PLANT DISEASE DETECTION AND PESTICIDE PREDICTION USING DEEP LEARNING

A PROJECT REPORT SUBMITTED

IN PARTIAL FULFILMENT FOR THE AWARD OF

THE DEGREE OF

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS & INSTRUMENTATION ENGINEERING

Submitted by

B. AMULYA	**18071A1005**
K. KEERTHANA	**18071A1026**
K. SAI BHAVANA	**18071A1029**

DEPARTMENT OF ELECTRONICS & INSTRUMENTATION ENGINEERING

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AICTE Approved; UGC Autonomous; JNTUH Affiliated; UGC "College with Potential for Excellence"; NAAC "A++" Grade

ISO 9001:2015 Certified; QS I.GAUGE "Diamond" Rated; NIRF 2019: 109th Rank Engineering (151–200 Band Overall)

NBA Accredited: CE, CSE, ECE, EEE, EIE, IT, ME, AE; JNTUH-Recognised Research Centres: CE, CSE, ECE, EEE, ME

2022

PLANT DISEASE DETECTION AND PESTICIDE PREDICTION USING DEEP LEARNING

A PROJECT REPORT SUBMITTED

IN PARTIAL FULFILMENT FOR THE AWARD OF

THE DEGREE OF

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS & INSTRUMENTATION ENGINEERING

Submitted by

B. AMULYA	**18071A1005**
K. KEERTHANA	**18071A1026**
K. SAI BHAVANA	**18071A1029**

Under the Guidance of
Mrs. D. SWETHA
Assistant Professor, Dept. of EIE, VNR VJIET

DEPARTMENT OF ELECTRONICS & INSTRUMENTATION ENGINEERING

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AICTE Approved; UGC Autonomous; JNTUH Affiliated; UGC "College with Potential for Excellence"; NAAC "A++" Grade
ISO 9001:2015 Certified; QS I.GAUGE "Diamond" Rated; NIRF 2019: 109th Rank Engineering (151–200 Band Overall)
NBA Accredited: CE, CSE, ECE, EEE, EIE, IT, ME, AE; JNTUH-Recognised Research Centres: CE, CSE, ECE, EEE, ME

2022

DEPARTMENT OF ELECTRONICS & INSTRUMENTATION ENGINEERING

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

AICTE Approved; UGC Autonomous; JNTUH Affiliated; UGC "College with Potential for Excellence"; NAAC "A++" Grade

ISO 9001:2015 Certified; QS I.GAUGE "Diamond" Rated; NIRF 2019: 109th Rank Engineering (151–200 Band Overall)

NBA Accredited: CE, CSE, ECE, EEE, EIE, IT, ME; JNTUH-Recognised Research Centres: CE, CSE, ECE, EEE, ME

CERTIFICATE

This is to certify that the project titled **"Plant Disease Detection and Pesticide Prediction using Deep Learning"** is being submitted, by **B. AMULYA (18071A1005)**, **K. KEERTHANA (18071A1026)**, **K. SAI BHAVANA (18071A1029)**, in partial fulfilment of the requirement for the award of degree of **Bachelor of Technology in Electronics and Instrumentation Engineering**, to the Department of Electronics and Instrumentation Engineering at the **Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology** is a record of bona fide work carried out by them under my guidance and supervision. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree.

Mrs. D. SWETHA
Assistant Professor
Project Supervisor
Dept. of EIE, VNRVJIET
Hyderabad

Dr. R. Manjula Sri
Prof. and Head of the Dept.

Dept. of EIE, VNRVJIET
Hyderabad

External Examiner

ACKNOWLEDGEMENTS

This is an acknowledgement of the intense drive and technical competence of many individuals who contributed to the success of our project.

We express our special gratitude to our Project Guide, **Mrs. D. Swetha,** Assistant Professor, Department of Electronics & Instrumentation Engineering, VNRVJIET, for her guidance and help provided in successful completion of our project titled **"Plant Disease Detection and Pesticide Prediction Using Deep Learning "**. We thank her for the earnest cooperation she extended to us in executing the current project work.

We express our sincere thanks to **Dr. R. Manjula Sri,** Professor & Head of the Department of Electronics & Instrumentation Engineering, VNRVJIET, and to other faculty members in the Department for guiding us through our education at the Institute and for encouraging us all through. We particularly thank our mentors, **Dr. Kalapala Vidya Sagar**, Professor, **Dr. Kiran Chakravarthula**, Associate Professor, **Mrs. Shobhana Priscilla**, Sr. Assistant Professor for helping us through our journey at VNRVJIET. We also extend our gratitude to the Project Coordinator, **Dr. V. Nageshwar**, Associate Professor with the Department of EIE, for his valuable guidance and for streamlining the review process for our project work. Our thanks are also due to the other members of the Review Panel and all other faculty members.

We express our thanks to **Dr. C.D. Naidu**, Principal-VNRVJIET, for enabling us to use the Institute facilities and resources for the successful completion of our project work.

B. AMULYA

K. KEERTHANA

K. SAI BHAVANA

DECLARATION

We hereby declare that the project work titled **"Plant Disease Detection and Pesticide Prediction Using Deep Learning"** submitted, towards partial fulfilment of requirements for the degree of Bachelor of Technology in Electronics and Instrumentation Engineering, to the Department of Electronics & Instrumentation Engineering at the Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, is an authentic work and had not been submitted to any other University or Institute for any award of degree or diploma.

B. AMULYA	**K. KEERTHANA**	**K. SAI BHAVANA**
(18071A1005)	**(18071A1026)**	**(18071A1029)**

ABSTRACT

According to a report, the agriculture sector employs 70% of the Indian population. Natural disasters such as a wide range of diseases and pests have an impact on agricultural production, resulting in quality and quantity losses. Crop diseases are a significant hazard to agriculture. In any case, their fast distinguishing proof remains troublesome in many areas of the planet because of the absence of the important foundation. The project's goal is to identify a problem and recommend a solution for a diseased plant to be free of disease. The photographs of the plants should be uploaded to a website manually. Using CNN, the photos are analysed to detect plant illnesses. Deep Learning uses a CNN (Convolutional Neural Network). To categorise and detect the threat, a model is trained. Along with the identified disease, a suitable cure for the plant disease is provided. Crop output can be increased by taking the necessary precautions.

LITERATURE SURVEY

The authors **Dhapitha Nesarajan [1]** *et al.* (2020), in their paper entitled **"Coconut Disease prediction system using image processing and deep learning techniques"**, presented the classifiers used to recognize the pests by their food ways of behaving by developing an android mobile application. SVM and CNN were picked as the best and most fitting classifiers with 93.54% and 93.72% of accuracy separately. This paper proposes the identification of pest attack and supplement deficiency in the coconut leaves and examination of the illnesses. Coconut leaves monitorization has occurred after the utilization of pesticides and compost with the assistance of the best AI and picture handling methods.

The authors **Eusebio L. Mique, Jr [2]** *et al.* in their paper entitled **"Rice Pest and Disease Detection Using Convolutional Neural Network"**, introduced the pictures which were pre-processed and were utilized in training the model. An application is created with Computing techniques - Image processing, Supervised learning by classification. The model had achieved an accuracy of 90.9 percent. Through the created application, farmers were given data and systems on the best way to control and oversee rice pest infestation.

The authors **Sharadha P Mohanty [3]** *et al.* (2016), in their paper entitled **"Using deep learning for Image-based plant disease detection"**, presented a deep convolutional neural network model which was prepared to distinguish 14 crop species and 26 diseases utilizing a public dataset of 54,306 photographs of diseased and healthy plant leaves taken under controlled settings. On a held-out test set, the trained model accomplished an accuracy of 99.35 percent, showing the reasonableness of this system. Generally speaking, the strategy for preparing deep learning models on progressively immense and freely accessible picture datasets focuses on an unmistakable course toward far and wide worldwide cell phone helped crop infection diagnosis.

The authors **Santhosh Reddy[4]** *et al.* (2020), in their paper entitled **"Detection of Leaf Disease and Pesticide Recommendation using CNN"**, presented that Crop production is one of India's biggest issues, and this approach is being used to improve production and identify illness in the leaf. The state of the leaf provides information on the quality and quantity of the agricultural yield. As a result, they decided to use leaves to tackle many of the challenges that farmers face.In their project, they proposed a system for preprocessing and feature extraction from leaf images. The leaf photos in this set come from the Plant Village

dataset. Then they utilized CNN to detect diseases and make pesticide recommendations. Tensor flow technology is primarily used for this. So they employed multiple layers of a convolution neural network here, such as convolution, pooling, and fully connected output. Their findings show that tensor flow can provide excellent accuracy and effectiveness. Pesticide recommendation is a significant improvement over previous treatment.

The authors **Punam Bedi [5]** *et al.* (2021), in their paper entitled **"Plant Disease Detection using Hybrid Model based on convolutional autoencoder and convolutional neural network"**, proposed a clever hybrid model in view of Convolutional Autoencoder (CAE) network and Convolutional Neural Network (CNN) for automatic plant sickness detection. The proposed hybrid model is applied to distinguish Bacterial Spot sickness present in peach plants utilizing their leaf pictures, be that as it may, it very well may be utilized for any plant illness identification. The examinations acted in this paper utilized an openly accessible dataset named PlantVillage to get the leaf pictures of peach plants. The proposed framework accomplishes 99.35% training accuracy and 98.38% testing accuracy utilizing just 9,914 training parameters.

The authors **Mirjana Karanovic [6]** *et al.* (2019), in their paper entitled **"Solving Current Limitations of Deep Learning Based Approaches for Plant Disease Detection"**, introduced two methodologies that were utilized to expand the quantity of pictures in the dataset: traditional augmentation techniques and state-of-the-art style generative antagonistic organizations. A few examinations were directed to test the effect of preparing in a controlled climate and use, all things considered, circumstances to precisely distinguish plant illnesses in a mind boggling foundation and in different circumstances remembering the recognition of various illnesses for a solitary leaf. At long last, a clever two-stage engineering of a brain network was proposed for plant illness grouping zeroed in on a genuine climate. The prepared model accomplished an accuracy of 93.67%.

The authors **Prabhjot Kaur[7]** *et al.* (2022), in their paper entitled **"Recognition of Leaf Disease Using Convolutional Hybrid Neural Network by Applying Feature Reduction"**, proposed that due to time constraints and the complexity of diseases, physically checking plant illnesses is troublesome. Programmed portrayal of plant sicknesses is broadly required in the field of agricultural inputs. In light of the performance of all image-processing technologies, the best choice is made for this job. Plant diseases in grapevines are the subject of this study. The four illnesses experienced in grape plants are leaf blight, black rot, stable,

and black measles. A few past research recommendations that pre-owned AI calculations to identify a couple of sicknesses in grape plant leaves have been submitted; however, no one has yet proposed a total location of every one of the four infections. The photographs were gathered from the plant village dataset to teach the EfficientNet B7 deep architecture using transfer learning. The obtained features are down-sampled using a Logistic Regression approach after the transfer learning. Finally, after 92 epochs, state-of-the-art classifiers identify the most discriminant highlights with the greatest constant accuracy of 98.7%. A reasonable classifier for this application is likewise suggested in view of the recreation results. A fair comparison of existing procedures confirms the effectiveness of the suggested technique.

The authors **Heri Andrianto [8]***et al.* (2019), in their paper entitled **"Smartphone Application for Deep Learning-Based Rice Plant Disease Detection"**,introduced a profound learning-based rice sickness recognition framework, which comprises of an AI application on a cloud server and an application on a cell phone. The cell phone application capabilities to catch pictures of rice plant leaves, send them to the application on the cloud server and get order brings about the type of data on the kinds of plant diseases.The execution of the rice plant sickness recognition framework with VGG16 design has a train exactness worth of 100 percent and a test precision worth of 60%.

The authors **Bo Wang [9]** *et al.* (2022), in their paper entitled **"Identification of Crop Diseases and Insect Pests based on Deep Learning"**,proposed a yield sickness and pest identification model in view of profound gaining according to the viewpoint of biological and natural security to handle the issues of different sorts of harvest sicknesses and nuisances, fast spread speed, and quite a while of manual identification of illnesses and irritations. To begin, field sampling is used to acquire crop images, and preprocessing of image is finished using nearest-neighbor interpolation. The AlexNet model's network structure is then upgraded. Different neuron nodes and experimental settings are set by optimizing the whole connection layer. Finally, agricultural diseases and pests are identified using the upgraded AlexNet model. The average recognition accuracy and recognition time of aromatic pear diseases and insect pests are 96.26 percent and 321 ms, respectively, according to the experimental study of the suggested model based on the produced data set, which is superior to other comparable models. This method's recognition accuracy on different data sets is not less than 91 percent, indicating that it is portable.

The authors **Usha Kiruthika [10]** *et al. (2019)* in their paper entitled **"Detection and Classification of Paddy Crop Disease using Deep Learning Techniques"**, introduced a viable strategy for the recognizable proof of paddy leaf infection. The proposed approaches include pre-handling of an info picture and the paddy plant illness type is perceived utilizing the Gray-Level Co-event Matrix (GLCM) procedure and classifiers to be specific Artificial Neural Networks are utilized for better exactness of location.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Objective

- The primary target of this task is to distinguish the plant disease and provide the remedy to increase the yield.
- The aim of the project is to get the highest accuracy by training the model using the CNN algorithm.
- To help the farmer by giving him the necessary precautions to safeguard the plants.

Fig 1.1 Diseased tomato leaf

1.2 Outline

India is a developed country, with farming utilizing more than 70% of the populace. Ranchers have a great many choices with regards to picking proper yields and insect sprays for their plants. Thus, crop harm would bring about a huge loss of result, influencing the economy.Plant's leaves, which are unguarded, display disease symptoms first. From the outset of their life cycle until they are fit to be collected, the yields should be checked for sicknesses. The agricultural land mass and productivity determine a country's economic progress. Agriculture is the primary source of income for the majority of the population. Farmers grow a variety of crops depending on soil fertility and resource availability.

Fig 1.2 Early blight in potato plant

1.3 Motivation

Human society now has the ability to generate enough food to feed more than 7 billion people, thanks to modern technologies. Climate change, pollinator decrease, plant diseases, and other factors, however, continue to pose a danger to food security. Plant infections are not only an overall risk to food security, however, they can likewise have destroying consequences locally. Smallholder ranchers whose livelihoods are dependent on healthy crops may suffer as a result.

Crop illnesses are a significant risk to food security, but due to an absence of diagnostic facilities in numerous areas of the planet, early detection is difficult. Plant diseases have been recognised manually with the use of early experience since antiquity, although this can lead to errors and ineffective treatments. However, with the help of the trained model the errors can be resolved with the highest accuracy.

Fig 1.3 Farmer examining the diseased leaf

1.4 Scope for the Work

- Albeit this assortment contains an enormous number of photographs of different plant species and their diseases, the foundation is straightforward and clear. The actual environment, then again, ought to be examined in a practical setting.
- An additional productive method for seeing illness spots in plants ought to be created on the grounds that it will set aside cash by lessening the utilization of fungicides, pesticides, and herbicides that aren't required.
- Since the seriousness of plant infections fluctuates after some time, DL models ought to be improved or acclimated to permit them to recognize and arrange diseases during their whole life cycle.
- To figure out the components that impact the ID of plant illnesses, for example, dataset classes and sizes, learning rate, enlightenment, etc, a far reaching review is required.
- The DL design ought to be viable across an assortment of lighting conditions, consequently the datasets ought to incorporate photos gathered in an assortment of field situations notwithstanding this present reality.

CHAPTER 2
PROJECT OVERVIEW

2.1 Existing System

There are a number of disease detection methodologies available. Artificial Neural Networks and K-means have proven to be the most effective so far.

One of those researches that has proven to be useful is Santosh Reddy, Madhura G. Prasad, M. Mamatha, P. Sanhitha, E. Shilpa proposed a model for “Detection of Leaf Disease and Pesticides Recommendation using CNN”.

The model is designed to detect diseases and make pesticide recommendations. Tensor flow technology is primarily used for this. So they employed multiple layers of a convolution neural network here, such as convolution, pooling, and fully connected output layer.

2.1.1 Issues in existing system

- In some cases, the execution actually needs precision in terms of results. More work on optimization is essential.

2.1.2 Limitations in existing system

- There are a couple of illnesses that have been covered. Accordingly, work should be extended to make extra progress on diseases.

2.2 Proposed System

The CNN method was chosen for this project out of all the image classification algorithms available. The design of the algorithm is the primary consideration while selecting CNN.

2.2.1 Benefits in proposed system

- For image classification and recognition, CNN is a more powerful and accurate algorithm
- It does not rely on manual image processing techniques. It finds the relevant features without the need for human intervention.
- Because of its architecture and excellent accuracy, the CNN method is widely used for picture categorization and recognition.

CHAPTER 3

METHODOLOGY

3.1 Introduction

There are many illness detection methodologies available. Artificial Neural Networks and K-means have proven to be the most effective yet. In this project, we primarily employ Disease detection using a convolutional neural network. This project also includes a pesticide suggestion. It is not necessary to design systems automatically. The very best Detecting or identifying the leaf as critical challenges here. Is that a healthy or unhealthy leaf? If it's a sickly leaf, remove it. The disease's name should be displayed. The speedy Pesticides recommendations are made on the spot along with the precautions.

The majority of pesticides used by farmers are toxic to humans. Farmers, on the other hand, can monitor the leaves daily. The tasks to be completed include detecting diseases and prescribing pesticides, but the other crucial goal is to meet them as quickly as possible. Therefore we are using the Convolutional Neural Network(CNN).

3.1.1 Artificial Neural Network(ANN)

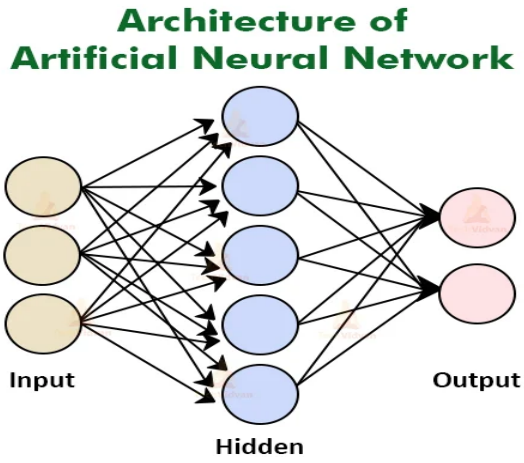


Fig 3.1.1 Architecture of Artificial Neural Network

Artificial neural networks (ANNs), also known as neural networks (NNs), are a type of artificial neural network. SNNs are a subset of ML that is at the core of calculations in view of deep learning. Their name and configuration depend on the human cerebrum reenacting the manner in which natural neurons speak with each other. Similar to humans, a brain with

neurons that are interconnected; artificial neural networks are artificial neurons that are linked together in different layers of the network.

Each node, or artificial neuron, is connected to another and has a specific function, weight, and cutoff if any individual node's output surpass the specified approach. When a node receives a value, it is actuated and starts sending data to the network's next layer. Unless specified, no data is transferred to the next network layer. As a result, the binary data obtained is then compared to the actual output If it differs, it adjusts its results until it produces a result that corresponds to the actual output.

Artificial neural networks are composed of node layers, every one of which contains an input layer and one or more hidden layers.

- **Input Layer:**

  As the name suggests, it acknowledges inputs in different formats determined by the developer.

- **Hidden Layer:**

The hidden layer shows up between the input and result layers. It does all of the calculations to uncover hidden features and patterns. It takes an input and computes the weighted sum of the inputs, along with a bias. A transfer function is used to address this computation. It determines the weighted total, which is then fed into an activation function to produce the output. Activation functions decide if a node should fire or not. The nodes which are terminated are the ones in particular who come to the result layer.

- **Output Layer:**

An ANN first undergoes a training phase in which it figures out how to perceive information patterns, whether visually, audibly, or textually. During this supervised stage, the network looks at its actual result to what was generally anticipated to deliver the ideal outcome.The distinction between the two outcomes is back propagation adjustment which means that the network operates in reverse by adjusting the weight of its connections from the output unit to the input units until the distinction between the actual and wanted result is reached and creates the littlest measure of mistake conceivable.

3.1.2 Convolutional Neural Network(CNN)

Convolutional Neural Network, also known as CNN or ConvNet, is a type of deep neural network that is repeatedly used to analyze visual imagery. It takes an image as input and converts it into a format that is simpler to process while holding the elements that are utilized for prediction. They are otherwise called shift-invariant artificial neural network or space invariant artificial neural networks(SIANN).

CNNs are a kind of neural network that works in image processing. A binary representation of visual information is a digital picture. It comprises of a lattice like arrangement of pixels with pixel values demonstrating how brilliant and what variety every pixel ought to be.

CNN's main feature is detecting image features such as patterns, brightness, and color spots. The CNN algorithm can detect advanced features in an image by extracting these basic features from the input image.

3.1.3 Architecture and Methodology of CNN

A CNN typically contains three layers. They are

1. Convolutional layer
2. Pooling layer
3. Fully Connected layer

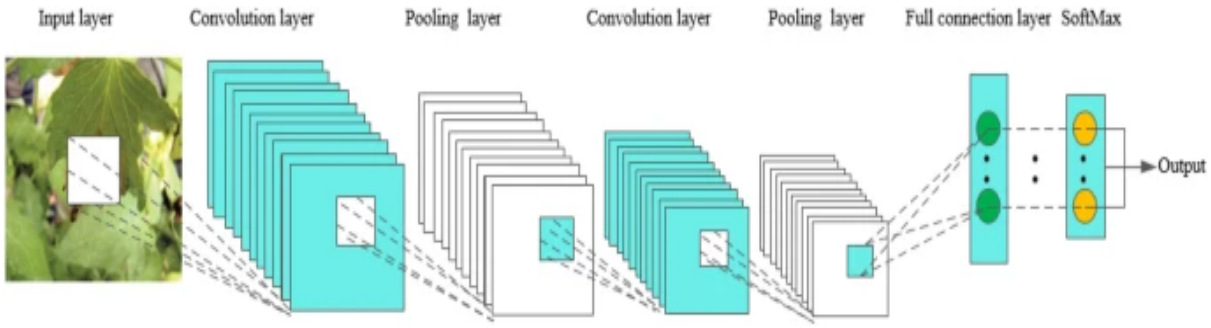


Fig 3.1.2 Architecture of Convolutional Neural Network

- **Convolutional layer:**

CNN's most important layer is the Convolutional layer. To obtain the convolved feature, the kernel is applied to the input image. The obtained convolved feature is fed into the next layer as input. Each Convolutional layer includes a collection of filters known as Convolutional kernels.

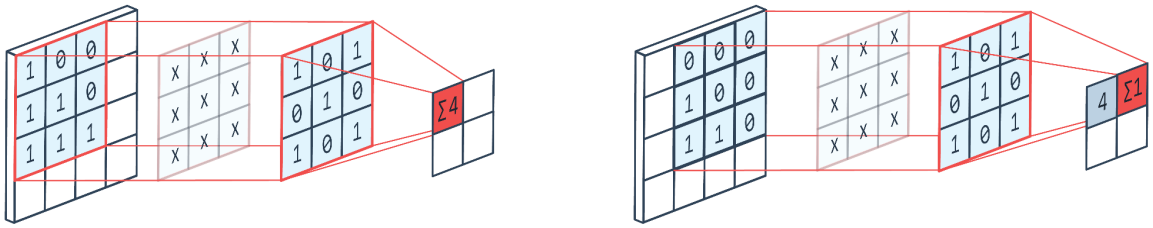


Fig 3.1.3 Convolution Layer

A kernel is a sort of channel that is utilized to extract images from pictures. The kernel matrix is shifted across the input data. It does dot product on the specific region of the image under consideration, which is the same size as the kernel. The convolved feature, also known as an Activation Map, is the result of the dot product. A kernel matrix, also known as stride, is typically three or five bytes in size. In addition, the number of Convolutional layers that can be present in CNN is limited to three, sixteen, or sixty-four kernels.

- **Pooling Layer:**

A filter is slid over the input sent to this layer in the pooling layer. At some points, the network's output is replaced. The Convolutional layer and pooling layer are stacked one after the other in a typical CNN model. Pooling layers are used to reduce parameter values.

There are three sorts of pooling layers utilized in CNN design. They are

1. Max Pooling
2. Average Pooling
3. Global Pooling

Max Pooling:

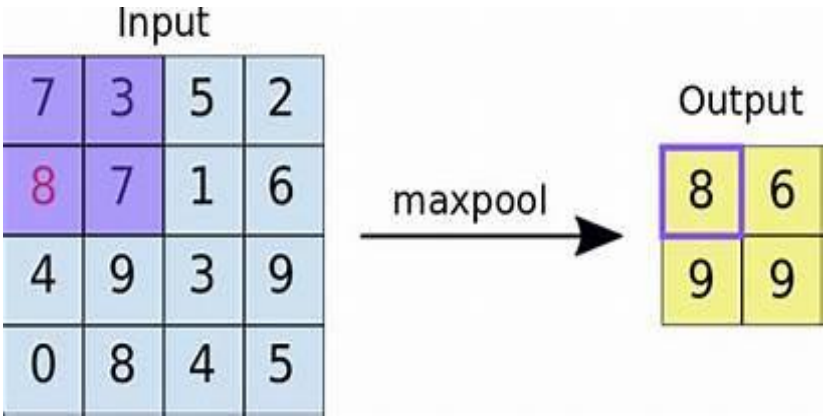


Fig 3.1.4 Max Pooling Layer

Max Pooling is a pooling activity where the greatest worth of the subregion is considered from the feature map. The result of this layer is the feature map that comprises of the prominent features of the past feature map.

Average Pooling:

Average Pooling is a pooling layer that involves the typical worth of the subregions in the feature map. This pooling layer would deliver a feature map that is the average of the features in the past feature map.

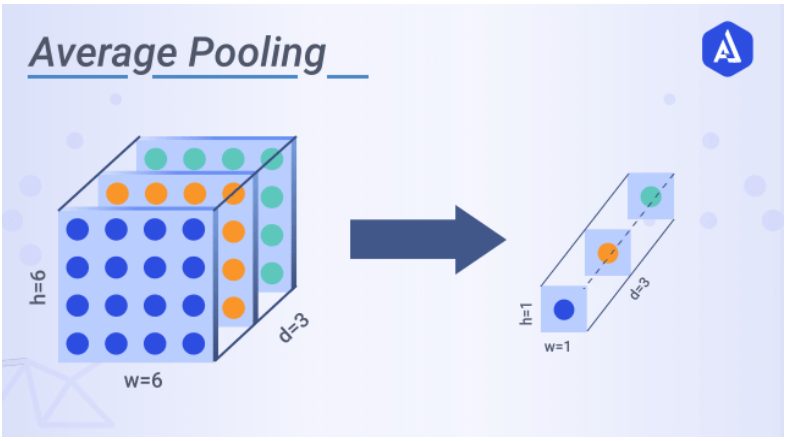


Fig 3.1.5 Average Pooling Layer

Global Pooling:

Global Pooling is the pooling layer that reduces the dimensionality of the image or feature map from three-dimensional to one-dimensional.

Fully Connected Layer:

The fully connected layer receives the final output of the stacked Convolutional layers and pooling layers. The fully connected layer is typically positioned before the output layer and is one of the last layers in a CNN design. This layer's input image is also fed to some more fully connected layers. These layers are where image classification takes place. The representation between input and output is mapped by the fully connected layer. The activation function is another important parameter of the CNN model. They are utilized to learn and imprecise any kind of continuous and complex connection between network factors. At the end of the day, it figures out which model data ought to be terminated forward and which shouldn't at the network's end. It brings nonlinearity into the network. There are several popular activation functions, including the ReLU, SoftMax, tanH, and Sigmoid functions. Each of these functions has a particular application. The sigmoid and SoftMax functions are liked for a binary classification CNN model, as well as for multi-class classification. SoftMax is commonly used.

Here we mainly used the max-pooling layer to get the prominent features from the input image.

SoftMax and ReLu are used for the multi-class classification of the CNN model.

3.2 Software Requirements

3.2.1 Libraries

Numpy

Huge, multi-layered clusters and networks are upheld by NumPy, a library for the Python programming language, alongside a significant number of undeniable level numerical tasks that may be performed on these arrays. Jim Hugunin and a number of other developers worked together to produce Numeric, the predecessor of NumPy. Travis Oliphant developed NumPy in 2005 by heavily altering Numeric to incorporate the capabilities of the rival Numarray. Numerous people have contributed to the open-source program NumPy. A project sponsored financially by NumFOCUS is NumPy.

MatplotLib

For the Python programming language and its NumPy mathematical science expansion, Matplotlib is a charting library. With the assistance of generally useful GUI tool stash like Tkinter, wxPython, Qt, or GTK, it offers an article situated API for implanting plots into applications. It isn't prescribed to utilize the procedural "pylab" interface, which depends on a state machine (like OpenGL) and was made to look like the MATLAB interface intently. Matplotlib is utilized by SciPy. With the choice to utilize Python and the advantage of being free and open-source, Matplotlib is made to be all around as easy to understand as MATLAB.

Tensor Flow

A complete open-source machine learning platform is called TensorFlow. Researchers can advance the state-of-the-art in ML thanks to its extensive, adaptable ecosystem of tools, libraries, and community resources, while developers can simply create and deploy ML-powered apps. TensorFlow was initially created for enormous mathematical calculations without remembering profound deep learning.

Keras

A Python interface for artificial neural networks is provided by the open-source software package known as Keras. The TensorFlow library interface is provided by Keras. Additionally, it permits the deployment of distributed deep learning model training on clusters of graphics processing units.

To make a managing picture and text information simpler and to smooth out the coding expected to make profound brain organizations, Keras incorporates different executions of broadly utilized brain network building pieces like layers, goals, activation functions, and optimizers. Convolutional and recurrent neural networks are supported by Keras in addition to traditional neural networks. Other widely used utility layers are supported, including dropout, batch normalization, and pooling.

Users of Keras can productize deep models on web-enabled devices, Java Virtual Machines, and smartphones (iOS and Android).

CV2

OpenCV-Python, often known as "Unofficial pre-built CPU-only OpenCV packages for Python," uses the module import name cv2. Traditional OpenCV requires unnecessary, lengthy processes that involve creating the module from scratch.

Here CV2 is used for color conversion , image resizing , to convert (or) store image in an array.

Python Imaging Library (PIL)

PIL is the Python Imaging Library which gives the python translator picture-altering capacities. This Library is chiefly used to plot the design of the trained model which is suggested by Keras.

We can likewise utilize a variety of names. On the off chance that the variety contention is excluded, the picture is loaded up with nothing (this normally compares to dark). On the off chance that the variety is None, the picture isn't instated. This can be helpful on the off chance that you will glue or attract things in the picture.

Sklearn

For the Python programming language, Sklearn is a free AI library. Support-vector machines, irregular timberlands, slope helping, k-implies, and DBSCAN are only a couple of the order, relapse, and bunching calculations it offers. It is additionally worked to work with Python's NumPy and SciPy logical and mathematical libraries.

Flask

Flask is a Python-based web application system. Flask is based on Werkzeug's WSGI tool compartment and the Jinja2 format motor.

The Web Server Gateway Interface (WSGI) has been embraced as a norm for creating Python web applications. WSGI is a norm for making a general connection point between a web server and a web application.

Jinja2 is a well-known Python templating motor. To create dynamic pages, a web templating framework consolidates a layout with a particular information source.

Werkzeug

It's a WSGI tool compartment with demands, reaction objects, and other utility capabilities. This permits you to fabricate a web structure on top of it. Werkzeug is one of the groundworks of the Flask structure.

Hist

The type of graph which is plotted between loss vs epoch and accuracy vs epoch and then the both graphs are compared where the loss acquired is low and the accuracy will be high.

fit

An automatic procedure called fitting ensures that your machine learning models have the unique parameters best suited to accurately tackle the particular real-world business problem you're trying to solve.

Epoch

The term "epoch" is applied to the number of passes the AI calculation has made across the full traiing dataset. Regularly, datasets are coordinated into groups (particularly when how much information is extremely huge).

Hyper Text Markup Language

HTML is a markup language that is utilized to make the construction of web pages that are shown on the World Wide Web (www). It incorporates Tags and Attributes that are utilized to make pages. We can likewise utilize Hyperlinks to associate various pages.

Cascading Style Sheets

A CSS record contains style decides that the program deciphers and afterward applies to the comparing components in your document.A selector and statement block involve a style rule set.

Overfitting

When a model is prepared with a lot of information, it starts to gain from the clamor and wrong information sections in our informational index. The model then neglects to accurately order the information because of an excessive number of subtleties and commotion. Non-parametric and non-straight techniques are the reasons for overfitting because these sorts

of AI calculations have more opportunity to build the model given the dataset and can make unreasonable models. If we have straight information, we can stay away from overfitting by utilizing a direct calculation, or we can utilize choice tree boundaries like the maximal profundity.

3.2.2 Activation Functions

ReLu

The rectifier or ReLU (Rectified Linear Unit) activation function is a type of activation function used in artificial neural networks that is defined as the argument's positive side:

The displaystyle f(x)=x+=max(0,x) expression

where x is a neuron's input. This is also referred to as a ramp function, and in electrical engineering, it is comparable to half-wave rectification.

Contrasted with the generally utilized activation capabilities, for example, the strategic sigmoid (which is motivated by the likelihood hypothesis), it empowers better training of deeper networks.

SoftMax

The softmax capability is a complex rendition of the strategic capability, frequently known as softargmax or standardized dramatic capability. It is regularly utilized as the last enactment capability of a brain organization to standardize the result of an organization to a likelihood dispersion over anticipated yield classes and is used in multinomial calculated relapse.

3.2.3 Optimizer

An optimizer is a method or procedure that changes brain network properties like loads and learning rates. In this manner, it assists with decreasing total loss and raising accuracy. A deep learning model typically has a considerable number of limits, making the task of picking the real loads for the model testing.It features the significance to choose an improvement calculation that is suitable for your application. Accordingly, before diving deeply into the subject, understanding these algorithms is indispensable.

Adam

Adam is a different optimization algorithm that can be used to train deep learning models instead of stochastic gradient descent. Adam creates an optimization technique that can

handle sparse gradients on noisy situations by combining the best features of the AdaGrad and RMSProp algorithms.

3.3 Data Collection

Images can be downloaded from the website called Kaggle.com using the keywords plant and disease names. Here we have used 3 crops with diseases and they are collected from PlantVillage Dataset. They are Tomato with nine diseases, Potato with two diseases and Pepper with one disease and one healthy leaf for each crop.Total Images collected are 20,639.

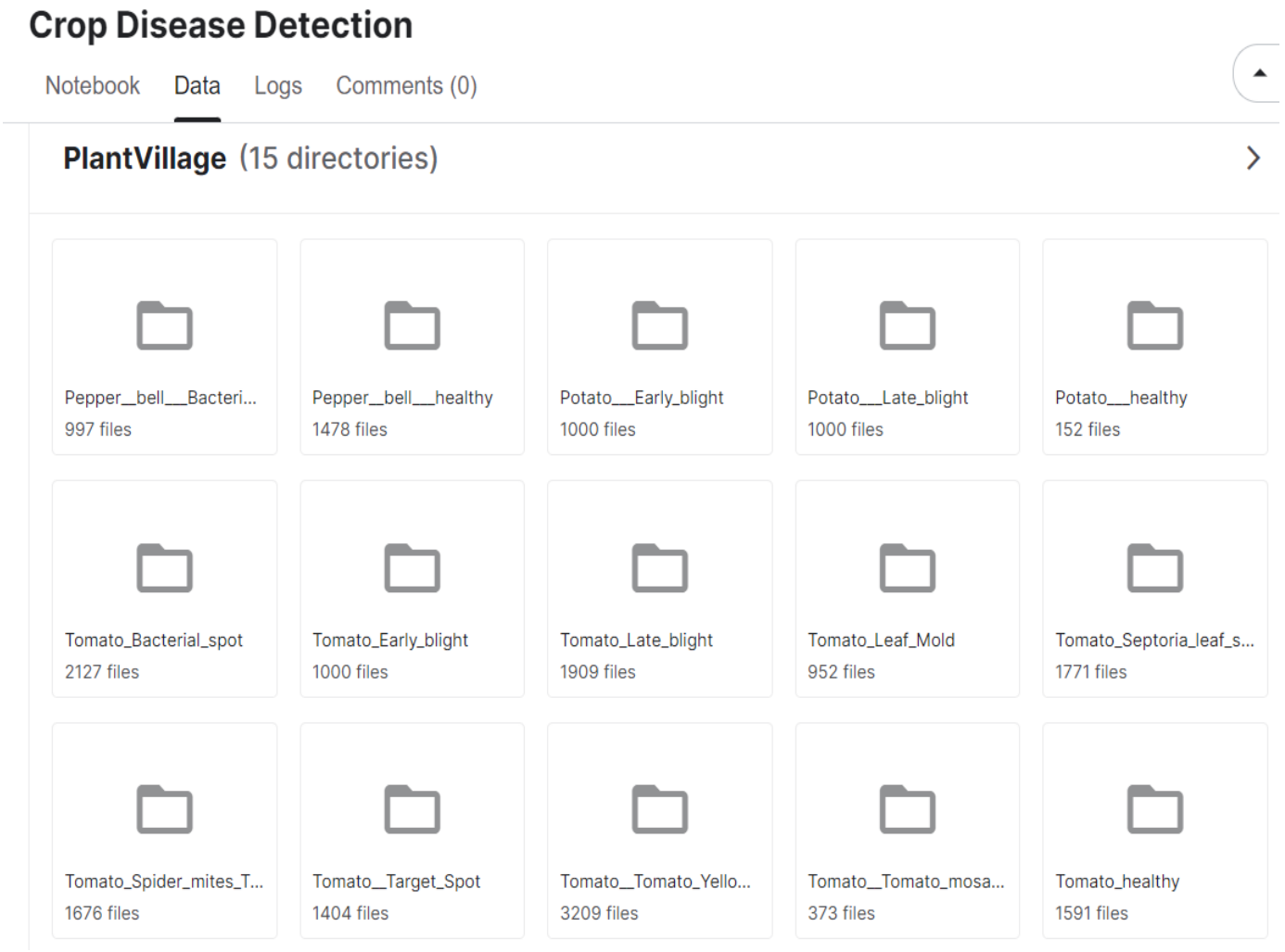


Fig 3.3.1 Dataset Collection

3.3.1 Pepper Bell

The crop plant Pepper Bell is considered for disease detection. The total number of images considered for this plant is 2475 out of which diseased leaf images considered are 997 and the healthy images considered are 1478.

S.No	Disease	No. of Images
1.	Bacterial Spot	997
2.	Healthy	1478

Table 3.3.1 Diseases for Pepper Bell

Fig 3.3.2 Pepper Bell Bacterial Spot

3.3.2 Potato

The crop plant Potato is considered for disease detection. The total number of images considered for this plant is 2152 out of which diseased leaf images considered are 2000 and the healthy images considered are 152. There are 2 diseases that are affecting the potato crop.

Fig 3.3.3 Potato Early Blight

Fig 3.3.4 Potato Late Blight

S.No	Disease	No. of images
1.	Early Blight	1000
2.	Late Blight	1000
3.	Healthy	152

Table 3.3.2 Diseases for Potato

3.3.3 Tomato

The crop plant Potato is considered for disease detection. The total number of images considered for this plant is 16012 out of which diseased leaf images considered are 14421 and the healthy images considered are 1591. There are 9 diseases that are affecting the tomato crops.

S.No	Disease	No. of images
1.	Bacterial spot	2127
2.	Early blight	1000
3.	Late blight	1909
4.	Leaf Mold	952
5.	Septoria Leaf Spot	1771
6.	Spider mites	1676
7.	Target Spot	1404
8.	Yellow Leaf Curl Virus	3209
9.	Mosaic Virus	373
10.	Healthy	1591

Table 3.3.3 Diseases for Tomato Crop

The labels given below are according to table 3.3.3 , refer the table 3.3.3 for the disease names of the tomato crop.

1

2

3

4




5 **6** **7** **8**


9 **10**

The total number of images in the database is 20639 and as mentioned above for each crop the classes are classified based on their respective diseases . The table below will give a clear picture of the image classes.

Table 3.3.4 Dataset used for Plant Disease Detection & Pesticide Prediction

S.No	**Plant Name**	**Disease**	**Class Name**	**No. of Images**
1.	Pepper Bell	Bacterial Spot	C0	997
2.	Pepper Bell - Healthy	—	C10	1478
3.	Potato	Early Blight	C13	1000
4.	Potato	Late Blight	C11	1000
5.	Potato - Healthy	—	C1	152
6.	Tomato	Bacterial Spot	C4	2127
7.	Tomato	Early Blight	C8	1000
8.	Tomato	Late Blight	C12	1909
9.	Tomato	Leaf Mold	C2	952
10.	Tomato	Septoria Leaf Spot	C5	1771

11.	Tomato	Spider mites	C7	1676
12.	Tomato	Target Spot	C9	1404
13.	Tomato	Yellow Leaf Curl Virus	C3	3209
14.	Tomato	Mosaic Virus	C14	373
15.	Tomato - Healthy	—	C6	1591

3.4 Implementation

The images are first collected from the database and then the below steps are to be followed ,refer the fig 3.4 for better understanding.

Image Acquisition

The image is chosen from the database during the training phase. The image can be acquired from the camera in real-time during the testing phase, but in this application, there is a special folder on the laptop where the image will be collected and given to the algorithm to be analyzed.

Image Pre-Processing

Image pre-processing is a technique for resizing a selected image to a specific dimension. We're going to resize the image to 64 x 64 pixels. This is critical since all of the leaves in the database should have the same size, making detection easier. Because we use colorful photos, no color conversion is required. The produced image will be given to the algorithm for testing and training purposes after pre-processing.

Training

The Convolutional Neural Network will be trained in this step to create an image categorization model. We'll use and modify the DensNet architecture to support our various categories (classes). The project's dataset is split into two sets: training, which comprises 80% of the dataset, and validation, which comprises the remaining 20%.

Testing

The test set for predicting whether a leaf is unhealthy or healthy and includes the name of the disease will be used to assess the performance of the classifier in this phase. The classifier

receives the test data as input, and based on the attributes gleaned from the train data, it determines whether the leaf is healthy or ill.

Training Loss

The training loss is a metric that actions how well a profound learning model matches the training data. That is, it assesses the model's blunder on the training data. The training set is a subset of the dataset that was utilized to initially prepare the model. The training loss is determined computationally by adding the number of errors for each example in the training set.

It's likewise significant that the training loss is determined after each cluster. The training loss is usually represented by plotting a bend.

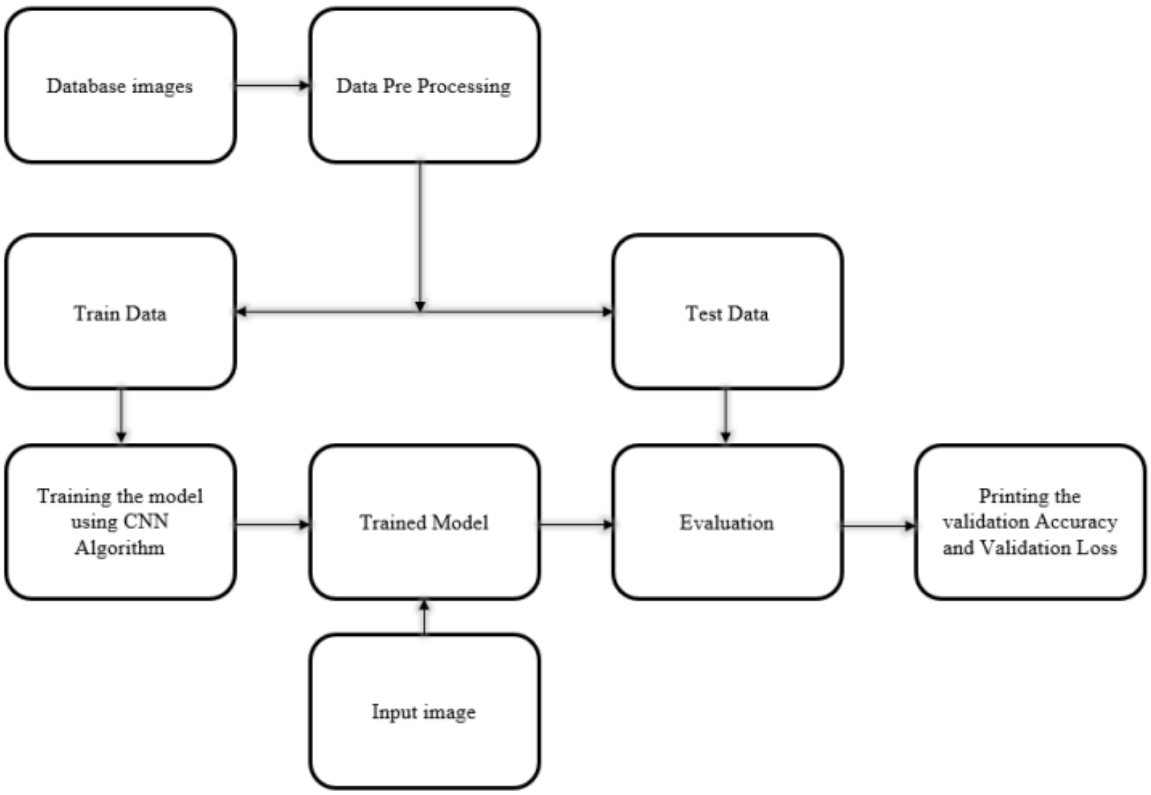


Fig 3.4.1 Block Diagram

Validation Loss

Validation loss is a measurement for assessing a deep learning model's presentation on the validation set. The validation set is a subset of the dataset set to the side to test the model's exhibition. The validation is not entirely set in stone from the amount of the slip-ups for each example in the validation set, like the training loss.

After each epoch, the validation loss is also calculated. This tells us whether the model needs to be fine-tuned or adjusted further. We normally achieve this by showing a learning curve for the validation loss.

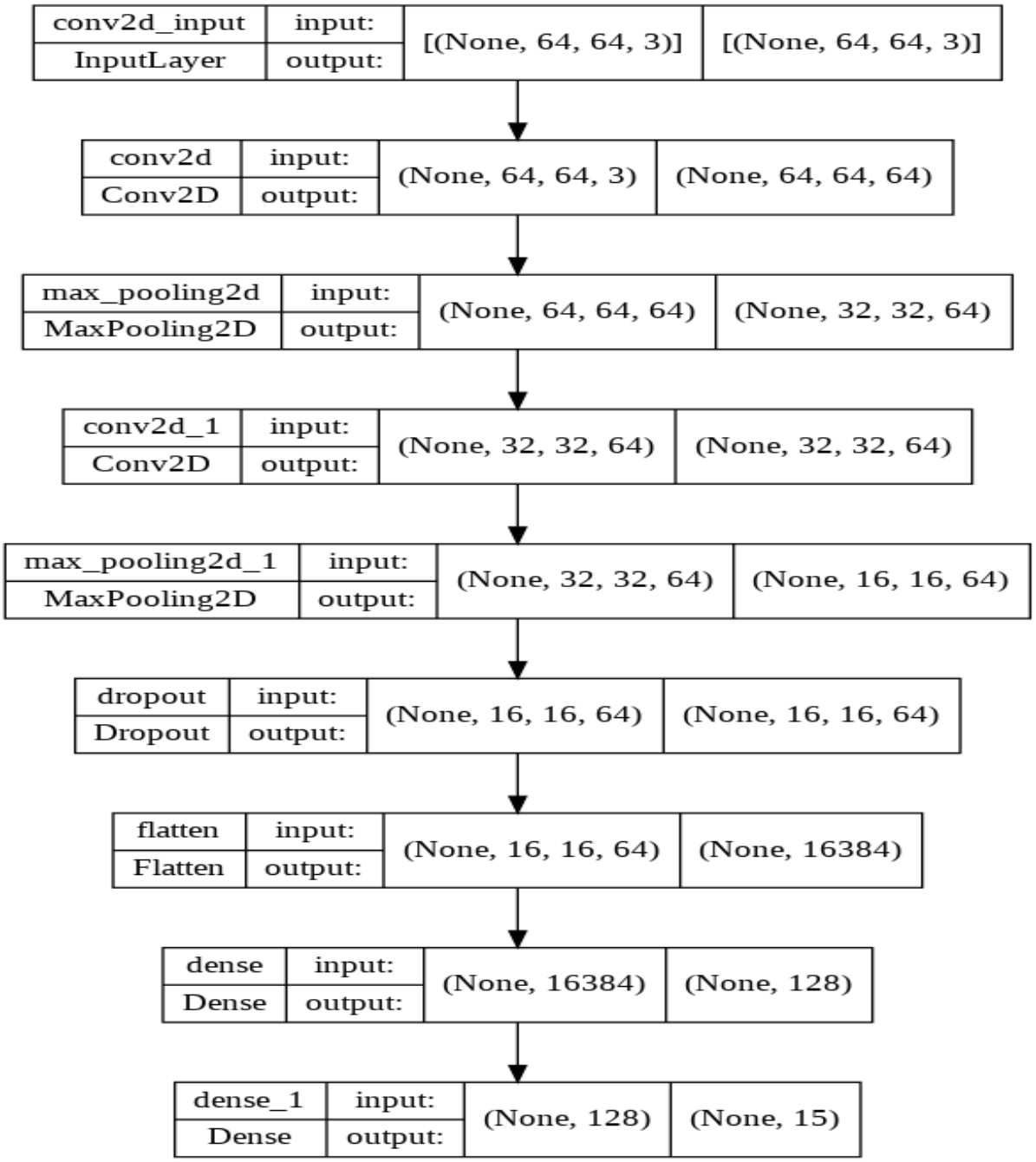
Training Accuracy

Training accuracy alludes to the prepared model recognizing free pictures that were not utilized in training, while test accuracy alludes to the prepared model distinguishing autonomous pictures that were not utilized in training.

Validation Accuracy

The validation accuracy, which you work out on the data set you don't use for training yet use (during the preparation cycle) for approving (or "testing") your model's speculation limit or for "early end," is frequently alluded to as the test (or testing) exactness.

3.4.1 CNN Architecture



Dropout

To reduce co-adaptation, we use dropout while training the Neural Network. Dropout includes haphazardly switching off a portion of a layer's neurons at each training step by focusing out the neuron values. The dropout rate, rd, is the negligible portion of neurons that

are focused out. The excess neurons' qualities are duplicated by 1/(1-rd) with the goal that the absolute amount of the neuron values stays consistent.

Flatten

In Python, we sometimes need a one-dimensional array instead of a two-dimensional or multi-dimensional array. The numpy module includes a function named numpy.ndarray.flatten() for this purpose, which returns a one-dimensional duplicate of the array rather than a two-dimensional or multi-dimensional array.

Densenet

A DenseNet is a kind of convolutional neural network that utilizes Dense Blocks to interface all layers (with matching element map sizes) straightforwardly to one another, subsequent in thick associations between layers. Each layer takes additional contributions from every single going before level and gives its own element guides to all following layers to keep up with the feed-forward nature.

CHAPTER 4
RESULTS AND DISCUSSION

A CNN model is built and trained to detect plant diseases for potato, tomato and Pepper. The CNN algorithm is chosen over other algorithms due to its high accuracy in results. This accuracy may vary contingent upon the quantity of pictures given. It can be increased by increasing the number of epochs.The number of epochs we used are 10 .

```
hist = model.fit(X_train, y_train,
          batch_size = batch_size, epochs = nb_epochs,
          verbose = 1, validation_data = (X_test, y_test))
```

```
Epoch 1/10
452/452 [==============================] - 17s 13ms/step - loss: 1.3312 - accuracy: 0.5780 - val_loss: 0.7809 - val_accuracy: 0.7492
Epoch 2/10
452/452 [==============================] - 6s 13ms/step - loss: 0.6273 - accuracy: 0.7910 - val_loss: 0.6011 - val_accuracy: 0.8018
Epoch 3/10
452/452 [==============================] - 5s 11ms/step - loss: 0.4369 - accuracy: 0.8518 - val_loss: 0.5789 - val_accuracy: 0.8029
Epoch 4/10
452/452 [==============================] - 4s 9ms/step - loss: 0.3493 - accuracy: 0.8827 - val_loss: 0.4519 - val_accuracy: 0.8499
Epoch 5/10
452/452 [==============================] - 5s 11ms/step - loss: 0.2900 - accuracy: 0.9007 - val_loss: 0.3181 - val_accuracy: 0.8939
Epoch 6/10
452/452 [==============================] - 5s 11ms/step - loss: 0.2311 - accuracy: 0.9222 - val_loss: 0.3690 - val_accuracy: 0.8826
Epoch 7/10
452/452 [==============================] - 4s 9ms/step - loss: 0.2016 - accuracy: 0.9310 - val_loss: 0.4238 - val_accuracy: 0.8695
Epoch 8/10
452/452 [==============================] - 4s 9ms/step - loss: 0.1776 - accuracy: 0.9391 - val_loss: 0.3322 - val_accuracy: 0.8916
Epoch 9/10
452/452 [==============================] - 4s 9ms/step - loss: 0.1471 - accuracy: 0.9487 - val_loss: 0.3974 - val_accuracy: 0.8738
Epoch 10/10
452/452 [==============================] - 4s 9ms/step - loss: 0.1393 - accuracy: 0.9528 - val_loss: 0.2869 - val_accuracy: 0.9116
```

Fig 4.1 Accuracy for 15 class labels combined together

When the test data is given as input to the model with the number of epochs provided is 10, the accuracy achieved for 15 class labels together is 95%.

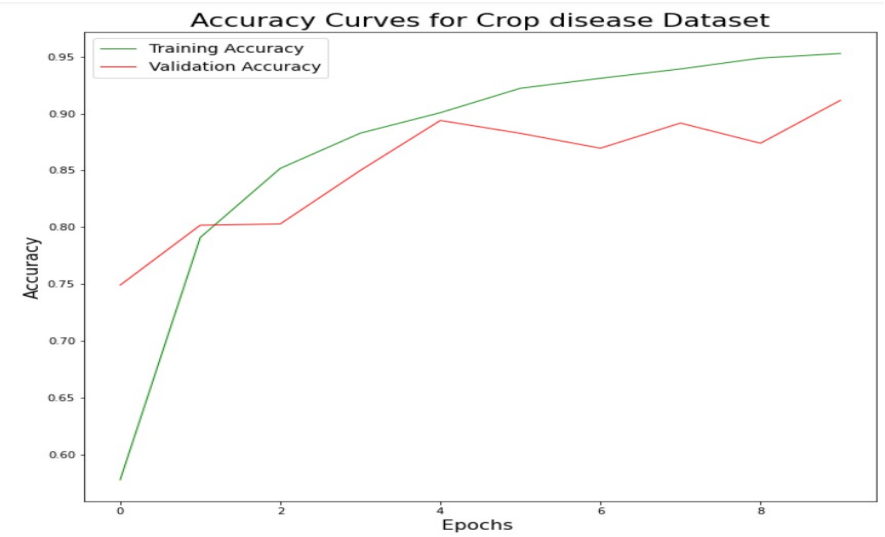


Fig 4.2 Accuracy Vs Epochs

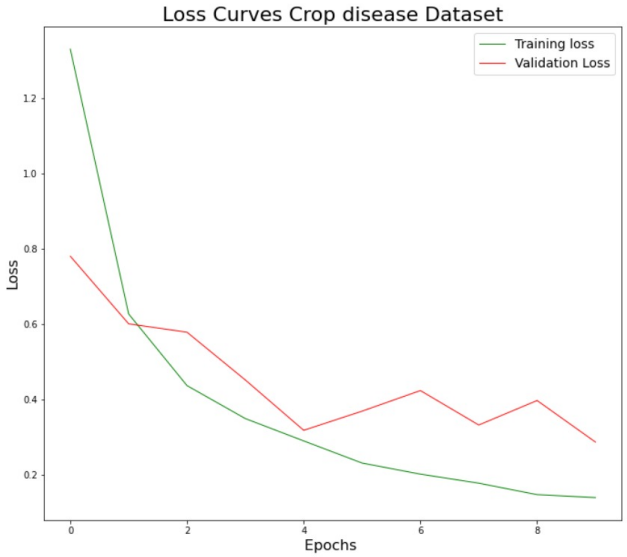


Fig 4.3 Loss Vs Epochs

The Accuracy Vs Epoch graph represents the validation accuracy and training accuracy where accuracy increases by increasing the epochs

The Loss Vs Epoch graph represents the validation loss and training loss where loss decreases from first epoch to last epoch

Here the adam optimizer will reduce the loss by acting as an optimizer.

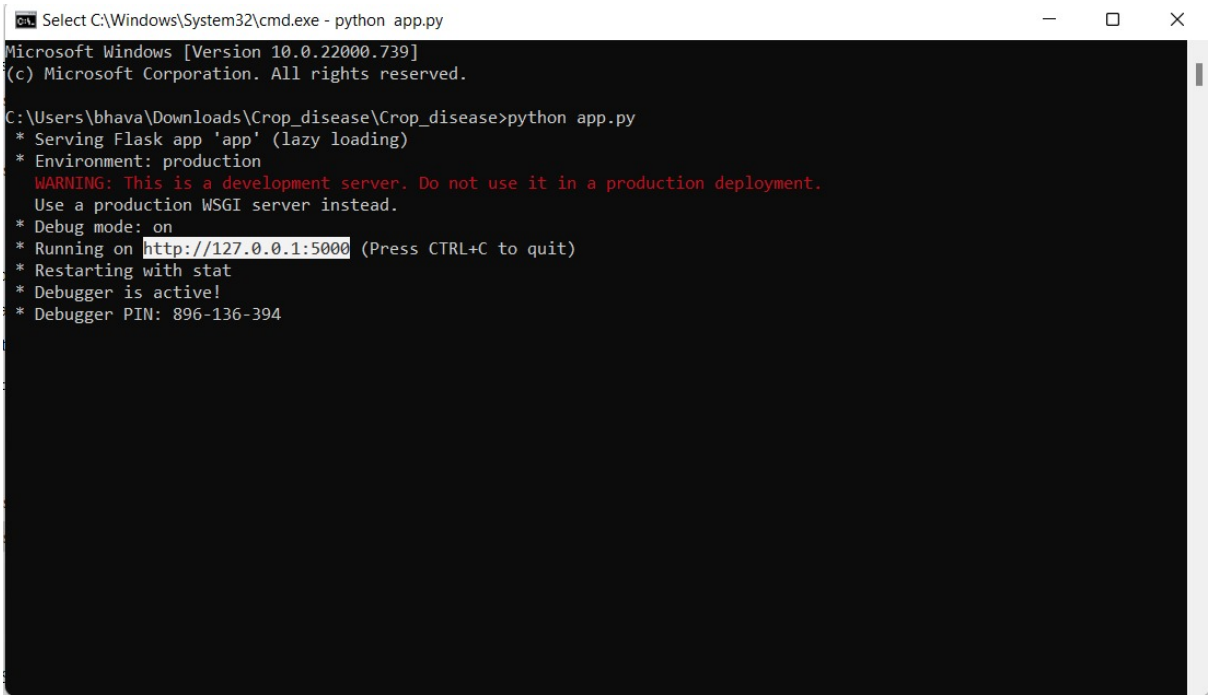
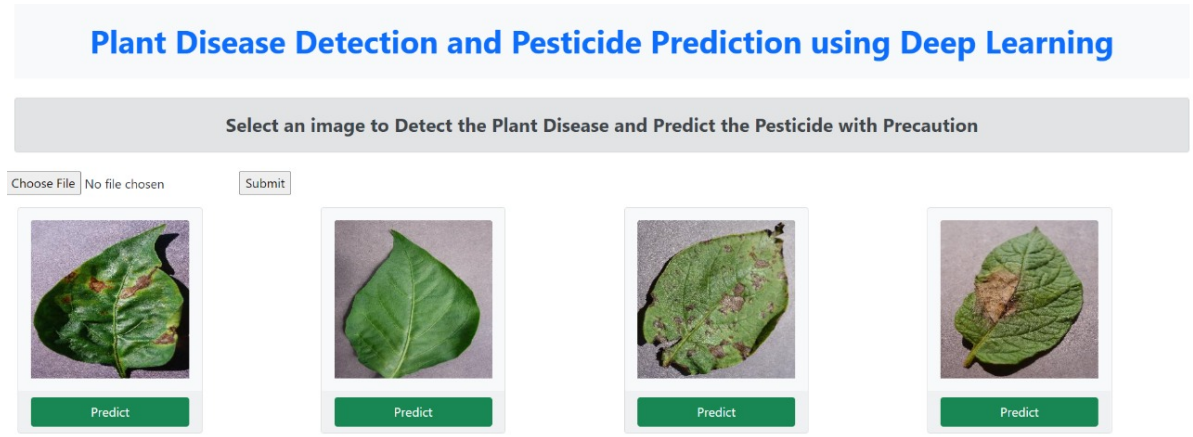


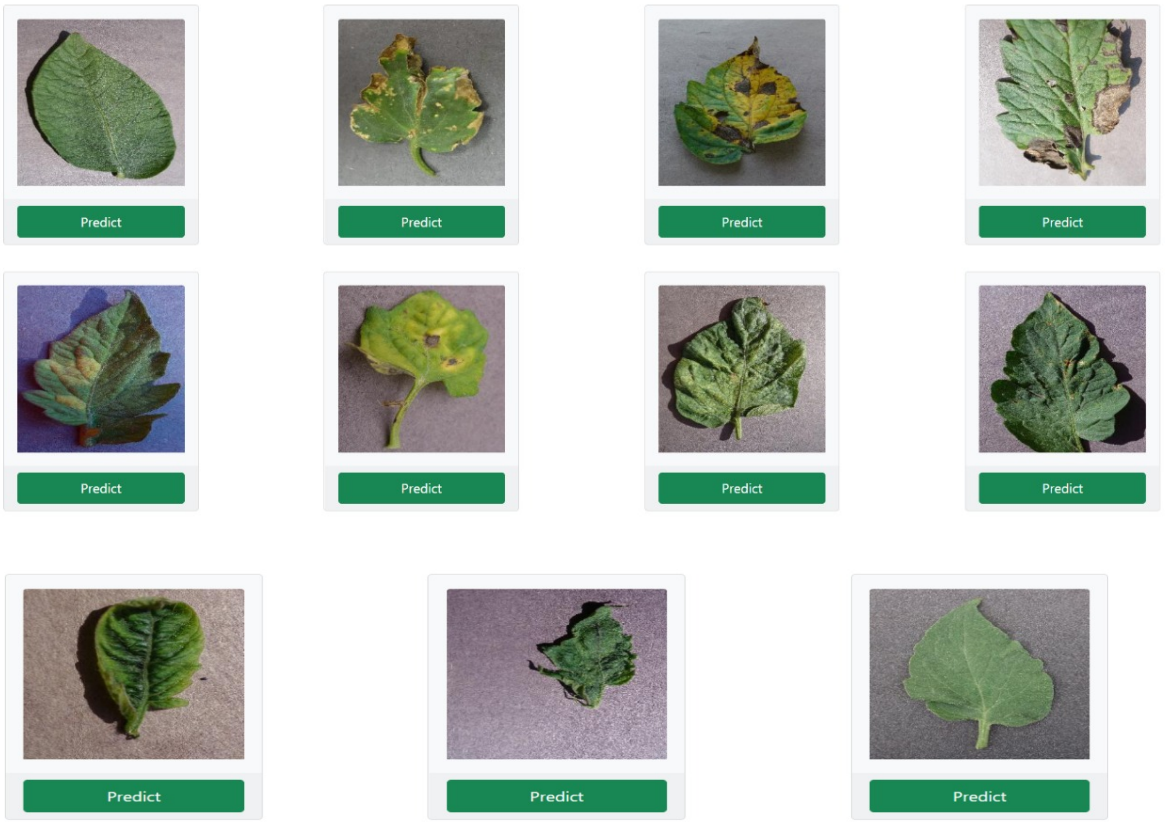
Fig 4.4 Command Prompt

The path **http: //127.0.0.1:5000** from Command Prompt will be copied and pasted in to the web browser as shown below

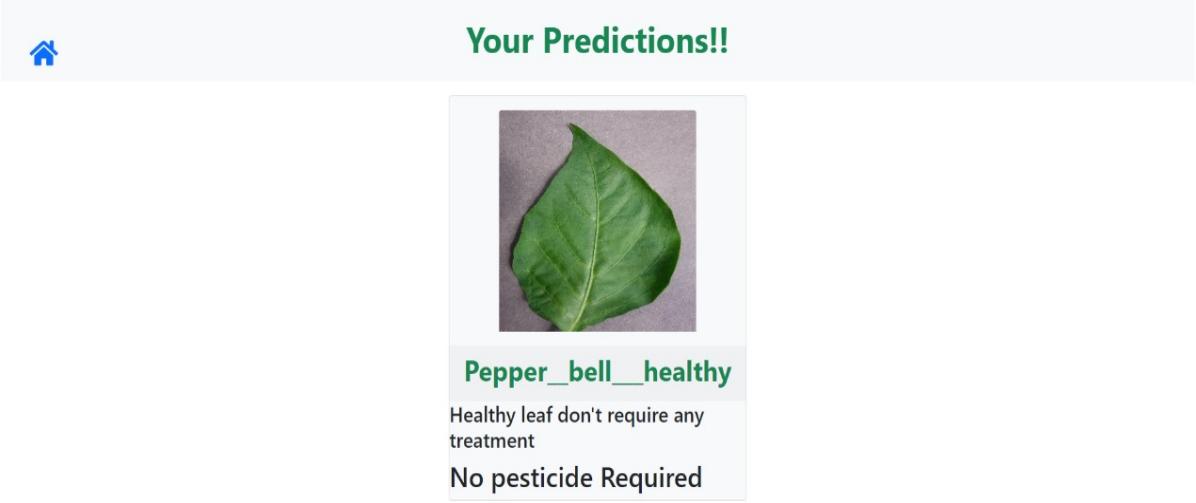
Fig 4.5 Path or IP Address

After giving the path the web application will be opened in the web browser as shown below





The above images represent 15 class labels (Table 3.3.4) which are used in this project and each label refers to different crops like Pepper Bell , Potato and Tomato . Giving the data/image to “Choose File” option and upon submitting the trained model , it will be able to detect the disease and predict the pesticide along with the precaution as shown below.



Your Predictions!!

Potato___Early_blight

Alternaria solani is a fungal pathogen that produces a disease in tomato and potato plants called early blight. The pathogen produces distinctive bullseye patterned leaf spots and can also cause stem lesions and fruit rot on tomato and tuber blight on potato

Penthiopyrad

CHAPTER 5
CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

The suggested framework assumes that deep learning is a fundamental mechanism for achieving reasonable and strong solutions. The images of leaf diseases are organised using a convolution neural system. With the highest level of precision, it distinguishes between infected and non-infected leaves. The proposed framework is effective due to the multiclass classification of various diseases and healthy leaves from yields like pepper, potato, and tomato. The method detects diseases in their early stages and alerts farmers to the necessary actions to be done. The technique greatly facilitates the development of high-quality crops and reduces the losses experienced by farmers as a result of contaminated crops. It ensures that a healthy crop is produced at the end of each cycle.

5.2 Future Scope

- For efficient result processing, the disease detection system can be integrated into a cloud system.
- Integration of an automated disease detection system with soil measurement sensors.

REFERENCES

1. Pranali K. Kosamkar, V.Y.Kulkarni, Krushna Mantri, Shubham Rudrawar, Shubhan Salmpuria, Nishant Gadekar “Leaf disease detection and recommendation of pesticides”, 2018 Fourth ICCUBEA 2018

2. M. Bhange and H. A. Hingoliwala, "Smart Farming: Pomegranate Disease Detection Using Image Processing", Procedia Computer Science, vol. 58, pp. 280-288, 2015.

3. S. Ramesh and D. Vydeki, "Rice blast disease detection and classification using machine learning algorithm", Proceedings - 2nd International Conference on Micro-Electronics and Telecommunication Engineering ICMETE 2018, pp. 255-259, 2018.

4. V. Singh and A. K. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques", Information Processing in Agriculture, vol. 4, no. 1, pp. 41-49,2017.

5. S. S. Chouhan, A. Kaul and U. P. Singh, "A deep learning approach for the classification of diseased plant leaf images", Proceedings of the 4th International Conference on Communication and Electronics Systems ICCES 2019, no. Icces, pp. 1168-1172, 2019.

6. M. Arsenovic, M. Karanovic, S. Sladojevic, A. Anderla and D. Stefanovic, "Solving current limitations of deep learning based approaches for plant disease detection", Symmetry, vol. 11,no. 7, 2019.

7. H. Durmus, E. O. Gunes and M. Kirci, "Disease detection on the leaves of the tomato plants by using deep learning", 2017 6th International Conference on Agro-Geoinformatics Agro-Geoinformatics 2017, 2017.

8. S. S. Shinde and M. Kulkarni, "Review Paper on Prediction of Crop Disease Using IoT and Machine Learning", International Conference on Transforming

Engineering Education ICTEE 2017, pp. 5-8, 2018.

9. J. P. Shah, H. B. Prajapati and V. K. Dabhi, "A survey on detection and classification of rice plant diseases", 2016 IEEE International Conference on Current Trends in Advanced Computing ICCTAC 2016, pp. 1-8, 2016.

10. Y. Fang and R. P. Ramasamy, "Current and prospective methods for plant disease detection", Biosensors, vol. 5, no. 3, pp. 537-561, 2015.

11. A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey", Computers and Electronics in Agriculture, vol. 147, no. July 2017, pp. 70-90, 2018.

12. K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis", Computer and Electronics in Agriculture, vol. 145, no. January, pp. 311-318, 2018.

13. Y. Lu, S. Yi, N. Zeng, Y. Liu and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks", Neurocomputing, vol. 267, pp. 378-384, 2017.

14. Halil Durmuu, Ece Olcay Güneu, “Disease Detection on the Leaves of the TomatoPlants by Using DeepLearning,” I.T.U. TARBIL Environmental Agriculture Informatics Applied Research Center, 2017.

15. Vyshnavi.G.K.P, Sirpa.M.R , “Healthy and Unhealthy PlantLeaf Identification and Classification Using HierarchicalClustering,” IRJET, Vol.03, 2016.

Fig A.1 Project Expo at Show And Tell

APPENDIX A

PROGRAMMING

Connecting Google Colab with Kaggle in order to to download the data from kaggle.

```
# Run this cell and select the kaggle.json file downloaded
# from the Kaggle account settings page.
#
from google.colab import files
files.upload()
```

Choose Files kaggle.json

- **kaggle.json**(application/json) - 66 bytes, last modified: 6/7/2022 - 100% done

```
Saving kaggle.json to kaggle (1).json
{'kaggle.json': b'{"username":"karthik12s","key":"56a984ed14e695d673666653747fb61e"}'}
```

```
[3] !ls -lha kaggle.json
```

```
-rw-r--r-- 1 root root 66 Jun 19 15:47 kaggle.json
```

```
[4] !pip install -q kaggle
```

```
[5] # The Kaggle API client expects this file to be in ~/.kaggle,
    # so move it there.
    !mkdir -p ~/.kaggle
    !cp kaggle.json ~/.kaggle/

    # This permissions change avoids a warning on Kaggle tool startup.
```

```
[6] !kaggle datasets list
```

ref	title	size	lastUpdated	downloadCount	voteCount	usabilityRating
victorsoeiro/netflix-tv-shows-and-movies	Netflix TV Shows and Movies	2MB	2022-05-15 00:01:23	8968	277	1.0
devansodariya/student-performance-data	Student Performance Dataset	7KB	2022-05-26 13:55:09	4628	152	0.9705882
iamsouravbanerjee/software-professional-salaries-2022	Salary Dataset - 2022	526KB	2022-06-15 17:13:05	1735	46	1.0
paradisejoy/top-hits-spotify-from-20002019	Top Hits Spotify from 2000-2019	94KB	2022-05-31 07:20:57	6907	163	1.0
surajjha101/stores-area-and-sales-data	Supermarket store branches sales analysis	10KB	2022-04-29 11:10:16	6893	189	1.0
mayureshkoli/police-deaths-in-usa-from-1791-to-2022	Police deaths in USA from 1791 to 2022	698KB	2022-06-09 17:47:57	827	29	1.0
saddamazyazy/go-to-college-dataset	Go To College Dataset	12KB	2022-05-20 10:46:02	1816	46	1.0
gunapro/student-behavior	Student Behavior	5KB	2022-06-03 13:45:54	1565	43	0.8235294
odins0n/amex-parquet	Amex Competition Data in Parquet Format	9GB	2022-05-25 23:20:19	944	124	1.0
ahmedshahriarsakib/usa-real-estate-dataset	USA Real Estate Dataset	5MB	2022-06-04 21:40:25	953	36	1.0
kkhandekar/cost-of-living-index-by-city-2022	Cost of Living Index by City 2022	15KB	2022-06-03 02:50:24	927	30	1.0
ruchi798/parquet-files-amexdefault-prediction	Feather & Parquet Files : AMEX-Default Prediction	22GB	2022-05-26 05:46:53	671	102	1.0
azminetoushikwasi/ucl-202122-uefa-champions-league	UCL ⚽ 2021-22 🌟 Players Data | Champions League	55KB	2022-06-13 08:36:43	1485	48	1.0
prasertk/cities-with-the-best-worklife-balance-2022	Cities with the Best Work-Life Balance 2022	5KB	2022-06-01 09:15:52	810	29	1.0
kuchhbhi/play-store-apps-may2022	Play Store Apps May-2022	468KB	2022-05-19 07:57:19	826	29	1.0
aryashah2k/sample-students-grades-dataset-tutorial-notebook	Sample Students Grades Dataset Tutorial Notebook	360B	2022-06-07 09:11:25	250	27	1.0
dansbecker/melbourne-housing-snapshot	Melbourne Housing Snapshot	451KB	2018-06-05 12:52:24	89136	1092	0.7058824
imoore/age-dataset	Age dataset: life, work, and death of 1.22M people	34MB	2022-06-07 08:56:52	675	51	1.0
ankanhore545/dropout-or-academic-success	Predict Dropout or Academic Success	103KB	2022-06-05 11:32:43	617	28	0.9117647
odins0n/top-20-play-store-app-reviews-daily-update	Top 20 Play Store App Reviews (Daily Update)	16MB	2022-06-18 22:21:15	130	24	1.0

```
[7] !kaggle datasets download -d emmarex/plantdisease
```

```
Downloading plantdisease.zip to /content
 96% 628M/658M [00:02<00:00, 321MB/s]
100% 658M/658M [00:02<00:00, 275MB/s]
```

```
[8]  # kaggle datasets download -d emmarex/plantdisease
```

```
[9]  !unzip \*.zip
```

```
  inflating: plantvillage/PlantVillage/Tomato_healthy/e0987875-8fe2-49ee-8f01-e29281c3d790___RS_HL 0239.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e0b87789-7d59-4bd1-bc78-ec3a942021f4___RS_HL 9894.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e0d2d9f6-b29e-4cd8-8c2b-7d462271ceb3___RS_HL 9787.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e0d4c623-8305-4ded-8164-d8fcd5bc2016___RS_HL 0448.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e110eaa3-3f80-477f-ad86-18b7e308b764___RS_HL 0349.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e12936ac-7354-434c-b857-1e742e383dd5___RS_HL 9878.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1400c81-62df-44b6-97fd-e210b66dae0b___RS_HL 9672.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1436a17-dca7-4687-a0ec-31a28389063e___RS_HL 0529.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1517f9b-6bab-415b-b8f5-1f52dd544b2d___RS_HL 0203.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1571f83-5f00-4653-a176-2033f27405fa___GH_HL Leaf 353.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1860bf2-2656-4dba-b9de-f80932bca552___RS_HL 0398.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1959167-b7b2-49e3-b5ce-4c5665022ada___RS_HL 0127.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1aeb847-e87b-4f24-aca7-c6e06b6c07ca___GH_HL Leaf 185.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1d129fb-f5ae-41e1-b097-445d792e4beb___RS_HL 0473.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e1eaf28e-7726-4e87-a1f4-f1ccd44355d7___RS_HL 0405.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2085395-2d3c-4698-ad7f-da9fdc73c4d1___GH_HL Leaf 254.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e225eb60-d018-4014-a4f5-eadd0aad06ef___GH_HL Leaf 438.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e25bc3a8-dd00-4c02-9f4e-81e6cf682729___RS_HL 0101.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e283f6ea-c756-47ca-ac84-57a38138b571___GH_HL Leaf 343.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2869c8b-aedf-4074-bb99-28e6ed2b4223___RS_HL 9673.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e28f30cb-755a-459b-b653-96d44b3b01b5___RS_HL 9926.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2991a66-412d-4841-8dc0-524e38338a82___GH_HL Leaf 517.1.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2bd556e-fe0f-4a97-819c-83dbc4a61ac5___GH_HL Leaf 429.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2bde260-6fac-4259-8b88-c7378f1ea730___RS_HL 0511.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2cd8a78-ae65-4743-9ecf-839647cac550___RS_HL 9833.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e2f520b1-ac21-445b-9a3f-cb6e506e3044___RS_HL 0381.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e30f24b2-c298-4853-8fda-9e67b3738a55___GH_HL Leaf 287.JPG
  inflating: plantvillage/PlantVillage/Tomato_healthy/e336eb74-a19d-4461-871e-3124c5e0d047___RS_HL 0558.JPG
```

Processing

```
[10] import numpy as np  #array manipulation
     # import pandas as pd  #sheets
     import tensorflow as tf
     from tensorflow import keras # framework for deep learning applications
     import os # to access files
     import cv2 # image manipulation
     from PIL import Image # image plot
     import random # for shuffling
```

```
[11] label_dict={0:'Pepper__bell___Bacterial_spot', 1:'Potato___healthy', 2:'Tomato_Leaf_Mold',
                 3:'Tomato__Tomato_YellowLeaf__Curl_Virus', 4:'Tomato_Bacterial_spot', 5:'Tomato_Septoria_leaf_spot',
                 6:'Tomato_healthy', 7:'Tomato_Spider_mites_Two_spotted_spider_mite', 8:'Tomato_Early_blight',
                 9:'Tomato__Target_Spot', 10:'Pepper__bell___healthy', 11:'Potato___Late_blight', 12:'Tomato_Late_blight',
                 13:'Potato___Early_blight', 14:'Tomato__Tomato_mosaic_virus'}
```

```
[12] pesticides = {
       "Apple___Apple_scab": ["Choose resistant varieties when possible. Rake under trees and destroy infected leaves to reduce the number of fungal spores available to start the disease c
         "liquid copper soap"
       ],


       "Apple___Black_rot": ["Hopefully, you are now well aware of the importance of getting rid of these sources of fungal infection.  This is the primary method of control. Remove the ca
         "captan and sulphar products(lime sulphur)"
       ],
```

```
[13] pesticides = {'Apple___Apple_scab': ['Choose resistant varieties when possible. Rake under trees and destroy infected leaves to reduce the number of fungal spores available to start t
       'liquid copper soap'],
      'Apple___Black_rot': ['Hopefully, you are now well aware of the importance of getting rid of these sources of fungal infection.  This is the primary method of control. Remove the car
       'captan and sulphar products(lime sulphur)'],
      'Apple___Cedar_apple_rust': ['Choose resistant cultivars when available. Rake up and dispose of fallen leaves and other debris from under trees. Remove galls from infected junipers.
       'myclobutanil(Immunox)'],
      'Apple___healthy': ["Healthy leaf don't require any treatment",
       'No pesticide Required'],
      'Blueberry___healthy': ["Healthy leaf don't require any treatment",
       'No pesticide Required'],
      'Cherry_(including_sour)___Powdery_mildew': ['The key to managing powdery mildew on the fruit is to keep the disease off of the leaves.  Most synthetic fungicides are preventative, r
       'copper based fungicide(green cure fungicide)'],
      'Cherry_(including_sour)___healthy': ["Healthy leaf don't require any treatment",
       'No pesticide Required'],
      'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot': ['Plant corn hybrids with resistance to the disease; crop rotation and plowing debris into soil may reduce levels of inoculum ir
       'foliar fungicide'],
      'Corn_(maize)___Common_rust_': ['The most effective method of controlling the disease is to plant resistant hybrids; application of appropriate fungicides may provide some degree on
       'stratego YLD'],
      'Corn_(maize)___Northern_Leaf_Blight': ['Follow proper tillage to reduce fungus inoculum from crop debris. Follow crop rotation with non host crop. Grow available resistant varieties
       'Delaro® fungicide'],
      'Corn_(maize)___healthy': ["Healthy leaf don't require any treatment",
       'No pesticide Required'],
      'Grape___Black_rot': ['Black rot (Guignardia bidwelli) is a fungal disease that occurs in grapes grown in a hot and humid climate. This fungal disease has the ability to ruin an enti
       'mancozeb + mycobutanil, imidacloprid or azadirachtin'],
      'Grape___Esca_(Black_Measles)': ['The fungi Phaeoacremonium aleophilum, Phaeomoniella chlamydospora[1] and Fomitiporia mediterranea[2] are associated with the disease.',
       'Bio - Tam 2.0'],
      'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)': ['The pathogen survives in infected plant residue in soil and seed borne.Favourable conditionsThe disease is more prevalent during June-
       'streptocycline(500 ppm)'],
      'Grape___healthy': ["Healthy leaf don't require any treatment",
       'No pesticide Required'],
```

```
[14] def get_label(x):
         return label_dict[x]

[15] # categories

[16] np.shape(cv2.imread("/content/plantvillage/PlantVillage/Potato___Early_blight/001187a0-57ab-4329-baff-e7246a9edeb0___RS_Early.B 8178.JPG"))

     (256, 256, 3)

[17] # categories

[18] decode_label_dict = {}
     for i in label_dict:
       decode_label_dict[label_dict[i]] = i

[19] decode_label_dict

     {'Pepper__bell___Bacterial_spot': 0,
      'Pepper__bell___healthy': 10,
      'Potato___Early_blight': 13,
      'Potato___Late_blight': 11,
      'Potato___healthy': 1,
      'Tomato_Bacterial_spot': 4,
      'Tomato_Early_blight': 8,
      'Tomato_Late_blight': 12,
      'Tomato_Leaf_Mold': 2,
      'Tomato_Septoria_leaf_spot': 5,
```

```
[20] path='/content/plantvillage/PlantVillage/'
     categories=os.listdir(path)
     training=[]
     categories
     for category in categories:
         class_num=decode_label_dict[category]
         for img in os.listdir(path+category):
             if img.endswith('.jpg') or img.endswith('.JPG'):
                 img_array=cv2.imread(path+category+'/'+img)
                 new_array=cv2.resize(img_array,(64,64)) # to reduce the bulkness of model
                 training.append([new_array, class_num])
     training[0]
```

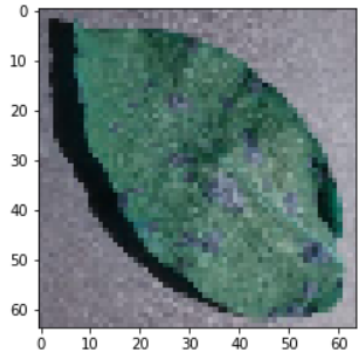
```
[array([[[112, 111, 121],
         [102, 100, 111],
         [117, 115, 128],
         ...,
         [157, 153, 164],
         [152, 148, 159],
         [147, 143, 154]],

        [[111, 109, 120],
         [112, 111, 121],
         [110, 109, 119],
         ...,
         [139, 135, 146],
         [152, 148, 159],
         [149, 145, 156]],

        [[108, 106, 118],
         [100,  99, 109],
```

```
[21] import matplotlib.pyplot as plt
     plt.imshow(training[0][0])
```

```
<matplotlib.image.AxesImage at 0x7f8c891fead0>
```




```
[22] np.shape(training[0][0])
```

```
(64, 64, 3)
```

```
[23] random.shuffle(training)
```

```
[24] np.shape(training[0][0])
```

```
(64, 64, 3)
```

```
[25] X=[]
     y=[]
     for features, label in training:
         X.append(features)
         y.append(label)
     # x1 = np.array(X)
     X=np.array(X).reshape(-1,64,64,3)
```

```
[26] np.shape(X)

     (20636, 64, 64, 3)
```

```
[27] X[0]

     array([[[101,  97, 108],
             [ 99,  95, 106],
             [112, 108, 119],
             ...,
             [146, 144, 150],
             [142, 140, 146],
             [138, 136, 142]],

            [[111, 107, 118],
             [113, 109, 120],
             [115, 111, 122],
             ...,
             [147, 145, 151],
             [148, 146, 152],
             [149, 147, 153]],
```

```
[28] y[0]

     13
```

```
[29] X=X.astype('float32')
     X/=255
     from keras.utils import np_utils
     Y=np_utils.to_categorical(y,15)
     print(Y[100])
     print(Y.shape)

     [0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
     (20636, 15)
```

```
[30] Y[0]

     array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
           dtype=float32)
```

```
[31] from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 42)
```

Model Building

```
[32] from keras.models import Sequential
     from keras.layers.core import Dense, Activation, Dropout, Flatten
     from keras.layers.convolutional import Convolution2D, MaxPooling2D
     from tensorflow.keras.optimizers import Adam
```

```
[33] model=tf.keras.Sequential([
         tf.keras.layers.Conv2D(64,(3,3), padding='same', activation=tf.nn.relu,input_shape=(64,64,3)),
         tf.keras.layers.MaxPooling2D((2, 2), strides=2),
         tf.keras.layers.Conv2D(64, (3,3), padding='same', activation=tf.nn.relu),
         tf.keras.layers.MaxPooling2D((2, 2), strides=2),
         tf.keras.layers.Dropout(0.5),
         tf.keras.layers.Flatten(),
         tf.keras.layers.Dense(128, activation=tf.nn.relu),
         tf.keras.layers.Dense(15,  activation=tf.nn.softmax)
     ])
```

```
[34] model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 64)        1792

 max_pooling2d (MaxPooling2D  (None, 32, 32, 64)       0
```

```
[34] _________________________________________________________________
      Layer (type)                Output Shape              Param #
     =================================================================
      conv2d (Conv2D)             (None, 64, 64, 64)        1792

      max_pooling2d (MaxPooling2D  (None, 32, 32, 64)       0
      )

      conv2d_1 (Conv2D)           (None, 32, 32, 64)        36928

      max_pooling2d_1 (MaxPooling  (None, 16, 16, 64)       0
      2D)

      dropout (Dropout)           (None, 16, 16, 64)        0

      flatten (Flatten)           (None, 16384)             0

      dense (Dense)               (None, 128)               2097280

      dense_1 (Dense)             (None, 15)                1935

     =================================================================
     Total params: 2,137,935
     Trainable params: 2,137,935
     Non-trainable params: 0
     _________________________________________________________________
```

```
[35] from IPython.display import Image
     from tensorflow.keras.utils import plot_model
     plot_model(model, show_shapes=True, show_layer_names=True, to_file='model.png')
     # Image(retina=True, filename='model.png')
```

```
[36] batch_size = 32
     # nb_classes =4
     nb_epochs = 10
     img_rows, img_columns = 64, 64
     img_channel = 3
     # nb_filters = 32
     nb_pool = 2
     nb_conv = 3
```

```
[37] model.compile(optimizer='Adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

```
[38] X_train=np.array(X_train)
     y_train=np.array(y_train)
     X_test=np.array(X_test)
     y_test=np.array(y_test)
```

```
[39] hist = model.fit(X_train, y_train,
               batch_size = batch_size, epochs = nb_epochs,
               verbose = 1, validation_data = (X_test, y_test))
```

```
Epoch 1/10
452/452 [==============================] - 17s 12ms/step - loss: 1.2969 - accuracy: 0.5826 - val_loss: 0.7091 - val_accuracy: 0.7666
Epoch 2/10
452/452 [==============================] - 5s 11ms/step - loss: 0.5985 - accuracy: 0.8006 - val_loss: 0.4767 - val_accuracy: 0.8412
Epoch 3/10
452/452 [==============================] - 4s 10ms/step - loss: 0.4277 - accuracy: 0.8537 - val_loss: 0.4937 - val_accuracy: 0.8364
Epoch 4/10
452/452 [==============================] - 5s 12ms/step - loss: 0.3570 - accuracy: 0.8809 - val_loss: 0.3779 - val_accuracy: 0.8774
```

```
[40] import matplotlib.pyplot as plt

       # Loss Curves
     history = hist
     plt.figure(figsize=(25, 10))
     plt.subplot(1, 2, 1)
     plt.plot(history.history['loss'],'-g',linewidth=1.0)
     plt.plot(history.history['val_loss'],'r',linewidth=1.0)
     plt.legend(['Training loss', 'Validation Loss'],fontsize=14)
     plt.xlabel('Epochs ',fontsize=16)
     plt.ylabel('Loss',fontsize=16)
     plt.title('Loss Curves Crop disease Dataset',fontsize=22)
     plt.savefig('Loss_curves.png')
     # Accuracy Curves
     plt.subplot(1, 2, 2)
     plt.plot(history.history['accuracy'],'-g',linewidth=1.0)
     plt.plot(history.history['val_accuracy'],'r',linewidth=1.0)
     plt.legend(['Training Accuracy', 'Validation Accuracy'],fontsize=14)
     plt.xlabel('Epochs ',fontsize=16)

     plt.ylabel('Accuracy',fontsize=16)
     plt.title('Accuracy Curves for Crop disease Dataset',fontsize=22)
     plt.savefig('Accuracy_curves.png')
     plt.show()
```

```
[41] score = model.evaluate(X_test, y_test, verbose = 0 )
     print("Test Score: ", score[0])
     print("Test accuracy: ", score[1])
```

```
Test Score:  0.2980339229106903
Test accuracy:  0.9118074774742126
```

```
[42] def get_pesticide_name(arr):
         arr = list(arr)
         ind = arr.index(max(arr))
         print("Disease Name :" ,label_dict[ind])
         for i in pesticides[label_dict[ind]]:
           print(i)
```

```
[43] y = model.predict(X_test).round(2)
```

```
[44] y[2]
```

```
array([0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
      dtype=float32)
```

```
[45] get_pesticide_name(y[14])
```

```
Disease Name : Potato___healthy
Healthy leaf don't require any treatment
No pesticide Required
```

```
[46] y_test[0]
```

```
7
```

```
[46]
```

```
[47] y[0]
```

```
array([0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
      dtype=float32)
```

```
[47]
```

```
[48] get_pesticide_name(y[4])
```

```
Disease Name : Tomato__Target_Spot
Do not plant new crops next to older ones that have the disease.Plant as far as possible from papaya, especially if leaves have small angular spots (Photo 5).Check all seedlings in the
Chlorothalonil
Azoxystrobin
```

```
[49] # for i in pesticides:
     #   if str(type(pesticides[i]))!= "<class 'list'>":
     #     pesticides[i] = [pesticides[i],"No pesticide Required"]
     #     print(i,str(type(pesticides[i])))
```

```
[50] pesticides
```

```
 'liquid copper soap '],
'Apple___Black_rot': ['Hopefully, you are now well aware of the importance of getting rid of these sources of fungal infection.  This is the primary method of control. Remove the ca
 'captan and sulphar products(lime sulphur)'],
'Apple___Cedar_apple_rust': ['Choose resistant cultivars when available. Rake up and dispose of fallen leaves and other debris from under trees. Remove galls from infected junipers.
 'myclobutanil(Immunox)'],
'Apple___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Blueberry___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Cherry_(including_sour)___Powdery_mildew': ['The key to managing powdery mildew on the fruit is to keep the disease off of the leaves.  Most synthetic fungicides are preventative,
 'copper based fungicide(green cure fungicide)'],
'Cherry_(including_sour)___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot': ['Plant corn hybrids with resistance to the disease; crop rotation and plowing debris into soil may reduce levels of inoculum i
 'foliar fungicide'],
'Corn_(maize)___Common_rust_': ['The most effective method of controlling the disease is to plant resistant hybrids; application of appropriate fungicides may provide some degree on
 'stratego YLD'],
'Corn_(maize)___Northern_Leaf_Blight': ['Follow proper tillage to reduce fungus inoculum from crop debris. Follow crop rotation with non host crop. Grow available resistant varietie
 'Delaro® fungicide'],
'Corn_(maize)___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Grape___Black_rot': ['Black rot (Guignardia bidwelli) is a fungal disease that occurs in grapes grown in a hot and humid climate. This fungal disease has the ability to ruin an ent
 'mancozeb + mycobutanil, imidacloprid or azadirachtin'],
'Grape___Esca_(Black_Measles)': ['The fungi Phaeoacremonium aleophilum, Phaeomoniella chlamydospora[1] and Fomitiporia mediterranea[2] are associated with the disease.',
 'Bio - Tam 2.0'],
'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)': ['The pathogen survives in infected plant residue in soil and seed borne.Favourable conditionsThe disease is more prevalent during June
 'Streptocycline(500 ppm)'],
'Grape___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Orange___Haunglongbing_(Citrus_greening)': ['A number of predators and parasites feed on ACP. The nymphs are killed by tiny parasitic wasps and various predators, including lady be
 'Montery Horticultural oils '],
```

```
[51] !pip install tensorflowjs
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting tensorflowjs
  Downloading tensorflowjs-3.18.0-py3-none-any.whl (77 kB)
     |████████████████████████████████| 77 kB 6.6 MB/s
Requirement already satisfied: tensorflow-hub<0.13,>=0.7.0 in /usr/local/lib/python3.7/dist-packages (from tensorflowjs) (0.12.0)
Requirement already satisfied: tensorflow<3,>=2.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflowjs) (2.8.2+zzzcolab20220527125636)
Requirement already satisfied: six<2,>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflowjs) (1.15.0)
Collecting packaging~=20.9
  Downloading packaging-20.9-py2.py3-none-any.whl (40 kB)
     |████████████████████████████████| 40 kB 7.6 MB/s
Requirement already satisfied: pyparsing>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging~=20.9->tensorflowjs) (3.0.9)
Requirement already satisfied: tensorboard<2.9,>=2.8 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (2.8.0)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (3.17.3)
Requirement already satisfied: flatbuffers>=1.12 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (2.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.1.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.1.2)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.21.6)
Requirement already satisfied: absl-py>=0.4.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.1.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (0.26.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (57.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.6.3)
Requirement already satisfied: libclang>=9.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (14.0.1)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (4.1.1)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.14.1)
Requirement already satisfied: gast>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (0.5.3)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (0.2.0)
Requirement already satisfied: tensorflow-estimator<2.9,>=2.8 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (2.8.0)
Requirement already satisfied: keras<2.9,>=2.8.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (2.8.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (1.46.3)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<3,>=2.1.0->tensorflowjs) (3.1.0)
```

```
[52] import tensorflowjs as tfjs
```

```
[52]
```

```
[53] tfjs.converters.save_keras_model(model, "/content/sample_data/jsfiles")
```

```python
from flask import Flask,redirect,render_template,url_for,request
import cv2
import numpy as np
import os
import glob
import time
from werkzeug.utils import secure_filename
s='static/images/'

app=Flask(__name__)
@app.route('/', methods = ['GET', 'POST'])
def home():
    l = glob.glob('static/images/*')
    l=len(l)
    if request.method == 'POST':
        f = request.files['file']
        l = l+1
        f.save(s+secure_filename(str(l)+'.jpg'))
        print(s+str(l)+'.jpg')
        time.sleep(0.5)
        return redirect('/pred2')
    #print(img)
    t=l+1
    return render_template('index1.html',i=1,l1=t)
@app.route('/<name>')
def pred(name):
    c=s+name+'.jpg'
    print(c)
    img = cv2.resize(cv2.imread(c,1),(64,64))
    img = np.array(img)
    img=img/255
    #img = img.reshape(img.shape[0],img.shape[1],1)
    #img=np.array([img,img])
    img = img.reshape(1,img.shape[0],img.shape[1],3)
    img = np.array(img).tolist()
    return render_template('index.html',im=img,s=c)
@app.route('/pred2')
```

```python
@app.route('/pred2')
def pred3():
    l = glob.glob('static/images/*')
    l=len(l)
    c=s+str(l)+'.jpg'
    print(c)
    img = cv2.resize(cv2.imread(c,1),(64,64))
    img = np.array(img)
    img=img/255
    #img = img.reshape(img.shape[0],img.shape[1],1)
    #img=np.array([img,img])
    img = img.reshape(1,img.shape[0],img.shape[1],3)
    img = np.array(img).tolist()
    return render_template('index.html',im=img,s=c)
if __name__=='__main__':
    app.run(debug=True)

```

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Predictions!!</title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="shortcut icon" href="#">
     <script type="text/javascript" src="../static/dltp.js"></script>
    <!-- Import the webpage's stylesheet -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-giJF6kkoqNQ00vy+HMDP7azOuL0xtbfIcaT9wjKH
    <!-- Font Awesome -->
  <script src="https://kit.fontawesome.com/7654ab4b19.js" crossorigin="anonymous"></script>
  </head>
  <body>
    <div class="container-fluid">
    <h1  class="text-center fw-bolder py-4 text-success bg-light"> Your Predictions!!</h1>
    <div class="alert alert-warning alert-dismissible fade show text-center my-4 mx-5" role="alert">
  <p class="fs-5">Check the console (Press F12) to see predictions!</p>
  <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
  </div>
  <div id="feedback">
    <a href="/" class="fixed-top m-5"><span><i class="fas fa-home fa-2x"></i></span></a>
  </div>

    <div class="card  m-3 bg-light w-25 d-block mx-auto" >
      <div class="card-body w-75 d-block mx-auto">
        <img src="{{s}}" class="card-img-top"</img>
      </div>
      <div class="card-footer border-top-0">
        <h2 class="text-center fw-bolder text-success" id="p1">Hang On!! Our Model is predicting Plant Disease!!</h2>
      </div>
      <div>
```

```
    </div>

    <!-- Import TensorFlow.js library -->
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs/dist/tf.min.js" type="text/javascript"></script>
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>
    <!-- Import the page's JavaScript to do some stuff -->
    <script type="text/javascript" {{ url_for('static', filename='app.js')}}></script>
        <script type="text/javascript">


        </script>
    <script>
    // Load our saved model from current directory (which will be
    // hosted via Firebase Hosting)
    async function predict() {
      // const img = new Image();
      // img.src = '../static/2.jpg';
      // const im = tf.browser.fromPixels(img);
      // im.reshape([3,48, 48,1]);
      // console.log(tf.tensor(im).shape());
      //  const im1=[im,im];
       function myFunc(vars) {
    return vars
    }
var dict = {0:'Pepper__bell___Bacterial_spot', 1:'Potato___healthy', 2:'Tomato_Leaf_Mold',
            3:'Tomato__Tomato_YellowLeaf__Curl_Virus', 4:'Tomato_Bacterial_spot', 5:'Tomato_Septoria_leaf_spot',
            6:'Tomato_healthy', 7:'Tomato_Spider_mites_Two_spotted_spider_mite', 8:'Tomato_Early_blight',
            9:'Tomato__Target_Spot', 10:'Pepper__bell___healthy', 11:'Potato___Late_blight', 12:'Tomato_Late_blight',
            13:'Potato___Early_blight', 14:'Tomato__Tomato_mosaic_virus'};
var pesticides = {'Apple___Apple_scab': ['Choose resistant varieties when possible. Rake under trees and destroy infected leaves to reduce the number of fungal
  'liquid copper soap'],
 'Apple___Black_rot': ['Hopefully, you are now well aware of the importance of getting rid of these sources of fungal infection.  This is the primary method of
  'captan and sulphar products(lime sulphur)'],
 'Apple___Cedar_apple_rust': ['Choose resistant cultivars when available. Rake up and dispose of fallen leaves and other debris from under trees. Remove galls
  'myclobutanil(Immunox)'],
 'Apple___healthy': ["Healthy leaf don't require any treatment",
  'No pesticide Required'],
```

```
'Blueberry___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Cherry_(including_sour)___Powdery_mildew': ['The key to managing powdery mildew on the fruit is to keep the disease off of the leaves.  Most synthetic fungicides are
 'copper based fungicide(green cure fungicide)'],
'Cherry_(including_sour)___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot': ['Plant corn hybrids with resistance to the disease; crop rotation and plowing debris into soil may reduce levels
 'foliar fungicide'],
'Corn_(maize)___Common_rust_': ['The most effective method of controlling the disease is to plant resistant hybrids; application of appropriate fungicides may provide
 'stratego YLD'],
'Corn_(maize)___Northern_Leaf_Blight': ['Follow proper tillage to reduce fungus inoculum from crop debris. Follow crop rotation with non host crop. Grow available resi
 'Delaro® fungicide'],
'Corn_(maize)___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Grape___Black_rot': ['Black rot (Guignardia bidwelli) is a fungal disease that occurs in grapes grown in a hot and humid climate. This fungal disease has the ability
 'mancozeb + mycobutanil, imidacloprid or azadirachtin'],
'Grape___Esca_(Black_Measles)': ['The fungi Phaeoacremonium aleophilum, Phaeomoniella chlamydospora[1] and Fomitiporia mediterranea[2] are associated with the disease.
 'Bio - Tam 2.0'],
'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)': ['The pathogen survives in infected plant residue in soil and seed borne.Favourable conditionsThe disease is more prevale
 'Streptocycline(500 ppm)'],
'Grape___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Orange___Haunglongbing_(Citrus_greening)': ['A number of predators and parasites feed on ACP. The nymphs are killed by tiny parasitic wasps and various predators, inc
 'Montery Horticultural oils '],
'Peach___Bacterial_spot': ["It's Easy To Misdiagnose Bacterial Spot On Peaches. Bacterial spot is a major disease for peach and nectarine growing areas experiencing wa
 'copper, oxytetracycline(Mycoshield and generic equivalents), and syllit + captan'],
'Peach___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Pepper__bell___Bacterial_spot': ['For plant beds and flats in the greenhouse, keep the house as dry as possible and avoid splashing water. Spray with fixed coppers (i
 'copper pesticides Cuprofix Ultra'],
'Pepper__bell___healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Potato___Early_blight': ['Alternaria solani is a fungal pathogen that produces a disease in tomato and potato plants called early blight. The pathogen produces distin
 'Penthiopyrad'],
'Potato___Late_blight': ["If infected tubers make it into the storage bin, there's a very high risk to the storage life of that bin. Once in storage, there isn't much
 'Metalaxyl a phenylamide'],
```

```
'Potato__healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Raspberry__healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Soybean__healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Squash__Powdery_mildew': ['Use resistant varieties when they are available. Thin plants to proper spacing so each leaf gets good exposure to sun and fresh air. Plan
 'neem oil, sulfur and stylet oil, copper fungicides'],
'Strawberry__Leaf_scorch': ["The best way to treat this disease is to alter watering practices and allow for dryer conditions. If that isn't possible, you can treat
 'Tata Taqat, Fungicide, Captan 70 % , Hexaconazole 5 % WP, 500 Gm(or) garlic oil spray'],
'Strawberry__healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required'],
'Tomato_Bacterial_spot': ['Bacterial spot can be a devastating disease when the weather is warm and humid. The disease can affect all above-ground parts of tomato and
 'copper - tolerant Xanthomonas strains'],
'Tomato_Early_blight': ['Prune or stake plants to improve air circulation and reduce fungal problems.Make sure to disinfect your pruning shears (one part bleach to 4
 'Bonide® Copper Fungicide Dust'],
'Tomato_Late_blight': ['When it rains, water hits the ground, splashing soil and spores onto the lower leaves of plants, where the disease shows its earliest symptoms
 'Bonide 811 Copper 4 E Fungicide.'],
'Tomato_Leaf_Mold': ['Managing Tomatoes With Leaf Mold. If you grow tomatoes in a greenhouse or high tunnel, you are more likely to have problems with leaf mold of to
 'chlorothalonil,maneb, mancozeb and copper formulations.'],
'Tomato_Septoria_leaf_spot': ['Removing infected leaves. Remove infected leaves immediately, and be sure to wash your hands thoroughly before working with uninfected
 'Fungonil and Daconil'],
'Tomato_Spider_mites_Two_spotted_spider_mite': ['Spray infested plants with horticultural oil or insecticidal soap to kill spider mites. Follow product recommendation
 'bifenthrin and permethrin'],
'Tomato__Target_Spot': ['Do not plant new crops next to older ones that have the disease.Plant as far as possible from papaya, especially if leaves have small angular
 'Chlorothalonil',
 'Azoxystrobin'],
'Tomato__Tomato_YellowLeaf__Curl_Virus': ['Maintain good weed control in the field and surrounding areas. Prevent the spread of any whiteflies to healthy plants. Toma
 'azadirachtin and pyrethrin insectiside'],
'Tomato__Tomato_mosaic_virus': ['Inspect transplants prior to purchase. Choose only transplants showing no clear symptoms. Avoid planting in fields where tomato root
 'Bon - Neem'],
'Tomato_healthy': ["Healthy leaf don't require any treatment",
 'No pesticide Required']}
function indexOfMax(arr) {
    if (arr.length === 0) {
        return -1;
```

```
    }

    var max = arr[0];
    var maxIndex = 0;

    for (var i = 1; i < arr.length; i++) {
        if (arr[i] > max) {
            maxIndex = i;
            max = arr[i];
        }
    }

    return maxIndex;
}

    var myVar = myFunc({{im|tojson}});

    // Relative URL provided for my-model.json.
    const model = await tf.loadLayersModel('../static/model1/model.json');
    // Once model is loaded, let's try using it to make a prediction!
    // Print to developer console for now.

    const tensor1 = model.predict(tf.tensor(myVar));
    const values = tensor1.dataSync();
    const arr = Array.from(values);
    console.log(values);
    console.log(arr);
    var t=indexOfMax(arr);
    console.log(t);
    document.getElementById("p1").innerHTML=dict[t];
    document.getElementById("p3").innerHTML=pesticides[dict[t]][0] ;
    document.getElementById("p4").innerHTML=pesticides[dict[t]][1] ;
}

predict();
```

```
</script>
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js" integrity="sha384-q2kxQ16AaE6UbzuKqyBE9/u/KzioAlnx2maXQHiDX9d4/zp8Ok3f+M7DPm+Ib6IU"
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.min.js" integrity="sha384-pQQkAEnwaBkjpqZ8RU1fF1AKtTcHJwFl3pblpTlHXybJjHpMYo79HY3hIi4NKx
  </body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <link rel="shortcut icon" href="#">
    <title>Plant Disease Detection & Pesticide Prediction</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-giJF6kkoqNQ00vy+HMDP7azOuL0xtbfIcaT9wjKH
  </head>
  <body>
    <div class="container-fluid">
      <h1 class="text-center fw-bolder py-4 text-primary bg-light"> Plant Disease Detection and Pesticide Prediction using Deep Learning</h1>
      <div class="alert alert-secondary text-center fw-bold fs-4 my-4" role="alert">
        Select an image to Detect the Plant Disease and Predict the Pesticide with Precaution
      </div>
    </div>
    <div class="row">
      <form action = "/" method = "POST"
        enctype = "multipart/form-data">
        <input type = "file" name = "file" />
        <input type = "submit"/>
      </form>
      {%for i in range(1,11) %}
        <div class="col-lg-3 col-md-6">
          <div class="card  m-3 bg-light" style="width:65%;">
            <div class="card-body">
              <img src="static/images/{{i}}.jpg" class="card-img-top" id='im'</img>
            </div>
            <div class="card-footer border-top-0">
              <a href="{{i}}" class="btn btn-success d-block mx-auto">Predict</a>
            </div>
          </div>
        </div>
      {%endfor%}
        </div>

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js" integrity="sha384-q2kxQ16AaE6UbzuKqyBE9/u/KzioAlnx2maXQHiDX9d4/zp80k3f+M7DPm+Ib
```

```
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js" integrity="sha384-q2kxQ16AaE6UbzuKqyBE9/u/KzioAlnx2maXQHiDX9d4/zp80k3f+M7DPm+Ib
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.min.js" integrity="sha384-pQQkAEnwaBkjpqZ8RU1fF1AKtTcHJwFl3pblpTlHXybJjHpMYo79HY3hIi14UK
  </body>
</html>
```