Summer Project On

Plant Disease Detection

By

Prashant Singh (2021510064)

Under the guidance of

Internal Supervisor

Prof. Aarti Karande

Department of Master Of Computer Application
Sardar Patel Institute of Technology
Autonomous Institute Affiliated to Mumbai University

2022-23

CERTIFICATE OF APPROVAL

This is to certify that the following student

Prashant singh (2021510064)

Have satisfactorily carried out work on the project entitled

"Plant Disease Detection"

Towards the fulfilment of project, as laid down by Sardar Patel Institute of Technology during year 2022-23.

Project Guide:
Prof. Aarti Karande

PROJECT APPROVAL CERTIFICATE

This is to certify that the following student

Prashant Singh (2021510064)

Have successfully completed the Project report on

"Plant Disease Detection",

which is found to be satisfactory and is approved

at

SARDAR PATEL INSTITUTE OF TECHNOLOGY,
ANDHERI (W), MUMBAI

INTERNAL EXAMINER EXTERNAL EXAMINER

HEAD OF DEPARTMENT PRINCIPAL

Contents

Abstract

India is a rapidly developing nation, and agriculture is the backbone of the country's early growth. Agriculture is struggling to meet its needs as the global population grows at a rapid rate. Climate change, pollinator decline, crop pests, lack of irrigation, and other factors continue to pose a threat to food security. The benefit of using our project is that crop diseases can be monitored by detecting them as soon as they appear on the crops.This project aids in concentrating on the visually targeted quality of crop.

This study is focused on the identification of plant diseases. The segmentation, feature extraction, and classification techniques are used to detect plant diseases. Photos of leaves from various plants are taken with a digital camera or similar unit, and the images are used to classify the affected region in the leaves. To detect plant disease, we use a Convolution neural network and a Deep neural network in the proposed framework. This paper proposes a framework that employs low-cost, open-source software to achieve the task of reliably detecting plant disease.

Objectives

This cross platform application is used

- To detect the diseased part of the plant leaf using the machine learning model build with the help of TensorFlow.
- It is Used for Detecting weather a plant leaf is healthy or unhealthy by utilizing classical Machine Learning Algorithm and Pre-processing the data using Image Processing.

List of Figures

List of Tables

1 Introduction

1.1 Problem Definition

Plants are highly susceptible to diseases that inhibit plant development, which has an effect on the farmer's ecology. The use of an automated disease detection technique is advantageous in detecting a plant disease at an early stage.

1.2 Objectives and Scope

1.2.1 Objectives

This cross Platform application is used

- To detect the diseased part of the plant leaf using the machine learning model build with the help of TensorFlow.
- It is Used for Detecting weather a plant leaf is healthy or unhealthy by utilizing classical Machine Learning Algorithm and Pre-processing the data using Image Processing.

1.2.2 Scope

The farmers can use this application effortlessly and predict whether plant is diseased or not.

In the application the farmer must provide plant image in order for detection of disease.

The System is being made for reducing the financial loss that farmer faces and helping farmers to detect the disease part of plant and saving crops.

1.3 Existing System

The current approach for detecting plant disease is simple naked eye observation by plant experts,which can be used to detect and identify plant diseases. In these circumstances, the suggested technique is useful for tracking vast fields of crops.Furthermore, in some nations, farmers lack adequate facilities or are unaware that they can contact experts. As a result, consulting experts is not only more expensive but also more time consuming. In those circumstances, the suggested technique for tracking a large number of plantswould be useful
Some of the disadvantages of existing system are as follows :

- Only humans are capable of predicting diseases.
- The procedure is extremely slow
- Consumption of time and space is also very high and the price is also high.

1.4 Proposed System

This study is focused on the identification of plant diseases. The segmentation, feature extraction, and classification techniques are used to detect plant diseases. Photos of leaves from various plants are taken with a digital camera or similar unit, and the images are used to classify the affected region in the leaves. To detect plant disease, we use a Convolution neural network and a Deep neural network in the proposed framework. This paper proposes a framework that employs low-cost,open-source software to achieve the task of reliably detecting plant disease.

Some of the advantages of our system are as follows :

- Detects related images with a low-cost camera and opencv.
- Opencv aids in the efficient analysis of images

1.5 System Requirements

Table 1.5.1: System Requirements

Operating System	Windows, MacOS, Linux
Hardware	Functioning camera
RAM	Minimum 4 GB RAM
Storage	Minimum 250 GB Storage Space

2 Software Requirement Specification (SRS) and Design

2.1 Purpose

The purpose of our project is to develop an UI application that can help farmers to easily detect the diseased part of the plant
This can save lots of efforts of farmers and it will be help them to save thier crop and money.This app will keep track of all the information regarding the diseased part of the plant.

2.2 Definition

To build a Plant Disease Detection Application so the farmers can easily detect the diseased part of the plant.

2.3 Overall Description

2.3.1 Product Functions

The product function includes:

1. Image acquisition.
2. Image pre-processing.
3. Image enhancement.
4. Image segmentation.
5. Image analysis
6. Feature extraction.
7. Disease classification.

3 List Of Modules

3.1 Image Acquisition:

The first step is to gather data from a publicly accessible repository. The picture is used as the input for further processing.We've chosen the most common image domains so that we can accept any format as input to our method, including.bmp,.jpg, and.gif. The camera feeds the real-time images directly. Since most leaves colour varies from red to green for exact segmentation, a white background is provided for further study, proper visibility, and easy image analysis. Cotton images are captured using an image capturing system in this process. The picture is taken in such a way that any distortion is avoided. The photo was not taken in direct sunlight because it would distort the picture.

3.2 Image Pre-processing:

The use of computer algorithms to perform image processing on digital images is known as image pre-processing. We can detect the plant by analysing the image with a specific algorithm. We use a similar approach for image processing and detection with a specific algorithm. The image quality is critical in this process; we can't use the algorithm if the image isn't clear.

3.2.1: Infected Plant

3.3 Image Enhancement:

The process of modifying digital images so that the effects are more appropriate for display or further image processing is known as image enhancement. Any of the following can be used to improve an image:

Histogram Equalization.

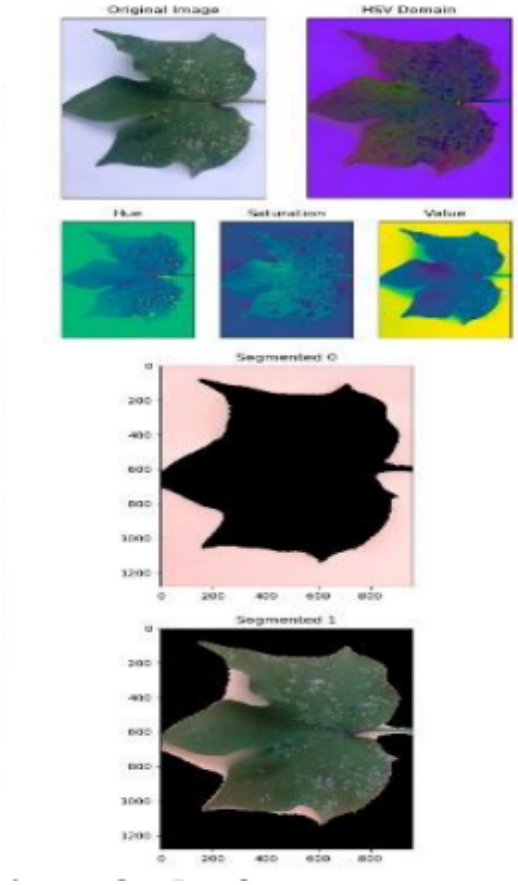
Noise removal using filters.

Unsharp mask filtering.

Decorrelation stretch etc.

3.4 Image Segmentation:

The method of segmenting a digital image into multiple segments is known as image segmentation (sets of pixels, also known as image objects). Image segmentation is used to make image identification and analysis simpler by dividing the image into several segments and analysing each segment individually. Color, texture, and intensity are all common characteristics among the various segments.



3.5 Image Analysis:

In this step, image segmentation is used to locate the region of interest. The technique used in segmentation is region-based segmentation, which uses the colour of the leaf to distinguish between healthy and diseased regions of the plant leaf.

3.6 Feature Extraction:

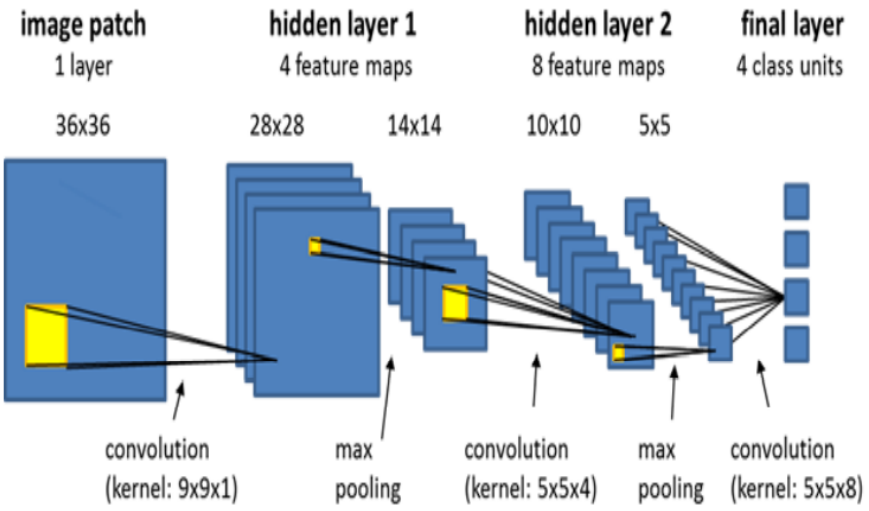
Feature extraction is a part of the dimensionally reduction method in machine learning, which divides and reduces a large collection of raw data into smaller classes. When we have a large amount of data and need to minimise the number of resources while avoiding errors, this step is critical. As a result, function extraction aids in the extraction of the best feature from large data sets by selecting and combining variables into functions

3.7 Disease Classifications::

It is the method of using our qualified deep learning model to recognise plant disease. A digital camera or equivalent system should be used to take an image of the contaminated plant's leaf. Opencv was used to scan the image. Then it determines what kind of plant it is. It determines what kind of disease the plant has after finding it.

4 Training and Keras Model Creation Using CNN

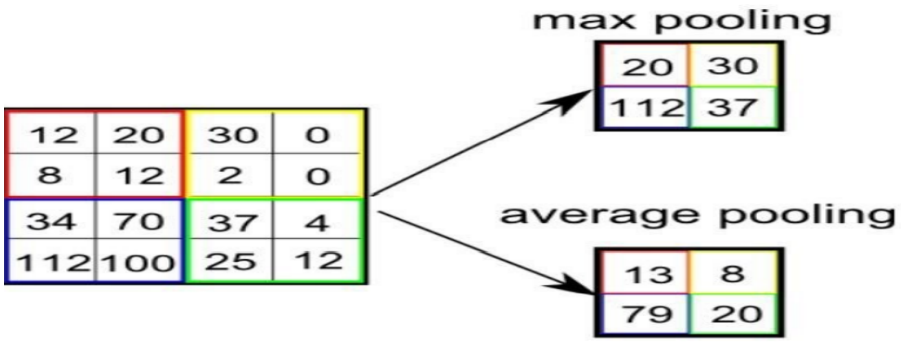
4.1 Convolutional Neural Network(CNN)



4.1.1: Convolutional Neural Network

Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions(number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores. 1. Convolutional Layer: In convolution layer I have taken a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration I slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As I continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some colour. 2. Pooling Layer: We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters. There are Two types of pooling a. Max Pooling: In max pooling we take a window size [for example window of size 2*2], and only taken the maximum of 4 values. Well lid this window and continue this process, so well finally get an activation matrix half of its original Size. b. Average Pooling: In average pooling we take average of all Values in a window.

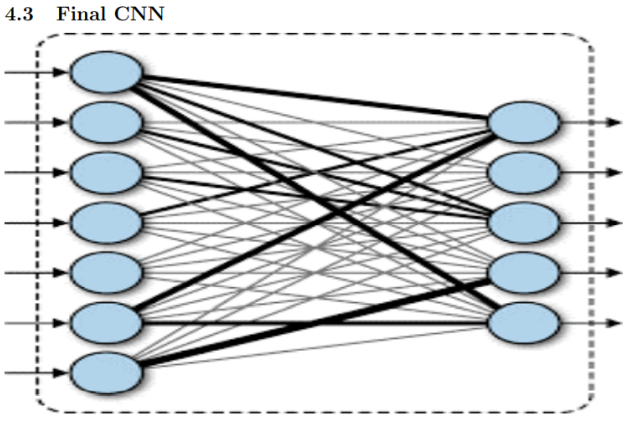
4.2 CNN Pooling



4.2.1: CNN pooling

3. Fully Connected Layer: In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons.

4.3 Final CNN

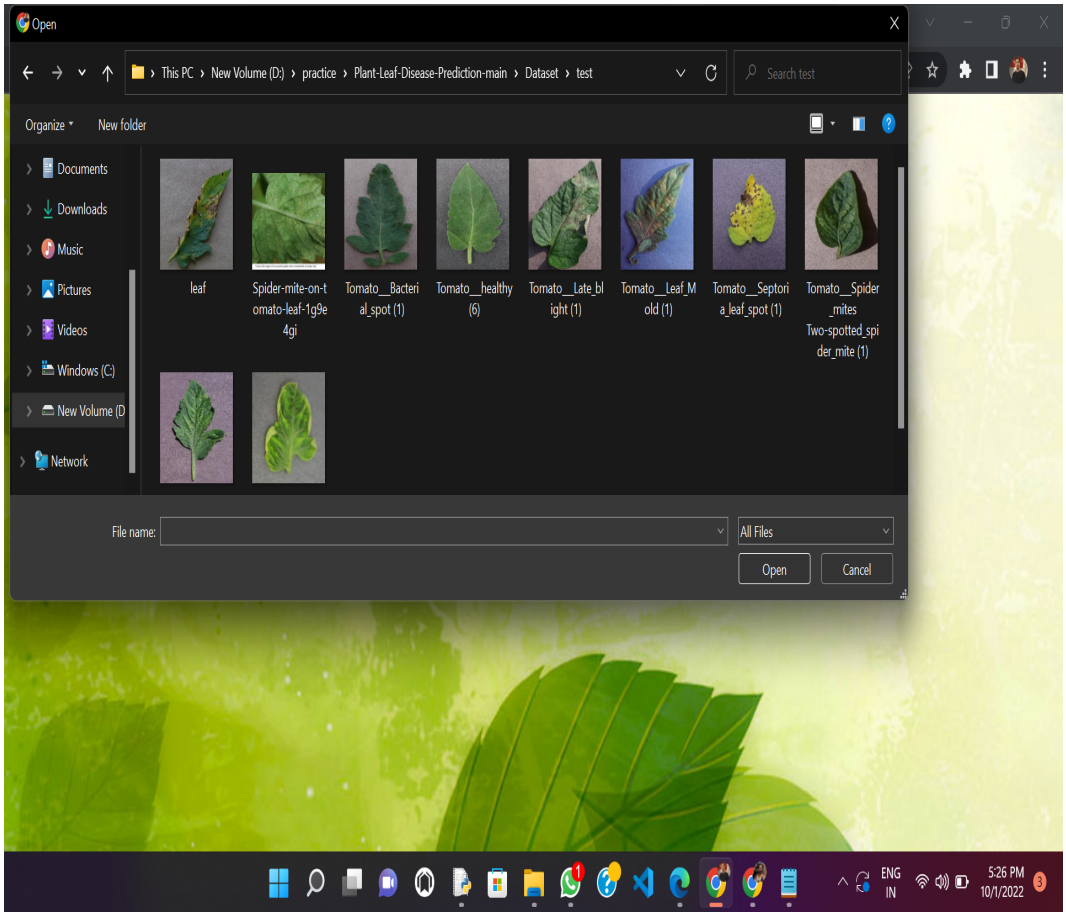


4.3.1: Final CNN

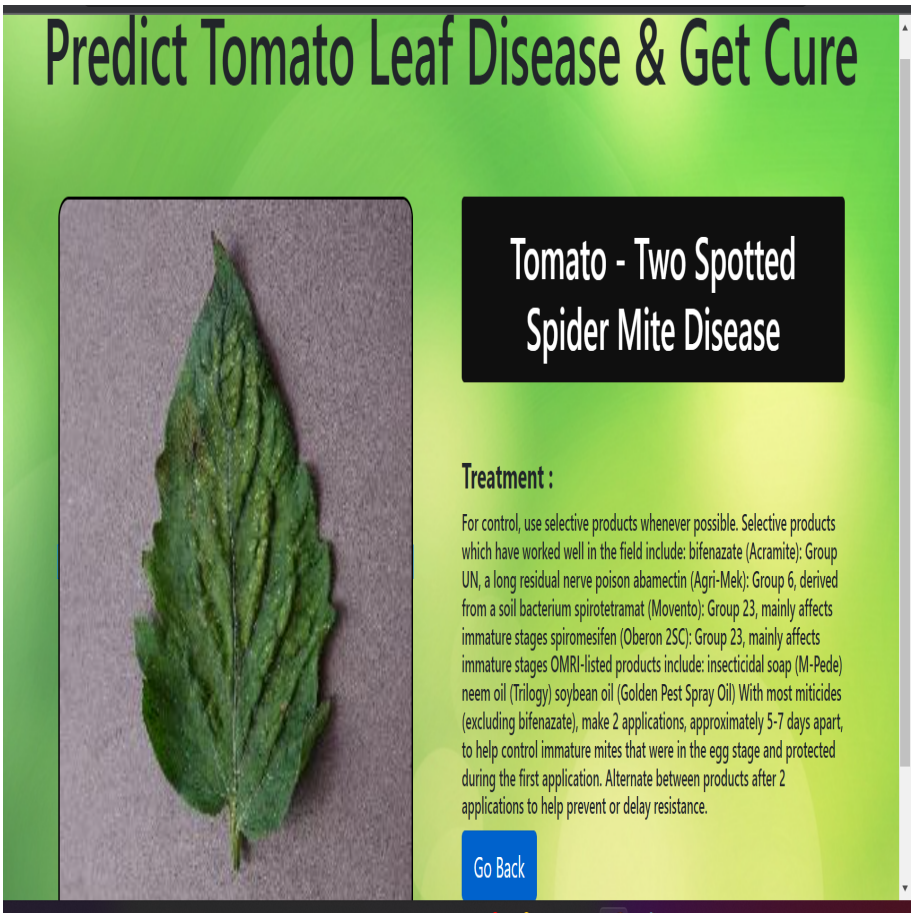
5 Project Implementation and Testing

5.1 Open the Web Application

5.2 Select Plant Image

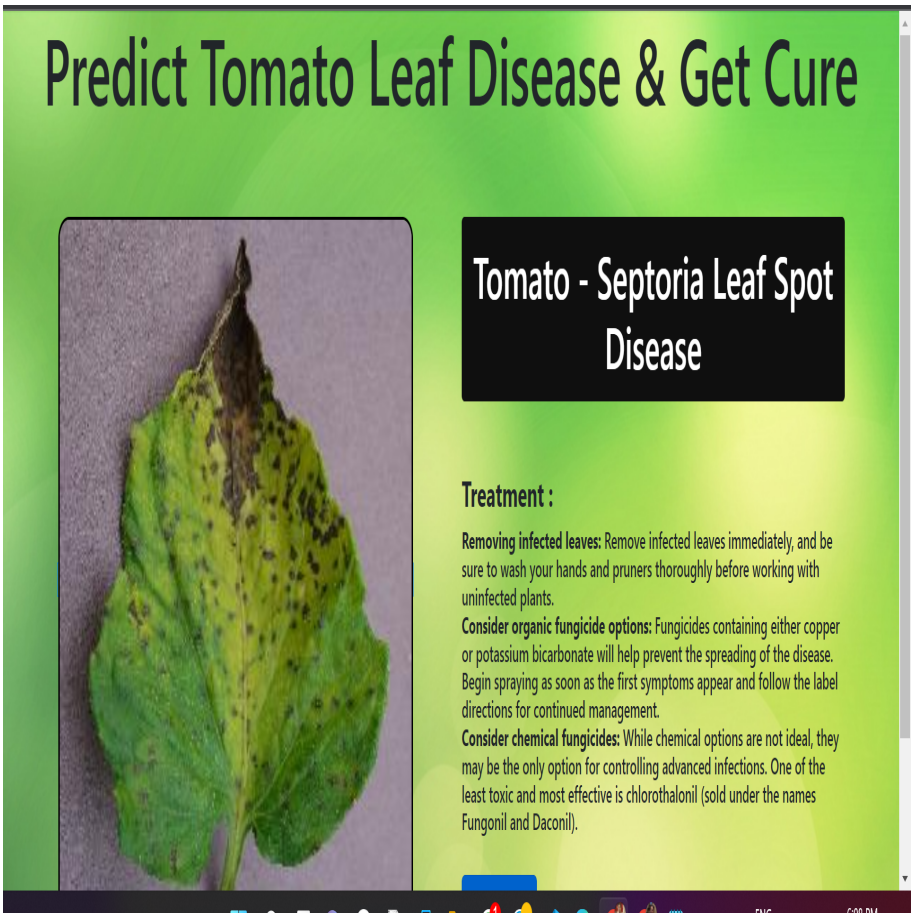


5.3 Profile



5.3.1: Predicted Result

5.4 Profile



5.4.1: Predicted Result

5.5 Code 1

Training.py - D:\practice\Plant-Leaf-Disease-Prediction-main\Training.py (3.10.4)

File Edit Format Run Options Window Help

```
from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession

config = ConfigProto()
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf
tf.compat.v1.disable_eager_execution()
import matplotlib.pyplot as plt
import numpy as np
import os

#basic cnn
# Initialising the CNN
classifier = Sequential()

# Step 1 - Convolution
classifier.add(Conv2D(32, (3, 3), input_shape = (128, 128, 3), activation = 'relu'))

# Step 2 - Pooling
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a second convolutional layer
classifier.add(Conv2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Step 3 - Flattening
classifier.add(Flatten())

# Step 4 - Full connection
classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dense(units = 10, activation = 'sigmoid'))
```

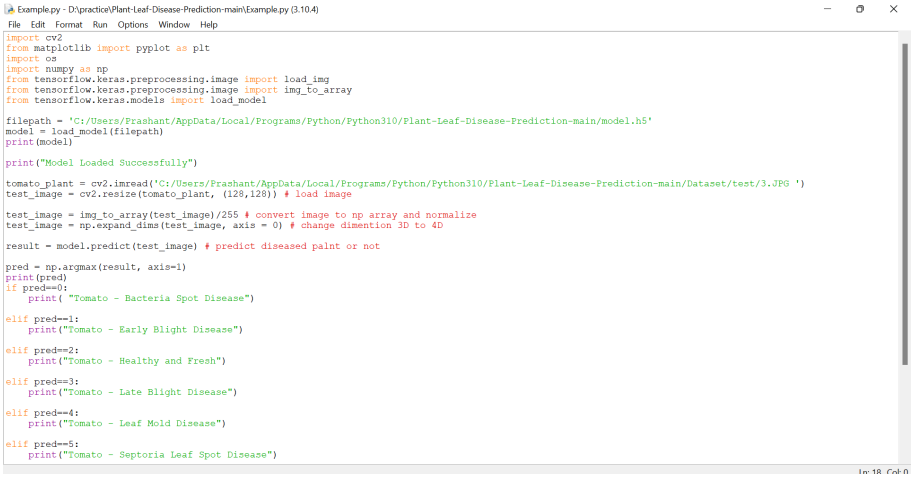
5.6 Code 2

Training.py - D:\practice\Plant-Leaf-Disease-Prediction-main\Training.py (3.10.4)

File Edit Format Run Options Window Help

```
# Step 4 - Full connection
classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dense(units = 10, activation = 'sigmoid'))

# Compiling the CNN
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])


train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255)

training_set = train_datagen.flow_from_directory(r'D:\practice\Plant-Leaf-Disease-Prediction-main\Dataset\train', # relative path from working directoy
                                                 target_size = (128, 128),
                                                 batch_size = 6, class_mode = 'categorical')
valid_set = test_datagen.flow_from_directory(r'D:\practice\Plant-Leaf-Disease-Prediction-main\Dataset\val', # relative path from working directoy
                                             target_size = (128, 128),
                                             batch_size = 3, class_mode = 'categorical')

labels = (training_set.class_indices)
print(labels)


classifier.fit_generator(training_set,
                         steps_per_epoch = 20,
                         epochs = 50,
                         validation_data=valid_set

                         )

classifier_json=classifier.to_json()
with open("model1.json", "w") as json_file:
    json_file.write(classifier_json)
# serialize weights to HDF5
    classifier.save_weights("my_model_weights.h5")
    classifier.save("model.h5")
    print("Saved model to disk")
```

5.7 Code 3



Example.py - D:\practice\Plant-Leaf-Disease-Prediction-main\Example.py (3.10.4)

File Edit Format Run Options Window Help

```
import cv2
from matplotlib import pyplot as plt
import os
import numpy as np
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model

filepath = 'C:/Users/Prashant/AppData/Local/Programs/Python/Python310/Plant-Leaf-Disease-Prediction-main/model.h5'
model = load_model(filepath)
print(model)

print("Model Loaded Successfully")

tomato_plant = cv2.imread('C:/Users/Prashant/AppData/Local/Programs/Python/Python310/Plant-Leaf-Disease-Prediction-main/Dataset/test/3.JPG ')
test_image = cv2.resize(tomato_plant, (128,128)) # load image

test_image = img_to_array(test_image)/255 # convert image to np array and normalize
test_image = np.expand_dims(test_image, axis = 0) # change dimention 3D to 4D

result = model.predict(test_image) # predict diseased palnt or not

pred = np.argmax(result, axis=1)
print(pred)
if pred==0:
    print( "Tomato - Bacteria Spot Disease")

elif pred==1:
    print("Tomato - Early Blight Disease")

elif pred==2:
    print("Tomato - Healthy and Fresh")

elif pred==3:
    print("Tomato - Late Blight Disease")

elif pred==4:
    print("Tomato - Leaf Mold Disease")

elif pred==5:
    print("Tomato - Septoria Leaf Spot Disease")
```

Ln: 18 Col: 0

5.8 Code 4

leaf.py - D:\practice\Plant-Leaf-Disease-Prediction-main\leaf.py (3.10.4)

File Edit Format Run Options Window Help

```
#Import necessary libraries
from flask import Flask, render_template, request

import numpy as np
import os

from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model

filepath = 'D:\practice\Plant-Leaf-Disease-Prediction-main\model.h5'
model = load_model(filepath)
print(model)

print("Model Loaded Successfully")

def pred_tomato_dieas(tomato_plant):
  test_image = load_img(tomato_plant, target_size = (128, 128)) # load image
  print("@@ Got Image for prediction")

  test_image = img_to_array(test_image)/255 # convert image to np array and normalize
  test_image = np.expand_dims(test_image, axis = 0) # change dimention 3D to 4D

  result = model.predict(test_image) # predict diseased palnt or not
  print('@@ Raw result = ', result)

  pred = np.argmax(result, axis=1)
  print(pred)
  if pred==0:
      return "Tomato - Bacteria Spot Disease", 'Tomato-Bacteria Spot.html'

  elif pred==1:
      return "Tomato - Early Blight Disease", 'Tomato-Early_Blight.html'

  elif pred==2:
      return "Tomato - Healthy and Fresh", 'Tomato-Healthy.html'

  elif pred==3:
      return "Tomato - Late Blight Disease", 'Tomato - Late_blight.html'
```

5.9 Code 5

leaf.py - D:\practice\Plant-Leaf-Disease-Prediction-main\leaf.py (3.10.4)

File Edit Format Run Options Window Help

```
    elif pred==7:
        return "Tomato - Tomoato Yellow Leaf Curl Virus Disease", 'Tomato - Tomato_Yellow_Leaf_Curl_Virus.html'
    elif pred==8:
        return "Tomato - Tomato Mosaic Virus Disease", 'Tomato - Tomato_mosaic_virus.html'

    elif pred==9:
        return "Tomato - Two Spotted Spider Mite Disease", 'Tomato - Two-spotted_spider_mite.html'


# Create flask instance
app = Flask(__name__)

# render index.html page
@app.route("/", methods=['GET', 'POST'])
def home():
        return render_template('index.html')


# get input image from client then predict class and render respective .html page for solution
@app.route("/predict", methods = ['GET','POST'])
def predict():
    if request.method == 'POST':
        file = request.files['image'] # fetch input
        filename = file.filename
        print("@@ Input posted = ", filename)

        file_path = os.path.join(r'D:\practice\Plant-Leaf-Disease-Prediction-main\static\upload', filename)
        file.save(file_path)

        print("@@ Predicting class......")
        pred, output_page = pred_tomato_dieas(tomato_plant=file_path)

        return render_template(output_page, pred_output = pred, user_image = file_path)

# For local system & cloud
if __name__ == "__main__":
    app.run(threaded=False,port=8080)
```

Ln: 1 Col: 0

6 Limitations

- Currently this application can be used on desktop only.
- Model is trained with limited dataset.
- For now limited plants disease can be detected.

7 Future Enhancements

- Making this application for mobile phones.
- Add more types of plant for detection.
- Continuous training - while the app is being used and user is happy with predictions made, it will account that in predicting next plant disease.
- By experimenting with different plants like potato,onion,carrot I hope to add new feature like detecting type of tree based on its leaves.
- I am considering enhancing the preprocessing more accurately.

8 User Manual

Step 1 – open the Web Application

Step 2 – Select the image from your device for disease detection

Step 3 – Submit the image.

Step 4 – Click on Predict and get the result.

9 Conclusion

The proposed system tracks the cultivated field on a regular basis. The CNN and DNN algorithms are used to identify crop diseases at an early stage. Machine learning methods are used to train the model, which aids in making appropriate disease decisions. To contain infected diseases, the farmer is advised to use pesticides as a cure. In the future, the proposed scheme could be expanded to provide additional facilities such as nearby government markets, pesticide price lists, and a nearby open market, among others..This project presents a review of various disease classification strategies for crop disease detection, as well as an algorithm for image segmentation that can be used for automated detection and classification of plant leaf diseases in the future. The best results were obtained with very little computational effort, demonstrating the efficacy of the proposed algorithm in recognising and classifying crop diseases. Another benefit of this approach is that plant diseases can be detected at an early stage, or even at the beginning. Convolution neural network and Deep neural network algorithms may be used to increase recognition rates in the classification process.

10 Bibliography

10.1 Web References

[1.] http://developer.android.com/docs/
[2.] https://www.simplilearn.com/tutorials/deep-learning-tutorial
[3.] https://www.youtube.com/watch?v=dGtDTjYs3xc&list=PLeo1K3hjS3ut49PskOfLnE6WUoOp_2lsD
[4.] https://stackoverflow.com/
[5.] https://
[6.] https://www.geeksforgeeks.org/deeplearning/
[7.] [https://teachablemachine.withgoogle.com/train/image
[8.] https://blog.paperspace.com/deploying-deep-learning-models-flask-web-python/