PLANT DISEASE DETECTION USING DEEP LEARNING-WEB APP

A PROJECT REPORT

Submitted by

JAYAKRISHNA K

MURALI R

ESWARAMOORTHY B

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING,GUINDY

ANNA UNIVERSITY : CHENNAI 600 025

JUNE 2022

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**PLANT DISEASE DETECTION USING DEEP LEARNING-WEB APP"** is the bonafide work of "**JAYAKRISHNA K (2019103023), MURALI R (2019103038), ESWARAMOORTHY B (2019103014)"** who carried out the project under my supervision.

DATE: 04.06.2022

PLACE: Chennai

SIGNATURE

Mr.T.M.Thiyagu

SUPERVISOR

Teaching Fellow

Dept. Of Computer Science & Engineering

Anna University,
Chennai - 600025.

ABSTRACT

In India, Agriculture plays an essential role because of the rapid growth of population and increasing demand for food. Therefore, the demand for crop yield increases. One fo main reasons for low crop yield is diseases caused by bacteria, virus and fungus.But the detection of disease takes long time as it requires a visit by biologists.But with the help of our project,anyone can easily identify a disease by uploading a picture of the concerned plant via a web app and with the help of machine learning,we can classify the disease earlier and thus can be treated later.Machine learning methods can be used for disease detection because it mainly applies on data themselves and gives priority to outcomes of the certain task.Here,in this survey it has been observed that Convolutional Neural Network(CNN) gives high accuracy and detects more number of diseases of multiple crops.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

Abbreviations	Expansion
CNN	Convolutional Neural Network
ROI	Region of Interest

CHAPTERS

CHAPTER 1: INTRODUCTION

1.1 OVERVIEW

One of main reasons for low crop yield is diseases caused by bacteria, virus and fungus.But the detection of disease takes a long time as it requires a visit by biologists. But with the help of our project, anyone can easily identify a disease by uploading a picture of the concerned plant via a web app.Many Machine learning algorithms can be used for disease detection because it mainly applies on data themselves and gives priority to outcomes of the certain task.Convolutional Neural Network(CNN) gives high accuracy and proven to be very good in classifying diseases.

1.1.1 CONVOLUTIONAL NEURAL NETWORK ALGORITHM

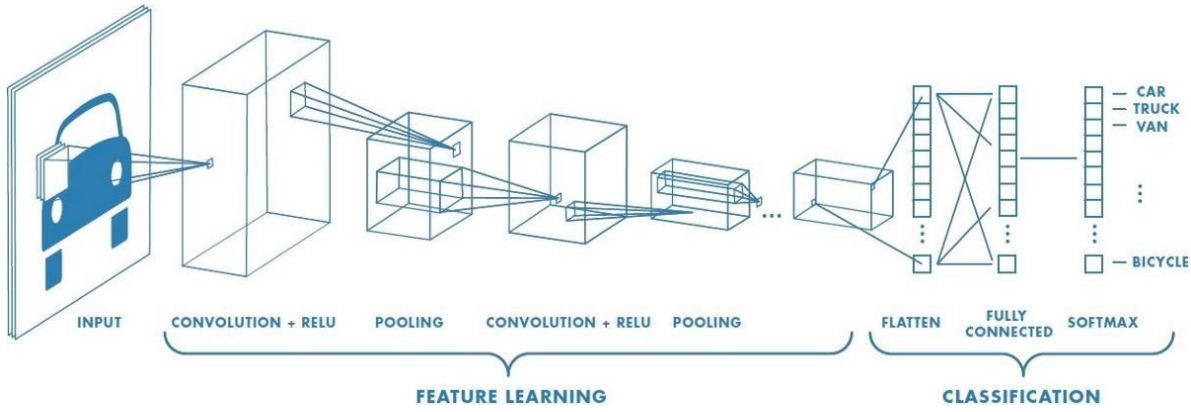

Figure 1.1.1 CNN Architecture Diagram

A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include layers that perform convolutions.

Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This is followed by other layers such as pooling layers, fully connected layers, and normalization layers

1.1.2 FLASK

Flask is a web application framework written in Python. It is developed by Armin Ronacher, who leads an international group of Python enthusiasts named Pocco. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

1.1.3 WSGI

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

1.1.4 Werkzeug

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

1.1.5 Jinja2

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have a built-in abstraction layer for database handling, nor does it have form validation support. Instead, Flask supports the extensions to add such functionality to the application. Some of the popular Flask extensions are discussed later in the tutorial.

1.2 PROBLEM STATEMENT

The proposed research work is carried on three crops: Potato, tomato, and pepper. In a survey, 61.33% of farmers cultivating potatoes reported blight as one of the major reasons for their crop failure. According to The Hindu, in the year 2020, Tamil Nadu faced a huge crisis for tomato crops approx. 60% of crops failed due to a virus. India has a 40% share of total pepper production globally. Pepper has also many herbal benefits in throat infections. India is the 2nd largest producer of tomatoes and potatoes followed by China which is the largest producer of these crops. Improving fertilization and automating the system for disease detection can improve the crop production of our country.

1.3 OBJECTIVES

- To detect unhealthy regions of plant leaves.
- Classification of plant diseases using texture features.
- To analyze the leaf infection.
- To make this service available on Web App which can run on low level configuration devices.

CHAPTER 2: LITERATURE SURVEY

2.1 Literature of Related Works

Malvika Ranjan et al. in the paper **—"Detection and Classification of leaf disease using Artificial Neural Network"** proposed an approach to detect diseases in plants utilizing the captured image of the diseased leaf. Artificial Neural Network (ANN) is trained by properly choosing feature values to distinguish diseased plants and healthy samples. The ANN model achieves an accuracy of 80%.

According to the paper **—"Detection of unhealthy regions of plant leaves and classification of plant leaf diseases using texture features"** by S. Arivazhagan, disease identification process includes four main steps as follows: first, a color transformation structure is taken for the input RGB image, and then by means of a specific threshold value, the green pixels are European Journal of Molecular & Clinical Medicine ISSN 2515-8260 Volume 7, Issue 07, 2020 1607 detected and uninvolved, which is followed by segmentation process, and for obtaining beneficial segments the texture statistics are computed. At last, a classifier is used for the features that are extracted to classify the disease.

Kulkarni et al. in the paper **—"Applying image processing technique to detect plant diseases"** , a methodology for early and accurate plant diseases detection, using artificial neural network (ANN) and diverse image processing techniques. As the proposed approach is based on ANN classifier for classification and Gabor filter for feature extraction, it gives better results with a recognition rate of up to 91%.

In paper **—"Plant disease detection using CNN and GAN"** , by Emaneul Cortes, an approach to detect plant disease using Generative Adversarial networks has been proposed. Background segmentation is used for ensuring proper feature extraction and output mapping. It is seen that using Gans may hold promise to classify diseases in plants, however segmenting based on background did not improve accuracy.

In the paper **—"Convolutional Neural Network based Inception v3 Model for Animal Classification"** , Jyotsna Bankar et al. have proposed use of inception v3 model in classifying animals in different species. Inception v3 can be used to classify objects as well as to categorize them, this capability of inception v3 makes it instrumental in various image classifiers.

2.2 Summary of Related works

From study of above classification techniques we come up with the following conclusion. The k-nearest-neighbor method is perhaps the simplest of all algorithms for predicting the class of a test example. An obvious disadvantage of the k-NN method is the time complexity of making predictions.Using CNN we are improving the accuracy to a higher level.The convolutional neural network(CNN) model is trained to identify healthy and diseased plants of 14 diseases. CNN is found competitive with the best available machine learning algorithms in classifying high-dimensional data sets.

CHAPTER 3:SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

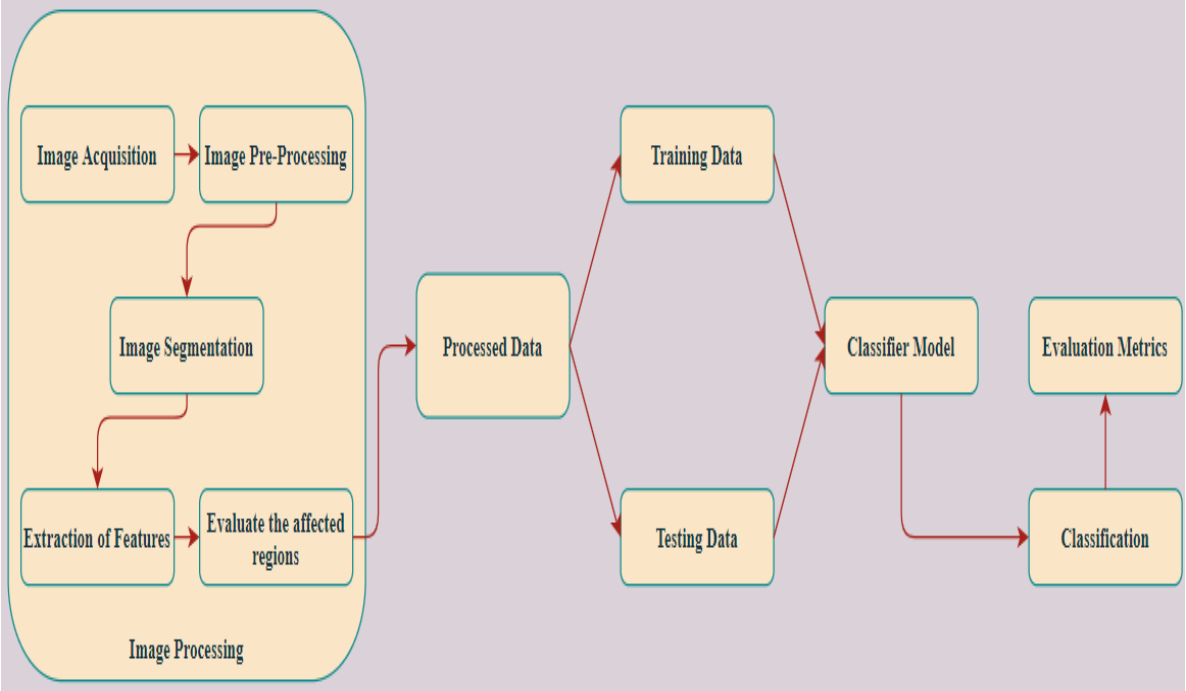

Figure 3.1.1 Module Diagram of the ML Methodology

3.2 System Requirements

3.2.1 Hardware Specification

The System doesn't require any specialized hardware as it doesn't carry out a large amount of processing.

The basic requirements are:

• **Processors**: Intel's Atom® processor or Intel's Core™ processor i3 or above

• **Disk space**: Recommended disk space is 1 GB

• **RAM**: 2GB or more

3.2.2 Software Specification

The project requires an operating system and other software needed for the execution of this application. Operating System provides the underlying environment for the execution of the application. Additional software is needed for the development.

• **Operating systems**: Microsoft Windows 7 or above, Apple's macOS, & Linux

• **Python versions**: Python 3.6.X and above

• **Libraries**:Keras,Tensorflow,flask and other general purpose libraries.

- The project has been developed in a google colab.
- The following packages have been imported and used for the project..

1)Numpy:

A library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

2) Cv2(open cv): OpenCV is a library of bindings designed to solve computer vision problems.

3)Tensorflow: TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

4) Os: The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc. It is also possible to automatically perform many operating system tasks.

5) Sklearn: A free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including SVM, , gradient boosting, K-means, and

DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries Numpy and SciPY.

6) Keras: Keras is an open-source neural network library written in Python. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

7) Matplotlib: A plotting library for the Python programming language and its numerical mathematics extension.

CHAPTER 4:DETAILED ARCHITECTURE

4.1 LIST OF MODULES

1)Image processing

2)Feature Extraction

3)Model Training

4)Classification

5)Web App Implementation

4.1.1 IMAGE PROCESSING

Image preprocessing is required to resize captured images from high resolution to low resolution. The image resizing can be done through the process of interpolation.Batch Normalization is also used to normalize data.
The original image will be let free of an unsharp, or smoothed, version of an image by eliminating them.

4.1.2 FEATURE EXTRACTION

Feature Extraction is one of the most interesting steps of image processing to reduce the efficient part of an image or dimensionality reduction of interesting

parts of an image as a compact feature vector.Feature reduction representation is useful when the image size is large and required to rapidly complete tasks such as image matching and retrieval.This part is handled by our CNN Layer which is used for feature detection and for selection we use Max Pooling and dropout Layer.

4.1.3 MODEL TRAINING

It is done after model curation which is done on the basis of all above information since we are using deep learning most of the task or almost all tasks are handled by our model itself so to perform all above operation we define layers in our model.After curation of Model we train our model by setting our Hyper-parameters.After training part is performed our model is ready for the process of Image Classification.

4.1.4 CLASSIFICATION

It consists of a model that contains predefined patterns that are compared with detected objects to classify them in a proper category.Classification will be executed on the basis of spectral defined features such as density, texture etc. Image classification is performed using Convolution Neural Networks and Deep Learning, and it has introduced the Convolution Neural Network (CNN) as a new area in machine learning and is applied to detect crop disease through classification of image.

CHAPTER 5: IMPLEMENTATION AND RESULTS

5.1. ML IMPLEMENTATION

In the automation of multiple processes, machine learning plays a critical role. The proposed architecture was designed with that goal in mind, and it is based on machine learning methodologies. Especially in the case of detecting and categorizing images into various disease categories. This section has been structured such that the topic begins with the device specifications and data acquisition for the data used in the currently suggested approach. The second point of discussion would be image segmentation. Feature extraction would be the focus of the debate. The fourth point of discussion would be the classification method.

5.1.1 IMAGE PROCESSING

```
# Display images for different species
def plot_defects(defect_types, rows, cols):
    fig, ax = plt.subplots(rows, cols, figsize=(12, 12))
    defect_files = train['File'][train['Disease Type'] ==
defect_types].values
    n = 0
    for i in range(rows):
        for j in range(cols):
            image_path = os.path.join(data_dir, defect_files[n])
            ax[i, j].set_xticks([])
            ax[i, j].set_yticks([])
            ax[i, j].imshow(cv2.imread(image_path))
            n += 1
# Displays first n images of class from training set
plot_defects('Tomato_Bacterial_spot', 5, 5)
```

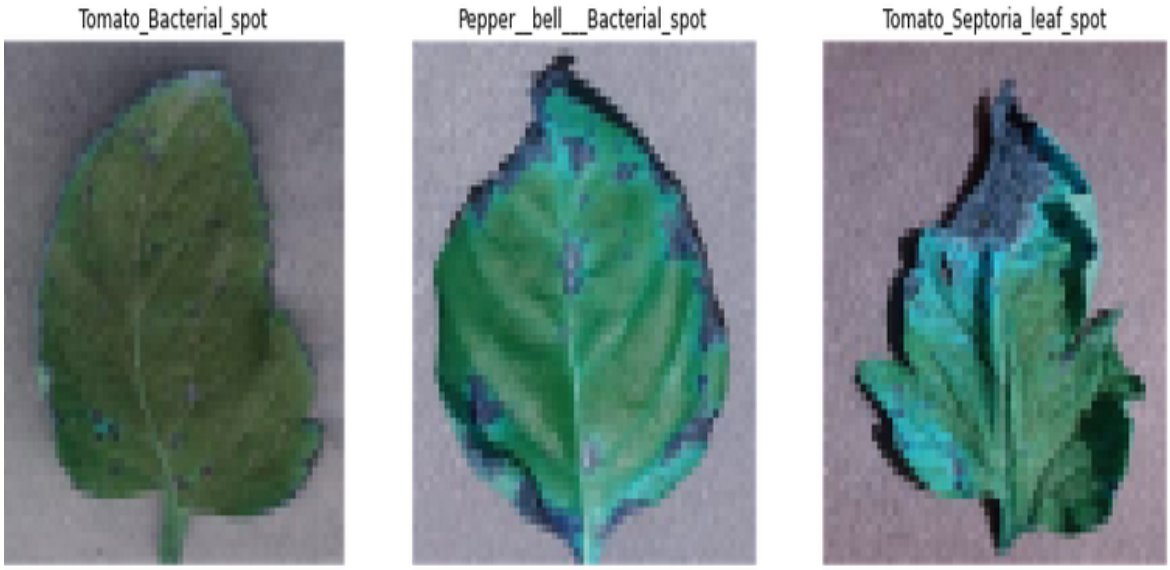

Figure 5.1.1 Image Preprocessing

5.1.2 Feature Extraction

```
densenet = DenseNet121(weights='imagenet', include_top=False)
   input = Input(shape=(SIZE, SIZE, N_ch))
   x = Conv2D(3, (3, 3), padding='same')(input)
   x = densenet(x)
   x = GlobalAveragePooling2D()(x)
   x = BatchNormalization()(x)
   x = Dropout(0.5)(x)
   x = Dense(256, activation='relu')(x)
   x = BatchNormalization()(x)
   x = Dropout(0.5)(x)
   output = Dense(15,activation = 'softmax', name='root')(x)
 # model
    model = Model(input,output)

    optimizer = Adam(lr=0.002, beta_1=0.9, beta_2=0.999, epsilon=0.1,
decay=0.0)
    model.compile(loss='categorical_crossentropy',
optimizer=optimizer, metrics=['accuracy'])
    model.summary()

    return model
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-a
29089792/29084464 [==============================] - 0s 0us/step
29097984/29084464 [==============================] - 0s 0us/step
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 64, 64, 3)]       0

conv2d (Conv2D)              (None, 64, 64, 3)         84

densenet121 (Functional)     (None, None, None, 1024)  7037504

global_average_pooling2d (G  (None, 1024)              0
lobalAveragePooling2D)

batch_normalization (BatchN  (None, 1024)              4096
ormalization)

dropout (Dropout)            (None, 1024)              0

dense (Dense)                (None, 256)               262400

batch_normalization_1 (Batc  (None, 256)               1024
hNormalization)

dropout_1 (Dropout)          (None, 256)               0

root (Dense)                 (None, 15)                3855

=================================================================
Total params: 7,308,963
Trainable params: 7,222,755
Non-trainable params: 86,208
```

Figure 5.1.2 Feature Extraction

5.1.3 MODEL TRAINING

```
model = build_densenet()
annealer = ReduceLROnPlateau(monitor='val_accuracy', factor=0.5,
patience=5, verbose=1, min_lr=1e-3)
checkpoint = ModelCheckpoint('model.h5', verbose=1,
save_best_only=True)
datagen = ImageDataGenerator(rotation_range=360,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
vertical_flip=True)
datagen.fit(X_train)
hist = model.fit_generator(datagen.flow(X_train, Y_train,
batch_size=BATCH_SIZE),
steps_per_epoch=X_train.shape[0],
              epochs=EPOCHS,
              verbose=2,
```

```
                callbacks=[annealer, checkpoint],
                validation_data=(X_val, Y_val))
```

5.1.4 CLASSIFICATION

```
model = load_model('model.h5')
Y_pred = model.predict(X_val)
Y_pred = np.argmax(Y_pred, axis=1)
Y_true = np.argmax(Y_val, axis=1)
cm = confusion_matrix(Y_true, Y_pred)
plt.figure(figsize=(12, 12))
ax = sns.heatmap(cm, cmap=plt.cm.Greens, annot=True, square=True,
xticklabels=disease_types, yticklabels=disease_types)
ax.set_ylabel('Actual', fontsize=40)
ax.set_xlabel('Predicted', fontsize=40)
```

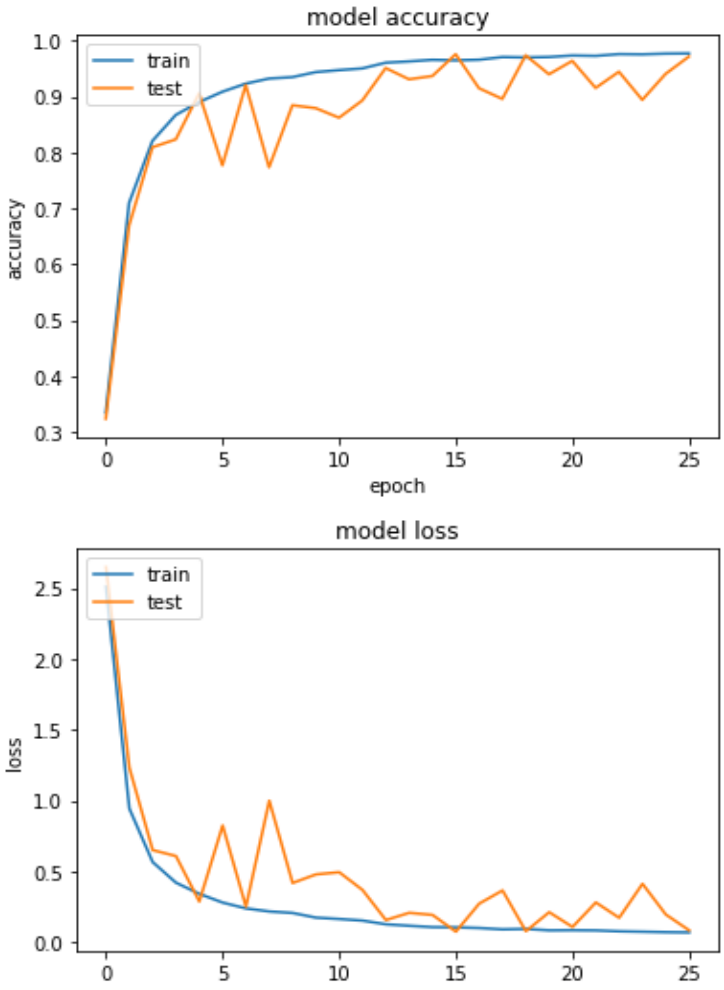

Figure 5.1.3 Accuracy plot

5.2 WEB APP IMPLEMENTATION

Simple web app is created using flask which is a micro-framework for web app creation using python.For this one python file is created which is connected with multiple Javascript, HTML and CSS since I have created a small web app rather than a fancy and complex web app because our aim is just to present our model functionality so I have used one file of each type rather than multiple files

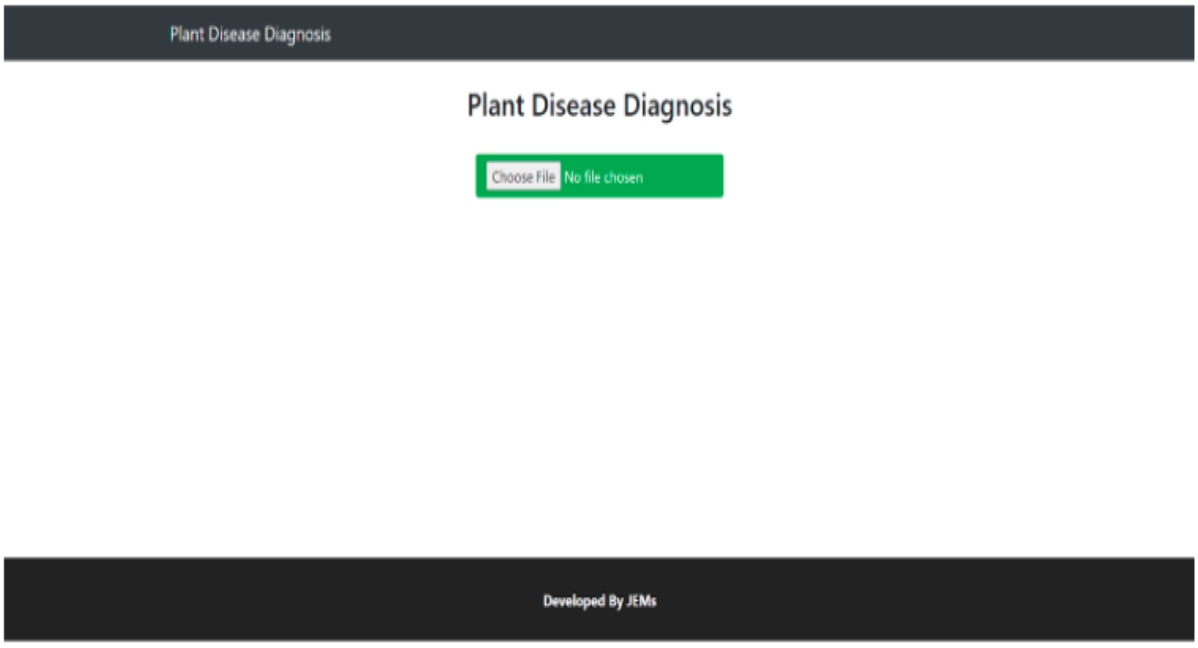

Figure 5.2.1 Home page

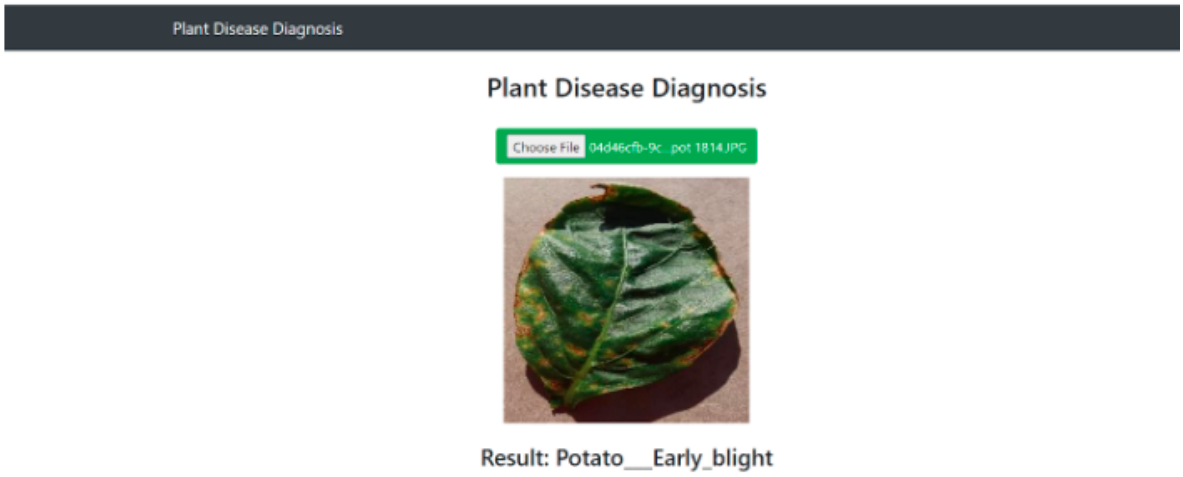

Figure 5.2.2 Prediction page-1

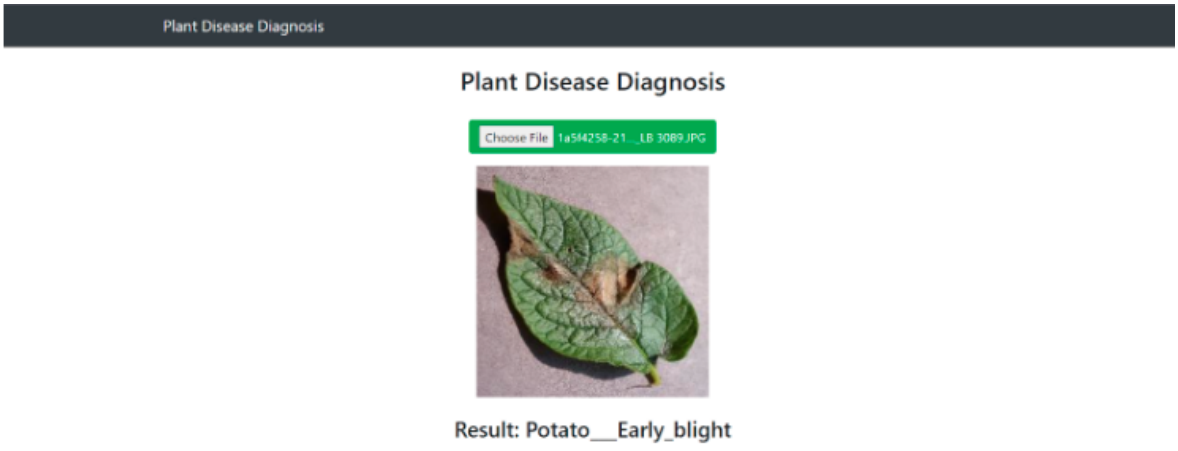

Figure 5.2.3 Prediction page-2

CHAPTER 6: PERFORMANCE METRICS

6.1 Evaluation Metrics

Confusion matrix is extensively used to classify model performance justification. The confusion matrix operates on a set of testing datasets. The true values of familiarity here .For each type of dataset, the entries in the matrix are True Positive (TP) rate, False Positive (FP) rate, True Negative (TN) rate, False Negative (FN). The accuracy is the division of the absolute number of predictions and the predictions that were correct.

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN)$$

$$\text{Recall} = TP/TP+FN$$

$$\text{Precision} = TP/TP+FP$$

$$\text{F1 Score} = 2* \frac{\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}}$$

TABLE 1: Evaluation Metrics of Potato Disease analysis using CNN

Category Name	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
Late Blight	91.07	95.41	93.29	94.71
Early Blight	98.36	94.71	96.43	96.84
Healthy	98.93	98.62	98.76	96.43
Overall	96.12	96.25	96.16	95.99

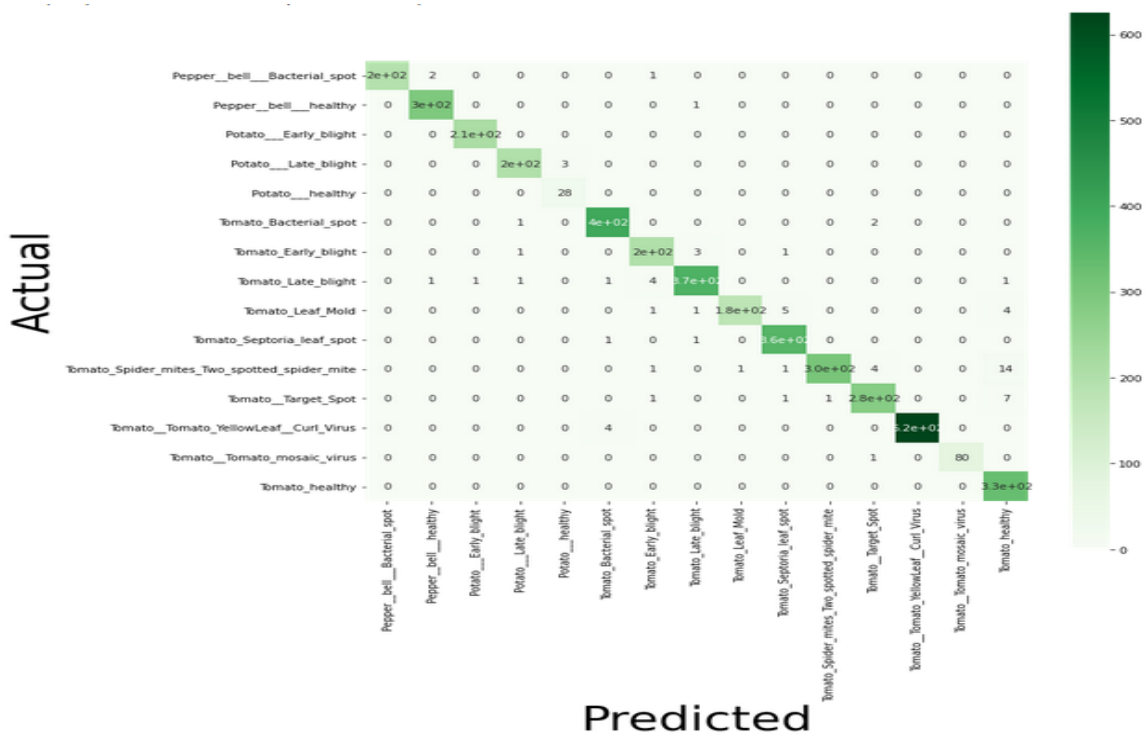

Figure 6.1.2 Confusion matrix obtained for cnn model

CHAPTER 7:CONCLUSION AND REFERENCES

7.1 CONCLUSION

Based on the results observed after implementing our Convolutional Neural Network model, it can be safe to say that our model is able to identify and classify the various plant diseases present in the dataset with good accuracy. Furthermore, the accuracy of our model can be increased by increasing the epoch value but at the loss of processing time. To make the plant disease detection process cater to more plants and diseases, it would be more useful to increase the size of our dataset by collecting images of more plants and their diseases.

7.2 Future Scope

In future we can add more classes of leaves and disease type

The disease detection system can be integrated in a cloud system for efficient result processing.

REFERENCES

[1] Sardogan, M., Tuncer, A., and Ozen, Y.: Plant Leaf Disease Detection and Classification Based on CNN with the LVQ Algorithm. In: 3rd Int. Conf. Comput. Sci. Eng. (2018) 382–385

[2] Wallelign, S., Polceanu, M., and Buche, C.: Soybean plant disease identification using a convolutional neural network. In: Proc. 31st Int. Florida Artif. Intell. Res. Soc. Conf. FLAIRS 2018 (2018), 146–151

[3] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., and Stefanovic, D.: Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification. Comput. Intell. Neurosci. 2016(2016)

[4] Fuentes, A., Yoon, S., Kim, S. C., and Park, D. S.: A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors (Switzerland) 17 (2017).

[5] Arivazhagan, S. and Ligi, S. V.: Mango Leaf Diseases Identification Using Convolutional Neural Network. Int. J. Pure Appl. Math., 120(2018) 11067–11079

[6] Oppenheim, D. and Shani, G.: Potato Disease Classification Using Convolutional Neural Networks. Adv. Anim. Biosci., 8 (2017), 244–249

[7] Barbedo, J. G. A.: Factors influencing the use of deep learning for plant disease recognition. Biosyst. Eng., 172 (2018) 84–91

[8] Brahimi, M., Boukhalfa, K., and Moussaoui, A.: Deep Learning for Tomato Diseases: Classification and Symptoms Visualization. Appl. Artif. Intell., 31 (2017) 299–315

[9] Shrivastava, V. K., Pradhan, M. K., Minz, S., and Thakur, M. P.: Rice plant disease classification using transfer learning of deep convolutional neural networks. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch., 42 (2019) 631–635