TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

ADVANCED COLLEGE OF ENGINEERING AND MANAGEMENT

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

BALKHU, KATHMANDU

A Major Project Proposal Report On

PLANT DISEASE DETECTION USING IMAGE PROCESSING

PROJECT CODE: EX755

Submitted By:

BISHAL GODAR(24907)

SADIKSHYA SUBEDI(24922)

SHITAL KUMAR NYAUPANE(24923)

SWETA ADARSH(24926)

A project report submitted to the department of Electronics and Computer Engineering in the partial fulfillment of the requirements for degree of Bachelor of Engineering in Computer Engineering

Lalitpur, Nepal

Supervised By:

Er. Krishna Kumar Jha

11 March 2022

ADVANCED COLLEGE OF ENGINEERING AND MANAGEMENT
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LETTER OF APPROVAL

The undersigned certify that they have read and recommended to the Institute of Engineering for acceptance, a project report entitled "PLANT DISEASE DETECTION USING IMAGE PROCESSING" submitted by

Bishal Godar 24907
Sadikshya Subedi 24922
Shital Kumar Nyaupane 24923
Sweta Adarsh 24926

In partial fulfillment for the degree of Bachelor in Electronics and Computer Engineering.

……
Supervisor
Er. Krishna Kumar Jha
Electronic Engineer
Ministry of Communication and Information Technology

……
External Examiner

……
Er. Ajaya Shrestha
Head of Department
Senior Lecture
Department of Computer and Electronics

COPYRIGHT

ACKNOWLEDGEMENT

We take this opportunity to express my deepest and sincere gratitude to our supervisor
Er. Krishna Kumar Jha, Electronic Engineer, Ministry of Communication and Information Technology, for his insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also for his constant encouragement and advice throughout our Bachelor's programme.

We express our deep gratitude to Er. Ajaya Shrestha , Head of Department of Electronics and Computer Engineering , Advance College of Engineering and Management, Er. Amit Kumar Rauniyar, Project Coordinator, Head of Department of Electronics and Computer Engineering , Advance College of Engineering and Management, for their regular support, co-operation, and coordination.

The in-time facilities provided by the department throughout the Bachelors program are also equally knowledgeable.

We would like to convey our thanks to the teaching and non-teaching staff of the Department of Electronics & Communication and Computer Engineering, ACEM for their invaluable help and support throughout the period of Bachelor’s Degree. We are also grateful to all our classmates for their help, encouragement and invaluable suggestions.

Finally, yet more importantly, we would like to express our deep appreciation to our grandmother, parents, sister and brother for their perpetual support and encouragement throughout the Bachelor’s degree period.

Bishal Godar (24907)
Sadikshya Subedi (24922)
Shital Kumar Nyaupane (24923)
Sweta Adarsh (24926)

ABSTRACT

In the agriculture sector, one of the major problems in the plants is its diseases. The plant diseases can be caused by various factors such as viruses, bacteria, fungus etc. Most of the farmers are unaware of such diseases. That's why the detection of various diseases of plants is very essential to prevent the damages that it can do to the plants itself as well as to the farmers and the whole agriculture ecosystem. Regarding these practical issues, this research aimed to classify and detect the plant's diseases automatically especially for the tomato plant. As per the hardware requirement, Raspberry Pi is the major computing unit. Image processing is the key process of the project which includes image acquisition, adjusting image ROI, feature extraction and convolution neural network (CNN) based classification. Here, the Python programming language, OPENCV library, is used to manipulate raw input images. To train on CNN architecture and create a machine learning model that can predict the type of diseases, image data is collected from the authenticated online source. As a result , few diseases that usually occur in tomato plants such as Late blight, Early Blight and Healthy leaf are detected.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

CNN : Convolutional Neural Network

MATLAB : MATrix LABoratory

MuPAD : Multi Processing Algebra Data

LABVIEW : Laboratory Virtual Instrument Engineering Workbench

CHAPTER 1

INTRODUCTION

1.1 Background

Agriculture is always the backbone of developing nations. To make people in such a country economically well and strong we need to harvest good quality and quantity from agriculture. Every year a huge amount of crops gets damaged due to bad weather, viruses and different crops diseases.Farmers spend thousands of rupees on disease management, often without adequate technical support, resulting in poor disease control, pollution and harmful results. In addition, plant disease can devastate natural ecosystems, compounding environmental problems caused by habitat loss and poor land management.

1.2 Motivation

Digital object detection, image processing or machine vision has advanced rapidly in recent decades and is a very important aspect of artificial intelligence and the interaction between physical-machine-based theory and advanced technology These innovations are commonly used in industry and pharmacy, but rarely in agricultural or natural rainforest-related fields. Despite the importance of using digital image processing to identify plant diseases, as this has been studied for at least 30 years, the work done indicates that it is rather timid. There are too many practices which are common. Any plant form could be interpreted by the ideal shape. This is definitely impossible due to the current level of wealth.

1.3 Statement of the problem

Since many farmers in Nepal suffer from the problem of degrading food and cash crop production due to infection of different pathogens and lack of proper insecticides. Due to this they face heavy loss in production and in the market while distribution of crops. People lack proper knowledge about the plant disease and their cure. Farmers completely depend upon phytologists but there is a lack of technically skilled phytologists who can integrate technology as well as plant biology, however the impacts of crop breeding for resource-poor farmers have

been disappointing. Much greater emphasis is required to address reasons for the gap between potential and actual yields achieved by farmers.

1.4 Project objectives

The objective of this project is to reduce the number of plants dying due to improper identification of disease. After identification of the disease, it suggests the name of pesticide to be used. Identify the insects and pests responsible for the epidemic and be able to narrow the problem down to several possibilities which will require further study in the laboratory before they can make a final diagnosis. This system is focused on research aimed at improving food security by reducing crop losses, particularly for low-income farmers. This project aims at providing direct technical support to the farmers in real time while working in their fields without depending on the phytologists.

1.5 Significance of the study

The study of plant diseases is important as they cause loss to the plant as well as plant production. The various types of losses occur in the field, in storage or any time between sowing and consumption of produce. The diseases are responsible for direct monetary loss and material loss. This helps to minimize loss of crops as a result of unidentification diseases.

CHAPTER 2

LITERATURE REVIEW

Machine learning image processing technology is developing rapidly and is widely used in all aspects, including the agricultural field [1]. Applying machine learning and image processing technology to crop disease recognition has incomparable advantages over traditional manual diagnosis and recognition methods [2]. People only need to collect a small number of disease image samples. The process involves the following steps: firstly, the dataset is pre-processed; secondly, the feature extraction algorithm is used to extract the features of the disease area in the image; lastly, the obtained feature information is sent to the classifier for training and the model parameters are obtained [3]. The generated model can be used to detect the disease category. Training with a large number of datasets is time consuming due to the lack of datasets and the poor generalization ability of the model. Moreover, the development of agricultural modernization towards the direction of intelligence has highlighted the shortcomings of these traditional image detection methods [4].

The development of new technology has enabled not only discovering the characteristics of things artificially but also collecting a large amount of data, designing algorithms and programming, mining laws from data and building models by using advanced computer hardware facilities. Deep learning is a representative branch of artificial intelligence.[5] Although this concept was only introduced in 2012, various network models have been produced after its development [6]. At present, deep learning is widely used in various fields, especially in computer vision. It efficiently solves the tasks of image classification, object detection and semantic segmentation. Compared with the traditional pattern detection method, the disease detection method based on the deep convolutional network (CNN) abandons the sophisticated image pre-processing and feature extraction operations and uses the end-to-end structure to simplify the detection process.[7] CNN can be used to train the model prediction results, which not only can save time and workforce but also can make a real-time judgment, as long as a large number of crop disease image datasets can be obtained. This way greatly reduces the great losses caused by the disease.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 HARDWARE COMPONENTS

Component Name	Quantity	Description
Raspberry Pi	1	It is a tiny and affordable computer that can be used to record HD video and high-resolution photos.
Pi Camera	1	It is a portable light weight camera that supports Raspberry Pi.

Table 3.1 Requirement Analysis Table

3.1.1 Raspberry Pi

Figure 3.1.1 Raspberry Pi

Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The Raspberry Pi project originally leaned towards the promotion of teaching basic computer science in schools and in developing countries. The original model became more popular than anticipated, selling outside its target

market for uses such as robotics. It is widely used in many areas, such as for weather monitoring, because of its low cost, modularity, and open design. It is typically used by computer and electronic hobbyists, due to its adoption of HDMI and USB devices.

The Raspberry Pi 4 Model B is the latest version of the low-cost Raspberry Pi computer. The Pi isn't like your typical device; in its cheapest form it doesn't have a case, and is simply a credit-card sized electronic board -- of the type you might find inside a PC or laptop, but much smaller.

The Raspberry Pi 4 can do a surprising amount. Amateur tech enthusiasts use Pi boards as media centers, file servers, retro games consoles, routers, and network-level ad-blockers, for starters. However that is just a taste of what's possible. There are hundreds of projects out there, where people have used the Pi to build tablets, laptops, phones, robots, smart mirrors, to take pictures on the edge of space, to run experiments on the International Space Station -- and that's without mentioning the more wacky creations -- tea bag dunker anyone?

3.1.2 Pi Camera

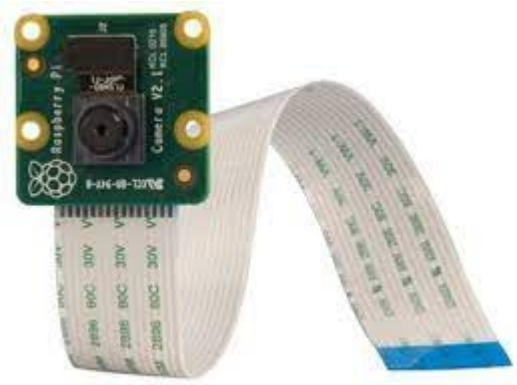

Figure 3.1.2 Pi Camera

The Pi Camera v2 is a high quality 8 megapixel Sony IMX219 image sensor custom designed add-on board for Raspberry Pi, featuring a fixed focus lens. In terms of still images, the camera is capable of 3280 x 2464 pixel static images, and also supports 1080p30, 720p60 and 640x480p90 video.

The Pi camera module is a portable lightweight camera that supports Raspberry Pi. It communicates with Pi using the MIPI camera serial interface protocol. It is normally used in image processing, machine learning or in surveillance projects.

3.2 SOFTWARE COMPONENTS

3.2.1 Python

Figure 3.2.1 Python

Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming. Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter,map and reduce functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

3.2.2 TensorFlow

Figure 3.2.2 Tensorflow

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 in 2015.

TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from

Google. In December 2017, developers from Google, Cisco, RedHat, CoreOS, and CaiCloud introduced Kube Flow at a conference. Kubeflow allows operation and deployment of TensorFlow on Kubernetes. In March 2018, Google announced TensorFlow.js version 1.0 for machine learning in JavaScript. In Jan 2019, Google announced TensorFlow 2.0. It became officially available in Sep 2019. In May 2019, Google announced TensorFlow Graphics for deep learning in computer graphics.

As TensorFlow's market share among research papers was declining to the advantage of PyTorch, the TensorFlow Team announced a release of a new major version of the library in September 2019. TensorFlow 2.0 introduced many changes, the most significant being TensorFlow eager, which changed the automatic differentiation scheme from the static computational graph, to the "Define-by-Run" scheme originally made popular by Chainer and later PyTorch.[Other major changes included removal of old libraries, cross-compatibility between trained models on different versions of TensorFlow, and significant improvements to the performance on GPU.

TensorFlow provides stable Python (for version 3.7 across all platforms) and C APIs; and without API backwards compatibility guarantee: C++, Go, Java, JavaScript and Swift (archived and development has ceased). Third-party packages are available for C#, Haskell, Julia, MATLAB, R, Scala, Rust, OCaml, and Crystal.

"New language support should be built on top of the C API. However, not all functionality is available in C yet." Some more functionality is provided by the Python API.

3.2.3 Kaggle Dataset

Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

Kaggle got its start in 2010 by offering machine learning competitions and now also offers a public data platform, a cloud-based workbench for data science, and Artificial Intelligence education. Its key personnel were Anthony Goldbloom and Jeremy Howard. Nicholas Gruen was the founding chair succeeded by Max Levchin. Equity was raised in 2011 valuing the company at $25 million. On 8 March 2017, Google announced that they were acquiring Kaggle.

Kaggle has run hundreds of machine learning competitions since the company was founded. Competitions have ranged from improving gesture recognition for Microsoft Kinect to making a football AI for Manchester City to improving the search for the Higgs boson at CERN. Several academic papers have been published on the basis of findings made in Kaggle competitions. A key to this is the effect of the live leaderboard, which encourages participants to continue innovating beyond existing best practice.

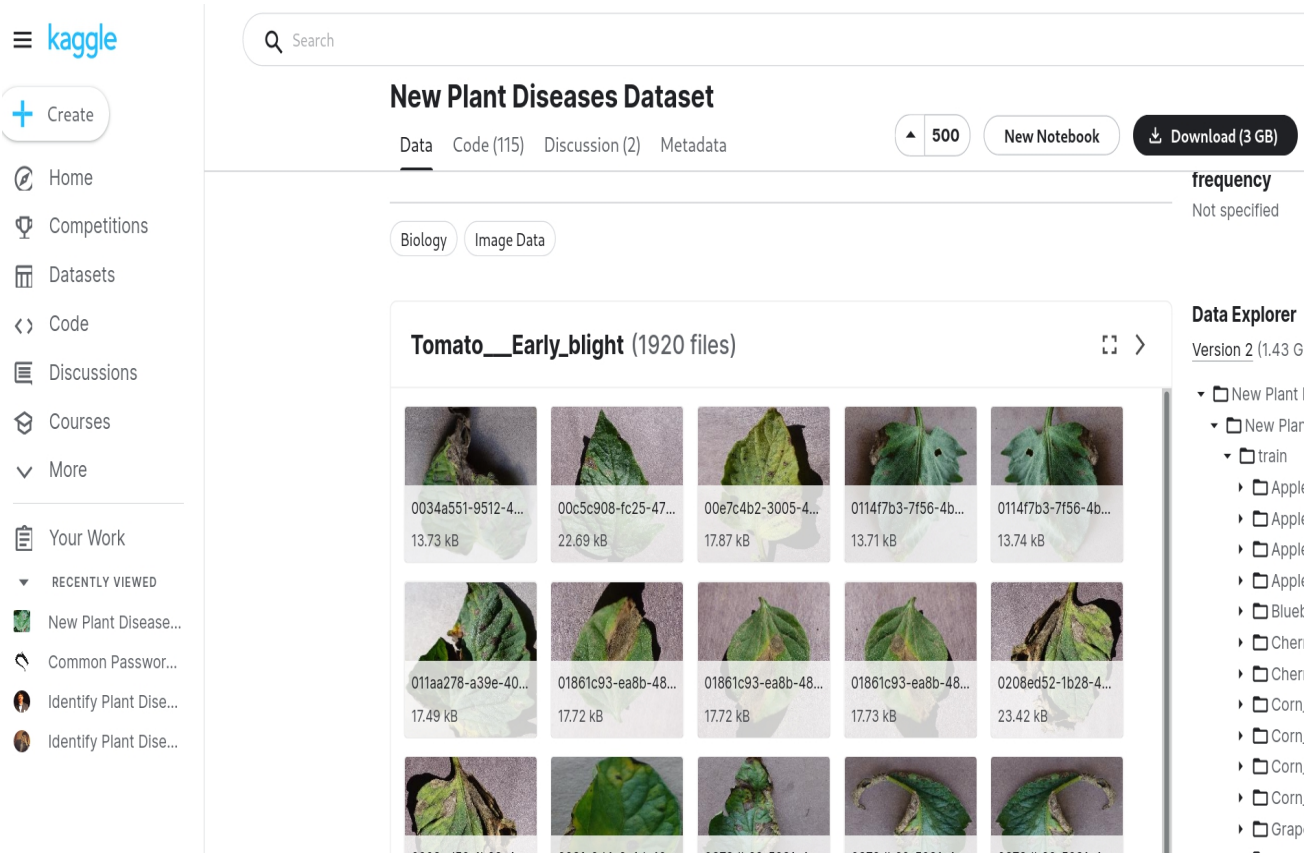

Figure 3.2.3 Kaggle Dataset

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE

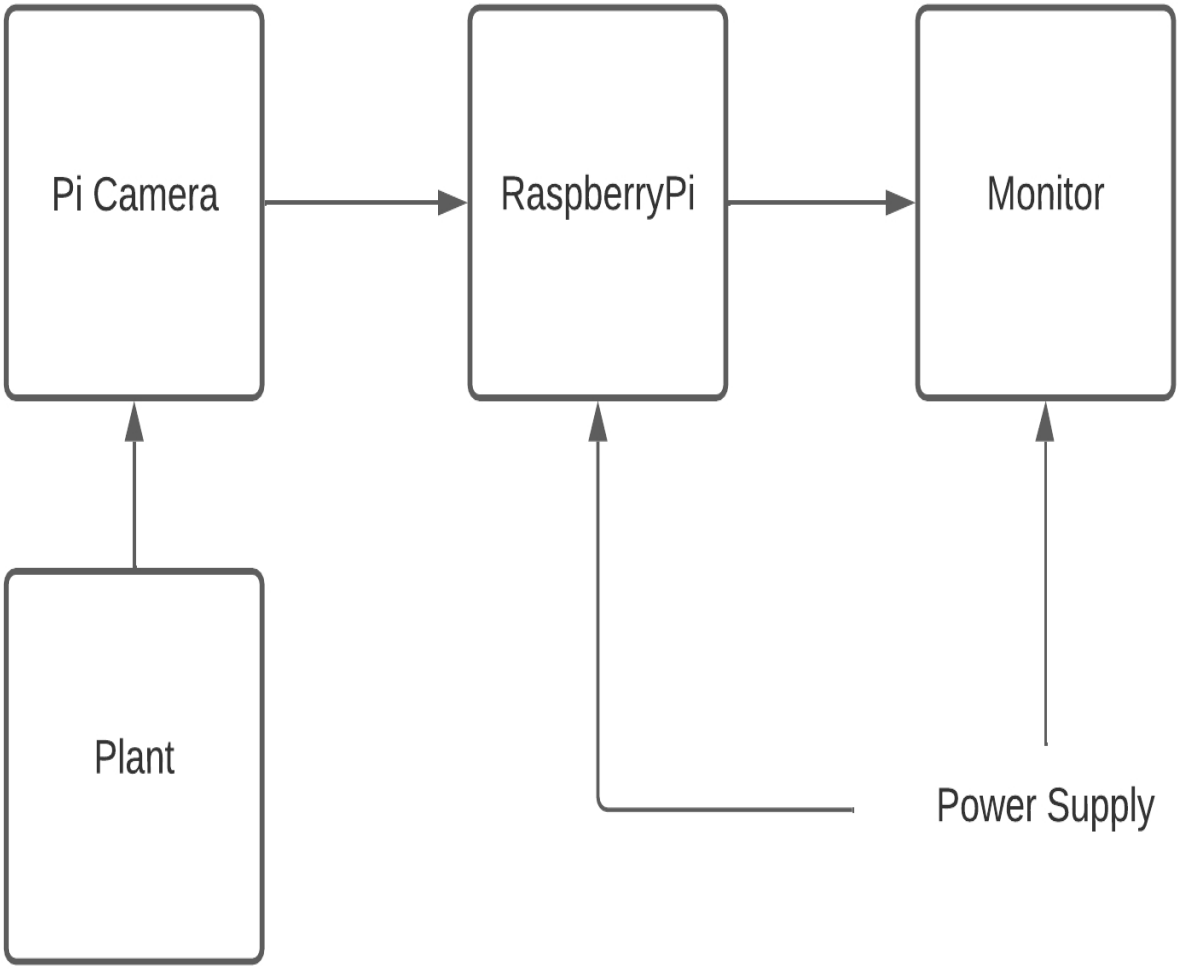

Figure 4.1 Block Diagram

Plant disease detection is a process through which an infection on the leaf of a plant can be detected. To check if the leaf is infected or not firstly an image is captured with the help of Pi Camera. The captured image is then sent on the internet which uploads on the local server. After that the input image is sent to the tensorflow image classifier. TensorFlow builds neural network models to classify images. Commonly, these will be Convolutional Neural Networks (CNN). TensorFlow is a powerful framework that lets you define, customize and tune many types of CNN architectures. After that tensorflow sends the result output to the local server and the classified result to the database. Database sends classified results to the display monitor of our desktop.

CHAPTER 5

METHODOLOGY

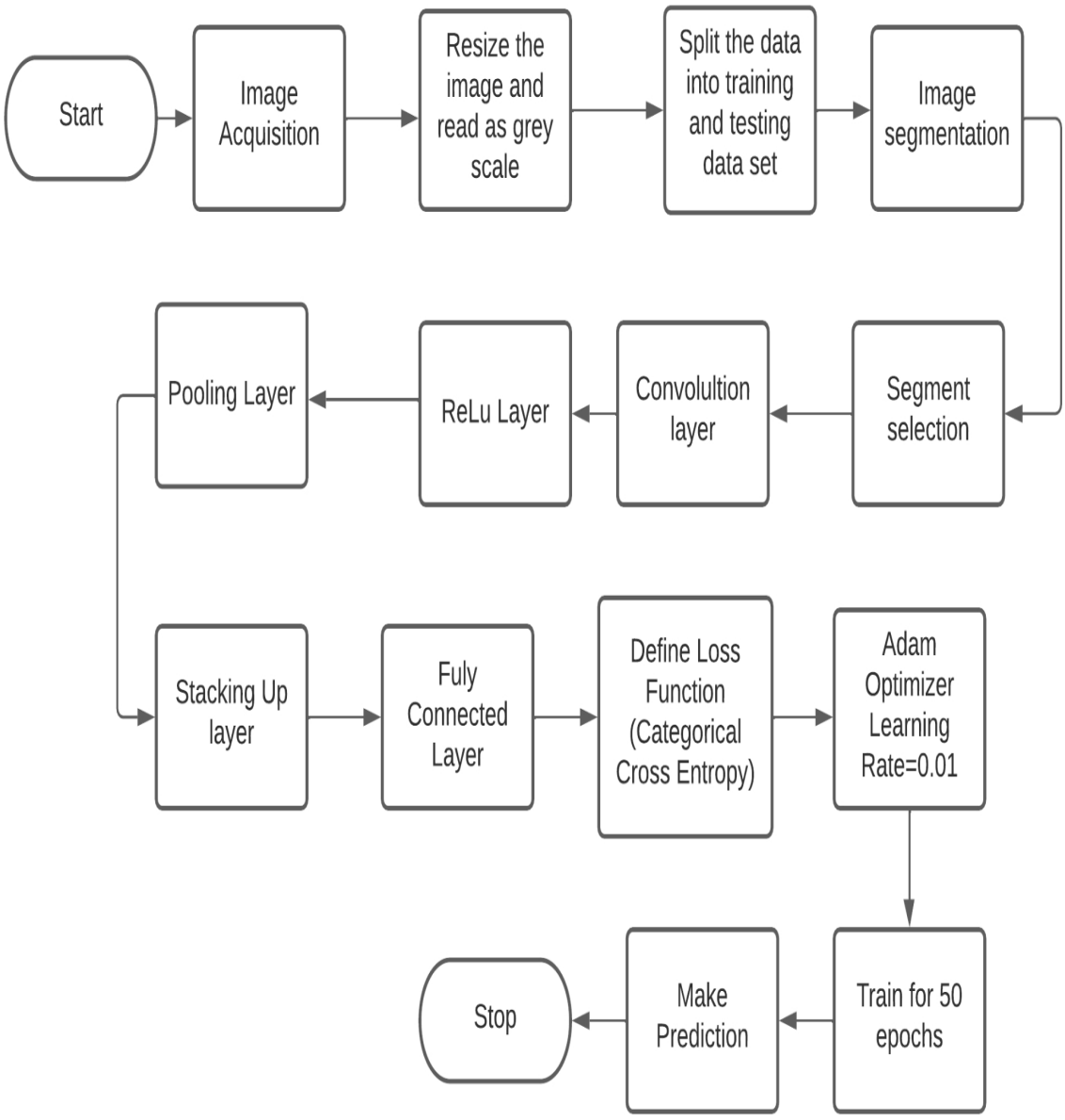

Figure 5.1.1 Detailed Block Diagram with every step.

i. Image Acquisition

We need various images of leaves to train the model. So we acquire high quality images from different sources and categorize them into different categories. For now we will be using the Plant Village dataset from Kaggle. It has up to 15 different directories and has up to 20K images. Hence we can directly train our model with this dataset and use it for predictions. Later we also can add different images and train models again to improve the model.

ii. Image Preprocessing

The aim of Image pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing, although geometric transformations of images (e.g. rotation, scaling, translation) are classified among pre-processing methods. The aim of image processing is to enhance the image data (features) by suppressing unwanted distortions and/or enhancement of some important image features in order that our models can benefit from this improved data to make computation on data. In this phase we usually improve image quality and we make the color correction to highlight features so that they could be simply caught by the classifier. We also reduce the size of the image to a certain limit so that we can convert the image to an array and feed it to a neural network. Image is pre-processed to enhance the image data that suppress undesired distortions, and enhances some image features important for further processing and analysis tasks. It includes color space conversion, image enhancement.

iii. Image Segmentation

Image segmentation is the process of finding boundaries of objects in images and enhancing those boundaries in such a way that they can become more highlighted and could be captured immediately by the classifier. Image segmentation is the best technique to improve image and characteristics of image. It also works well on low resolution and low quality images.

iv. Segment Selection

This is the process of choosing the correct segment from the image and using that selected segment for prediction.

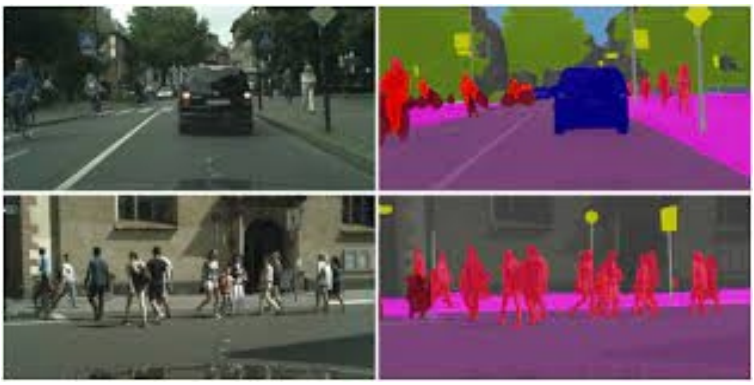

Figure 5.1.2 Image Segmentation

In the above figure, we can see that people on the road are marked In red color, we can select those people and detect that these are people. And this can be used for further classification.

v. Configuring Convolutional Neural Network

Convolutional neural networks (CNN) are great for photo tagging and recognizing patterns in image data. Here we will build a Convolutional Neural Network in TensorFlow Framework. Below is the figure that gives us an idea how CNN works.

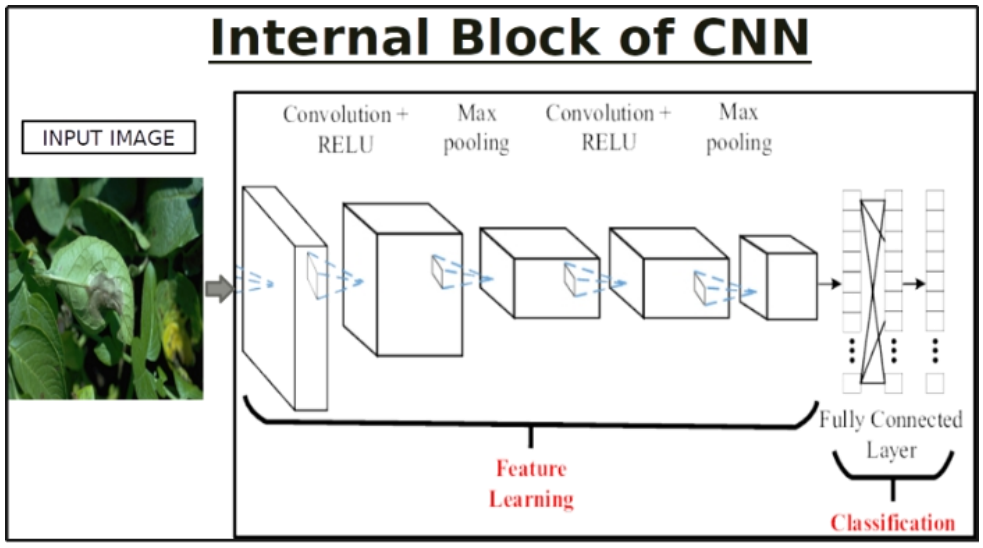

Figure 5.1.3 Internal Block of CNN

When an array of images is passed to the CNN layer it performs an operation called 'Convolution'. It is a linear operation that involves the multiplication of a set of weights with the input. This multiplication is called a filter or a kernel. This filter slides over the image and extracts important data from the image and passes it to a consecutive Convolutional layer.

CNNs are used for image classification and recognition because of their high accuracy. The CNN follows a hierarchical model which works on building a network, like a funnel, and finally gives out a fully-connected layer where all the neurons are connected to each other and the output is processed.

We will construct a new Convent step-by-step in this article to explain it further. In this example, we will be implementing the (Modified National Institute of Standards and Technology) MNIST data set for image classification. This data set contains ten digits from 0 to 9. It has 55,00 images — the test set has 10,000 images and the validation set has 5,000 images. We will be building a Convent made of two hidden layers or convolutional layers. Now let us look at one of the images and the dimensions of the images. Here is an image, which is the number 7.

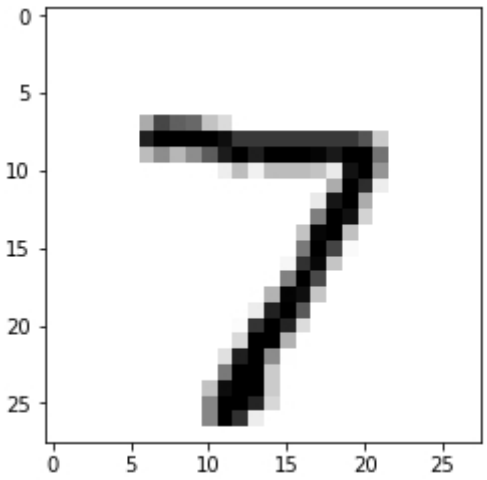

Figure 5.1.4 Training image example

This image's dimensions are 28×28 which are represented in the form of a matrix (28, 28). And the depth or the number of channels this image has is 1, since it is a grayscale image. If it was a color (for example, RGB) image, the number of channels would have been three.

Convolution Layer:

CNN compares the image piece by piece. The pieces that the CNN looks for are called features. We choose a feature and put it on the input image, if it matches then the input image is classified correctly. The feature is moved to every possible position of the image. The feature and the image are lined up and each image pixel is multiplied by the corresponding feature pixel. Thus, obtained pixel values are added up and then divided by the total number of pixels present in the feature. After this, a map is created and the obtained value is placed at that point. Now the filter is moved to every position of the image and a similar operation as described above is followed. Finally, a matrix can be obtained after performing the above operations for the feature. This obtained result is for a single feature. So similar operations are performed for every other feature. This is the first layer of the CNN, known as - Convolutional layer.

Formula that we use to create a feature map in convolutional layer can be defined as:

$$\frac{\Sigma(Xi * Yi)}{n}$$

Where X_i is the pixel value of feature from healthy leaf and Y_i is the pixel value of segment from diseased leaf and 'n' is the total number of pixels present in the feature.

ReLu Layer:

After the convolutional layer, the ReLU layer comes into play. In this layer, every negative value from the filtered image is removed and replaced with zeros (0). This is done in order to prevent the values from summing up to zero. ReLU is nothing but an activation function. It will only activate a node if the input is above a certain threshold value and below the threshold value, the output is zero. Thus, it has a linear relationship with the dependent variable. So, the ReLU function is used for every filter. ReLu activation function can be defined as follows:

$$R(z)=\begin{cases} z, \text{ for } z > 0 \\ 0, \text{ for } z <= 0 \end{cases}$$

Max Pooling Layer:

After the ReLU layer, there is a Pooling layer where the values are shrieked into smaller sizes. A window of a particular size is taken and moved across the entire image. From the window, only the maximum value is taken.

For a feature map having dimensions $n_h * n_w * n_c$, the dimensions of output obtained after a pooling layer is given by the formula:

$$\frac{(nh - f + 1)}{s} * \frac{(nw - f + 1)}{s} * n_c$$

Where ,

n_h= height of feature map

n_w=width of the feature map

n_c= number of channels in the feature map

f=size of filter

s=stride length

Stacking Up Layers and Fully connected layer:

After completion of the Pooling layer, all the obtained values are stacked up. Again, to shrink the values to a higher extent, we perform all the above-described operations again and again. Finally, we use a fully connected layer. Here we take our filtered and shrink images and put the values in a single list. This is the final layer where actual classification happens.

Now when an input image is given and after it goes through Convolutional layer, ReLU layer, pooling layer and fully connected layer, a similar value in the form of list can be obtained. This

list of the input image is compared with the list obtained from the trained image. Now the sum of the values of the list is taken, the value that is closer to the trained image is classified as the true value and the image is identified.

Categorical Cross Entropy Loss Function:

In order to reduce error and improve the accuracy of the model, loss function is used after the image is classified. The loss function in a neural network quantifies the difference between the expected outcome and the outcome produced by the machine learning model. Categorical cross entropy is a loss function that is used in multi-class classification tasks. These are tasks where an example can only belong to one out of many possible categories, and the model must decide which one.

The categorical cross entropy loss function calculates the loss of an example by computing the following sum:

$$\text{Loss} = -\sum_{i=1}^{output\ size} y_i \log y_k$$

Where ,

y_k is the k'th scalar value in the model output, y_i is the corresponding target value, and output size is the number of scalar values in the model output. In this context, y_i is the probability that an event occurs and the sum of all 'k' is 1, meaning that exactly one event may occur. The minus sign ensures that the loss gets smaller when the distributions get closer to each other.

We will use the Softmax activation function in the output layer. Softmax is used as the last activation function of a neural network to normalize the output of a network. Here is the equation for the SoftMax activation function.

$$\text{Softmax}(z_i) = \frac{exp(zi)}{\sum_j \quad exp(zj)}$$

Here, the Z represents the values from the neurons of the output layer. The exponential acts as the nonlinear function. Later these values are divided by the sum of exponential values in order to normalize and then convert them into probabilities.

So, in brief, at first, we take a dataset and then define functions to predict the dependent variables. And then our images are converted into grayscale images. The data is then split into training and testing data. Now we use TensorFlow to build the model and calculate the loss function. Then to reduce loss, an optimizer is used with a defined learning rate. Then we train the neural network for a certain number of epochs and finally make the prediction.

5.2 Integration of Different Module

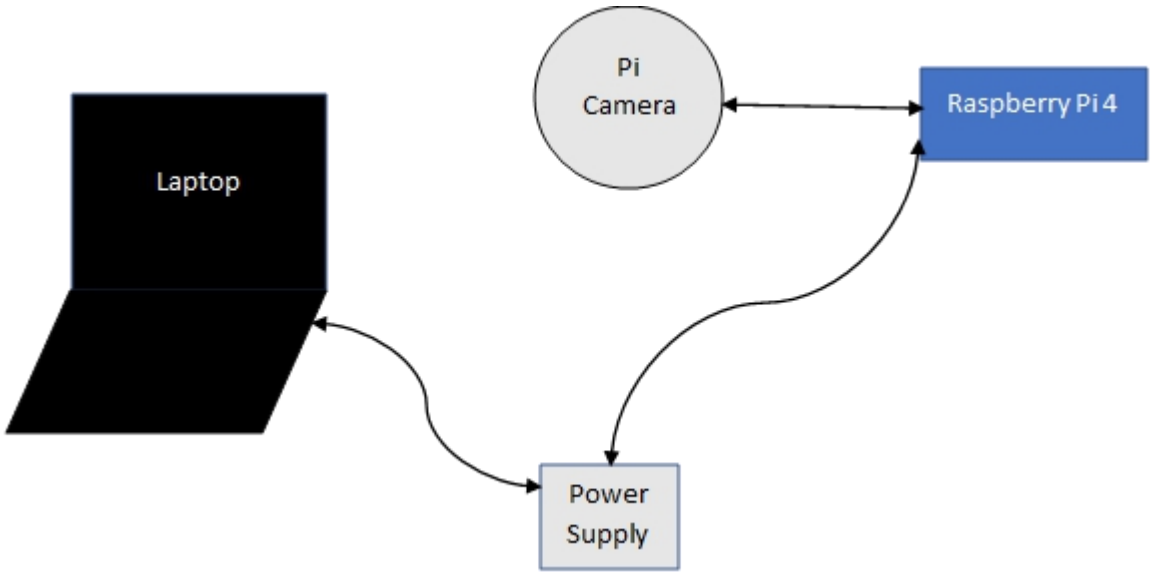

Figure 5.2 Integration of different module

Laptop is connected to the power supply, and the raspberry pi is connected to the laptop and power supply. Pi camera is connected to raspberry pi. All the components are dependent on each other.

CHAPTER 6

RESULT AND ANALYSIS

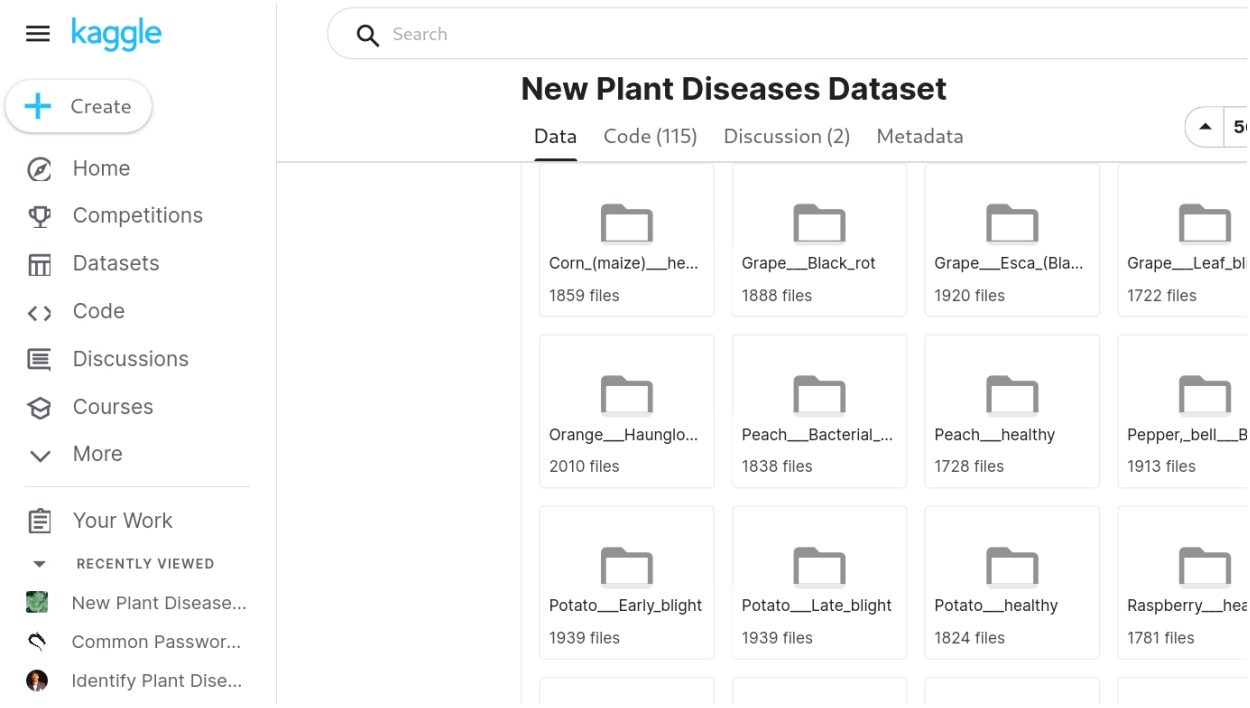

Figure 6.1.1 Dataset from kaggle

The datasets were downloaded from Kaggle. There are many varieties of datasets available in Kaggle. The dataset that we downloaded are of two diseases i.e. Early Blight and Late blight. Next one is Healthy Leaf. Training, testing and validation of those Datas are done so that the system gives 100% accurate results.Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

There are many varieties of datasets available in Kaggle. The dataset that we downloaded are of two diseases i.e. Early Blight and Late blight. Next one is Healthy Leaf. Training, testing and validation of those Datas are done so that the system may give very accurate results.

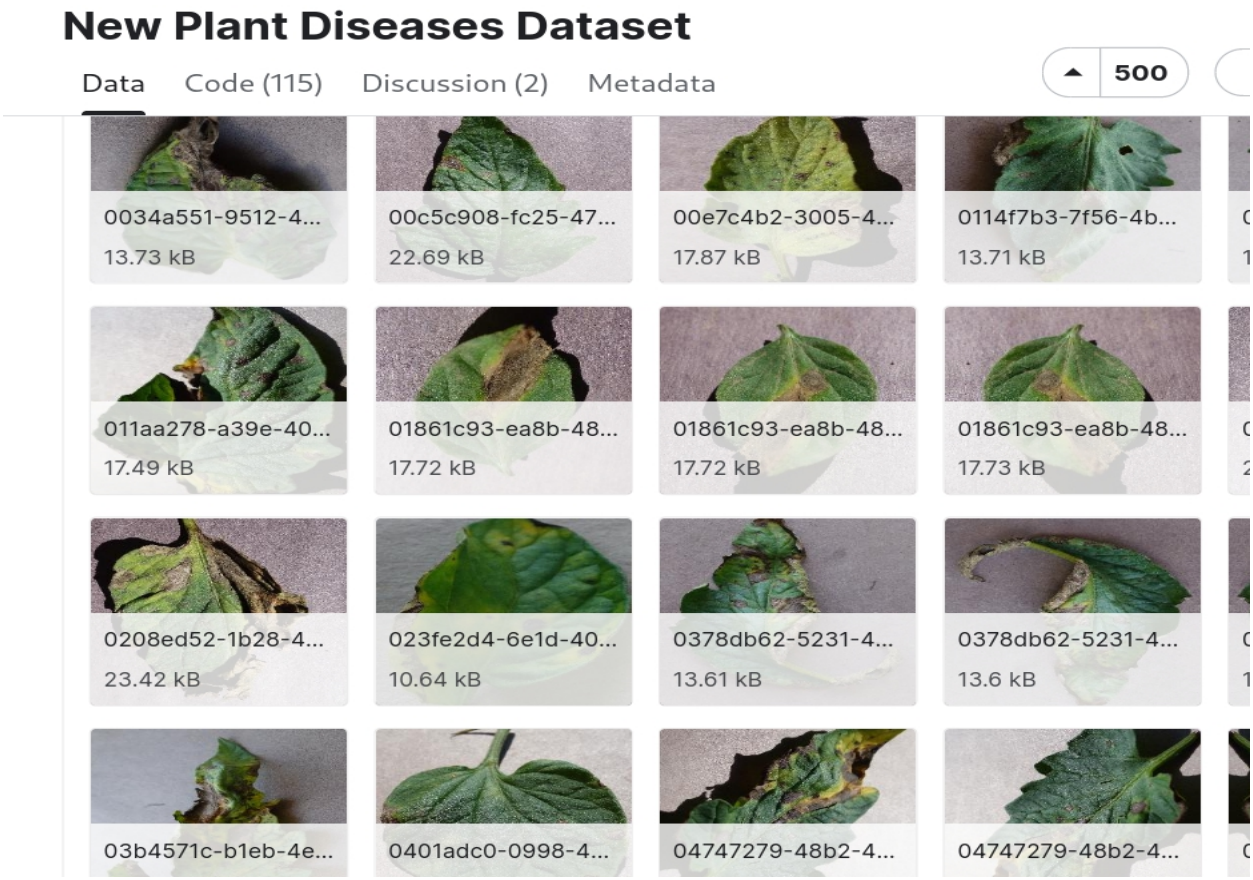

Figure 6.1.2 Image of leaf affected by early blight

Images from dataset, these images are of Late Blight. These images show every variety of late blight that can occur on the leaf of a tomato plant. The input images are compared with these images by training them. When the Data are trained and then tested so that the output is accurate. These datasets are very much useful in the field of data science and data analysis. With the help of these images data scientists and data analysts can come to a conclusion and if a plant is unhealthy, they suggest pesticides according to the disease. The designed system can detect the condition of a random input leaf. The system is trained and later on testing is done by sending input of a leaf. Firstly, it tells the actual state of a leaf and after that it predicts the condition of a leaf and later on coincides and tells the coincident percentage. The system works perfectly for the given Datas of a database.

Figure 6.1.3 .JPG Image of early blight

This dataset consists of about 87K rgb images of healthy and diseased crop leaves which is categorized into 3 different classes. The total dataset is divided into 80/20 ratio of training and validation set preserving the directory structure. A new directory containing 1000 test images is created later for prediction purpose.This dataset is recreated using offline augmentation from the original dataset.

Total number of dataset from kaggle=3000

Number of training dataset=80% of 3000= 2400

Number of testing dataset=10% of 3000= 300

Number of validation dataset=310% of 3000=300

Number of epocs	**Percentage accuracy(%)**
30	54
50	93
80	98

Table 6.1.1 Dataset Table

6.2 Model Accuracy and Model Loss

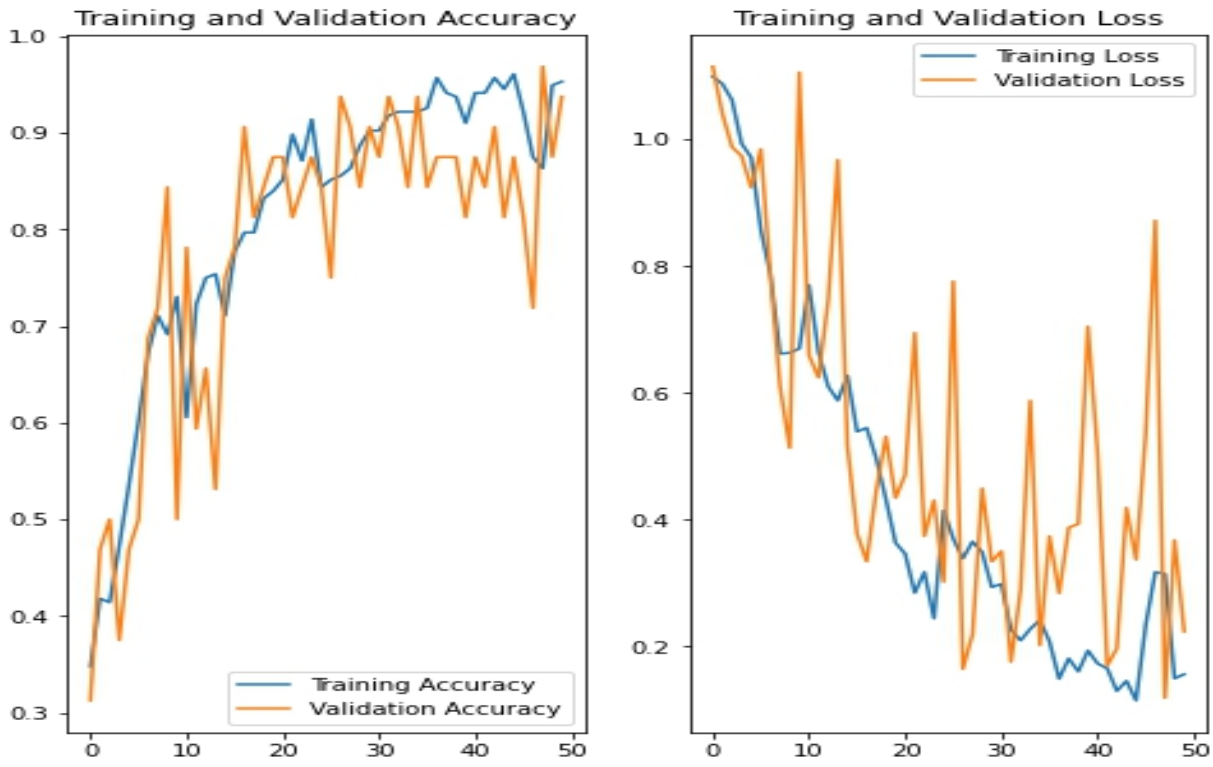

Figure 6.2.1 Model Accuracy and loss

The above graph shows the training accuracy and validation accuracy on the first graph while on the second graph it shows validation loss and training loss. In both cases the epochs run for 50 times.In case of an accuracy graph, accuracy is nearly equal to 1. Same case is there in case of loss graph, maximum loss at 50th epochs is nearly equal to 0. Here also not much fluctuation in values could be seen.

6.1.2 Model Summary

Layer(type)	Output Shape	Param #
Sequential (Sequential)	(32, 256, 256, 3)	0
Conv2d (Conv2D)	(32, 254, 254, 32)	896
Max_pooling2d(MaxPooling2d)	(32, 127, 127, 32)	0
Conv2d_1 (Conv2D)	(32, 125, 125, 64)	18496
Max_pooling2d_1(MaxPooling2D)	(32, 62, 62, 64)	0
Conv2d_2 (Conv2D)	(32, 60, 60, 64)	36928
Max_pooling2d_2 (MaxPooling2D)	(32, 30, 30, 64)	0
Conv2d_3 (Conv2D)	(32, 28, 28, 64)	36928
Max_pooling2d_3 (MaxPooling2D)	(32, 14, 14, 64)	0
Conv2d_4 (Conv2D)	(32, 12, 12, 64)	36928
Max_pooling2d_4 (MaxPooling2D)	(32, 6, 6, 64)	0
Conv2d_5 (Conv2D)	(32, 4, 4, 64)	36928
Max_pooling2d_5 (MaxPooling2D)	(32, 2, 2, 64)	0

flatten (Flatten)	(32, 256)	0
dense (Dense)	(32, 64)	16448
dense _1(Dense)	(32, 3)	195
Total params : 183, 747 Trainable params : 183, 747 Non-trainable params: 0		

Table 6.2.2 Model Summary

CHAPTER 7

TIME SCHEDULE

S.N.	Task	Semester			
		Seventh		Eighth	
		May-July	July-Sept	Oct-Nov	December
1	Project Analysis	[redacted]			
2	Requirement Analysis	[redacted]	[redacted]		
3	Designing		[redacted]	[redacted]	
4	Coding		[redacted]	[redacted]	[redacted]
5	Testing and Verification		[redacted]	[redacted]	[redacted]
6	Documentation	[redacted]	[redacted]	[redacted]	[redacted]

Table 7.1 Time Schedule

TOTAL COST

Hardware Components	**Cost**
Raspberry Pi 4	Rs 4700
Pi Camera	Rs 1550
Sd Card (32 GB)	Rs 900
Total	**Rs 7150**

Table 7.2 Total Cost

CHAPTER 8

CONCLUSION

In this way, by collecting data of various diseases of tomato plants and process them to train on CNN architecture to create a machine learning model, Late blight, Early Blight and Healthy leaf are the detected diseases. For detection purposes YOLO object detection algorithm built in darknet framework is used to train a model and predict diseases in tomato plants. Python programming language, OPENCV library is used to manipulate raw input images. Model is trained on the Nvidia 1080 gpu. This system is implemented in Raspberry pi, desktop based Graphical User Interface (GUI) is developed to capture image or video. In addition, It is found that using technique based data annotation and augmentation results in better performance. As a limitation; this system is only capable of detecting three classes of diseases and healthy plants. In order to detect other classes of diseases data has to be trained on current models. Algorithms will use transfer learning methods to classify other classes of diseases. The main challenge while developing object detection models on machine learning was to collect a large number of train images with different shapes, sizes, with different backgrounds, light intensity, orientation and aspect ratio. As per the recommendation; the further study can be done to detect all types of plant diseases, not only detection but also suggesting remedies for diseases. Finally, this system can be integrated with an IOT server to implement the system in rural and remote areas.

LIMITATIONS AND FUTURE WORK

LIMITATIONS

1. Since the design only detects two diseases till date, if a tomato leaf is suffering from another disease then this will not be detected.

FUTURE WORK

1. Other diseases can be combined.
2. Mobile applications can be developed.

REFERENCES

[1] Saradhambal.G, Dhivya R, Latha.S, R.Rajesh PLANT DISEASE DETECTION AND ITS SOLUTION USING IMAGE CLASSIFICATION, International Journal of Pure and Applied Mathematics,Volume 119 , No. 14 , 2018 , 879-884.

[2] P. Harini,L. V. Chandran , Disease Detection in Plant Leaves using K-Means Clustering and Neural Network, International Journal of Trend in Scientific Research and Development (IJTSRD) Volume 4 Issue 1, December 2019 .

[3] Yin Min Oo ,Nay Chi Htun,, Plant Leaf Disease Detection and Classification using Image Processing Spot , IEEE International Conference on Emerging Trends in Sci- enceInternational Journal of Research and Engineering ISSN: 2348-7860 (O), 2348-7852 (P) , Vol. 5 No. 9 , September-October 2018.

[4] Trimi Neha Tete, Sushma Kamlu ,Crop disease detection Plant Disease Detection Using Different Algorithms, Proceedings of the Second International Conference on Research in Intelligent and Computing in Engineering pp. 103–106, DOI: 10.15439/2017 ,R24 ACSIS, Vol. 10 ISSN 2300-5963

[5] Mr.V Suresh, D Gopinath, M Hemavarshini, K Jayanthan, Mohana Krishnan, Plant Disease Detection using Image Processing, International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 9 Issue 03, March-2020

[6] Vishnu S, A. Ranjith Ram, Plant Disease Detection Using Leaf Pattern: A Review , International Journal of Innovative Science, Engineering & Technology, ISSN 2348 – 7968 ,Vol. 2 Issue 6, June 2015 .

[7] Anne-Katrin Mahlen , Plant Disease Detection by Imaging Sensors - Parallels and Specific Demands for Precision Agriculture and Plant Phenotypes, Institute for Crop Science and

Resource Conservation (INRES) Phytomedicine, University of Bonn Meckenheimer Al 100 53115 Bonn, Germany