

Design of a Machine Learning Model for Predicting Plant Disease.

By

MHERUB AHSAN (161-15-726)

MD. OMUR KHYOM (161-15-723)

MD. MEHEDI HASAN NAYEEM (161-15-691)

This Report Presented in Partial Fulfillment of the necessities for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Ms. Naznin Sultana

Assistant Professor

Department of CSE

Daffodil International University



DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

December 2019

Design of a Machine Learning Model for Predicting Plant Disease.

By

MHERUB AHSAN (161-15-726)

MD. OMUR KHYOM (161-15-723)

MD. MEHEDI HASAN NAYEEM (161-15-691)

This Report Presented in Partial Fulfillment of the necessities for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Ms. Naznin Sultana

Assistant Professor

Department of CSE

Daffodil International University



DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

December 2019

APPROVAL

This Project titled “**Design of a machine learning model for predicting plant disease**”, submitted by **Mherub Ahsan(161-15-726)**, **Md.OmurKhyom(161-15-723)** and **Md. Mehedi Hasan Naeem(161-15-691)** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the necessities for the degree of B.Sc. in Computer Science and Engineering and approved on its vogue and contents. The presentation has been held on 26th November 2019.

BOARD OF EXAMINERS

Dr. Syed Akhter Hossain

Chairman

Professor and Head

Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. S M Aminul Haque

Internal Examiner

Associate Professor & Associate Head

Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Saif Mahmud Parvez

Internal Examiner

Lecturer

Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Mohammad Shorif Uddin

External Examiner

Professor

Department of Computer Science and Engineering
Jahangirnagar University

DECLARATION

I hereby declare that, I did this project under the supervision of **Naznin Sultana**, Assistant Professor, Department of CSE Daffodil International University. I'm also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:



Naznin Sultana

Assistant Professor

Department of CSE

Daffodil International University

Submitted by:

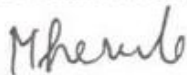


Md. Mehedi Hasan Nayeem

ID: -161-15-691

Department of CSE

Daffodil International University



Mherub Ahsan

ID: -161-15-726

Department of CSE

Daffodil International University



Md. Omur Khyom

ID: -161-15-723

Department of CSE

Daffodil International University

ACKNOWLEDGEMENT

First, we express our heartiest thanks and gratefulness to almighty God for His divine blessing which makes us possible to complete the final year project successfully. We are really grateful and wish our profound obligation to **Ms. Naznin Sultana, Assistant Professor**, Department of Computer Science & Engineering. Her Deep Knowledge & keen interest in the field of “Machine Learning” influenced us to build this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have created it potential to complete this project.

We would like to wish our heartiest gratitude to **Ms. Naznin Sultana, Assistant Professor**, and **Professor Dr. Syed Akhter Hossain** Head, Department of CSE, for his kind help to complete our project and also to all other faculty member and the staff of CSE department of Daffodil International University.

We would like to thank our entire course mate in Daffodil International University, who took part in this discussion while finishing the coursework. Finally, we must have acknowledged with due respect the constant support and patience of our parents.

ABSTRACT

Identification of diseases from the leaf images of a tomato leaf is one of the interesting research areas within the agriculture field, that machine learning ideas of computer field can be applied. This paper presents a model for the detection and classification of tomato leaf diseases based on the images of infected tomato leaf. We consider 10 tomato diseases named Bacterial_spot, Early_blight, late_blight, Leaf_Mold, Septoria_leaf_spot, Tomato_mosaic_virus, healthy, Spider_mites Two-spotted_spider_mite, Target_Spot, Tomato_Yellow_Leaf_Curl_Virus. It can also observe Healthy leafs. During this analysis, we used deep learning based model (CNN) for classification. First, we pre-process our image dataset very carefully as a result of preprocessing is that the most vital a part of our analysis. Then we train our model and validate according to the dataset. We test various techniques for this research but cnn works pretty well for our dataset, it gives an accuracy level of 87%.

TABLE OF CONTENTS

CONTENTS	PAGE
Board of examiners	I
Declaration	II
Acknowledgement	III
Abstract	IV
List of figures	VII
List of tables	VIII
 CHAPTER 1: INTRODUCTION	 1-3
1.1 Introduction	1
1.2 Motivation	1
1.3 Objective	2
1.4 Expected Outcome	2
1.5 Report Layout	3
 CHAPTER 2: BACKGROUND	 4-6
2.1 Introduction	4
2.2 Literature Review	4
2.3 Research Summary	5
2.4 Tools and Software	6
2.5 Scope of the Problem	6
2.6 Challenges	6
 CHAPTER 3: RESEARCH METHODOLOGY	 7-14
3.1 Introduction	7
3.2 Data Collection Procedure	7
3.3 Data Pre-processing	9
3.3.1 sklearn.preprocessing	9
3.3.2 ImageDataGenerator	9
3.3.3 keras.preprocessing	9
3.3.4 keras.preprocessing.image import img_to_array	10

3.3.5 sklearn.preprocessing .MultiLabelBinarizer	10
3.3.6 Image Labeling	10
3.3.7 .h5 file generation	10
3.3.8 Jason	10
3.3.9 pickle file	10
3.4 Model Configuration	11
3.5 model architecture	13
3.6 Parameter	14
3.7 Hyper parameter tuning	14

CHAPTER 4: EXPERIMENTAL RESULTS AND DISCUSSION

16-20

4.1 Introduction	16
4.2 Result/output	16

CHAPTER 5: CONCLUSION AND FUTURE SCOPE

21-22

5.1 Summary of study	21
5.2 Conclusions	21
5.3 Recommendations for Further Study	22

REFERENCES

23

APPENDIX

25

PLAGIARISM REPORT

30

LIST OF FIGURES

FIGURES	PAGE
Fig 3.4.1: Model configuration	12
Fig3.4.2: model loaded the data	12
Fig3.5.1: Basic cnn model	13
Fig3.5.2: input of cnn model	13
Fig3.6: Parameter statistics	14
Fig4.2.1: model accuracy	16
Fig4.2.2: Total Loss	17
Fig4.2.3: Confusion Matrix	18
Fig4.2.4: Tested Image	20

LIST OF TABELS

FIGURES	PAGE
Table 3.2.1: Detailed statistics of image collection	7
Table 3.2.2: Divided the whole dataset in training and testing set	8
Table 3.7 shows the tuning of hyperparameter	14
Table 4.2 shows classification report	19

CHAPTER 1

INTRODUCTION

1.1 Introduction

Bangladesh is eminent for Agriculture that means most of the people are engaged towards agriculture industry. The agriculture industry acts as a significant role in the economic sectors. Most people in this densely populated country Bangladesh are living on agriculture, contributing 19.6 percent to the national GDP and providing employment for 63 percent of the population. Plant diseases detection is one of the top research areas. Every disease has its own symptoms and color combination. We can easily separate these diseases from any plant leaf or fruit. From the beginning of digital computing, there have many techniques invented to identify disease or disease area. Some technique really does a good job to detect disease area of plant leaf or fruit. But when we come to find out exact disease and try to give them solution then it becomes harder than previous. Machine Learning actually change our thinking and analyzing life. Nowadays, almost every hard task can be solved using Image Processing and Machine Learning. Object Detection and Object identification become more reliable and accurate. So, we decided to solve the plant diagnosis problem using Machine Learning to get more accurate result and give farmer exact solution for disease prevention.

1.2 Motivation

Bangladesh is a developing country. Agriculture is one of the major sources of income from the early period of Bangladesh. But still, we are depending on the manual supervision system which follows a very slow process of managing the problems by naked eye while for the safe harvesting a quick and easy approach should be made to ensure the constant production of crops. Agricultural sector needs to digitalization. Lack of farmer's productive education on agriculture filed every year huge losses in this agricultural sector. For those problem solutions, we try to build a Machine Learning model that easily identified whatever plant is affected or not. For the betterment for the country we will do this project the agriculture sector may be rise very high and may produce a huge number of crops and vegetable fresh without any kind of diseases. Our research can also be identified that the plant is healthy or not. For purpose of those research is make more fast and accurate solution for farmers. In this research, we used ten diseases class.

The diseases are

- Bacterial_spot

- Early_blight
- late_blight
- Leaf_Mold
- Septoria_leaf_spot
- Spider_mitesTwo-spotted_spider_mite
- Target_Spot
 - Tomato_Yellow_Leaf_Curl_Virus
- Tomato_mosaic_virus •• Healthy

1.3 Objective

The objective of this research is to find out a technique which can help the farmer to detect the defected tomato leaf from the field. After identifying the diseases, farmers can also take necessary steps to solve these diseases using a chatbot. The proposed system is user-friendly, fast to execute and can be implemented easily in any system. The implementation for rural people is very hard because they have very much little knowledge. But smartphone is available nowadays in rural areas. That's why we will try to build an android application in future. By using this application, farmers can easily identify diseases. We will also make this in web platform for betterment and we will contact with agriculture ministry in future to use our app to detect the diseases of tomato which can be applied to other plants as well. It will save time and our energy because we can detect our plant diseases through the website to not go in any office or investing any money. I think it will help our country from huge economic loss.

1.4 Expected Outcome

Most of the people related with agriculture in our country living in rural areas. They have very little knowledge. For helping them, we selected those research areas. The outcome of our research to identify tomato diseases in real time by using tomato leaf and also give them step by step solution for the disease. We can create a python file and took an image and it will predict the diseases using the model and also return the image with blue shade of marking in the effected part. Our research also can identify the healthy potato. We will create an Android application for identified tomato diseases.

1.5 Report Layout

In chapter 1, we tried to cover basic concepts of “A machine learning model for predicting plant disease” and also discuss motivation, objective, and expected outcome of our research. In

chapter 2, focus on related works, the brief overview on summary, the scope of the problem and the challenges.

In chapter 3, discuss about research methodology

In chapter 4, describe the details of experimental results.

The final chapter 5, I have concluded about my evaluation result and also about some other features that can be included in future works for the better of my research work.

In the last portion of the report we add,

References

Appendix

Plagiarism report

List of figures

CHAPTER 2

BACKGROUND STUDY

2.1 Introduction

In this chapter, we will discuss about similar research's that was done by the others researchers so far. Deep dive into the machine learning and image processing research area. In our research, we used Convolution Neural Network (CNN). Using python programming language. Also, use several machine learning Framework. TensorFlow to our model. We collect dataset from the Internet. We have 18000 images but we used almost 6000 images because we don't have a high configure machine. For image preprocessing purpose we use several open source tools for labeling, noise reduce, size reduce etc. We used supervised learning algorithm where disease names are used to label the class. Earlier, many researchers have worked with image processing to ensure the use of technology in agriculture. But it is not being used in our country adequately. Several image processing techniques are being developed to ensure the compactness of methodologies available.

2.2 Literature Review

Many Image processing techniques were developed through many years of research for object detection and use of image classifiers as well.

In paper [1] A multiclass support vector machine model developed for classify potato disease detection. this model applied to 300 different images. the training set containing 180 images (60%) and the testing set containing 120 images (40%). For performance evaluation of the classification model, performance parameters such as accuracy, sensitivity, recall, and F1 score were calculated. Using this model 95% of the data were successfully classified. In paper [2] A design neural network for classify grape leaf. Then grape leaf disease segmentation is done using K-means clustering. The diseased portion from segmented images is identified. In paper [3] A deep CNN model was developed for classifying 13 leaf diseases. Model was able to detect leaf presence and distinguish between healthy leaves and 13 different diseases, which can be visually diagnosed. The complete procedure was described, respectively, from collecting the images used for training and validation to image preprocessing and augmentation and finally the procedure of training the deep CNN and finetuning. The experimental results on the developed model achieved precision between

91% and 98%, for separate class tests, on average 96.3%. In paper [4] authors described image processing technique for Apple fruit disease detection. Apple Bloch, Apple Rot and Apple Scab are used for fruit disease identification and the features used are global Color Histogram,

Local Binary Pattern, Color Coherence Vector and Complete Local Binary Pattern. K-means Clustering in $L^*a^*b^*$ space is used for defect segmentation. Multi-class SVM Classifier was used to train and test images. In paper [9] authors used K-Means clustering and image is converted from RGB color space to $L^*a^*b^*$ color space. Color features are Color coherence vector (CCV) and Global color histogram (GCH). Complete local binary pattern (CLBP), Gabor Features, Local binary pattern (LBP) and Local ternary pattern (LTP) texture features are used. Then authors performed feature level fusion by fusing more than two texture and color features also applied Random forest classifier on segmented image. Fruit disease is shown after classification by tagging every pixel by using this method. In paper [5] four different algorithms were tested to identify the diseases with various combination of Golden Delicious apple images using color information. The Mahalanobis distance between each pixel and both medians is computed and compared with the global model which is used next to enhance and filter images to detect various defects such as bruises, russet, scab, fungi or wounds of apples. A hybrid method is used for color segmentation which is based on seeded region growing algorithm. In paper [6] authors proposed a CNN model for predicting the plant diseases and it will predict also the stage of the diseases they get the accuracy of 98%. They use MATLAB for this project. In paper [7] authors proposed a web-based tool that has acquired an accuracy of 82% to identify pomegranate diseases. The system contained trained dataset of pomegranate fruits. In first step, feature extracting with color, morphology and CCV is done then clustered by K-means. Next, SVM classifier is used to classify the disease. In this section, we tend to discuss about some common connected work done by other research worker. There is no connected work has been done in Bangladesh yet. But abroad, we found very similar done by others researchers. We found many problems in those works most of the research has performance problems.

2.3 Research Summary

In this research, we collect image from GitHub. In this dataset almost 18000 images. Mainly ten tomato disease images contain in this dataset. Bacterial_spot, Early_blight, late_blight, Leaf_Mold, Septoria_leaf_spot, Spider_mites, Two-spotted_spider_mite, Target_Spot, Tomato_mosaic_virus, Tomato_Yellow_Leaf_Curl_Virus, Healthy. After downloading this

dataset, we pre-processed images. The all data we collected are already labeled. We use tensorflow for training and keras library for input the dataset. For training and testing we use cnn algorithm. In our algorithm the cnn use 4 layers

2.4 Tools and Software

We used Python syntax for this project. As a framework we used Keras, which is a high level neural network API written in Python. But Keras can't work by itself, it needs a backend for low-level operations. Thus, I installed a dedicated software library - Google's TensorFlow. As a development environment we used Jupyter notebook. And predefined library for preprocessing. We used Matplotlib library for visualization.

2.5 Scope of the Problem

The projected work will facilitate the farmers to predict whether the tomato plants are healthy or not. The whole process was difficult. Because our farmers are non-technical person. They don't have enough knowledge to do the total method easily. The whole process containing several steps like capturing images, entering this image on input and then run the whole model to test the images. But the research has an algorithm and the full step by step how to process data. However, this work be enforced several platforms like mobile apps, net apps etc. If we using the approach and try to build a web platform or mobile app that may be very useful for our non-technical farmers.

2.6 Challenges

The big challenge for this research is quality of dataset. We try several sources for locating a good dataset. When an extended search in internet then we found a dataset that is perfect for me. Others challenges are computation power. For training purpose, we need a high configuration pc. In training, many complex equations solve by a computer, but we don't have high configuration CPU. Training step very challenging for us. We test our dataset with several algorithms for improving accuracy. So, we need high computation power. After training validates it using test dataset. These tasks were difficult and challenging for me.

CHAPTER 03

RESEARCH METHODOLOGY

3.1 Introduction

In this session various steps are considered, initial of all, Data pre-processed, noise-reduction, and transfer it to TensorFlow appropriate format to train. After training step then take a look at validate images and send image for testing. And we show the test image that user randomly.

3.2 Data Collection Procedure

We collected the dataset from plants, villages. We collected over 18000 affected images.

These files were in standard image format ('jpg').

Table 3.2.1: Detailed statistics of image collection

Label	No Of images	Image Format	Image Size
Bacterial_spot	2127	jpg	256*256
Early_blight	1000	jpg	256*256
healthy	1591	jpg	256*256
Late_blight	1909	Jpg	256*256
Leaf_Mold	952	Jpg	256*256
Septoria_leaf_spot	1771	Jpg	256*256

Spider_mites Twospotted_spider_mite	1676	Jpg	256*256
Target_Spot	1404	Jpg	256*256
Tomato_mosaic_virus	373	Jpg	256*256
Tomato_Yellow_Leaf_Curl_Virus	573	Jpg	256*256

The only way to know how well a model the only way to know how well a model can perform to a new and unknown dataset is to test the model with the new dataset. To get better accuracy from the model we split the dataset into test and train set and the ratio of test-train was chosen to be 85 and 15%. However during this case, our model accuracy is 64%. Then we split it to 80%, 20 % then the model accuracy is pretty much well. Table 3.2.2: Divided the whole dataset into training and testing set

Label	No of images	training set	testing set
Bacterial_spot	2127	1700	427
Early_blight	1000	800	200
healthy	1591	1272	318
Late_blight	1909	1527	382

Leaf_Mold	952	760	191
Septoria_leaf_spot	1771	1416	355
Spider_mites Twospotted_spider_mite	1676	1340	336
Target_Spot	1404	1123	281
Tomato_mosaic_virus	373	298	75
Tomato_Yellow_Leaf_Curl_Virus	5337	4270	1067

3.3 Data Pre-processing

Preprocessing refers to all the transformations on the raw data before it is fed to the machine learning model.

3.3.1 sklearn.preprocessing

We use `sklearn.preprocessing`. The **sklearn.preprocessing** package provides many common utility functions and transformer classes to change raw feature vectors into an illustration that's additional appropriate for the downstream estimators. In general, learning algorithms benefit from standardization of the dataset. The following subjects are going to be handled:

- Missing values
- Polynomial features
- Categorical features

- Numerical features
- Custom transformations
- Feature scaling
- Normalization

3.3.2 ImageDataGenerator

For image augmentation we use this library function `keras.preprocessing.image import ImageDataGenerator`. Image data augmentation is used to expand the training dataset so as to enhance the performance and ability of the model to generalize. Image data augmentation is supported in the Keras deep learning library via the `ImageDataGenerator` class. The Keras deep learning neural network library provides the potential to suit models using image data augmentation via the `ImageDataGenerator` class.

3.3.3 keras.preprocessing

Preprocessed data may be a vital issue for dataset. Keras Preprocessing is that the data preprocessing and data augmentation module of the Keras deep learning library. It provides utilities for operating with image data, text data, and sequenced data.

3.3.4 keras.preprocessing.image import img_to_array

Keras provides the `img_to_array()` function for changing a loaded image in PIL format into a NumPy array for using with the deep learning models. The API also provides the `array_to_img()` function that can be used for converting a NumPy array of pixel data into a PIL image.

3.3.5 sklearn.preprocessing .MultiLabelBinarizer

`sklearn.preprocessing .MultiLabelBinarizer`. class `sklearn.preprocessing .MultiLabelBinarizer` (classes=None, sparse_output=False)[source] Transform between iterable of iterables and a multilabel format. Although a list of sets or tuples is a very intuitive format for multilabel data, it is unwieldy to method.

3.3.6 Image Labeling

The dataset we have used in our experiment was already labeled. So, the only thing we have done is to import those data in the model. There are many techniques to import data in our model and we use the keras library function to import.

3.3.7 .h5 file generation

An H5 file is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data. Two commonly used versions of HDF exclude HDF4 and HDF5 (developed to improve upon limitations of the HDF4 library). We generate the .h5 file from our train part and that we send it to test part to check for model prediction.

3.3.8 .Jason

Returns an illustration of the model as a JSON string, however that the representation does not include the weights, only the architecture. And it matches the label of the dataset from the train part to test part. We can call the Jason file in the other python file and matched with the new test image.

3.3.9 pickle file

We saved our model as .pkl file. Python pickle module is used for serializing and deserializing a Python object structure. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The thought is that the character stream contains all the information necessary to reconstruct the object in another python script. We use this pickle file to match our total model with the new image.

3.4 Model configuration

One of the most popular techniques used in improving the accuracy of image classification is Convolutional Neural Networks (CNNs for short). Convolutional Neural Network: A special type Neural Networks that works in the same way of a regular neural network except that it has a convolution layer at the beginning. Instead of feeding the entire image as an array of

numbers, the image is broken up into a number of tiles, the machine then tries to predict what each tile is. Finally, the computer tries to predict what's in the picture based on the prediction of all the tiles. This allows the computer to parallelize the operations and detect the object regardless of where it is located in the image. There are a few basic things about an Image Classification problem that you must know before you deep dive in building the convolutional neural network. We know that the machine's perception of an image is completely different from what we see. In fact, it is only numbers that machines see in an image. Each pixel in the image is given a value between 0 and 255. Thus, for the machine to classify any image, it requires some preprocessing for finding patterns or features that distinguish an image from another. Here's a look at the key stages that help machines to identify patterns in an image:

Convolution: Convolution is performed on an image to identify certain features in an image.

Convolution helps in blurring, sharpening, edge detection, noise reduction and more on an image that can help the machine to learn specific characteristics of an image.

Pooling: A convoluted image can be too large and therefore needs to be reduced. Pooling is mainly done to reduce the image without losing features or patterns.

Flattening: Flattening transforms a two-dimensional matrix of features into a vector of features that can be fed into a neural network or classifier.

Full-Connection: Full connections simply refer to the process of feeding the flattened image into a neural network.

Model configuration is a very important part of my research. In this section, we define 10 class of tomato leaf, batch size, learning rate, and other parameters. Below figure are shown the config file. we construct a CNN with 2 convolution layers followed by 2 FC layer and then 1 classifier. Here is where we construct our CNN network, we import the all dependencies we need and we define the batch size.

Fig 3.4.1 shows the model configuration

```

model = Sequential()
inputShape = (height, width, depth)
chanDim = -1
if K.image_data_format() == "channels_first":
    inputShape = (depth, height, width)
    chanDim = 1
model.add(Conv2D(32, (3, 3), padding="same", input_shape=inputShape))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(1024))
model.add(Activation("relu"))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(n_classes))
model.add(Activation("softmax"))

```

Fig3.4.1:Modelconfiguration

The total dataset is imported and the model load it completely fig 3.4.2 shows how the dataset was imported from directory

```

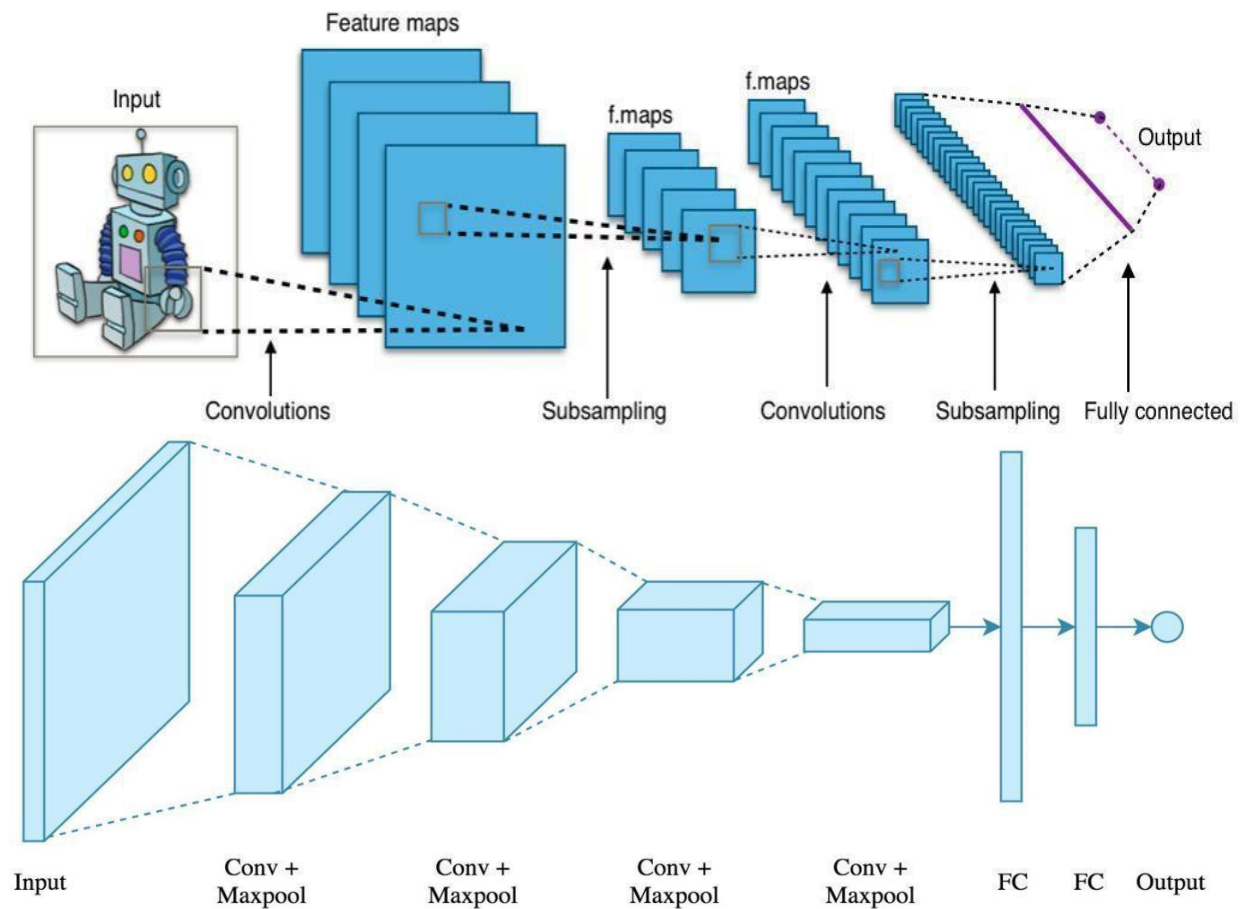
print(f"Error : {e}")

[INFO] Loading images ...
[INFO] Processing Tomato__Bacterial_spot ...
[INFO] Processing Tomato__Early_blight ...
[INFO] Processing Tomato__healthy ...
[INFO] Processing Tomato__Late_blight ...
[INFO] Processing Tomato__Leaf_Mold ...
[INFO] Processing Tomato__Septoria_leaf_spot ...
[INFO] Processing Tomato__Spider_mites Two-spotted_spider_mite ...
[INFO] Processing Tomato__Target_Spot ...
[INFO] Processing Tomato__Tomato_mosaic_virus ...
[INFO] Processing Tomato__Tomato_Yellow_Leaf_Curl_Virus ...
[INFO] Image loading completed

```

Get Size of Processed Image

Fig3.4.2:modelloadedthedata



3.5 Model Architecture:

TensorFlow is a powerful tool that makes it easy to create, train, and deploy Machine Learning models. We use CNN Model to our dataset for image processing. Below figure show the basic of CNN. In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. Multilayer perceptron's usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. Fig 3.5.1 shows the basic CNN model.

Fig 3.5.1: Basic CNN model

As per Fig 3.5.2 the model takes input and shows output

Fig 3.5.2: Input of CNN model

```

activation_7 (Activation)      (None, 10)                   0
=====
Total params: 58,097,546
Trainable params: 58,094,666
Non-trainable params: 2,880

```

3.6 Parameter:

From our dataset we loss parameter almost 2% data from total parameter. Total parameter about=58097546totalparameterabout=58094666Nontrainableparameter=2880

Fig3.6Showstheparameter

Fig3.6:Parameterstatistics

3.7 Hyper parameter Tuning:

A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters are learned. Basically hyperparameter is tuned for changingtheperformanceofthemodel.Thehistofhyperparameteraregiven

Table3.7showsthetuningofhyperparameter

No	Nameofhyperparameter	Tuning
1	EPOCHS	We used 199 times. Much more the epochs much better the accuracy.
2	Batchsize	Weusedthe32,therearealso 64,128etc
3	Depth	Wetookdepthsize3.
4	INIT_LR	Weuseditfor1
5	Width,heigh	Both(256,256)
6	test_size	0.2
7	random_state	42
8	rotation_range	25
9	width_shift_range	0.1
10	height_shift_range	0.1

11	shear_range	0.2
12	zoom_range	0.2
13	horizontal_flip	true
14	fill_mode	nearest
15	model	Sequential
16	inputShape	height,width,depth
17	Dropout	0.25
18	Spitedintox_test,x_train	80%,20%

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction:

Above all experiment and training was done on my test data. This is the final step of our work with real data. We formed a very good model accuracy which was good 87%. We train our model only for 150 epochs for our hardware limitation.

4.2 Result/Output:

Model Graph: Now it's the most important part of our model. As we discussed above, all things are for preparing our data to feed CNN model. Every step is related with some hidden layer. Each step processes its data and results should be the input data for next steps layers. To improve our result, we will alter the details of the learning process. We were selected 4000 training steps to get better accuracy. But we think our model is needed around 200k training step. But our hardware limitation it would be possible for us.

In figure 4.2.1 show the model accuracy for our model. It's measures the ratio of correct predictions to the total number of cases validation.

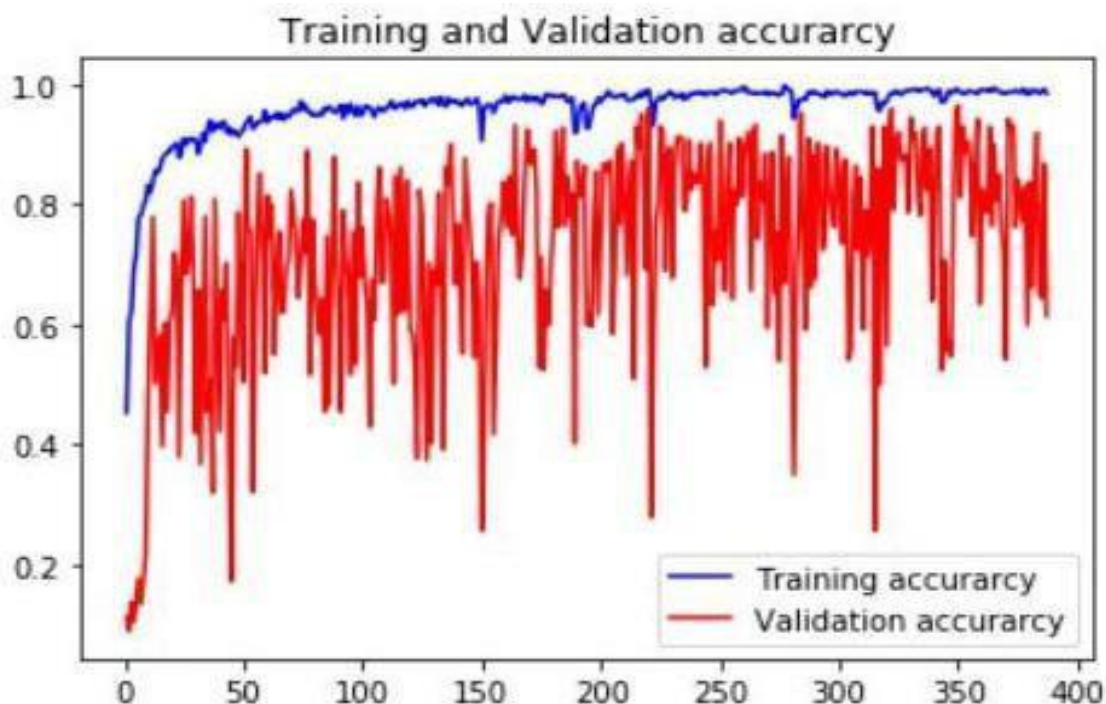
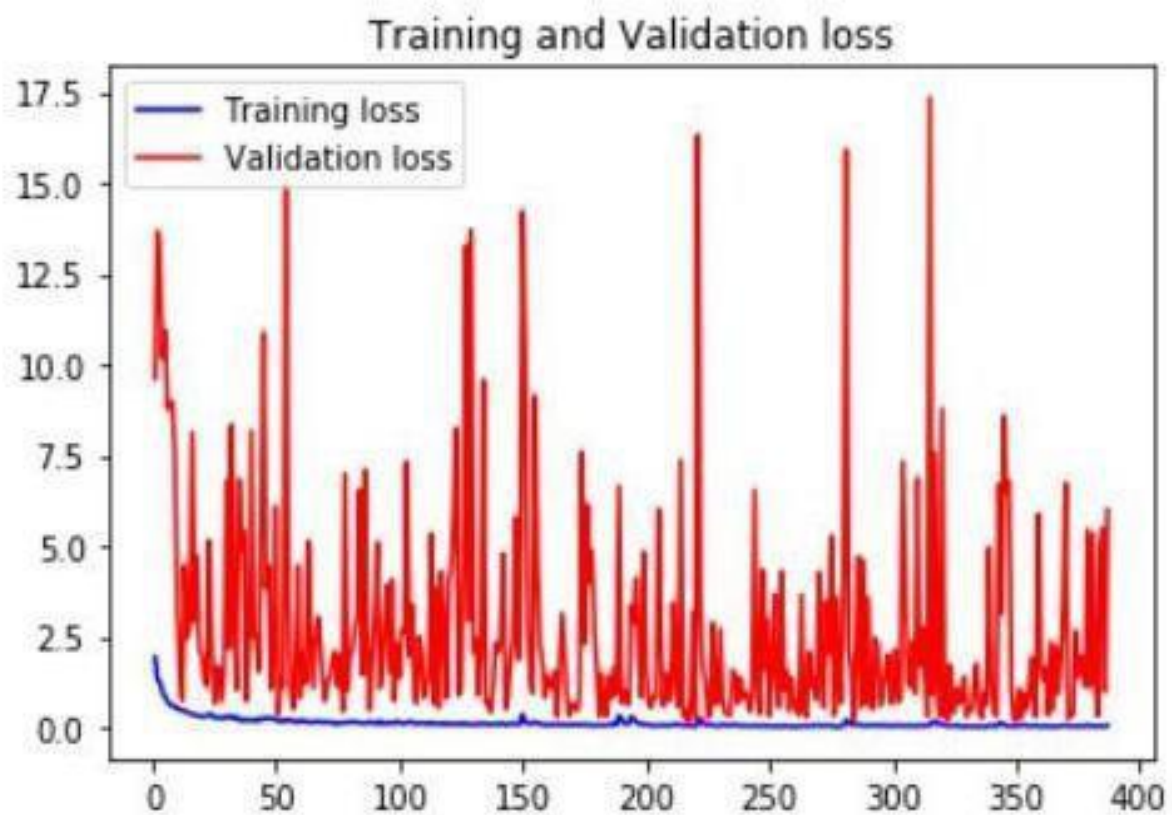


Fig4.2.1:model accuracy



In figure 4.2.2 shows the total loss in training session. The loss is calculated on training and validation and its interpretation is how well the model is doing for these two sets. Unlike accuracy, loss is not a percentage. It is a summation of the errors made for each example in training or validation sets.

Fig4.2.2: Total Loss

Confusion Matrix for Tomato: A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

Fig4.2.3 shows confusion matrix

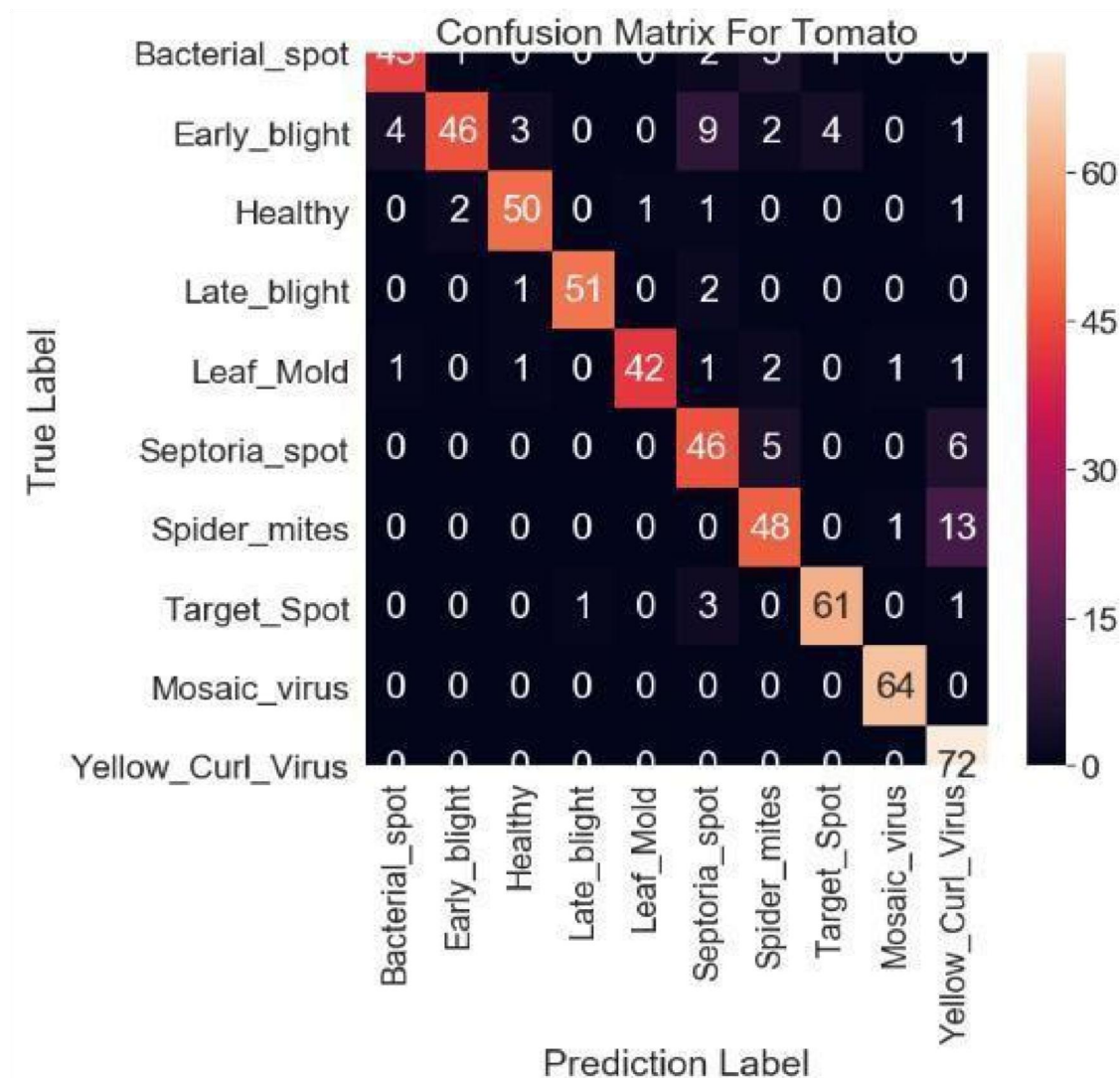


Fig4.2.3:ConfusionMatrix

Classification report: A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False. More specifically, True Positives, False Positives, True negatives and False Negatives are used to predict the metrics of a classification report as shown below:

Table 4.2 shows classification report

Classes	Precision	Recall	Fi-score	Support
Class0	0.90	0.83	0.86	52
Class1	0.94	0.67	0.78	69
Class2	0.91	0.91	0.91	55

Class3	0.98	0.94	0.96	54
Class4	0.98	0.86	0.91	49
Class5	0.72	0.81	0.76	57
Class6	0.77	0.77	0.77	62
Class7	0.92	0.92	0.92	66
Class8	0.97	1.00	0.98	64
Class9	0.76	1.00	0.86	72
Accuracy			0.87	600
Macroavg	0.88	0.87	0.87	600
Weightedavg	0.88	0.87	0.87	600

Output: We split our dataset with 80% as train & validation and 20% for testing the diseases.

In test part, model will predict tested images with label which is getting from .Json file. Here

we our model will show the test file and we call the `x_test` file for showing the image. Fig

4.2.4 shows the output

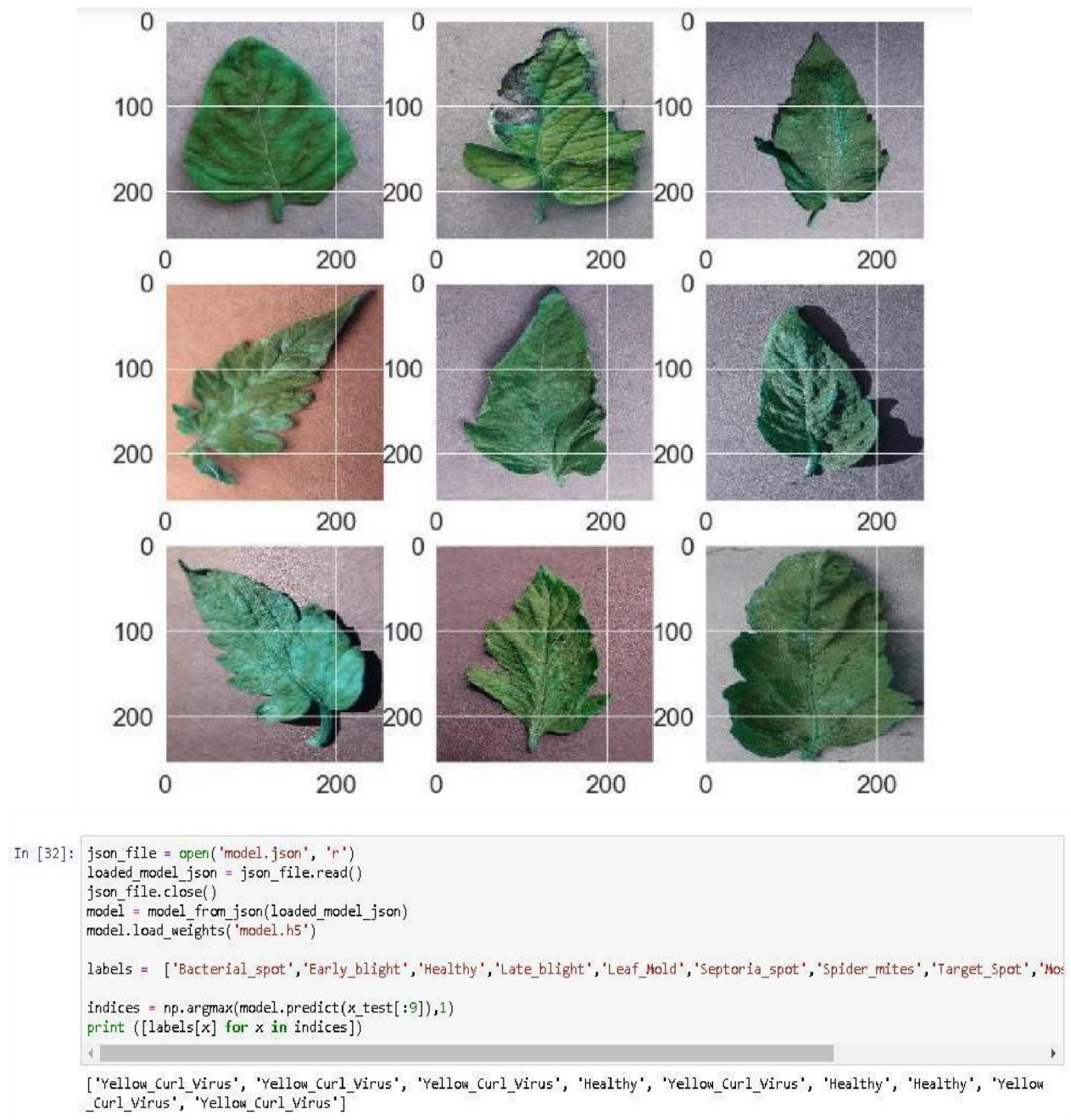


Fig 4.2.4: Tested Image

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Summary of the Study

we are trying to represent a Machine learning approach for the detection of tomato disease from tomato leaf using convolutional neural network model. The short summary of our whole work is given below.

I First stage

- Data collection and preprocessing
- Feature reduction, divided data into train and test set

II Second stage

- Training the dataset using model.
- Classify the 10 diseases

III. Third stage

- Compile the model and fit the model

IV. Last and final stage

- Find the accuracy & predicted test images.

Finetuning the model operator to get better accuracy and solution.

5.2 Conclusion

In this work, we have experimented with the detection of tomato disease based on the images of tomato leaves. We considered ten classes in our experiment including healthy tomato as one of the classes. The classifier which is implemented in our experiment was Deep Learning Convolutional Neural Network model. We considered 4 hidden layers. Each hidden layer extracts different information from the image data and improves biases and weights to make classification and detection more accurate. Hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. During my research, we face many kinds of problems. We had to learn ML and deep learning techniques and Convolution Neural Networks. Our university curriculum doesn't have any related course. So early we face many problems to do this kind of project. Then we start an online course and start learning. Our supervisor helps me for finding a good dataset. This research results would help our farmers and in agriculture a lot.

5.3 Recommendations for Further Study

In this experiment, we have shown nine types of tomato disease and healthy leaf detection using 18000 images. In near future, we will try to build such model which will be able to classify almost every tomato disease. As well as we have a plan to develop a mobile app and web app as soon as possible as an extension of our model. And we are looking forward to communicate with agriculture ministry so this system can be used for greater good.

Reference:

- [1] Monzurul Islam, Anh Dinh, Khan Wahid, Pankaj Bhowmik "Detection of Potato Diseases Using Image Segmentation and Multiclass Support Vector Machine.", 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), Volume 2017 (2017), 5 PAGES
- [2] Sanjeev Sannakki, Vijay S Rajpurohit, VBNargund, Pallavi Kulkarni.: Diagnosis And Classification Of Grape Leaf Diseases Using Neural Network. IEEE-31661
- [3] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic. Deep Neural Networks Based Recognition of Plant Disease by Leaf Image Classification. Volume 2016 (2016), Article ID 3289801, 11 pages <http://dx.doi.org/10.1155/2016/3289801>.
- [4] Anand Singh Jalal, Shiv Ram Dubey "Detection and Classification of Apple Fruit Diseases Using Complete Local Binary Patterns" IEEE Third International Conference on Computer and Communication Technology, pp.978-07695-4872, 2012.
- [5] Antoni Buades, Bartomeu Coll, Jean-Michel Morel "A non-local algorithm for image denoising", 2005, IEEE, 14 pages.
- [6] LEAF DISEASE DETECTION USING IMAGE PROCESSING Thete Vaishali V.1, Thakare Pradnya R.2, Kadlag
- [7] Gaurav B. 3, Prof. P.A. Chaudhari 4, pages 6 http://ijariie.com/AdminUploadPdf/LEAF_DISEASE_DETECTION_USING_IMAGE_PROCESSING_ijariie4143.pdf
- [8] Yu Sun and Jianxin Wang. Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning. Volume 2017, Article ID 2917536, 8 pages <https://doi.org/10.1155/2017/2917536> plant village, available at <<<https://plantvillage.psu.edu/>>>, last accessed on 02-11-2018 at 10:00 PM.
- [9] Deep learning is available at <<https://en.wikipedia.org/wiki/Deep_learning/>>, last accessed on 01-11-2019 at 6:00 pm.

- [10] Learn about tensorflow, available at<<<https://www.tensorflow.org/>>>, last accessed on23-11-2019at10:00AM.
- [11] Learn about tensorboard and tensorflow graph data visualization, available at <<https://www.tensorflow.org/guide/graph_viz/>>, last accessed on 25-04-2019 at 10:30AM.
- [12]Building a ConvolutionalNeuralNetwork using Tensorflow Library, available at <<<https://www.tensorflow.org/tutorials/keras/classification>>>,lastaccessedon10-03-2019 at11:00PM
- [13] Object Recognition with Google’s ConvolutionalNeuralNetworks, available at <<<https://medium.com/@williamkoehrsen/object-recognition-with-googlesconvolutionalneuralnetworks2fe65657ff90>>>
- [14] LearnaboutWikipedia,availableat<<<http://www.wikipedia.org/>>>
- [15] Knowabout<<<https://github.com/>>>

APPENDIX

Appendix A: Research Reflection

During my research, we face many kinds of problems. We had to learn ML and deep learning techniques and Convolution Neural Networks. Our university curriculum doesn't have any related course. So early we face many problems to do this kind of project. Then we start an online course and start learning.

Appendix B: Related issue

Imported library function

Import necessary packages

```
In [1]: import numpy as np
import pickle
import cv2
from os import listdir
from sklearn.preprocessing import LabelBinarizer
from tensorflow.keras.models import Sequential
from keras.layers.normalization import BatchNormalization
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers.core import Activation
from keras.layers.core import Flatten
from keras.layers.core import Dropout
from keras.layers.core import Dense
from keras import backend as K
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
from keras.preprocessing import image
from keras.preprocessing.image import img_to_array
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
```

Using TensorFlow backend.

Model Summary

```
In [14]: model.summary()
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 256, 256, 32)	896
activation_1 (Activation)	(None, 256, 256, 32)	0
batch_normalization_1 (Batch Normalization)	(None, 256, 256, 32)	128
max_pooling2d_1 (MaxPooling2D)	(None, 85, 85, 32)	0
dropout_1 (Dropout)	(None, 85, 85, 32)	0
conv2d_2 (Conv2D)	(None, 85, 85, 64)	18496
activation_2 (Activation)	(None, 85, 85, 64)	0
batch_normalization_2 (Batch Normalization)	(None, 85, 85, 64)	256
conv2d_3 (Conv2D)	(None, 85, 85, 64)	36928
activation_3 (Activation)	(None, 85, 85, 64)	0
batch_normalization_3 (Batch Normalization)	(None, 85, 85, 64)	256

dropout_2 (Dropout)	(None, 42, 42, 64)	0
conv2d_4 (Conv2D)	(None, 42, 42, 128)	73856
activation_4 (Activation)	(None, 42, 42, 128)	0
batch_normalization_4 (Batch Normalization)	(None, 42, 42, 128)	512
conv2d_5 (Conv2D)	(None, 42, 42, 128)	147584
activation_5 (Activation)	(None, 42, 42, 128)	0
batch_normalization_5 (Batch Normalization)	(None, 42, 42, 128)	512
max_pooling2d_3 (MaxPooling2D)	(None, 21, 21, 128)	0
dropout_3 (Dropout)	(None, 21, 21, 128)	0
flatten_1 (Flatten)	(None, 56448)	0
dense_1 (Dense)	(None, 1024)	57803776
activation_6 (Activation)	(None, 1024)	0
batch_normalization_6 (Batch Normalization)	(None, 1024)	4096
dropout_4 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 10)	10250
activation_7 (Activation)	(None, 10)	0

=====

Total params: 58,097,546
Trainable params: 58,094,666
Non-trainable params: 2,880

Model predicted (x test)

```
In [21]: model.predict(x_test)

Out[21]: array([[9.5271689e-01, 1.5521948e-03, 4.2496283e-02, ..., 1.3076633e-03,
 8.5263127e-05, 1.6868362e-03],
 [9.3078929e-01, 2.0331030e-03, 6.4354151e-02, ..., 1.0928203e-03,
 7.5830547e-05, 1.4821526e-03],
 [8.9779073e-01, 1.9668266e-03, 9.8844640e-02, ..., 9.0045738e-04,
 5.4768530e-05, 3.4645473e-04],
 ...,
 [8.9883637e-01, 2.6081468e-03, 9.7144879e-02, ..., 9.7753410e-04,
 6.1417362e-05, 2.2906535e-04],
 [9.4068742e-01, 2.0673431e-03, 5.6085683e-02, ..., 7.6615368e-04,
 3.1801814e-05, 2.8233606e-04],
 [9.3516463e-01, 2.4289112e-03, 6.0171023e-02, ..., 8.4805826e-04,
 6.1846920e-05, 1.1491876e-03]], dtype=float32)
```

Plot the train and validation curve

Plot the train and val curve

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
#Train and validation accuracy
plt.plot(epochs, acc, 'b', label='Training accuracy')
plt.plot(epochs, val_acc, 'r', label='Validation accuracy')
plt.title('Training and Validation accuracy')
plt.legend()

plt.figure()
#Train and validation loss
plt.plot(epochs, loss, 'b', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and Validation loss')
plt.legend()
plt.show()
```

Plagiarism

161-15-726

ORIGINALITY REPORT

27 %

SIMILARITY INDEX

21 %

INTERNET SOURCES

12 %

PUBLICATIONS

21 %

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to Daffodil International University

Student Paper

7 %

2

Santi Kumari Behera, Lipsarani Jena, Amiya Kumar Rath, Prabira Kumar Sethy. "Disease Classification and Grading of Orange Using Machine Learning and Fuzzy Logic", 2018 International Conference on Communication and Signal Processing (ICCSP), 2018

Publication

2 %

3

machinelearningmastery.com

Internet Source

1 %

4

muthu.co

Internet Source

1 %

5

www.aiproblog.com

Internet Source

1 %

6

stackoverflow.com

Internet Source

1 %

7

en.wikipedia.org

Internet Source

1 %

8	www.geeksforgeeks.org Internet Source	1%
9	ijraset.com Internet Source	1%
10	Submitted to St. Patrick's College Student Paper	1%
11	Monzurul Islam, Anh Dinh, Khan Wahid, Pankaj Bhowmik. "Detection of potato diseases using image segmentation and multiclass support vector machine", 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), 2017 Publication	1%
12	acadpubl.eu Internet Source	1%
13	www.cnblogs.com Internet Source	1%
14	Submitted to The University of the South Pacific Student Paper	1%
15	Submitted to Queen Mary and Westfield College Student Paper	1%
16	Submitted to Indian Institute of Management, Bangalore Student Paper	1%