INDEX

INTRODUCTION

Identification of the plant diseases is the key to preventing the losses in the yield and quantity of the agricultural product. The studies of the plant diseases mean the studies of visually observable patterns seen on the plant. Health monitoring and disease detection on plant is very critical for sustainable agriculture. It is very difficult to monitor the plant diseases manually. It requires tremendous amount of work, expertise in the plant diseases, and also require the excessive processing time. Hence, image processing and Machine learning techniques are used for the detection of plant diseases. Disease detection involves the steps like image acquisition, image pre-processing, image segmentation, feature extraction and classification.

MACHINE LEARNING

Machine learning (ML) is the study of algorithms and mathematical models that computer systems use to progressively improve their performance on a specific task. Machine learning algorithms build a mathematical model of sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning.

Machine learning tasks are classified into several broad categories. In supervised learning, the algorithm builds a mathematical model of a set of data that contains both the inputs and the desired outputs. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it contained the object. In special cases, the input may be only partially available, or restricted to special feedback. Semi-supervised learning algorithms develop mathematical models from incomplete training data, where portions of the sample inputs are missing the desired output

Plant disease identification by visual way is more laborious task and at the same time, less accurate and can be done only in limited areas. Whereas if automatic detection technique is used it will take less efforts, less time and become more accurate. In plants, some general diseases seen are brown and yellow spots, early and late scorch, and others are fungal, viral and bacterial diseases. Image processing is used for measuring affected area of disease and to determine the difference in the color of the affected area.

Image segmentation is the process of separating or grouping an image into different parts. There are currently many different ways of performing image segmentation, ranging from the simple thredholding method to advanced color image segmentation methods. These parts normally correspond to something that humans can easily separate and view as individual objects. Computers have no means of intelligently recognizing objects, and so many different methods have been developed in order to segment images. The segmentation process is based on various features found in the image. This might be color information, boundaries or segment of an image. Image is segmented using the K-means clustering technique.

Then unnecessary part (green area) within leaf area is removed. After that we calculate the texture features for the segmented infected object. Finally, the extracted features are passed through a pre-trained neural network.

IMAGE PROCESSING

Image processing is any form of processing for which the input is an image or a series of images or videos, such as photographs or frames of video. The output of image processing can be either an image or a set of characteristics or parameters related to the image. It also means "Analyzing and manipulating images with a computer". Image processing is performed these three steps:

- First, import images with optical devices like a scanner or a camera or directly through digital processing.
- Second, manipulate or analyze the images in some way. This step can include image improvement and data summary, or the images are analyzed to find rules that aren't seen by the human eyes. For example, meteorologists use this processing to analyze satellite photographs.
- Last, output the result of image processing. The result might be the image changed by some way or it might be a report based on analysis or result of the images.

Data Collection and Dataset Preparation

Images can be downloaded from the Internet using the keywords plant and disease names. Subsequently, all the images can be classified into different groups.

Bacterial spot

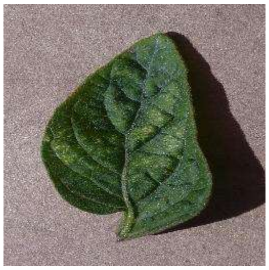
Early blight

Tomato healthy

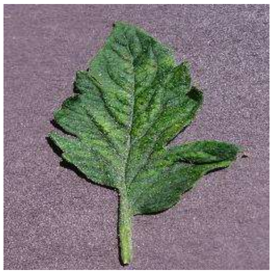
Late blight

Leaf Mold

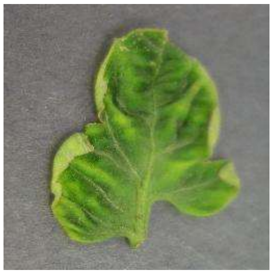
Septoria leaf spot



Spider mite



Mosaic virus



Yellow Leaf Curl Virus

Training

In this step, training the deep convolution neural network for making an image classification model will be done. CaffeNet architecture will be used and adjusted to support our different categories (classes). Rectified Linear Units (ReLU) will subsequently be used as substitute for saturating nonlinearities. This activation function adaptively will learn the parameters of rectifiers and improve accuracy at negligible extra computational cost

EXISTING SYSTEM

Leaf shape description is that the key downside in leaf identification. Up to now, several form options are extracted to explain the leaf form. However, there's no correct application to classify the leaf once capturing its image and identifying its attributes, however. In plant leaf classification leaf is classed supported its completely different morphological options.

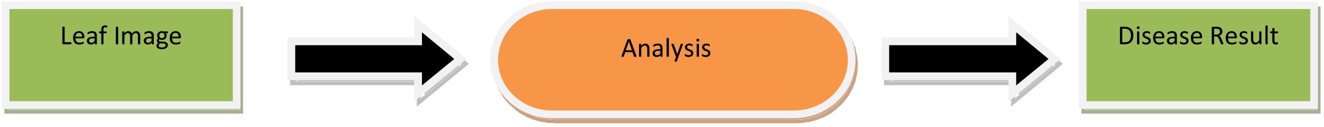
A number of the classification techniques used are:

- Fuzzy logic
- Principal component Analysis
- k-Nearest Neighbors Classifier

PROPOSED SYSTEM

The main purpose of proposed system is to detect the diseases of plant leaves by using feature extraction methods where features such as shape, color, and texture are taken into consideration. Convolution neural network (CNN), a machine learning technique is used in classifying the plant leaves into healthy or diseased and if it is a diseased plant leaf, CNN will give the name of that particular disease. Suggesting remedies for particular disease is made which will help in growing healthy plants and improve the productivity.

First the images of various leaves are acquired using high resolution camera so as to get the better results & efficiency. Then image processing techniques are applied to these images to extract useful features which will be required for further analysis. The basic steps of the system are summarized as:



The advantages of proposed algorithm are as follows:

- Use of estimators for automatic Initialization of cluster centers so there is no need of user input at the time of segmentation.
- The detection accuracy is enhanced with proposed algorithm.
- Proposed method is fully automatic while existing methods require user input to select the best segmentation of input image.
- It also provides environment friendly recovery measures of the identified disease.

OBJECTIVES

- To enhance the given input image by Image acquisition and Image preprocessing.
- Identify the affected part through texture analysis and Segmentation.
- Classify the healthy and affected leaf part by feature extraction and classification.
- Train the model by using testing data for accurate result

PURPOSE

India is an agricultural country, where most of the population depends on agricultural products. So the cultivation can be improved by technological support. Diseases may cause by pathogen in plant at any environmental condition. In most of the cases diseases are seen on the leaves of the plants, so the detection of disease plays an important role in successful cultivation of crops.

There are lots of techniques to detect the different types of diseases in plants in its early stages. Conventional methods of plant disease detection in naked eye observation methods and it are non-effective for large crops. Using digital image processing and machine learning the disease detection in plant is efficient, less time consuming and accurate. This technique saves time, efforts, labors and use of pesticides. Hope this approach will become a little contribution for agriculture fields.

SYSTEM REQUIREMENTS SPECIFICATION

Hardware Requirements :

Processor: 2.5 gigahertz (GHz) frequency or above.

RAM: A minimum of 4 GB of RAM.

Hard disk: A minimum of 20 GB of available space.

Monitor: Minimum Resolution 1024 X 768.

Software Requirements:

Operating System: Windows 7 and above.

Programming language: Python 2.7 and above. HTML, CSS, Java Script

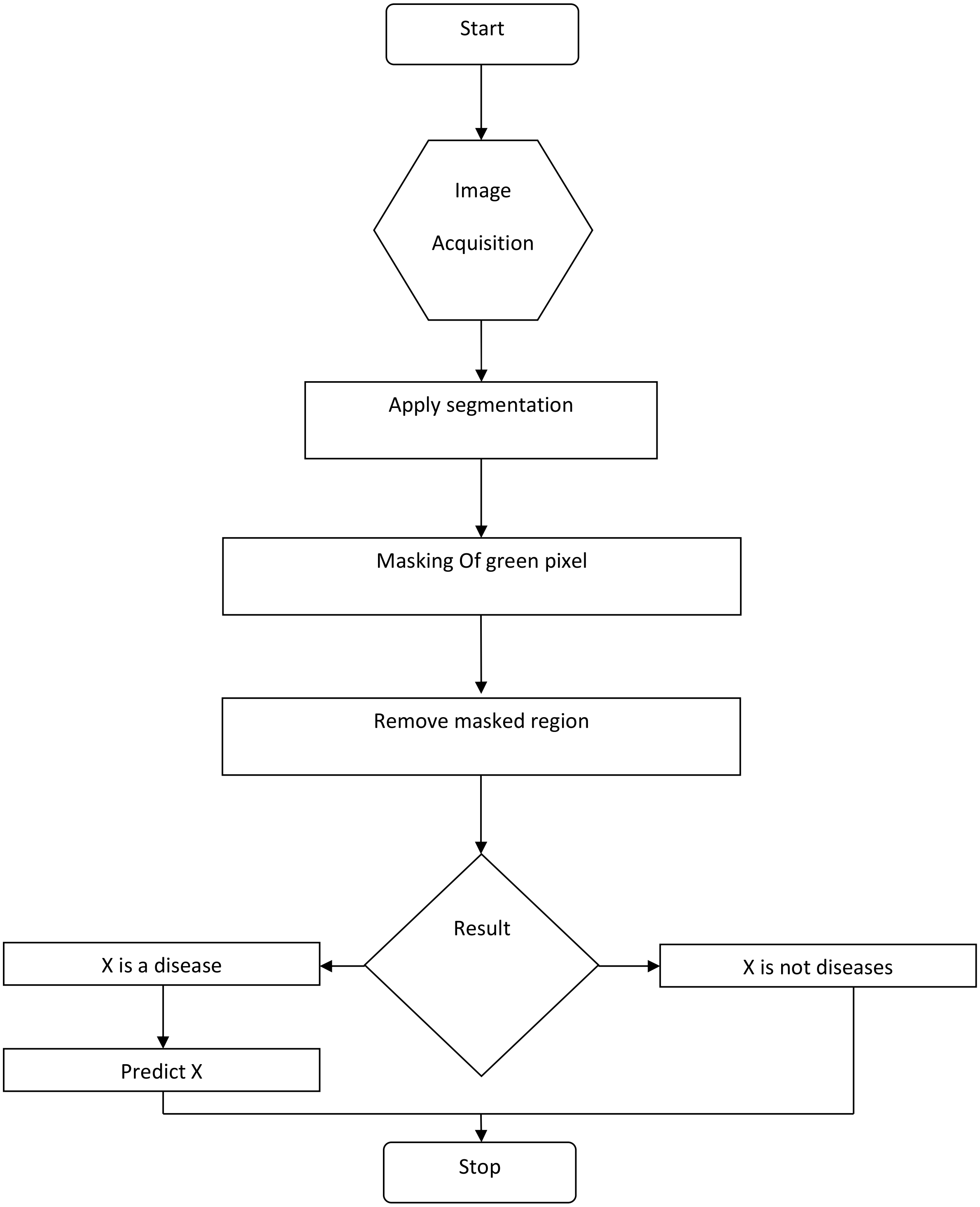
Platform: Jet Brains PyCharm Community Edition 2018.3.5 x64

Supporting libraries:

- Flask==2.0.1,
- itsdangerous==2.0.1
- Jinja2==3.0.1
- pip install Keras Preprocessing==1.1.2
- matplotlib==3.4.0
- matplotlib-inline==0.1.2
- MarkupSafe==2.0.1
- numpy==1.19.4
- scikit learn==0.24.1
- scipy==1.6.2
- sklearn==0.0
- streamlit==0.73.1
- tensorboard==2.4.1
- tensorboard-plugin-wit==1.8.0
- tensorflow==2.4.1
- tensorflow-estimator==2.4.0
- Werkzeug==2.0.1
- numpy==1.19.4
- pandas==1.2.0

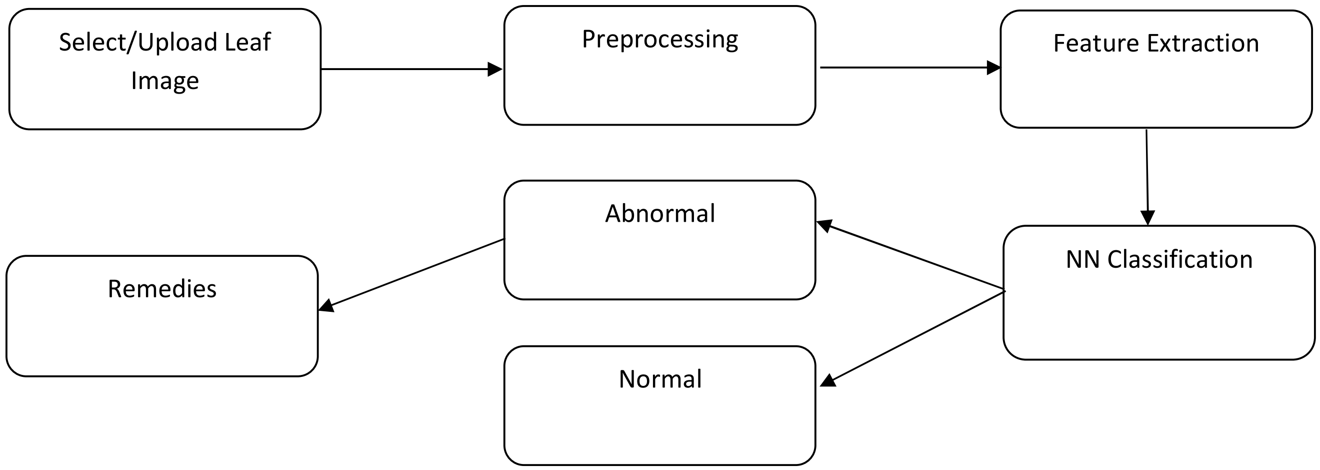
SYSTEM DESIGN

Data flow Diagram: A data flow Diagram is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated.



Activity Diagram

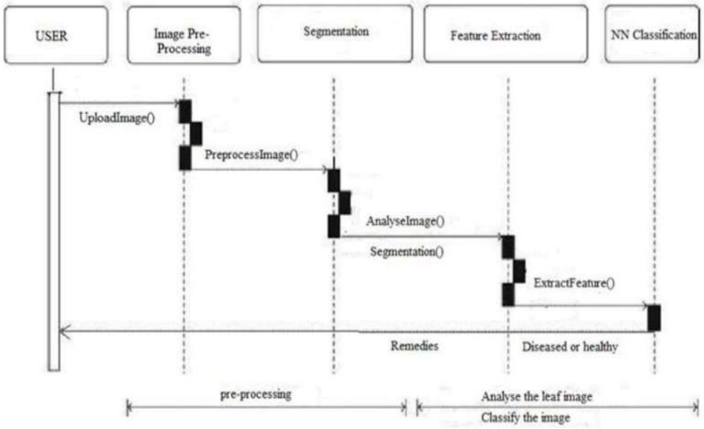
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as in operation of the system. The control flow is drawn from one operation to another.



The above flow represents the flow from one activity to another activity, the activity starts from input leaf image through digital camera, and then input leaf is preprocessed and extract the features like color, shape, texture and so on. Now, the processed image is classified as Normal or Abnormal, if Abnormal is found in the leaf, then remedies will be suggested.

Sequence Diagram

Sequence diagrams are sometimes called event diagrams or event scenarios. A sequence diagram shows as parallel vertical lines (lifelines), different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them in the order in which they occur.



1. Usage scenarios: A usage scenario is a description of a potential way your system is used. The logic of a usage scenario may be part of a use case, perhaps an alternate course. It may also be one entire pass through a use case, such as the logic described by the basic course of action or a portion of the basic course of action, plus one or more alternate scenarios. The logic of a usage scenario may also be a pass through the logic contained in several use cases.

2. The logic of methods: Sequence diagrams can be used to explore the logic of a complex operation, function, or procedure. One way to think of sequence diagrams, particularly highly detailed diagrams, is as visual object code

3. The logic of services: A service is effectively a high-level method, often one that can be invoked by a wide variety of clients. This includes web-services as well as business transactions implemented by a variety of technologies

Following are some common symptoms of bacterial and viral plant leaf diseases.

♦ **Bacterial disease symptoms**

Bacterial diseases include any type of illness caused by bacteria. Bacteria are a type of microorganism, which are tiny forms of life that can only be seen with a microscope. Other types of microorganisms include viruses, some fungi, and some parasites. Millions of bacteria normally live on the skin, in the intestines, and on the genitalia. The vast majority of bacteria do not cause disease, and many bacteria are actually helpful and even necessary for good health. These bacteria are sometimes referred to as “good bacteria” or “healthy bacteria.”

♦ **Viral disease symptoms**

Symptoms of viral diseases vary depending on the specific type of virus causing infection, the area of the body that is infected, the age and health history of the patient, and other factors. The symptoms of viral diseases can affect almost any area of the body or body system.

Symptoms of viral diseases can include:

- Flu-like symptoms (fatigue, fever, sore throat, headache, cough, aches and pains)

- Gastrointestinal disturbances, such as diarrhoea, nausea and vomiting

- Irritability

- Malaise (general ill feeling)

- Rash

Source Code

CSS

```
.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

.img-preview>div {
    width: 100%;
    height: 100%;
    background-size: 256px 256px;
    background-repeat: no-repeat;
    background-position: center;
}

input[type="file"] {
    display: none;
}

.upload-label{
    display: inline-block;
    padding: 12px 30px;
    background: #39D2B4;
    color: #fff;
    font-size: 1em;
    transition: all .4s;
    cursor: pointer;
}

.upload-label:hover{
    background: #34495E;
    color: #39D2B4;
}

.loader {
    border: 8px solid #f3f3f3; /* Light grey */
    border-top: 8px solid #3498db; /* Blue */
    border-radius: 50%;
    width: 50px;
    height: 50px;
    animation: spin 1s linear infinite;
}

@keyframes spin {
    0% { transform: rotate(0deg); }
    100% { transform: rotate(360deg); }
}
```

JavaScript

```
$(document).ready(function () {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();
            $('#result1').hide();
    // Upload Preview
    function readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function (e) {
                $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            }
            reader.readAsDataURL(input.files[0]);
        }
    }
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
                        $('#result1').text('');
        $('#result1').hide();
        readURL(this);
    });
    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);
        // Show loading animation
        $(this).hide();
        $('.loader').show();
        // Make prediction by calling api /predict
        $.ajax({
            type: 'POST',
            url: '/predict',
            data: form_data,
            contentType: false,
            cache: false,
            processData: false,
            async: true,
            success: function (data,data1) {
                // Get and display the result
                $('.loader').hide();
                $('#result').fadeIn(600);
                $('#result').text(' Result:  ' + data);
                                                    console.log('success')
    },
        });
    });
});
```

Python leaf Code

```
#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 =
'C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/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'

  elif pred==4:
      return "Tomato - Leaf Mold Disease", 'Tomato - Leaf_Mold.html'

  elif pred==5:
      return "Tomato - Septoria Leaf Spot Disease", 'Tomato - Septoria_leaf_spot.html'

  elif pred==6:
      return "Tomato - Target Spot Disease", 'Tomato - Target_Spot.html'

  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'] # fet input
        filename = file.filename
        print("@@ Input posted = ", filename)

        file_path = os.path.join('C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/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)
```

Training

```
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'))


# 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('C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/Dataset/train', # relative path from working directoy
                                                 target_size = (128, 128),
                                                 batch_size = 6, class_mode = 'categorical')
valid_set = test_datagen.flow_from_directory('C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/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")


'''
import cv2
from matplotlib import pyplot as plt
import os
import numpy as np


img = cv2.imread('C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Leaf_disease/data/d
(7)_iaip.jpg')
img_resize = cv2.resize(img, (128,128))



CV2 reads an image in BGR format. We need to convert it to RGB
b,g,r = cv2.split(img_resize)       # get b,g,r
rgb_img = cv2.merge([r,g,b])     # switch it to rgb



plt.imshow(rgb_img)
label_map = (training_set.class_indices)


print(label_map)
img_rank4 = np.expand_dims(rgb_img/255, axis=0)
```

```
classifier.predict(img_rank4)

h = list(label_map.keys())[classifier.predict_classes(img_rank4)[0]]

font = cv2.FONT_HERSHEY_DUPLEX

cv2.putText(img, h, (10, 30), font, 1.0, (0, 0, 255), 1)

cv2.imshow(h,img)


print(h)

'''
```

Example

```
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/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/model.h5'

model = load_model(filepath)

print(model)


print("Model Loaded Successfully")


tomato_plant = cv2.imread('C:/Users/Madhuri/AppData/Local/Programs/Python/Python38/Tomato_Leaf_Disease_Prediction/Dataset/test/Tomato___Bacterial_spot (1).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")

elif pred==6:
    print("Tomato - Target Spot Disease")


elif pred==7:
      print("Tomato - Tomoato Yellow Leaf Curl Virus Disease")
elif pred==8:
      print("Tomato - Tomato Mosaic Virus Disease")


elif pred==9:
      print("Tomato - Two Spotted Spider Mite Disease")
```

SYSTEM TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. There are various types of testing, explained in the following sections.

Testing Objective

- Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has a high probability of finding an as yet undiscovered error.

- A successful test one that uncovers an as –yet undiscovered error.

The above objectives imply a dramatic change in viewpoint. They move counter to the commonly held view that a successful test is one in which no errors are found. Testing cannot show the absence of detects; it can only show that software errors are present.

White Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, or structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality as in Black Box Testing Using white box testing methods, the software engineer can derive test cases that:

- Guarantee that all independent paths with in a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their validity.
- Logical errors and incorrect assumptions are inversely proportional to the probability that a program path will be executed.
- Often believe that logical path not likely to execute when, in fact, it may be executed on regular basis.

Levels of Testing

The different levels of testing that are to be conducted are:

- Code Testing

- Program Testing

- System Testing

Code Testing: The code test has been conducted to test the logic of the program. Here, we have tested with all possible combinations of data to find out logical errors. The code testing is done thoroughly with all possible data available with library.

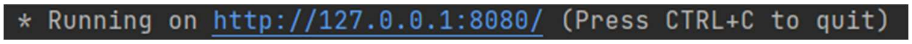
Program Testing: Program testing is also called unit testing. The modules in the system are integrated to perform the specific function. The modules have been tested independently, later Assembled and tested thoroughly for integration between different modules.

System Testing: System testing has been conducted to test the integration of each module in the system. It is used to find discrepancies between the system and its original objective. It is found that there is an agreement between current specifications and system documentation. Software Testing is carried out in three steps.

The first step includes unit testing where in each module is tested to provide his correctness, validity and also determine any missing operations. Errors are noted down and corrected immediately. Unit testing is the import and major part of the project. So errors are rectified easily in particular module and program clarity is increased. In this project entire system is divided into several modules and is developed individually. So unit testing is conducted to individual modules.

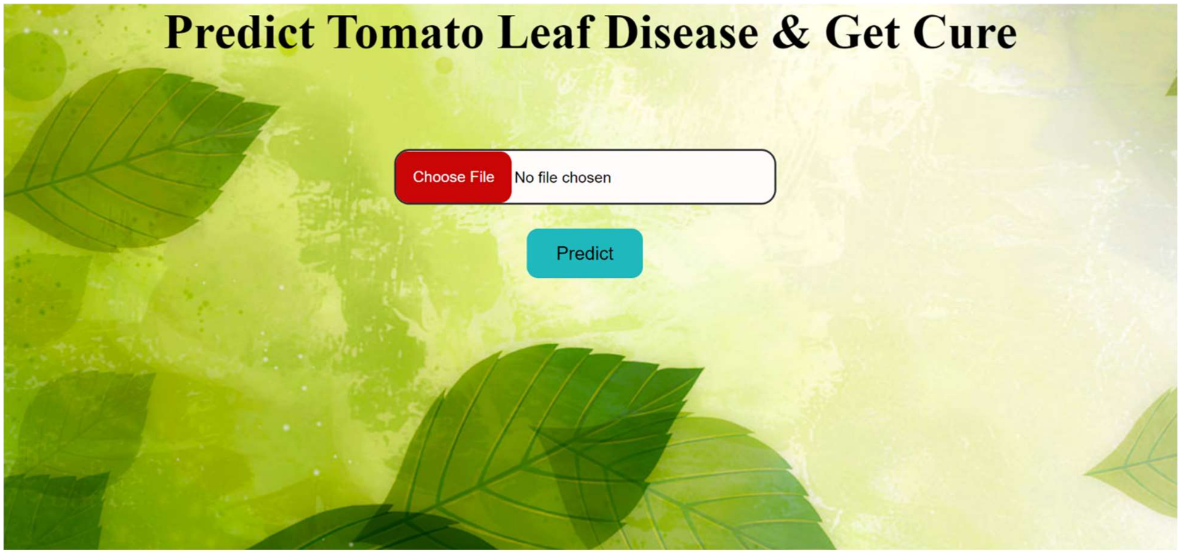
The second step includes integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole. The individual modules are clipped under this major module and tested again and the results are verified.

RESULTS

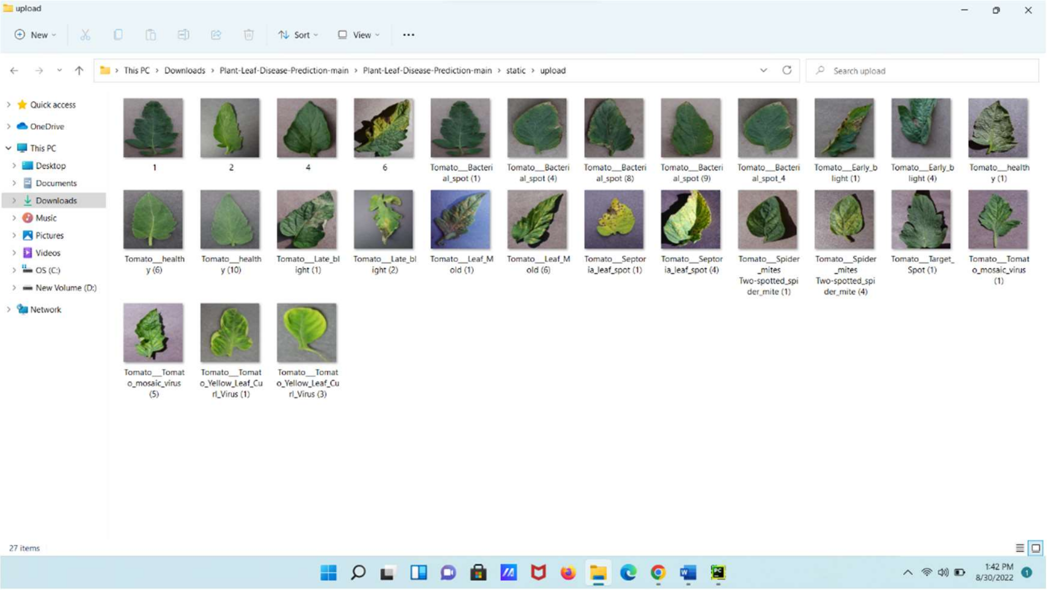


```
* Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
```

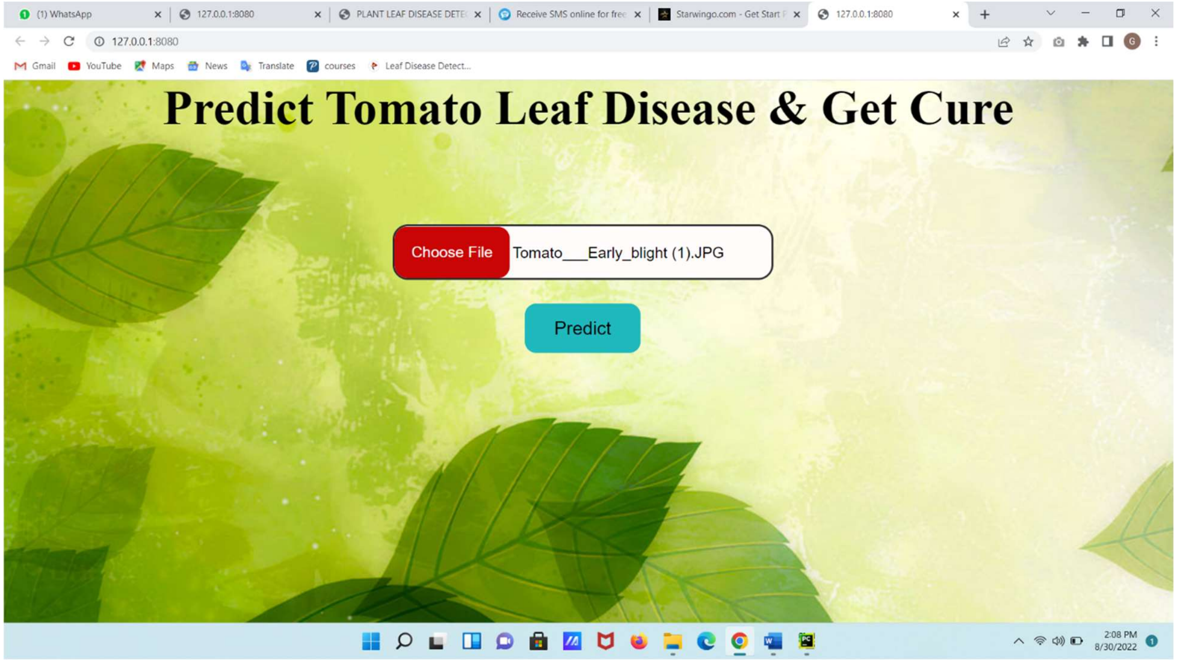
- After Debugging the program Click on Localhost: **http://127.0.0.1:8080/**



Depicts the first look of our front end. We have a text message called “Choose file” so that a user can understand to click the button. It has a button called “Choosefile” which can be used to browse the images on the system’s hard disk


upload
New
Sort
View
This PC › Downloads › Plant-Leaf-Disease-Prediction-main › Plant-Leaf-Disease-Prediction-main › static › upload
Search upload
Quick access
OneDrive
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
OS (C:)
New Volume (D:)
Network
1
2
4
6
Tomato__Bacterial_spot (1)
Tomato__Bacterial_spot (4)
Tomato__Bacterial_spot (8)
Tomato__Bacterial_spot (9)
Tomato__Bacterial_spot_4
Tomato__Early_blight (1)
Tomato__Early_blight (4)
Tomato__healthy (1)
Tomato__healthy (6)
Tomato__healthy (10)
Tomato__Late_blight (1)
Tomato__Late_blight (2)
Tomato__Leaf_Mold (1)
Tomato__Leaf_Mold (6)
Tomato__Septoria_leaf_spot (1)
Tomato__Septoria_leaf_spot (4)
Tomato__Spider_mites Two-spotted_spider_mite (1)
Tomato__Spider_mites Two-spotted_spider_mite (4)
Tomato__Target_Spot (1)
Tomato__Tomato_mosaic_virus (1)
Tomato__Tomato_mosaic_virus (5)
Tomato__Tomato_Yellow_Leaf_Curl_Virus (1)
Tomato__Tomato_Yellow_Leaf_Curl_Virus (3)
27 items
1:42 PM
8/30/2022

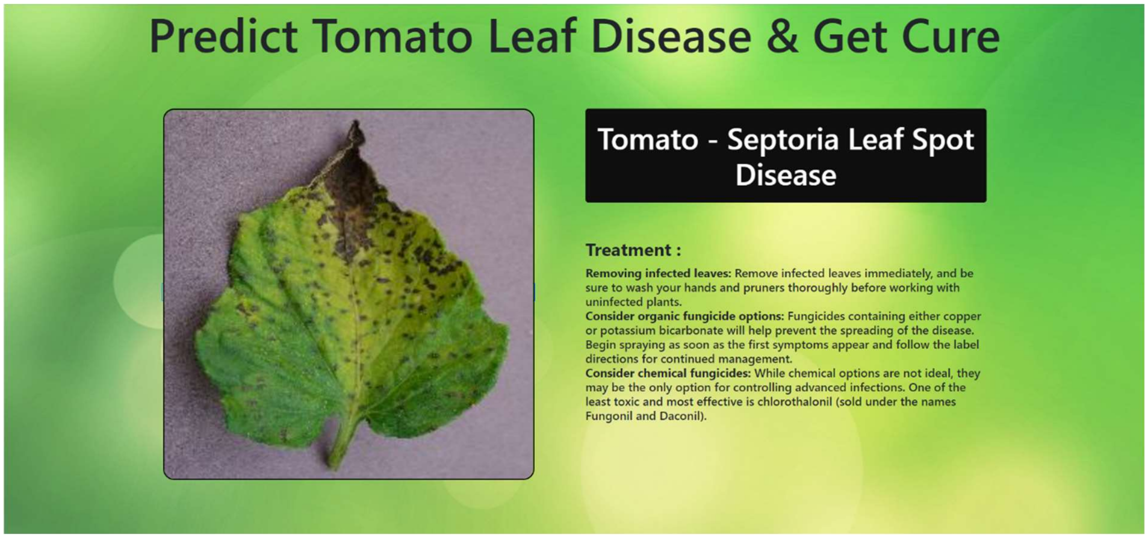
It shows the popup which appears when user clicks on 'get photo' button. The popup window will be having number of input images to be selected, this action should be confirmed with a double click or an open button.



Predict Tomato Leaf Disease & Get Cure

Choose File Tomato___Early_blight (1).JPG

After you select the leaf image Click on "Predict" to get leaf disease and get Cures



Predict Tomato Leaf Disease & Get Cure

Tomato - Septoria Leaf Spot Disease

Treatment :

Removing infected leaves: Remove infected leaves immediately, and be sure to wash your hands and pruners thoroughly before working with uninfected plants.

Consider organic fungicide options: Fungicides containing either copper or potassium bicarbonate will help prevent the spreading of the disease. Begin spraying as soon as the first symptoms appear and follow the label directions for continued management.

Consider chemical fungicides: While chemical options are not ideal, they may be the only option for controlling advanced infections. One of the least toxic and most effective is chlorothalonil (sold under the names Fungonil and Daconil).

Displays the status of leaf i.e. Healthy or unhealthy if it is unhealthy to get Disease name and it's Cure

CONCLUSION AND FUTURE WORK

CONCLUSION

There are number of ways by which we can detect disease of plants and suggest remedies for them. Each has some pros as well as limitations. On one hand visual analysis is least expensive and simple method, it is not as efficient and reliable. Image processing is a technique which is most spoken for very high accuracy and least time consumption are major advantages offered. The applications of K-means clustering and Neural Networks (NNs) have been formulated for clustering and classification of diseases that effect on plant leaves. Recognizing the disease accurately and efficiently is mainly the purpose of the proposed approach. The experimental results indicate that the proposed approach is a valuable approach, which can significantly support an accurate detection of leaf diseases in a little computational effort. Alongside the supply of cultivation tools, the farmers also need access to accurate information that they can use for efficient crop management and there is no better way than providing them a service that they can use through the software.

FUTURE WORK

- To improve recognition rate of final classification process hybrid algorithms like Artificial Neural Network, Bayes classifier, Fuzzy Logic can also be used.
- Mobile application can be developed which is handy and easy to use.
- An extension of this work will focus on automatically estimating the severity of the detected disease.
- As future enhancement of the project is to develop the open multimedia (Audio/Video) about the diseases and their solution automatically once the disease is detected.

References

1. A survey on crop disease detection using image processing technique for economic growth of rural area. Yashpal Sen1, Chandra Shekhar Mithlesh2, Dr. Vivek Baghel3

2. K. Elangoran, S. Nalini, 2011 “Detection and classification of leaf diseases using K- means-based segmentation and neural-networks-based classification.” Inform. Technol. J., 10: 267-275. DOI: 10.3923/itj.2011.267.275. 3. Sandesh Raut, Karthik Ingale, “Review on leaf disease detection using Image Processing techniques.” 4. “A Survey on Methods of Plant Disease Detection” Sagar Patil, Anjali Chandavale 5. T. RUMPF, A-K Mahlein, U sleiner, H. W. Dehne. "Texture analysis for diagnosing paddy disease." In International Conference on Electrical Engineering and Informatics, 2009. ICEEI'09., vol. 1, pp. 23-27. IEEE, 2009. 6. “Plant disease detection and classification using image processing and artificial neural networks.” Mr. Sanjay Mirchandani1, Mihir Pendse2, Prathamesh Rane3, Ashwini Vedula4 7. “Detection and Classification of Plant Leaf Diseases Using Image Processing Techniques:” Savita N. Ghaiwat, Parul Arora 8. “Image Processing Based Leaf Rot Disease, Detection of Betel Vine (Piper Betel.)” Amar Kumar Deya*, Manisha Sharmaa, M.R. Meshramb 9. “Advances in image processing for plant disease detection” Jayamala k Patil, Rajkumar 10. S Arivazhagan, R Newlin shebiah, S Ananthi, S Vishnu varthini “Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features.”