iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VIII **Month of publication:** August 2022

DOI: https://doi.org/10.22214/ijraset.2022.46394

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue VIII August 2022- Available at www.ijraset.com

Crop Health Card for Plant Diseases

Arfa Fathima[1], Pragathi. K[2], Shreshta Anil Kumar[3], Usha Lokesh[4], Asst. Prof. Dr. M.Rajeswari[5]
[1, 2, 3, 4, 5]*Electronics and Telecommunication Engineering Bangalore Institute of Technology Bangalore, Karnataka*

Abstract: ***Farmers have difficulty identifying, detecting, and treating plant diseases and their causes. Fruits are more susceptible to disease depending on the season and environmental conditions in which they grow during cultivation. The traditional method of predicting disease in fruits and plants is extremely difficult. Using the proposed model, a large dataset of different types of fruits and their diseases can be created, with a large number of images stored in different folders for processing the data set. Each data set is labeled with the type of disease and the affected fruit. This system is simple to train and test.***
Artificial Neural Networks are used to learn and categorize fruits and diseases. The proposed model can detect disease accurately and provide farmers with preventive measures and recommendations. Farmers all over the world will benefit from this system.

I. INTRODUCTION

The primary goal of this dissertation is to create a system that will allow farmers to accurately identify various crop diseases using images captured by smart phone cameras.
Deep learning has proven to offer significant opportunities for image classification and object recognition capabilities in recent years [1].
It entails training a neural network with images collected from actual crop leaves infected with a disease [2]. After training, the model can be integrated into a mobile app or deployed on a cloud server for use as a crop disease detector. Traditional computer vision algorithms and methods were primarily based on image processing algorithms and methods.
It was chiefly used for extracting image features such as detecting the corners, edges, and hue of objects. The main difficulty with this approach in traditional computer vision for a classification task is that you must select which features to look for in each given image.
When the number of class features increases, it becomes difficult to keep up. Deep learning DL has pushed the boundaries of what was previously possible in image classification. The goal of our work is to collect, identify, and classify various plant diseases.
To detect and classify the disease, use a machine learning algorithm and image processing. Calculate the precision. Our work entails automatically classifying and detecting plant diseases.
Agriculture is the cultivation of plants. Agriculture was a critical development in the rise of sedentary human civilization, as farming of domesticated species produced food surpluses that allowed people to live in cities. Plants were grown independently in at least 11 different parts of the world. Within the 12th century, industrial agriculture supported large-scale monoculture came to dominate agricultural output, despite the fact that approximately 2 billion people still relied on subsistence agriculture.
Food and fiber are two of the most important agricultural products. Grains, vegetables, and fruits are examples of food classes.
Agriculture employs more than one-third of the world's workforce, second only to the service sector. Plant breeding, agrochemicals such as pesticides and fertilizers, and technological advancements has increased crop yields significantly while causing ecological and environmental damage.

II. METHODOLOGY

To predict the disease of the fruit or the leaf of the fruit plant, several steps must be taken in the system, beginning with camera capture, training, segmentation, processing, calculating weights and mapping with datasets, and then predicting the disease that is present and providing suggestions to overcome the problem.

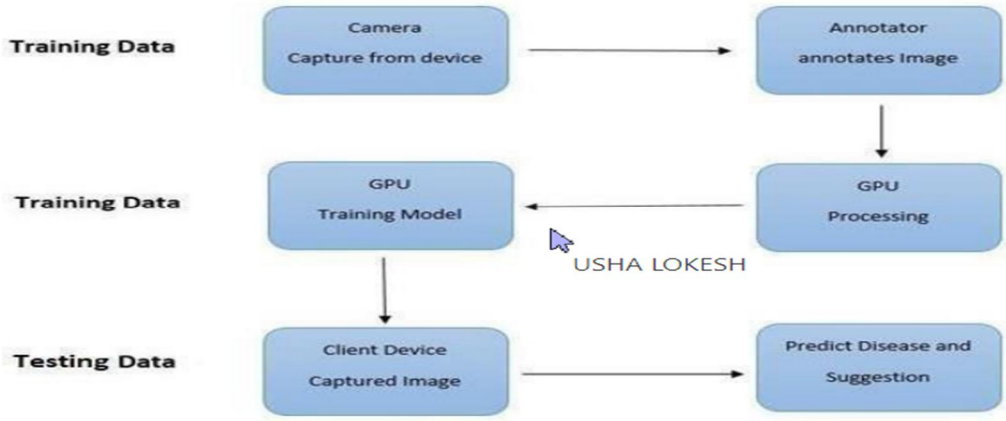


Figure 1 Methodology Process

As illustrated in Figure 1, top-level use cases are better suited to describing user interactions with the system. This is due to the fact that they provide information not only about the system's behavior, but also about the sequence of interactions that farmers typically perform in order to achieve a goal. After scanning some items on his Android phone, the farmer should proceed to the remedy page. The user can now remove or modify the diseased part that has been scanned until the scan is clear enough to begin the prediction process.

A. Image Recognition Using Deep Learning

Image recognition is the task of identifying images and categorizing them in one of several predefined distinct classes. So, image recognition software and apps can define what's depicted in a picture and distinguish one object from another.

Image recognition is one of the tasks in which deep neural networks (DNNs) excel. Neural networks are computing systems designed to recognize patterns. Their architecture is inspired by the human brain structure, hence the name. They consist of three types of layers: input, hidden layers, and output. The input layer receives a signal, the hidden layer processes it, and the output layer makes a decision or a forecast about the input data. Each network layer consists of interconnected nodes (artificial neurons) that do the computation.

The leading architecture used for image recognition and detection tasks is Convolutional Neural Networks (CNNs). Convolutional neural networks consist of several layers with small neuron collections, each of them perceiving small parts of an image. The results from all the collections in a layer partially overlap in a way to create the entire image representation. The layer below then repeats this process on the new image representation, allowing the system to learn about the image composition.

B. Implementation Of Image-Recognition Model

The first step is to import necessary packages for building the model. The packages include pandas, humpy, seaborn, matplotlib, tensorflow, keras and sklearn. Once the basic packages have been imported, deep learning packages are imported after tensorflow gpu has been installed using .whl file. tf.test.is_built_with_cuda() and tf.test.is_built_with_gpu_support() commands are used to test if cuda and gpu support have been enabled. Another OS module is imported to explore the dataset used for training the deep learning model. Os.listdir (train_dir) is used for checking contents of train and test directory. Plt.imshow(a1) command is used to check one image in directory to confirm if the correct directory has been accessed. For each image in the dataset, average size of the images is calculated. For loops are used to go through each image in training and testing directory and normalizing them using pillow. This step is optional since we can use OpenCV and in experimentation we have higher accuracy. Next, one image is accessed from each class like Apple_black_rot, Tomato_Late_Blight, Powdery_Mildew, Tomato_Bacterial_Spot, Grape_Esca, Corn_Northern_Leaf_Blight, Pepper_Bell_Bacterial_Spot bearing unique labels are in the right directories. The next important phase of the image recognition model is training the dataset. A tensorboard variable is setup to check the training epochs of deep learning models. A function is created to access all images across all twenty-nine classes and a training and test split is created using scikit-learn. Using the ImageDataGenerator() function, an image data generator is created. Using the image gen random transform(), image augmentation is added to the existing dataset.

Next, a model has to be defined for image recognition, once the dataset has been cleaned and prepared. The flow diagram and detailed explanation of the model is shown in figure. Image augmentation introduced to the image dataset include the rotation, horizontal flip, vertical flip, addition of sheer, padding and other such noise inducing opera tions which may introduce variety into the dataset and prevent overfitting on the image data and ensure a good validation accuracy for the trained model. In addition to pillow, openCV can be used to carry out important image processing operations such as background elimination and scikitlearn can be used to replace image data generators by invoking the train test split functionality, which would improve the efficiency.

The first step is to create Convolutional Neural Network of suitable architecture. The Sequential model is initialized with the following layers

1) 1 X DepthwiseConv2D with 24 filters
2) 1 X Conv2D with 32 filters.
3) 1 X MaxPool2D layer with 32 filters.
4) 1 X DepthwiseConv2D with 32 filters
5) 1 X Conv2D with 64 filters.
6) 1 X MaxPool2D layer with 64 filters
7) 1 X DepthwiseConv2D layer with 64 filters
8) 1 X Conv2D layer with 128 filters.
9) 1 X MaxPool2D layer with 128 filters.
10) 1 X Batch Normalization. • 1 X Flatten layer.
11) 1 X Densely Packed layer with 52 perceptrons with activation function = 'relu'.
12) 1 X Densely Packed layer with 29 neurons (number of classification) with "Softmax" Activation function.
13) Optimizer chosen is 'adam','rmsprop' is a suitable alternative. • Loss function is 'categorical cross entropy'.
14) Kernel regularize is 'l2'.

Model: "Plant_Leaf"

Layer (type)	Output Shape	Param #	Connected to
Input (InputLayer)	[(None, 256, 256, 3)	0	
DwC_1 (DepthwiseConv2D)	(None, 256, 256, 24)	120	Input[0][0]
PwC_1 (Conv2D)	(None, 256, 256, 32)	800	DwC_1[0][0]
MP_1 (MaxPooling2D)	(None, 128, 128, 32)	0	PwC_1[0][0]
DwC_2 (DepthwiseConv2D)	(None, 128, 128, 32)	160	MP_1[0][0]
PwC_2 (Conv2D)	(None, 128, 128, 64)	2112	DwC_2[0][0]
MP_2 (MaxPooling2D)	(None, 64, 64, 64)	0	PwC_2[0][0]
DwC_3 (DepthwiseConv2D)	(None, 64, 64, 64)	320	MP_2[0][0]
PwC_3 (Conv2D)	(None, 64, 64, 128)	8320	DwC_3[0][0]
MP_3 (MaxPooling2D)	(None, 32, 32, 128)	0	PwC_3[0][0]
DwC_4 (DepthwiseConv2D)	(None, 32, 32, 128)	640	MP_3[0][0]
PwC_4 (Conv2D)	(None, 32, 32, 256)	33024	DwC_4[0][0]
MP_4 (MaxPooling2D)	(None, 16, 16, 256)	0	PwC_4[0][0]
DwC_5 (DepthwiseConv2D)	(None, 16, 16, 256)	1280	MP_4[0][0]
PwC_5 (Conv2D)	(None, 16, 16, 512)	131584	DwC_5[0][0]
MP_5 (MaxPooling2D)	(None, 8, 8, 512)	0	PwC_5[0][0]
GAP_1 (GlobalAveragePooling2D)	(None, 32)	0	MP_1[0][0]
GAP_2 (GlobalAveragePooling2D)	(None, 64)	0	MP_2[0][0]
GAP_3 (GlobalAveragePooling2D)	(None, 128)	0	MP_3[0][0]
GAP_4 (GlobalAveragePooling2D)	(None, 256)	0	MP_4[0][0]
GAP_5 (GlobalAveragePooling2D)	(None, 512)	0	MP_5[0][0]
Concat (Concatenate)	(None, 992)	0	GAP_1[0][0] GAP_2[0][0] GAP_3[0][0] GAP_4[0][0] GAP_5[0][0]
Dropout_1 (Dropout)	(None, 992)	0	Concat[0][0]
Dense_1 (Dense)	(None, 512)	508416	Dropout_1[0][0]
Dropout_2 (Dropout)	(None, 512)	0	Dense_1[0][0]
Output (Dense)	(None, 38)	19494	Dropout_2[0][0]

Total params: 706,270
Trainable params: 706,270
Non-trainable params: 0

Figure 2 Model Summary

III. IMPLEMENTATION

This section describes the entire process of developing a deep learning model for crop disease recognition. This includes the specifics of the design as well as the entire process in detail. Starting with dataset collection, the proposed solution and assumptions are thoroughly developed. The implementation is discussed in figure 3

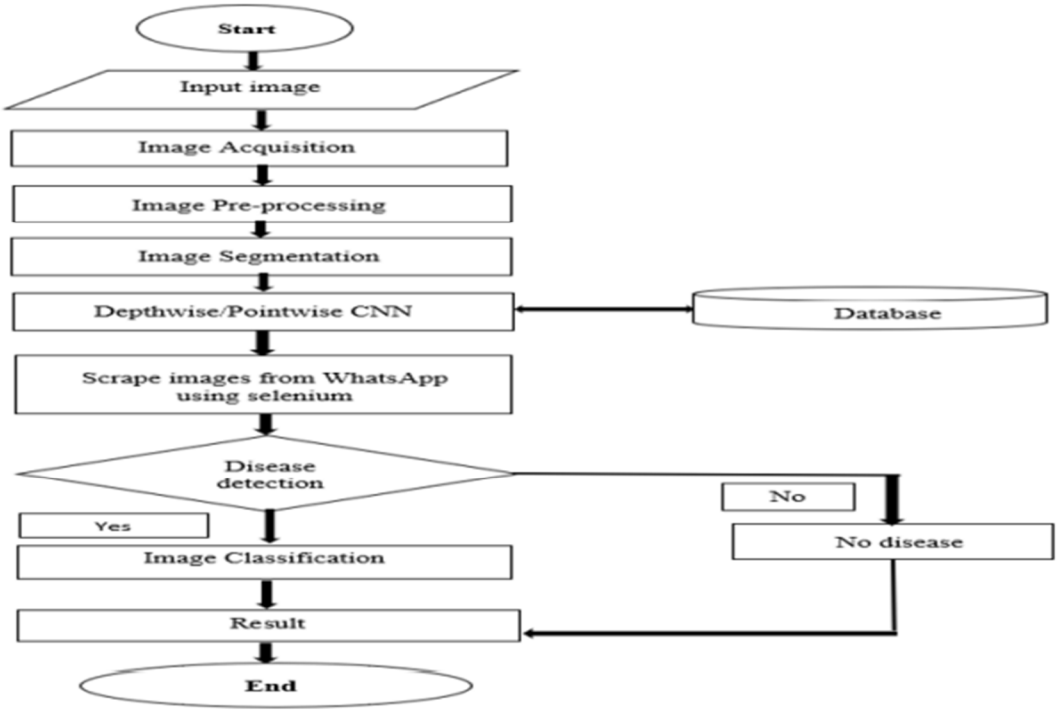


Figure 3 Implementation Flowchart

A. Building of Dataset

To train a deep learning network for our research problem, a dataset of diseased and healthy leaf images had to be acquired the dataset was put together by downloading high quality diseased images on the internet from various sources and taking pictures of leaves using a mobile phone camera. Plants are affected by diseases caused by various pathogenic fungi, bacteria, and viruses and to damage by parasitic worms and physiological disturbances also classified as diseases.

1) Image Acquisition: Image acquisition in image processing can be broadly defined as the action of retrieving an image from some source, usually a hardware-based source, so it can be passed through whatever processes need to occur afterward. Performing image acquisition in image processing is always the first step in the workflow sequence because, without an image, no processing is possible.

2) Image Pre-Processing: Image processing is a procedure to change an input image onto digital frame and performed a couple of process, with the true objective to get a redesigned picture or to separate some significant data from it. It is a sort of signal dispensation which input is image, like video edge or photo and result may be picture or characteristics related with that image. The image pre-processing includes filtering, color conversion and detail enhancement of image.

3) Image Segmentation: Image segmentation is the task of clustering parts of an image together that belong to the same object class. Image segmentation is an extension of image classification where, in addition to classification, we perform localization. Image segmentation thus is a superset of image classification with the model pinpointing where a corresponding object is present by outlining the object's boundary

4) Depth wise/Point wise CNN: Point wise Convolution is a type of convolution that uses a 1x1 kernel: a kernel that iterates through every single point. This kernel has a depth of however many channels the input image has from the database.

5) Scrape Images using from WhatsApp using Selenium: Selenium can be used to automate web browser interaction with Python. Basically, selenium pretends to be a real user, it opens the browser which is WhatsApp Web, “moves” the cursor around and clicks buttons if you tell it to do so and downloads the most recent image. The initial idea behind Selenium, is automated testing.

6) Disease Detection: It means representing the image into the different meaningful part that makes easy to analyze the images. The images are segmented into 3 clusters depending on the color variation. Among the three clusters, the disease affected part is chosen to extract features and if it classifies the given image as to which disease it is.

B. Dataset Division

For the purposes of training and testing the model, three separate datasets are required. In this process we are subdivided the dataset put together in the previous section into the following sets:

1) *Training Set:* This is the actual set used to train the model to learn its hidden parameters such as weights and biases.
2) *Validation Set:* The validation set is used to evaluate the model. This accomplished by manually fine-tuning model hyper parameters allowing the training process to be monitored and to detect over fitting. These include among others the learning rate, batch size and number of epochs.
3) *Test Set:* This set is used when the training phase has been completed to evaluate the performance and accuracy of the final trained model.

IV. SOFTWARE REQUIREMENTS

1) *Operating System:* Windows 8 above
2) *Programming Language:* Python, JavaScript, HTML, CSS
3) *IDE:* Jupyter Notebook, VSCode
4) *API:* WhatsAp

V. RESULTS

The project is divided into 2 phases. The training phase and the testing phase. The training data is a type of data builds up the machine learning algorithm. The data scientist feeds the algorithm input data, which corresponds to an expected output. The model evaluates the data repeatedly to learn more about the data's behavior and then adjusts itself to serve its intended purpose. After the model is built, testing data once again validates that it can make accurate predictions. If training and validation data include labels to monitor performance metrics of the model, the testing data should be unlabeled. Test data provides a final, real-world check of an unseen dataset to confirm that the ML algorithm was trained effectively.

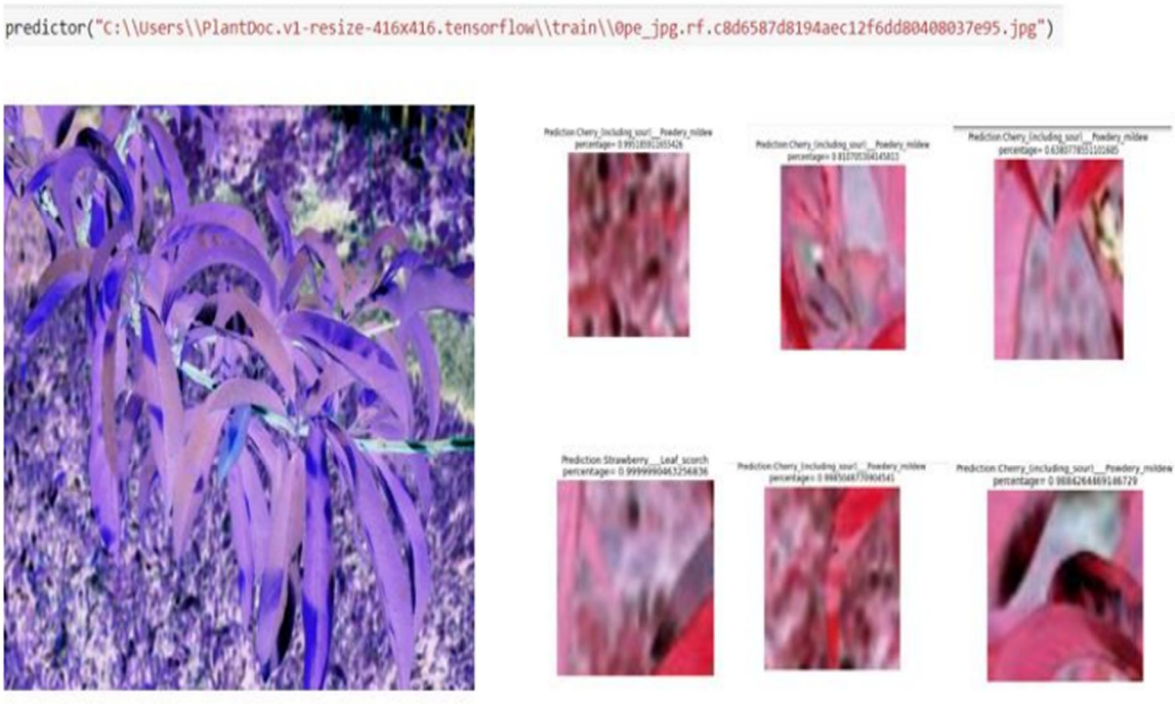


Figure 4 Jupyter Notebook Output

Figure 4 shows the output obtained on testing an image in Jupyter Notebook. The input image is divided into 25 frames and each frame is analyzed for the presence of disease. Disease is detected only in those frames where discoloration and rotting, is observed, the other frames are displayed as healthy and only diseased frames are shown as output.

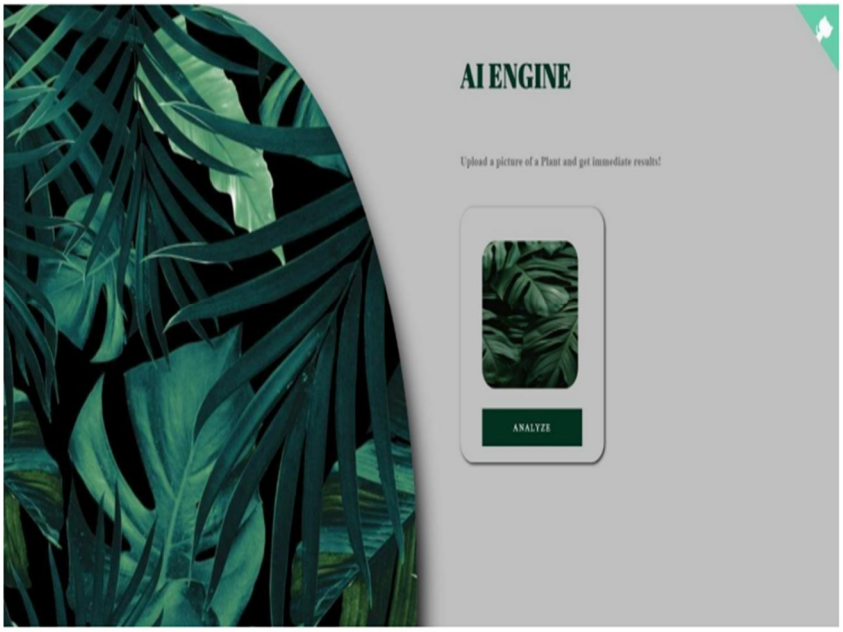


Figure 5 User Interface

Figure 5 illustrates the user interface of the Web application where the farmer or the user can input the image captured from the camera to help detect if a disease is present or not. The input image can be directly captured from the camera or can be an image from the existing dataset. The analyze button can be used to show the name of the disease.

Figure 6 Disease Prediction

Figure 6 shows the result of the input image, and the disease detected is Late blight in potato. If the plant given as an input has been affected by a particular disease, then the name will be displayed along with necessary remedy instructions and a list of pesticide and fungicides that can be applied in order to prevent further spread of the said disease.

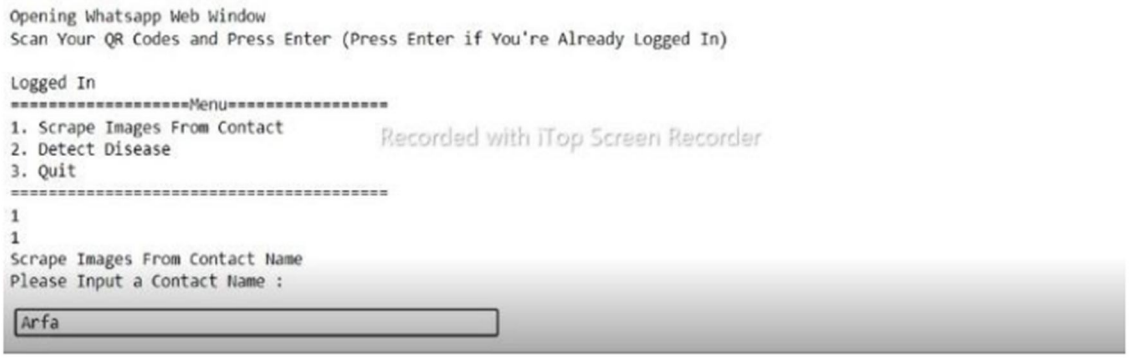


Figure 7 Menu Box

Figure 7shows the menu box for the WhatsApp API feature

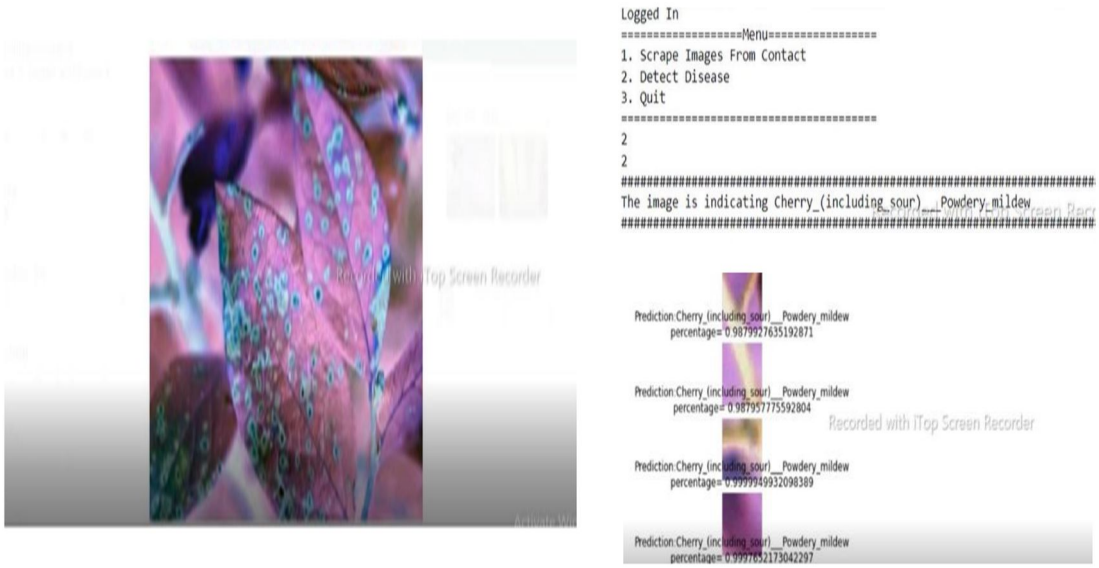


Figure 8WhatsApp API Feature

Figure 8 shows the usage of WhatsApp API feature for disease detection. The goal of the WhatsApp API is to use Python and Selenium to scrape all images from a certain contact or group.Through the WhatsApp web interface, PyWhatsApp is used to automate WhatsApp. The contacts we want to send messages or media attachments to can be added in any quantity (like Video or Images). One from Automation and the other for message scheduling, Selenium, Autoit, and Schedule have all been used.

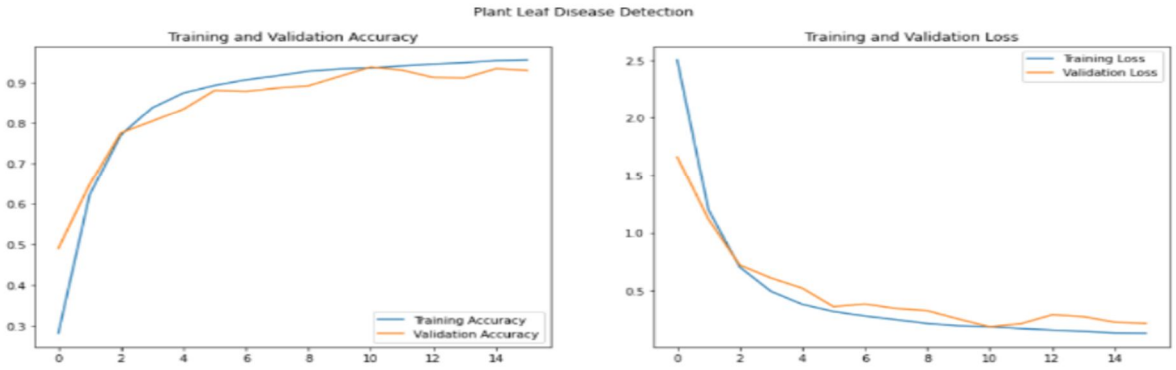


Figure 9 Training and Validation Accuracy and Loss

Figure 9 shows the graphs obtained in Jupyter notebook that specifies the training and validation accuracy and also the training and validation losses.

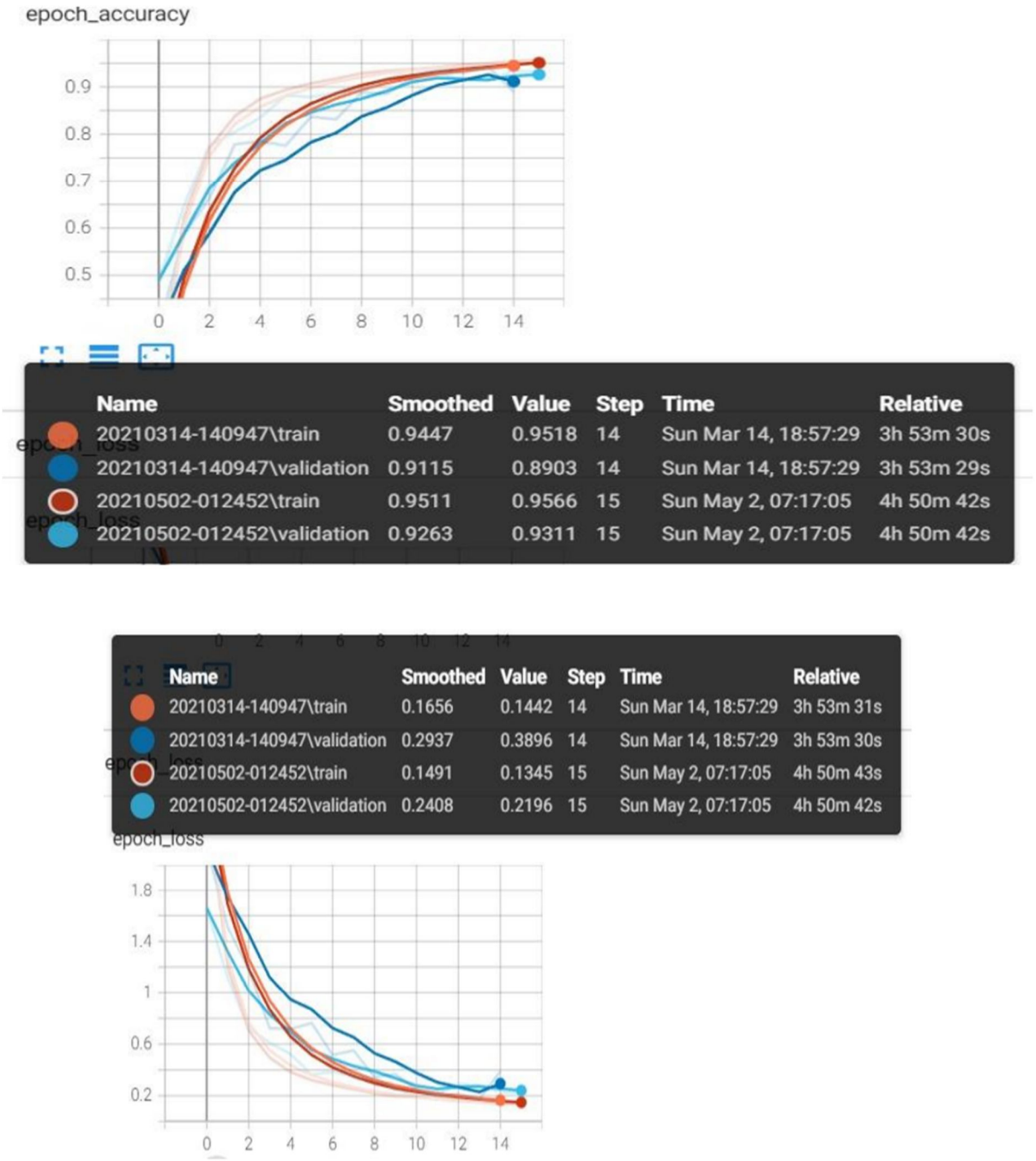


Figure 10 Epoch Accuracy and Loss

Figure 10shows the graphs generated in tensor board that details about the accuracy and loss per Epoch. The accuracy is 94.47% and the loss is 1.6%. Loss refers to the loss value over the training data after each epoch. This is what the optimization process is trying to minimize with the training so, the lower, the better. Accuracy refers to the ratio between correct predictions and the total number of predictionsin the training data. The higher, the better. The formula for Epoch accuracy and loss as follows: Total no. of correct predictions/ Total no. of predictions = Total no. of (true positives+ true negatives)/ Total no. of (true positives+ true negatives+ false positives+ false negatives).

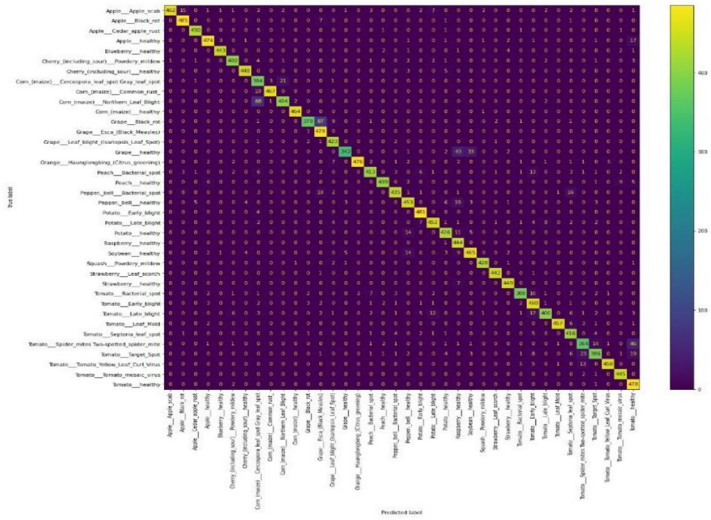


Figure 11 Datasets required for predictive analysis

Figure 11 shows the datasets required for predictive analysis. From the entire dataset, sample datasets are created for every filename, every disease prediction is made, in the 6th option there was a wrong prediction made with Apple_Scab being predicted as Potato_Late_Blight. So, the test accuracy is made where out of 32 predictions, 31 predictions were accurate, hence 96% accurate.

	Filename	Predicted classes
0	AppleCedarRust1.JPG	Apple___Cedar_apple_rust
1	AppleCedarRust2.JPG	Apple___Cedar_apple_rust
2	AppleCedarRust3.JPG	Apple___Cedar_apple_rust
3	AppleCedarRust4.JPG	Apple___Cedar_apple_rust
4	AppleScab1.JPG	Squash___Powdery_mildew
5	AppleScab2.JPG	Apple___Apple_scab
6	AppleScab3.JPG	Potato___Early_blight
7	CornCommonRust1.JPG	Corn_(maize)___Common_rust_
8	CornCommonRust2.JPG	Corn_(maize)___Common_rust_
9	CornCommonRust3.JPG	Corn_(maize)___Common_rust_
10	PotatoEarlyBlight1.JPG	Potato___Early_blight
11	PotatoEarlyBlight2.JPG	Potato___Early_blight
12	PotatoEarlyBlight3.JPG	Potato___Early_blight
13	PotatoEarlyBlight4.JPG	Potato___Early_blight
14	PotatoEarlyBlight5.JPG	Potato___Early_blight
15	PotatoHealthy1.JPG	Potato___healthy
16	PotatoHealthy2.JPG	Potato___healthy
17	TomatoEarlyBlight1.JPG	Tomato___Early_blight
18	TomatoEarlyBlight2.JPG	Tomato___Early_blight
19	TomatoEarlyBlight3.JPG	Tomato___Early_blight
20	TomatoEarlyBlight4.JPG	Tomato___Early_blight
21	TomatoEarlyBlight5.JPG	Tomato___Early_blight
22	TomatoEarlyBlight6.JPG	Tomato___Early_blight
23	TomatoHealthy1.JPG	Tomato___healthy
24	TomatoHealthy2.JPG	Tomato___healthy
25	TomatoHealthy3.JPG	Tomato___healthy
26	TomatoHealthy4.JPG	Tomato___healthy
27	TomatoYellowCurlVirus1.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus
28	TomatoYellowCurlVirus2.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus
29	TomatoYellowCurlVirus3.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus
30	TomatoYellowCurlVirus4.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus
31	TomatoYellowCurlVirus5.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus
32	TomatoYellowCurlVirus6.JPG	Tomato___Tomato_Yellow_Leaf_Curl_Virus

Figure 12 Confusion Matrix.

Figure 12 shows the confusion matrix. A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm. For example, the machine has identified 462 correct images of Apple_Scab, hence, highlighted in (0, 0), in (0, 2) 15 images of Apple_Scab has been wrongly detected as Apple_Black_Rot.

VI. CONCLUSION

This Methodology ensures that all types of identification problems are managed, such as assembly line product identification and fruit or vegetable disease identification. It reduces the number of plants that die as a result of incorrect disease identification. The main goal was to learn and design a machine learning application development within an android app, providing tensor flow framework implementation flexibility within any environment. This aided in the development of an exclusive Web application that employs machine learning. The development of Web

Application is a revolutionary addition to the project. A web application is feasible option for farmers to detect the disease as one can input the image captured from a mobile device directly to the PC and be able to view the results.

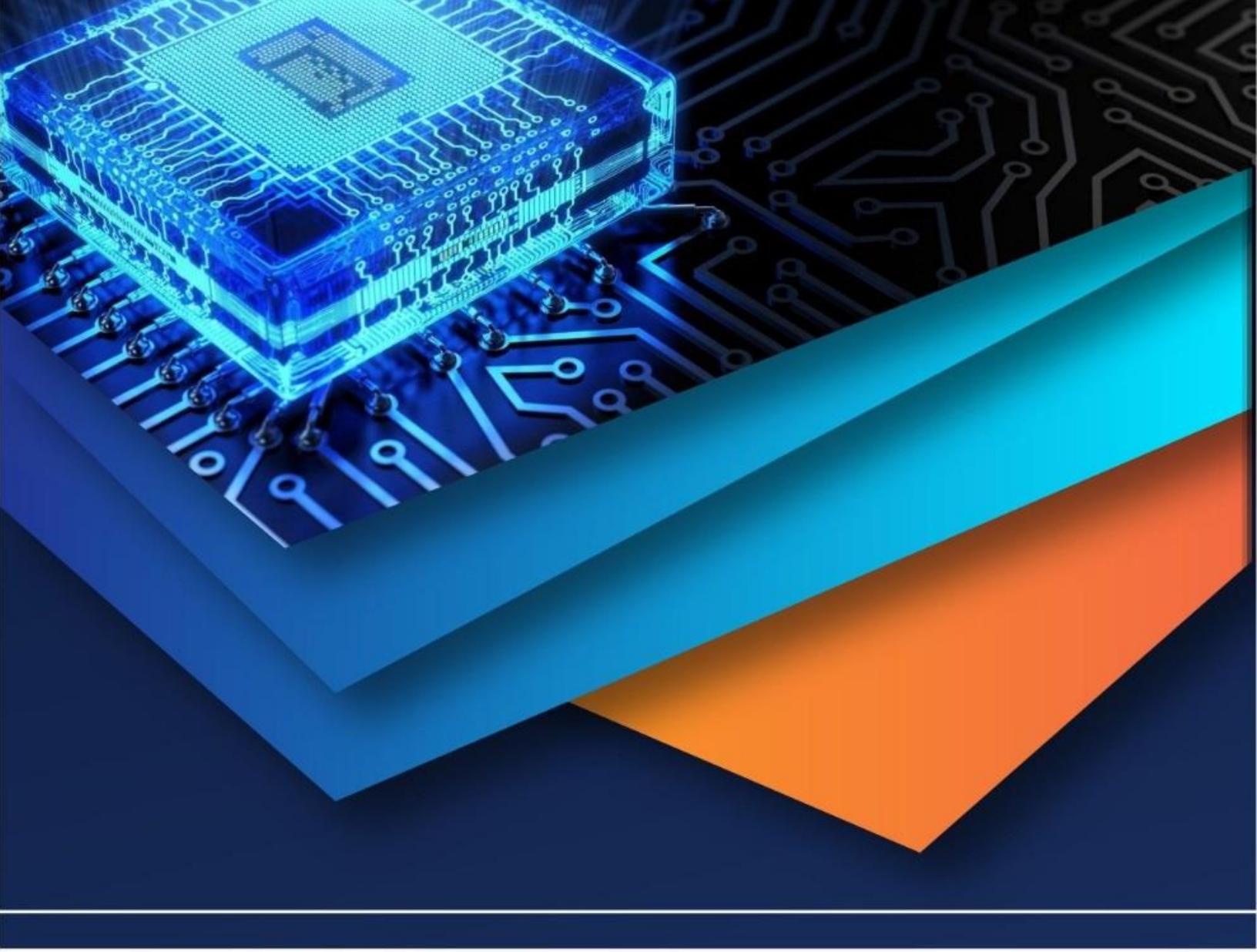
The work carried out during Project Phase-2 includes the collection of images of various plants to create a sufficient database, as well as the generation and testing of the code with hardware implementation. The design of the application's user interface is followed by the integration of data into the web application. Detection of plant disease through some automatic technique is beneficial as it reduces a large work of monitoring a big farm of crops, and at a very early stage itself, detects the symptoms of diseases, i.e., when they appear on plant leaves. This paper has sought to present a methodology for identifying and diagnosing plant leaf diseases, which will be valuable to farmers around the world in enhancing the health of their crops and treating diseased ones at an early stage. The work carried out has significance to the real-world categorization of crop disease and it involves both image processing and pattern recognition techniques.

VII. FUTURE SCOPE

1) Dataset used in the proposed method can be extended other plant diseases.
2) Other methods can be used for transfer learning.
3) This work is focused on front view of Paddy leaf but can be extended to side view and back view also.
4) Fusion of some other advance features will improve the output.

REFERENCES

[1] Savita N. Ghaiwat, Parul Arora, “Detection and Classification of Plant Leaf Diseases Using Image processing Techniques: A Review” 3, 2014.
[2] Rutu Gandhi ; Shubham Nimbalkar ; Nandita Yelamanchili ; Surabhi Ponkshe, “Plant disease detection using CNNs and GANs as an augmentative approach”, IEEE InternationalConference on Innovative Research and Development (ICIRD), 2018
[3] Prasanna Mohanty, David Hughes and Marcel Salathe, "Using Deep Learning for ImageBased Plant Disease Detection", 2016, Frontiers in Plant Science,7(September),[1419].
[4] S.Arivazhagan, R. Newlin Shebiah, S.Ananthi, S.Vishnu Varthini. 2013. “Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features”. Agric Eng Int: CIGR Journal
[5] Revathi, P.Hemalatha, “detection and classification of leaf disease using artificial neural net-work”, International Journal of Technical Research and Applications e-ISSN: 2320- 8163, Volume 3, Issue 3 (May-June 2017), PP. 331-333
[6] Arti N. Rathod, Bhavesh A. Tanawala, Vatsal H. Shah, “Leaf Disease Detection using Image Processing and Neural Network”, International Journal of Advance Engineering and Research Development (IJAERD) Volume 1,Issue 6,June 2014.
[7] Hulya Yalcin, 2021, “Green Leaf Disease Detection using CNN”, International Journal of Engineering Research & Technology (Ijert) Ncraem – 2021 (Volume 09 – Issue 15).
[8] Ms. Nilam Bhise, Ms. Shreya Kathet, Mast. Sagar Jaiswar, Prof. Amarja Adgaonkar. “Plant disease detection using machine learning”. Volume: 07 Issue: 07| July 2020. eISSN: 2395-0056.p-ISSN: 2395- 0072.International research journal of engineering and technology (IRJET).
[9] Aakanksha Rastogi, Ritika Arora, and Shanu Sharma, proposed "Leaf Disease Detection and Grading using Computer Vision Technology & Fuzzy Logic"2nd International Conference on Signal Processing and Integrated Networks2015.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY