, improving both accuracy and training time.

4.2 Design of the system:

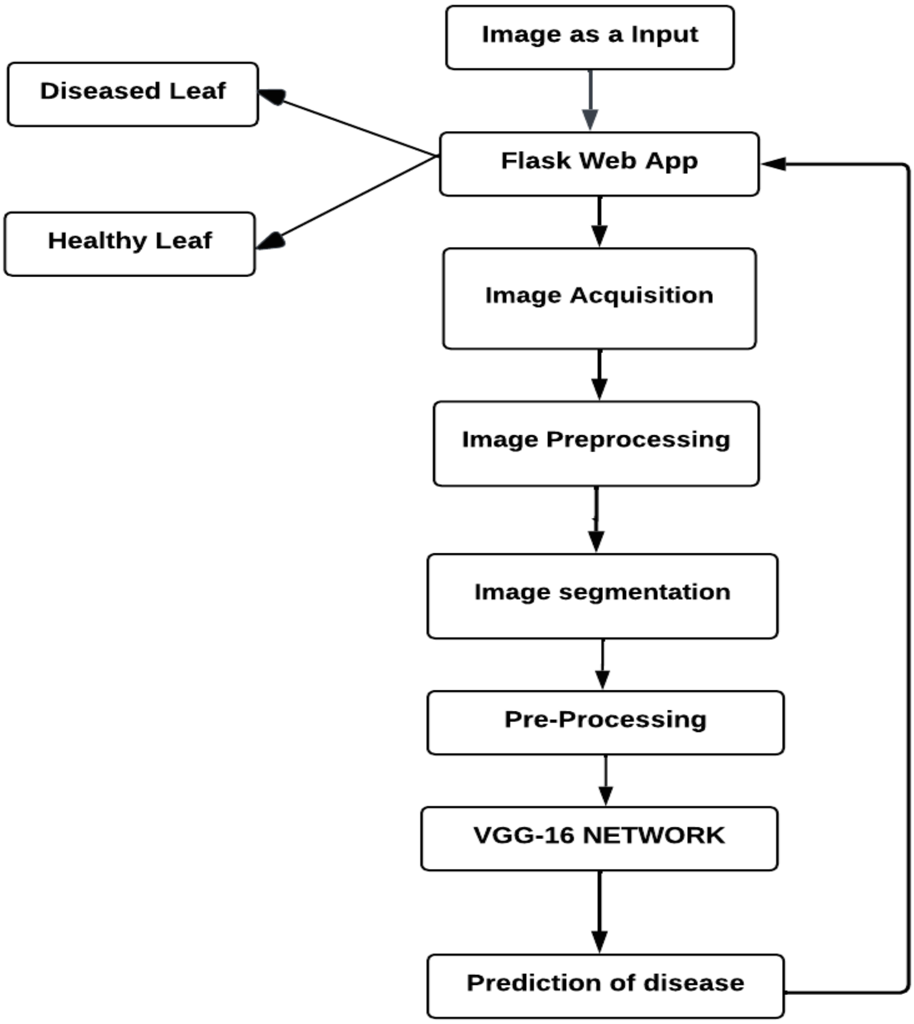
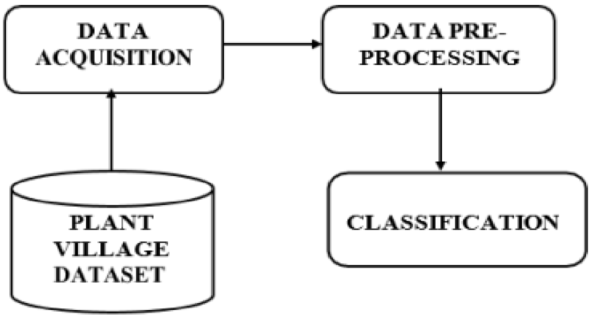


Fig: Flowchart Diagram of the Proposed System

A. System Architecture:

The proposed System architecture comprises of data acquisition from a huge dataset, processing at different convolutional layers and then the classification of plant diseases which declares if the plant image is of a healthy class or diseased class.



B. Data Flow Diagram

Data Flow Diagrams (DFDs) describe the processes of how the transfer of data takes place from the input till prediction of the corresponding output.

1.Data Flow Diagram – Level 0:

The DFD Level 0 depicts the users to input the image of the plant leaves. The system in turn detects and recognizes the plant leaf disease.

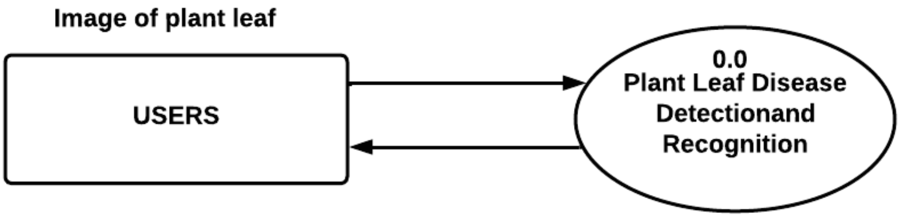


Fig: Data Flow Diagram – Level 0 for Proposed System

2. Data Flow Diagram – Level 1:

The Figure 4 displays the DFD Level 1, where the CNN model takes the image frrom the training dataset and then CNN model predicts the type of disease of the leaf.

Fig: Data Flow Diagram – Level 1 for Proposed System

3.Data Flow Diagram – Level 2:

DFD Level 2 goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning

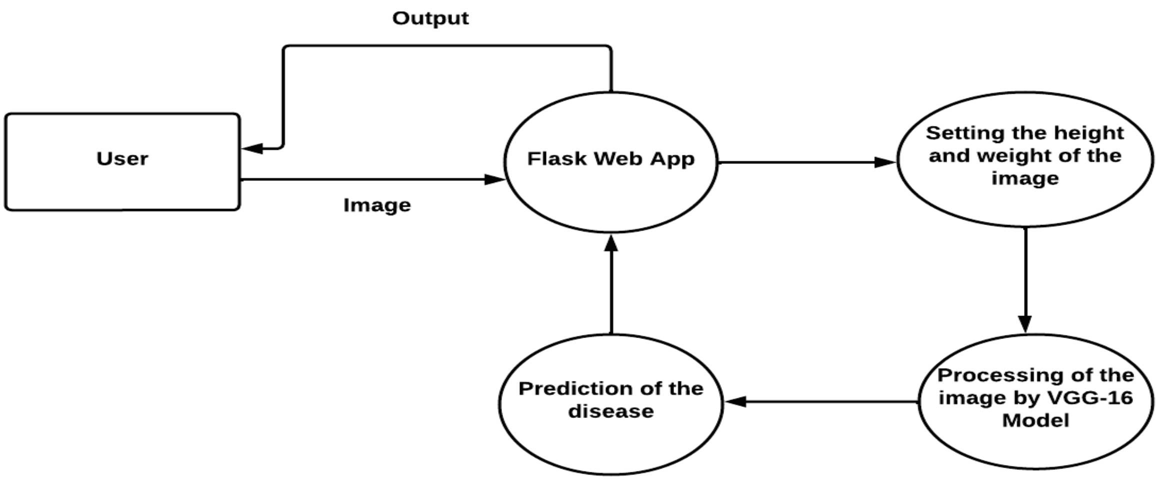
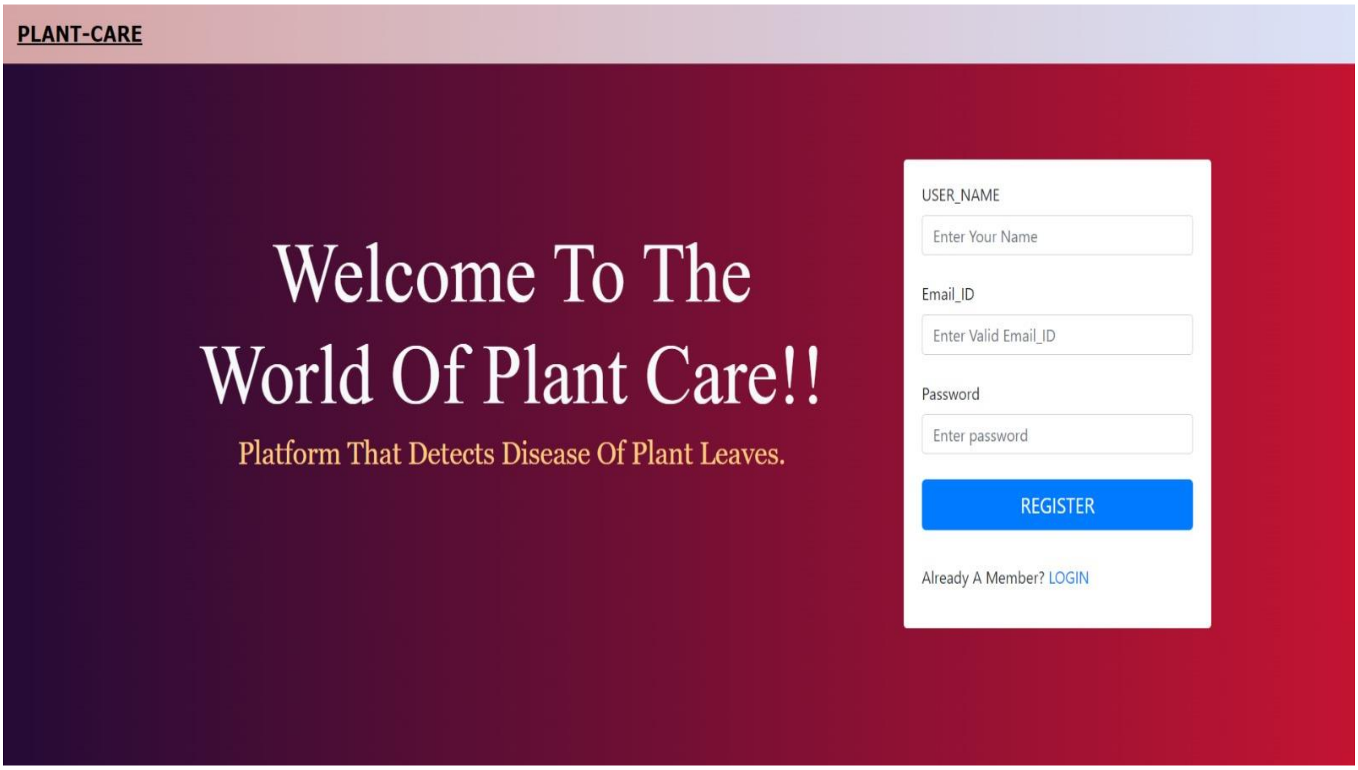
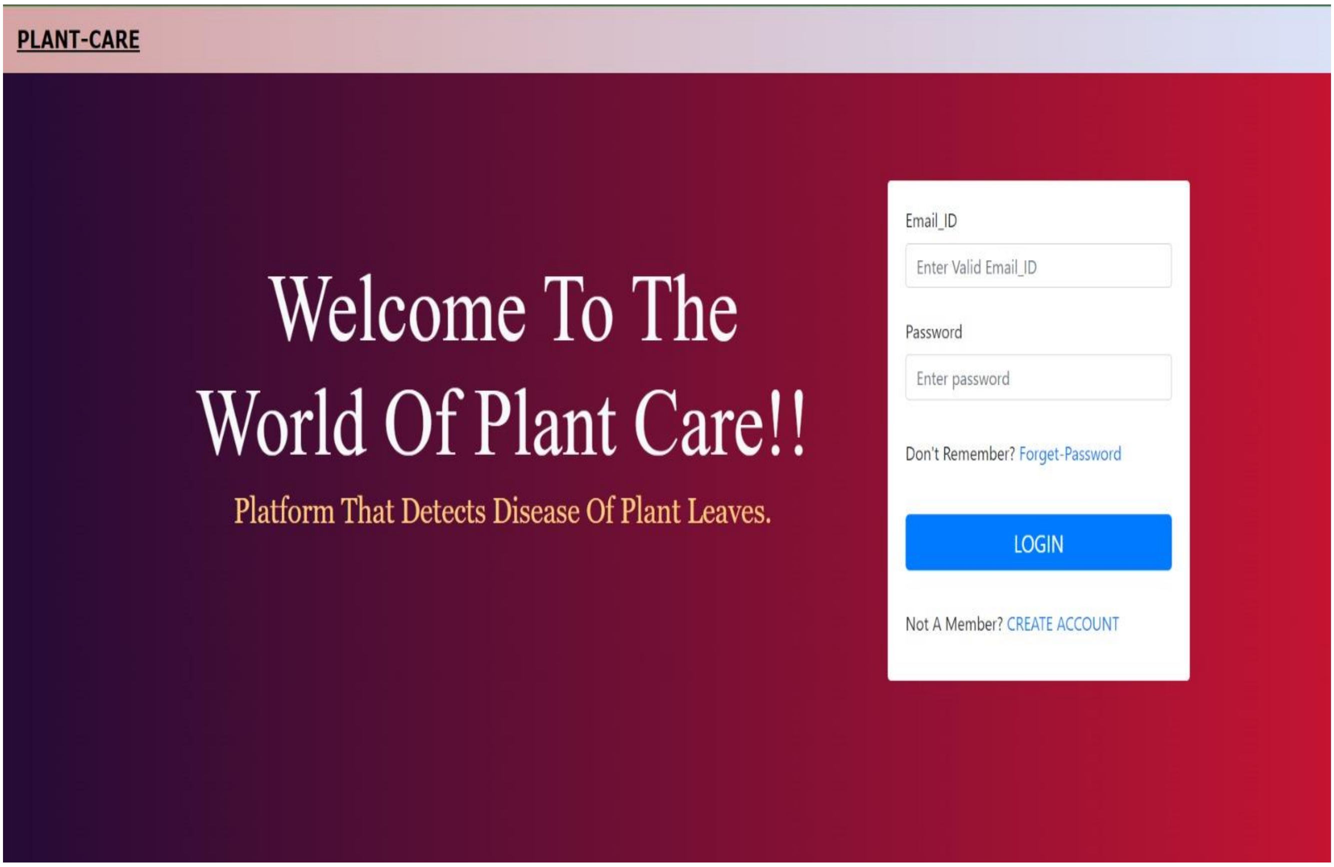


Fig: Data Flow Diagram – Level 2 for Proposed System

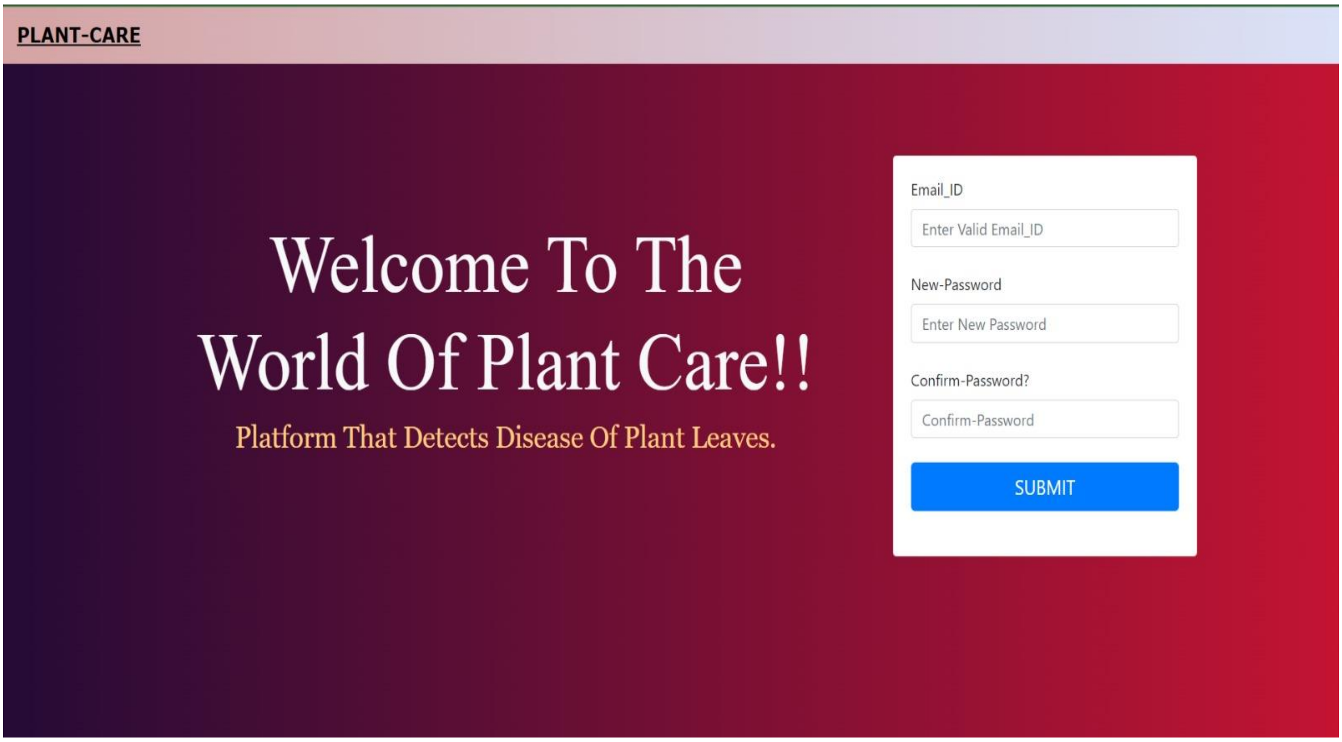
Login Page:



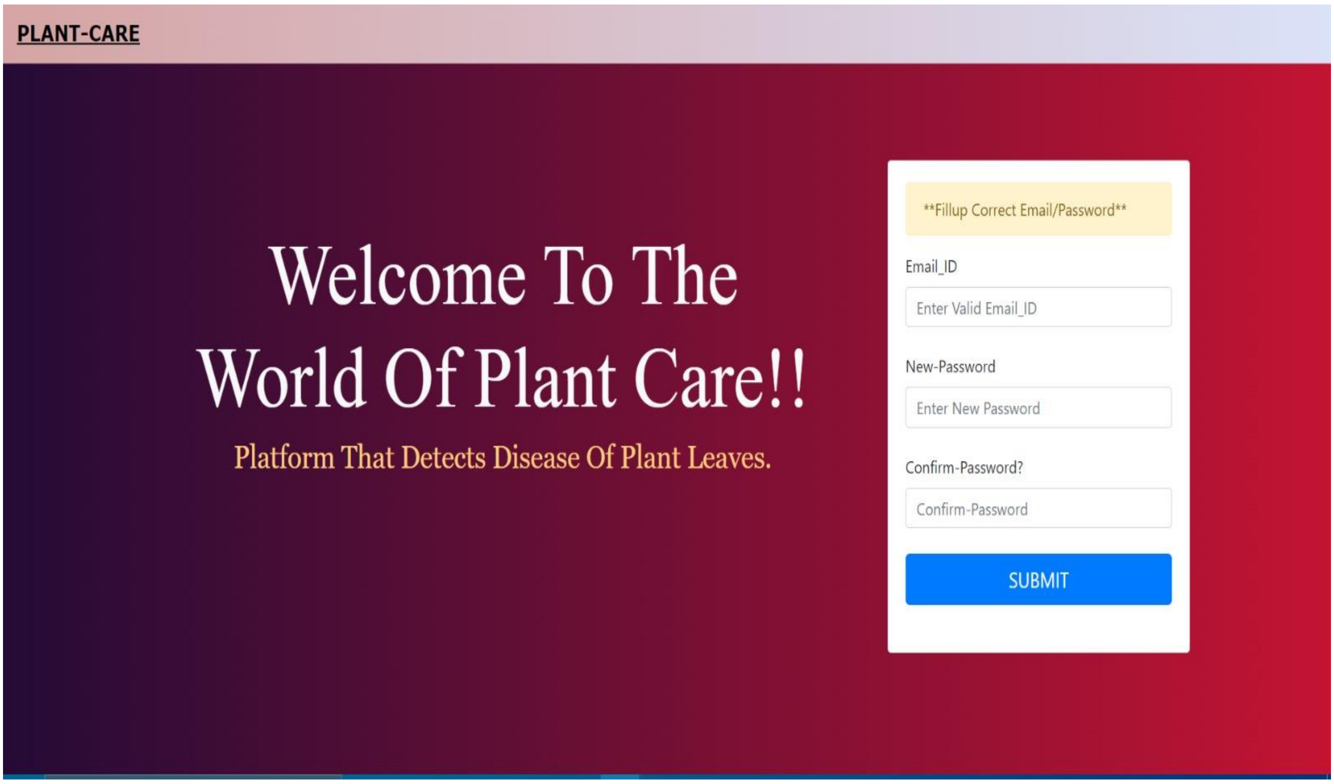
Register Page:

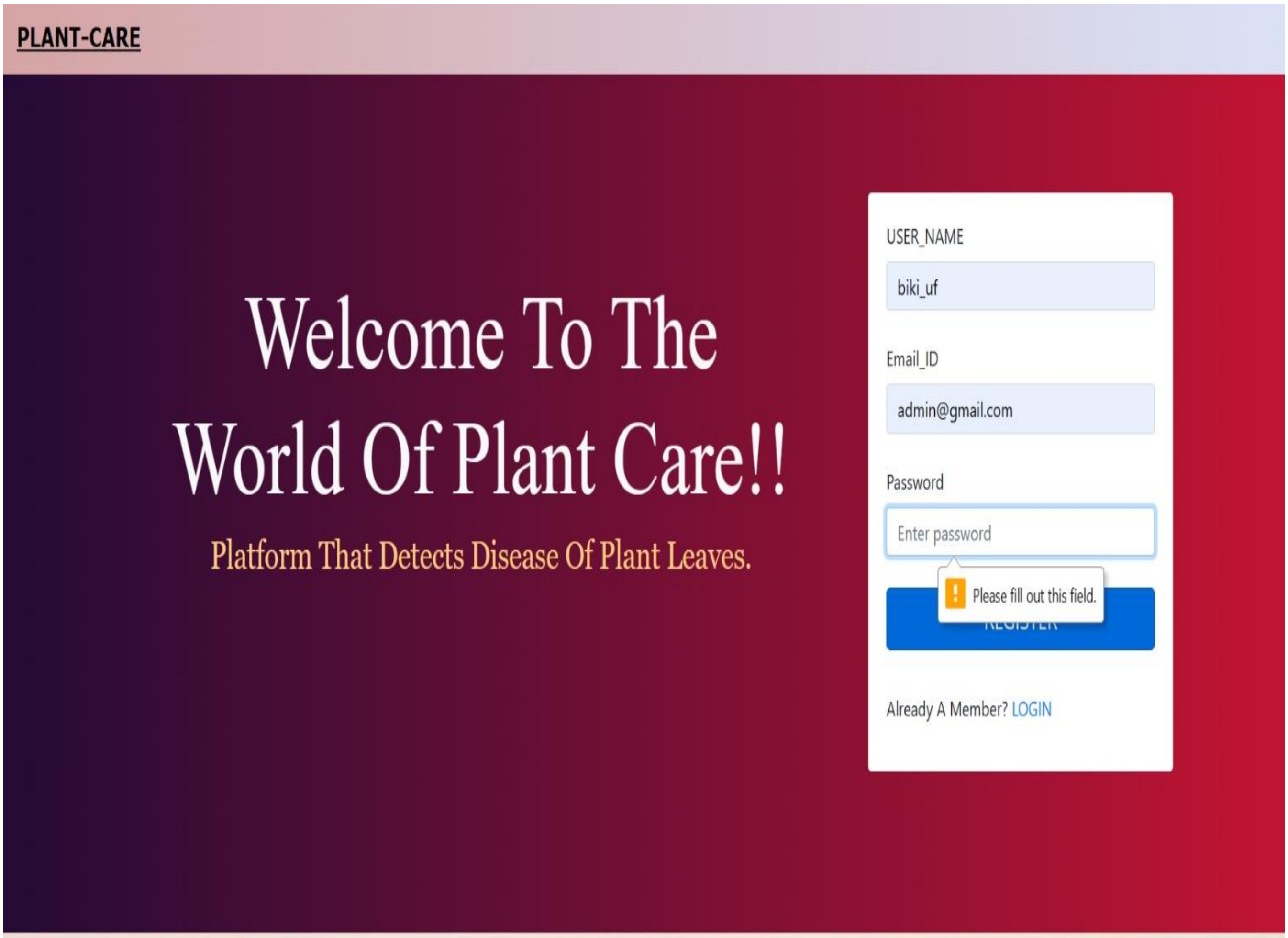


Forget/ Reset Password Page:

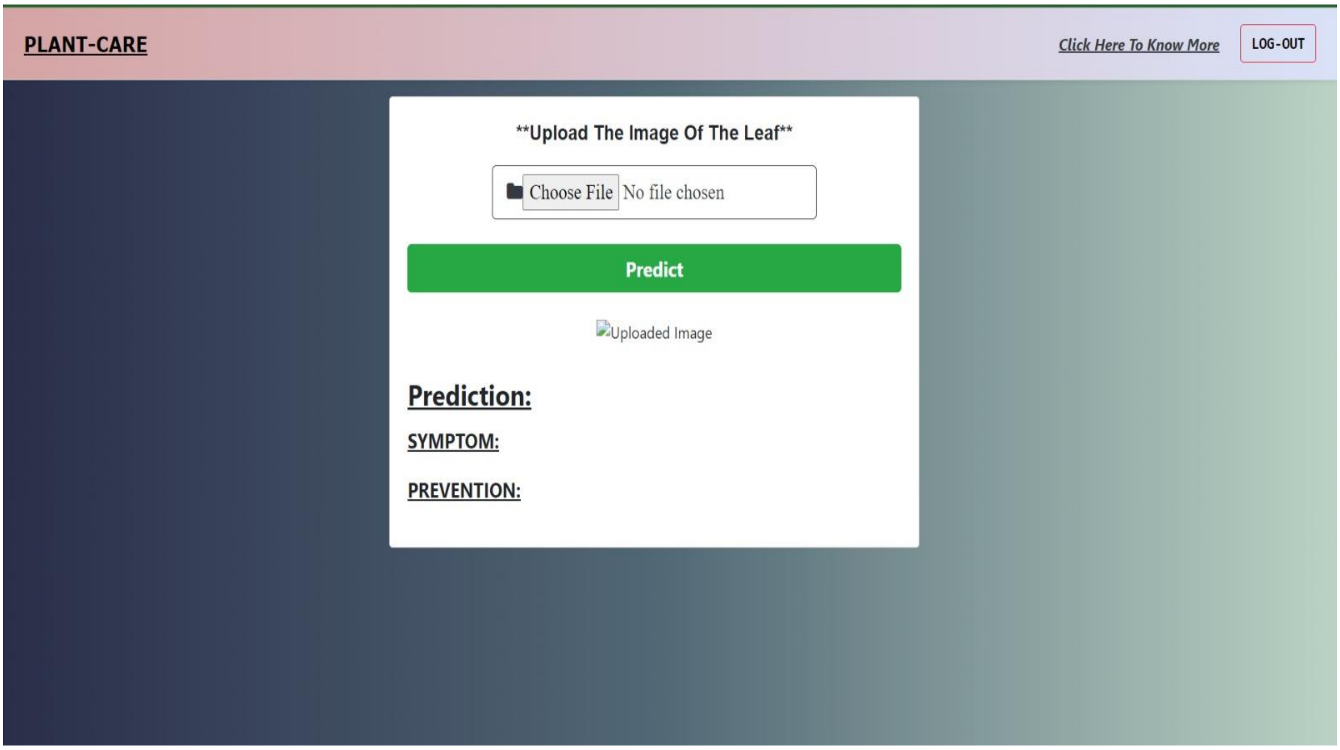


If you put something wrong information→

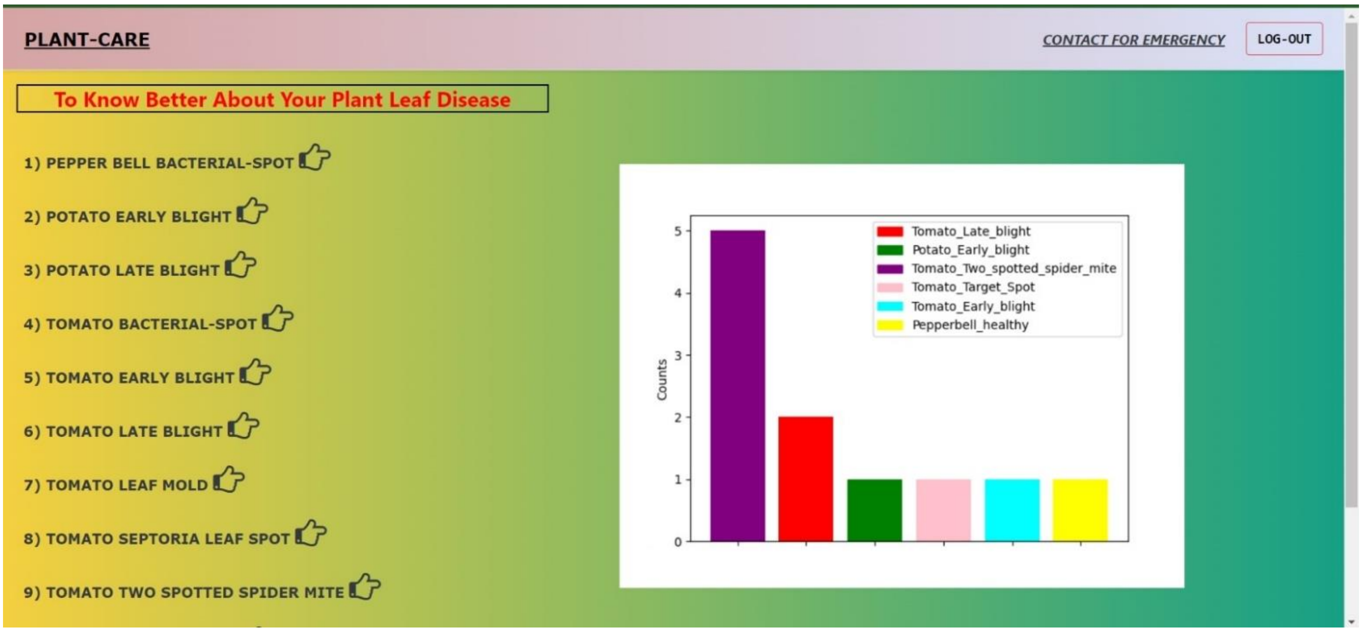




Upload image(Home Page)→



If you click on the “Click Here To Know More” then you will go to the details page whare you can see disease name and if you click on the disease name then you go to disease page. You can see here bar graph diagram of the survey. If you click on “LOG-OUT” button it will redirect you to login page→



4.3 Development:

For the application we made a web application using the deep learning model we created and saving the model in the model section in our web application. With regular auto updation in a server we can always get a new best model in regular intervals from the blockchain . The web application is made using HTML, CSS for frontend and Flask for server side. I have used fastai which is built on top of Pytorch. Dataset consists of 38 disease classes from PlantVillage dataset and 1 background class from Stanford’s open dataset of background images DAGS. 80% of the dataset is used for training and 20% for validation

A. Frontend:

For different users to make it easier for them to work easily and utilise the model more efficiently. The front end was made in a simpler UI. The simple UI will help in conversing with algorithms to opt the desired results.

FRONTEND	
HTML	**For basic layout and functions**
CSS	**For styling and improving UI**

The frontend uses basic knowledge of HTML and CSS

- We have created button and applied an animation for the button

- We created and applied icon for the page
- We applied CSS and the interface of the form.
- Created the basic interface of the page using HTML

B. Backend:

BACKEND	
MYSQL	**Store user information and data.**
FLASK	**Library used for creating the UI**

The backend was created using basic knowledge of Flask

- We created the synchronization between Flask (which used the model to provide output) , (which supported the interface between frontend and backend).
- Created form's backend using Flask, Mysql and simple error popup system.
- 1st user login, if Not Register then Create account, if forget password then reset password.
- Provided the incoming of image and display of result after going through the model which was trained previously.
- Upload of the image as file.

5: EXPERIMENTAL SETUP AND RESULT ANALYSIS

5.1 System Requirements:

Operating System: Windows (7 or higher) or any Unix based OS.

5.1.1 Hardware Requirements for running the application:

- RAM: 4GB or above
- GPU
- Disk space: At least 10GB
- Processor: 2.3 GHz i3 or higher

5.1.2 Software Requirements:

- Language: HTML, CSS, Python.
- Database: MYSQL.

5.2 RESULT ANALYSIS:

We have use Google Colab to perform plant disease detection with GPU in order to reduce the time required for training the model. Following steps are performed to classify healthy and diseased leaves using Transfer Learning:

1. The dataset is separated in training folders, testing folders and validation folders.

2. Upload the dataset on Google drive and mount the drive to the Google colab account.

3. Import the necessary and required libraries.

4. Load the VGGNet model with ImageNet weights.

5. Freeze the top layers and add new layers for transfer learning.

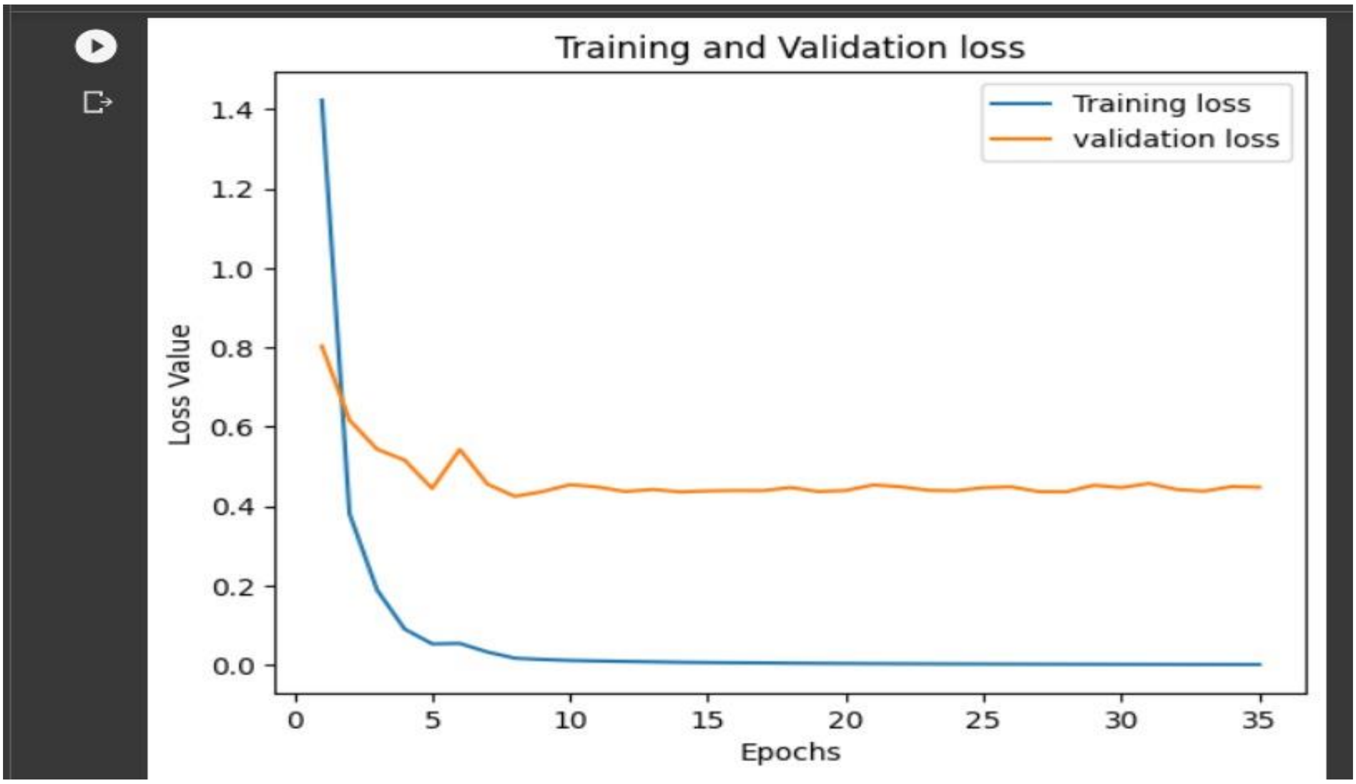
6. Provide the dataset as a directory to the ImageDataGenerator class.

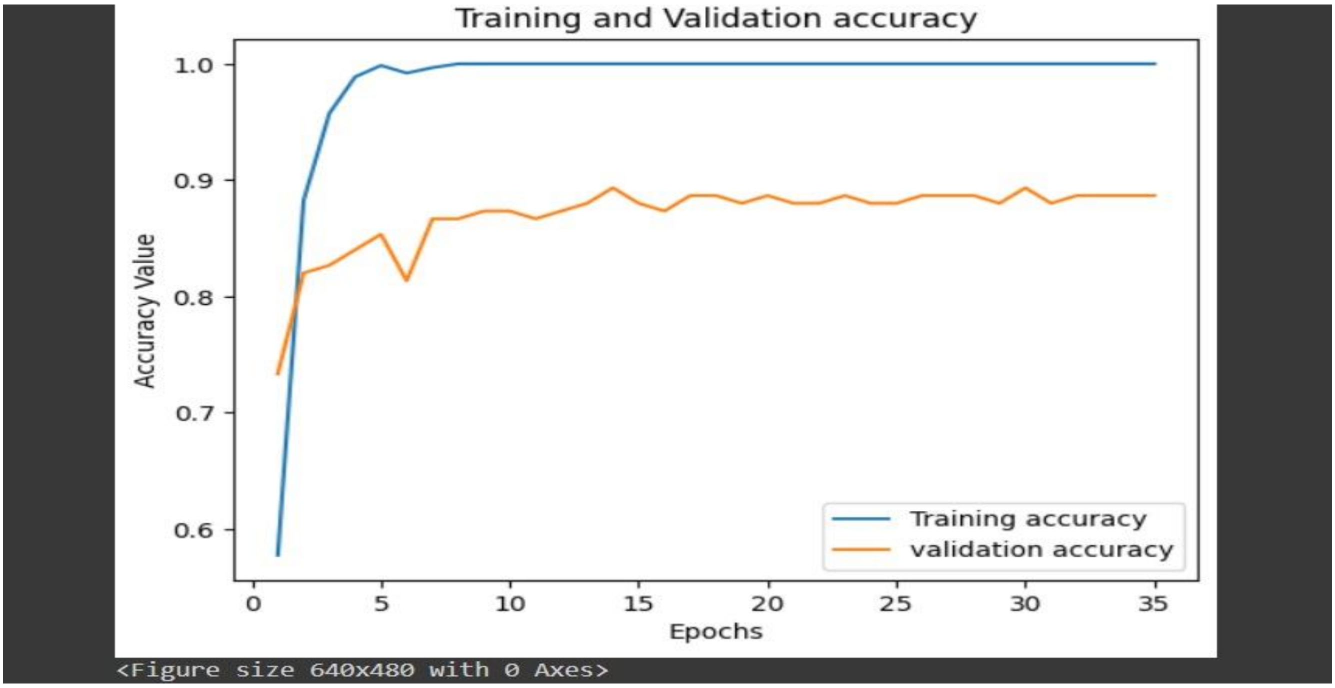
7. Compile and train the model.

8. Plot the accuracy and loss graphically.

9. Test the model by providing an input image.

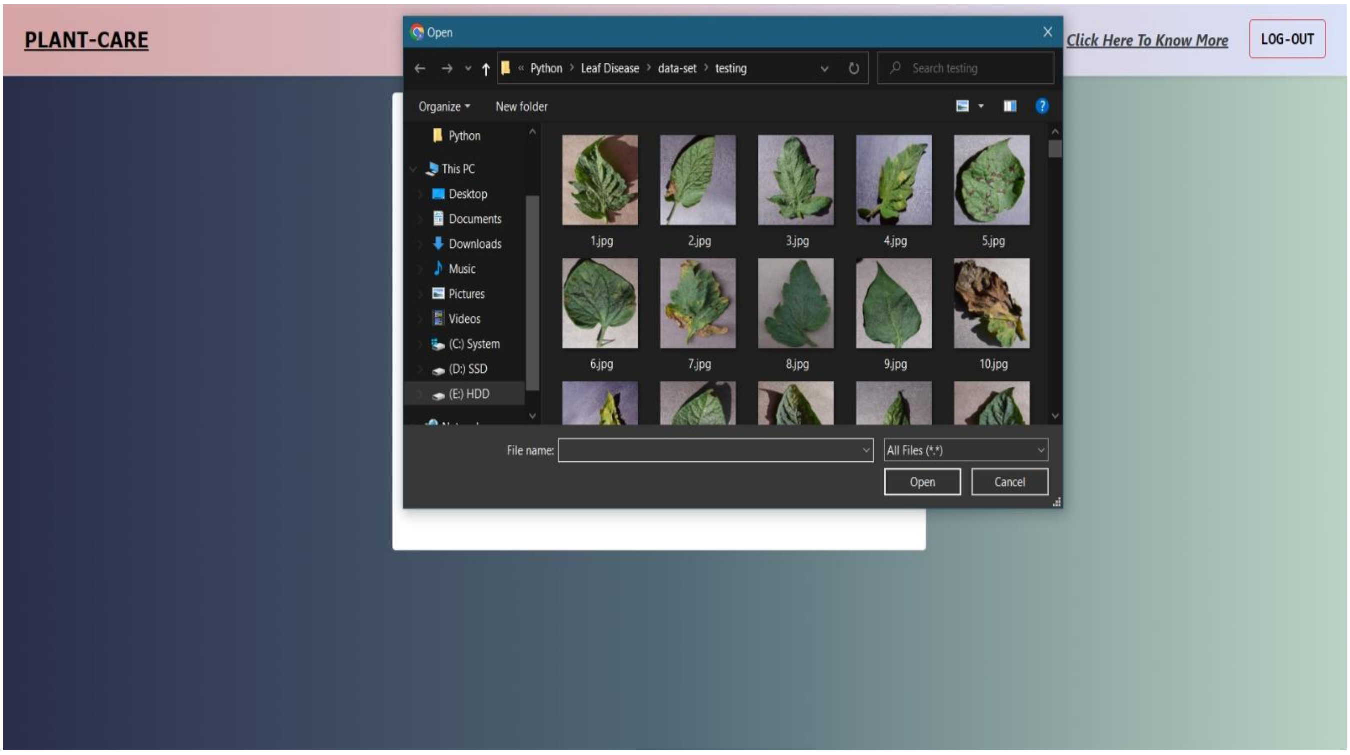
Screenshots of the generated graphs are given below:





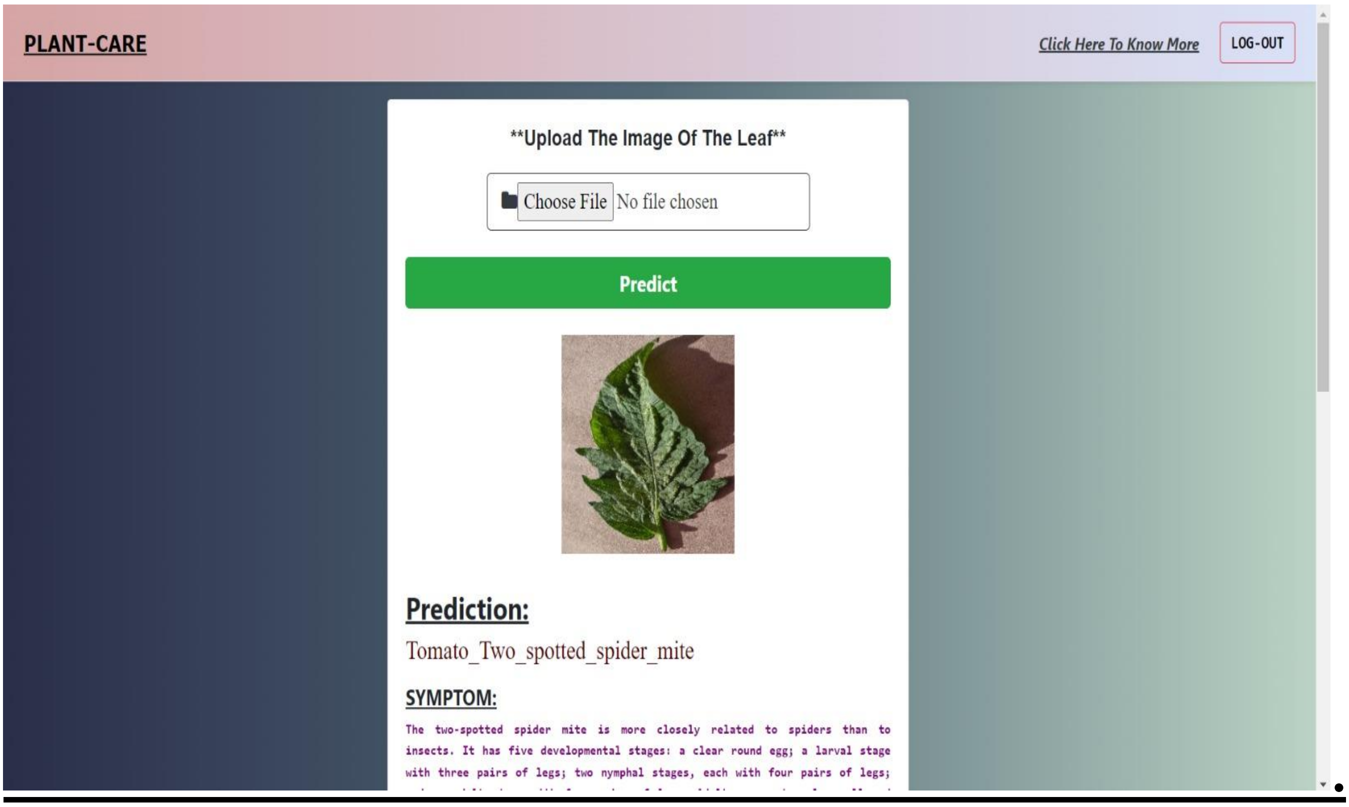
Input:

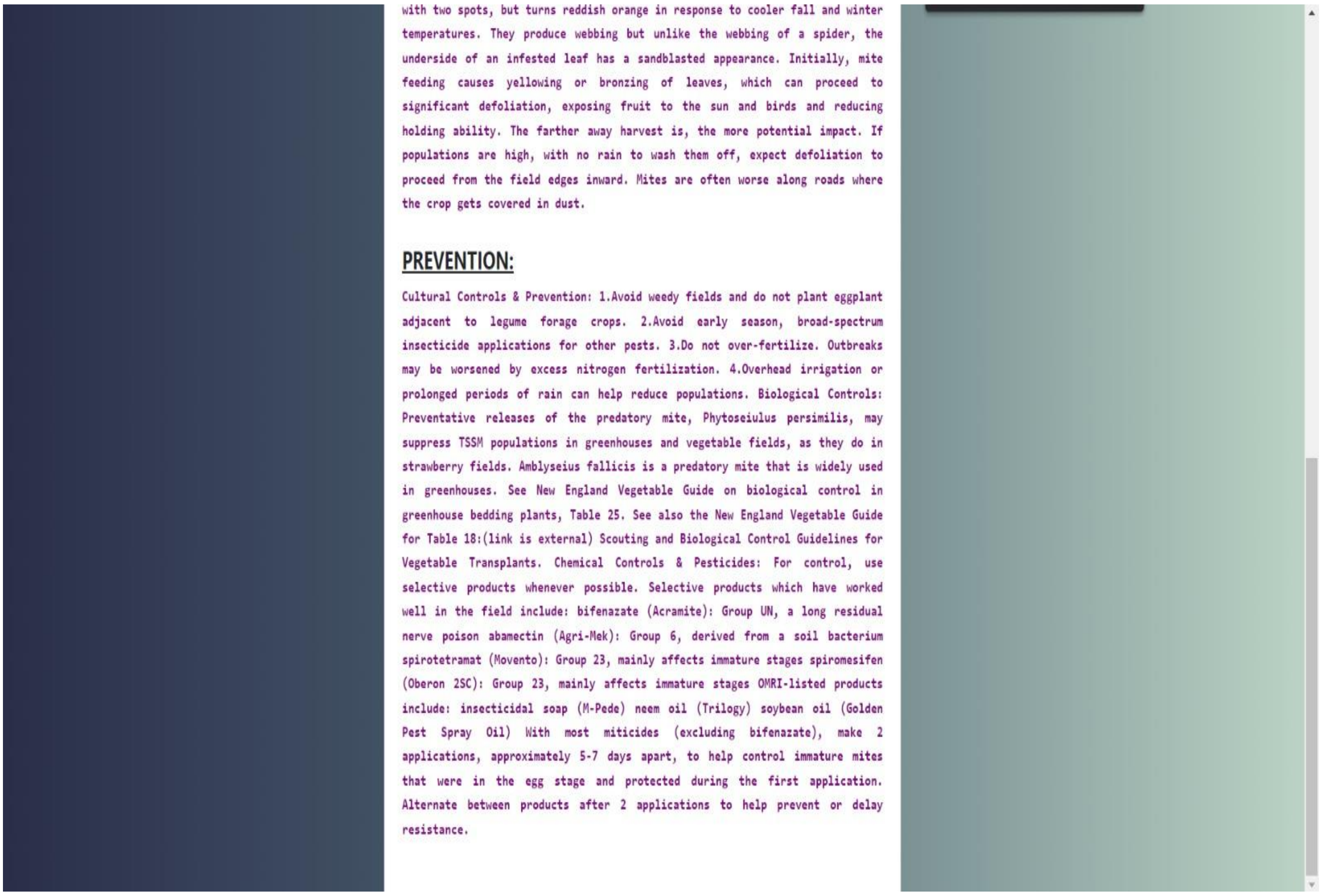
Choose an image→



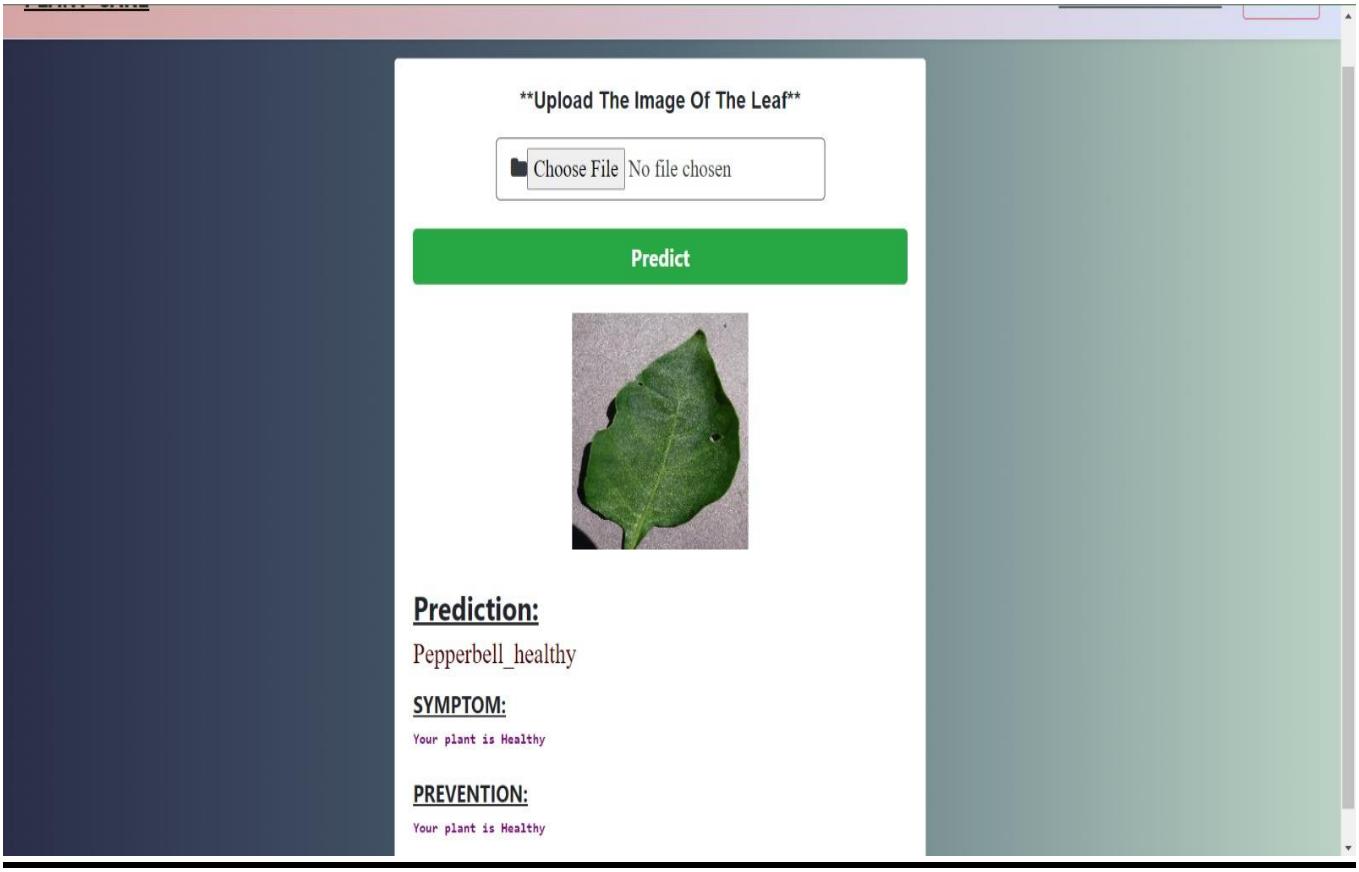
Output: (Prediction of that image)

Prediction of the image that have detect disease→





Prediction of the image that have detect healthy Plant→



6: CONCLUSION & FUTURE SCOPE

6.1. Conclusion:

In this study, we proposed a system for the detection of plant diseases through analyzing leaf images of plants to determine not only if they are healthy or diseased, but rather to classify which kind of disease exists in each crop type. Our model is based on a VGG-16 architecture that classifies 19 classes of plant diseases, according to the data acquired from the Plant Village dataset. The model was able to achieve a 88.67% accuracy with a loss of 0.4477.

Despite achieving a high accuracy with a low loss, our model faces some limitations since the input images must have certain illumination conditions and a complex background behind them due to the fact that they are collected from actual leaves from planted plants. These conditions pose as a challenge for any model used for plant disease detection and they can be considered as areas for improvement when designing or trying to enhance the existing model. Furthermore, in the future studies, our efforts will be focused on achieving more precise disease detection, particularly through training our machine learning model to identify the exact location of the disease on each leaf, especially if more than one disease is detected in one plant leaf. In addition to that, the plant disease dataset can be further increased to take into consideration even more plant diseases and to incorporate additional crop types. Moreover, we can consider some advanced methods to increase the accuracy of processing of leaf images by applying technologies like Faster region-based convolutional neural network (Faster R-CNN), which is a unified network designed for object recognition. Faster R-CNN creates a network that proposes a region for the detection which is then fed to the developed model for training, and after that according to the features, the optimal detection region is selected for classification purposes. Another method that can be implemented is the You Only Look Once YOLO technology which presents a very fast detection in real time with approximately 45 frames for second. Another technique similar to YOLO is the Single Shot Detector (SSD) which provides a fast detection of objects from a single frame. SSD achieves its high accuracy by producing detection at different scales and separates between the predictions by aspect ratio.

6.2. FUTURE ENCHANCEMENTS:

Finding paths with the help of this Artificial Convolutional Neural Network lets us help and find human paths and routes, such as sidewalks, parkways, forest paths. This project implements semantic segmentation approach and uses VGG16 pre-trained model.

Real-time disease detection: VGG-16 can be used to develop real-time disease detection systems that can be deployed in the field. This can help farmers to quickly detect and identify diseases, and take appropriate measures to prevent their spread.

Enhanced accuracy: As the dataset for plant diseases becomes larger, the accuracy of VGG-16 can be further improved. This can be achieved through the use of transfer learning, which involves fine-tuning the pre-trained VGG-16 model on a new dataset of plant disease images.

Multiclass classification: Currently, VGG-16 is used for binary classification (healthy vs. diseased plants). In the future, it could be extended to multiclass classification, which would allow the identification of specific diseases and their severity.

Integration with other technologies: VGG-16 can be integrated with other technologies such as drones, sensors, and IoT devices to provide a comprehensive solution for plant disease detection and management.

Extension to other domains: The VGG-16 model can also be extended to other domains such as animal health and human disease detection. Its ability to extract features from images makes it a useful tool for image recognition and classification tasks in various fields.

7: Bibliography

[1] Uddin, M., et al. "Cloud-connected flying edge computing for smart agriculture." Peer-to-Peer Networking and Applications 14.6 (2021): 3405-3415.

[2] Sharif, Z., Jung, L. T., & Ayaz, M. (2022, January). Priority-based Resource Allocation Scheme for Mobile Edge Computing. In 2022 2nd International Conference on Computing and Information Technology (ICCIT) (pp. 138-143). IEEE.

[3] N. Ashwin, U. K. Adusumilli, N. Kemparaju, and L. Kurra, "A machine learning approach to prediction of soybean disease," International Journal of Scientific Research in Science, Engineering and Technology, vol. 9, pp. 78-88, 2021.

[4] T. S. Xian and R. Ngadiran, "Plant diseases classification using machine learning," Journal of Physics: Conference Series, vol. 1962, p. 012024, 2021.

[5] P. Bedi and P. Gole, "Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network," Artificial Intelligence in Agriculture, vol. 5, pp. 90-101, 2021.

[6] S. Jeyalakshmi and R. Radha, "An effective approach to feature extraction for classification of plant diseases using machine learning," Indian Journal of Science and Technology, vol. 13, pp. 3295-3314, 2020.

[7] M. Lamba, Y. Gigras, and A. Dhull, "Classification of plant diseases using machine and deep learning," Open Computer Science, vol. 11, pp. 491-508, 2021.

[8] I. Ahmad, M. Hamid and S. Yousaf, "Optimizing Pretrained Convolutional Neural Networks for Tomato Leaf Disease Detection," Complexity, 2020.

[9] K. Akshai and J. Anitha, "Plant disease classification using deep learning," in International Conference on Signal Processing and Communication (ICPSC), 2021.

[10] Christian Szegedy and Wei Liu and Yangqing Jia and Pierre Sermanet and Scott Reed and Dragomir Anguelov and Dumitru Erhan and Vincent Vanhoucke and Andrew Rabinovich,"Going Deeper with Convolutions", 2014, https://arxiv.org/abs/1409.4842

[11] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[12] Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkor-eit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polo-sukhin, "Attention Is All You Need", 2017, https://arxiv.org/abs/1706.03762

[13] Bichen Wu and Chenfeng Xu and Xiaoliang Dai and Alvin Wan and Peizhao Zhang and Zhicheng Yan and Masayoshi Tomizuka and Joseph Gonzalez and Kurt Keutzer and Peter Vajda, "Visual Transformers: Token-based Image Representation and Processing for Computer Vision",2020.