

# DETECTION OF POTATO DISEASES USING DEEP LEARNING

BY

**MD. JABER HOSSAIN**

**ID: 171-15-8656**

AND

**MD. MAHEDI HASAN**

**ID: 171-15-8965**

AND

**MD. SAJIB SIKDER**

**ID: 171-15-9551**

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

Supervised By

**Md. Sazzadur Ahamed**

Sr. Lecturer

Department of CSE

Daffodil International University

Co-Supervised By

**Raja Tariqul Hasan Tusher**

Sr. Lecturer

Department of CSE

Daffodil International University



**DAFFODIL INTERNATIONAL UNIVERSITY**

**DHAKA, BANGLADESH**

**MAY 2021**

## **APPROVAL**

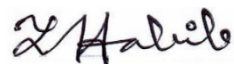
This Project titled “Detection of Potato Diseases using Deep Learning”, submitted by Md. Jaber Hossain, ID No: 171-15-8656, Md. Mahedi Hasan, ID No: 171-15-8965 and Md. Sajib sSikder, ID No: 171-15-9551 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 02/06/2021.

### **BOARD OF EXAMINERS**



**Dr. Touhid Bhuiyan**  
**Professor & Head**  
Department of CSE  
Faculty of Science & Information Technology  
Daffodil International University

**Chairman**



**Md. Tarek Habib**  
**Assistant Professor**  
Department of CSE  
Faculty of Science & Information Technology  
Daffodil International University

**Internal Examiner**



**Nusrat Jahan**  
**Senior Lecturer**  
Department of CSE  
Faculty of Science & Information Technology  
Daffodil International University

**Internal Examiner**



**Dr. Mohammad Shorif Uddin**  
**Professor**  
Department of Computer Science and Engineering  
Jahangirnagar University

**External Examiner**

## DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Md. Sazzadur Ahamed, Sr. Lecturer Department of CSE** Daffodil International University. We 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:



---

**Md. Sazzadur Ahamed**  
Sr. Lecturer  
Department of CSE  
Daffodil International University

### Co-Supervised by:



---

**Raja Tariqul Hasan Tusher**  
Sr. Lecturer  
Department of CSE  
Daffodil International University

### Submitted by:




---

**Md. Jaber Hossain**  
ID: 171-15-8656  
Department of CSE  
Daffodil International University



---

**Md. Mahedi Hasan**  
ID: 171-15-8965  
Department of CSE  
Daffodil International University



---

**Md. Sajib Sikder**  
ID: 171-15-9551  
Department of CSE  
Daffodil International University

## ACKNOWLEDGEMENT

First, we express our heartiest thanks and gratefulness to Almighty God for His divine blessing makes us possible to complete the final year thesis successfully.

We really grateful and wish our profound our indebtedness to **Md. Sazzadur Ahamed, Sr. Lecturer**, Department of CSE Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of “*Deep Learning*” to carry out this project. His 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 made it possible to complete this project.

We would like to express our heartiest gratitude to **Mr. Dr. Touhid Bhuiyan, Professor and Head**, Department of CSE, for his kind help to finish our project and also to 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 discuss while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

## **ABSTRACT**

Potato is one of the essential foods and root vegetables in the world and is also the staple food in some different nations. Globally It is the fourth-largest food and now the seventh-largest growing food in Bangladesh. Most of the time, the fungus infects potatoes. Our farmers' profit, food demand, and food security majorly depend on Identifying proper diseases and giving solutions. At present, Image processing technology mostly performing the detection of identifying diseases. So, we used six different architectures of convolutional neural networks (CNN) in the deep learning field. The architectures are MobileNet, Xception, Inception V3, VGG16, ResNet50, VGG19. From these architectures, one of the most popular CNN architecture is MobileNet that found the best result in our observation. For this approach, we used the potato image dataset. Here classified our dataset into three different classes where infected potatoes are two classes and a fresh potato class. This work can help to detect the disease of potatoes. This system will make it easier for farmers and researchers to find out potato diseases. It is much easier to Identify potato diseases in this method than to detect and classify the potato diseases manually, and it is possible to Identify potato diseases in a short time. This will be helpful for those who will be working with CNN.

# TABLE OF CONTENTS

<b>CONTENTS</b>	<b>PAGE</b>
Approval	i
Board of examiners	i
Declaration	ii
Acknowledgments	iii
Abstract	iv
<b>CHAPTER</b>	
<b>CHAPTER 1: INTRODUCTION</b>	<b>1-5</b>
1.1 Introduction	1
1.2 Motivation	2
1.3 Problem Statement	3
1.4 Working Procedure	3
1.5 Objective	4
1.6 Expected Outcome	4
<b>CHAPTER 2:. RELATED WORK AND BACKGROUND STUDY</b>	<b>6-7</b>
2.1 Related Work	6

<b>CHAPTER 3: BACKGROUND STUDY</b>	<b>8-2</b>
3.1 Introduction	8
3.2 Computer Vision	8
3.3 Machine Learning	8
3.4 Deep learning	9
3.5 Image Classification	9
3.6 Artificial neural network(ANN)	9
3.7 Random Forest	10
3.8 Support vector machine(SVM)	10
3.9 Convolutional neural network (CNN)	10
3.9.1 CNN Architectures	11
<b>CHAPTER 4: RESEARCH METHODOLOGY</b>	<b>12-18</b>
4.1 Dataset description	12
4.2 Data preprocessing	13
4.3 Data Processing	13
4.4 Data Augmentation	13
4.5 Model Visualization	14
4.6 Algorithm	14
4.7 CNN Model	14
4.7.1 VGG16	15

4.7.2 Inception v3	15
4.7.3 Resnet 50	16
4.7.4 MobileNet	17
4.7.5 VGG 19	18
4.7.6 Xception	18
<b>CHAPTER 5: RESULT ANALYSIS AND DISCUSSION</b>	<b>19-45</b>
5.1 Introduction	19
5.2 Dataset summary	19
5.3 Result analysis for VGG-19 model	21
5.3.1 Training and validation Accuracy for VGG-19	22
5.3.2 Test Accuracy for VGG-19	23
5.3.3 Confusion Matrix for VGG-19	24
5.3.4 Diagram of training and validation accuracy for VGG-19	24
5.3.5 Diagram of training and validation Loss for VGG-19	25
5.4 Result analysis for MobileNet model	26
5.4.1 Training and validation Accuracy for MobileNet:	26
5.4.2 Test Accuracy for MobileNet	27
5.4.3 Confusion Matrix of MobileNet	27
5.4.4 Diagram of training and validation accuracy for MobileNet	28
5.4.5 Diagram of training and validation Loss for MobileNet:	29



5.5 Result analysis for InceptionV3 Model:	29
5.5.1 Training and validation Accuracy	30
5.5.2 Test Accuracy for Inception V3	31
5.5.3 Confusion Matrix for Inception V3	31
5.5.4 Diagram of training and validation accuracy for Inception V3	32
5.5.5 Diagram of training and validation Loss for Inception V3	32
5.6 Result analysis for ResNet50 model	33
5.6.1 Training and validation Accuracy for ResNet50	33
5.6.2 Test Accuracy for ResNet50	34
5.6.3 Confusion Matrix of ResNet50	35
5.6.4 Diagram of training and validation accuracy for ResNet50	36
5.6.5 Diagram of training and validation Loss for ResNet50	36
5.7 Result analysis for Xception model	37
5.7.1 Training and validation Accuracy for Xception	37
5.7.2 Test Accuracy for Xception	38
5.7.3 Confusion Matrix of Xception	39
5.7.4 Diagram of training and validation accuracy for Xception	39
5.7.5 Diagram of training and validation Loss for Xception	40
5.8 Result analysis for VGG-16 model	41
5.8.1 Training and validation Accuracy for VGG-16	41
5.8.2 Test Accuracy for VGG-16	42

5.8.3 Confusion Matrix of VGG-16	42
5.8.4 Diagram of training and validation accuracy for VGG-16	43
5.8.5 Diagram of training and validation Loss for VGG-16	44
5.9 Discussion	44
5.10 Performance Comparison	45
<b>CHAPTER 6: CONCLUSION, RECOMMENDATION AND FUTURE WORK</b>	<b>46-47</b>
6.1 Conclusion	46
6.2 Recommendation	46
6.3 Future work	47
<b>REFERENCES</b>	<b>48</b>

## LIST OF FIGURES

<b>FIGURES</b>	<b>PAGE NO</b>
Figure 3.1: Basic CNN architecture	11
Figure 4.1: Diagram of data dataset preparation	12
Figure 4.2: Vgg 16 Architecture	15
Figure 4.3: Inception V3 Architecture	16
Figure 4.4: Resnet 50 Architecture	17
Figure 4.5: MobileNet Architecture	17
Figure 5.1: Training and validation accuracy of VGG-19	23
Figure 5.2: Test accuracy of VGG-19	23
Figure 5.3: Confusion Matrix of VGG-19	24
Figure 5.4: Diagram of training and validation accuracy for VGG-19	25
Figure 5.5: Diagram of training and validation loss for VGG-19	25
Figure 5.6: Training and validation accuracy for MobileNet	27
Figure 5.7: Test accuracy for MobileNet	27
Figure 5.8: Confusion Matrix of MobileNet	28
Figure 5.9: Diagram of training and validation accuracy for MobileNet	28
Figure 5.10: Diagram of training and validation loss for MobileNet	29
Figure 5.11: Training and validation accuracy of Inception V3	30
Figure 5.12: Test accuracy of Inception V3	31
Figure 5.13: Confusion Matrix of Inception V3	31
Figure 5.14: Diagram of training and validation accuracy for Inception V3	32
Figure 5.15: Diagram of training and validation loss for Inception V3	33
Figure 5.16: Training and validation accuracy for ResNet50	34
Figure 5.17: Test accuracy for ResNet50	35
Figure 5.18: Confusion Matrix of ResNet50	35
Figure 5.19: Diagram of training and validation accuracy for ResNet50	36
Figure 5.20: Diagram of training and validation loss for ResNet50	37
Figure 5.21: Training and validation accuracy for Xception	38

Figure 5.22: Test accuracy for Xception	38
Figure 5.23: Confusion Matrix of Xception	39
Figure 5.24: Diagram of training and validation accuracy for Xception	40
Figure 5.25: Diagram of training and validation loss for Xception	40
Figure 5.26: Training and validation accuracy for VGG-16	42
Figure 5.27: Test accuracy for VGG-16	42
Figure 5.28: Confusion Matrix of VGG-16	43
Figure 5.29: Diagram of training and validation accuracy for VGG-16	43
Figure 5.30: Diagram of training and validation loss for VGG-16	44

## LIST OF TABLES

<b>TABLES</b>	<b>PAGE NO</b>
Table 5.1: Data summary	19
Table 5.2: Dataset of fresh Photo	20
Table 5.3: Dataset of Common Scab Photo disease	20
Table 5.4: Dataset of Black Heart Photo disease	21
Table 5.5: Accuracy of our CNN models	45

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction

The demand for the rising of potatoes is increasing day by day around the world. As a result, people take fresh potatoes as food in their daily life. In the last year 2019, the production of potatoes was 423 million metric tons. Potatoes provide essential nutrients, and it gives the calories to the human body. It has many different diseases, such as fungus and bacteria. Potato production is reduced due to fungal diseases. Fungal diseases are mainly occurred due to Fungal infection. Potato production seriously decreases when Fungal infection on potatoes increases a lot. It damages the skin of the potatoes and decreases the growth of the potato, and the beginning of this infection is from the infected soil, weeds. As a result, the farmers have to face many losses. For these diseases, the production of potatoes is decreasing day by day. Due to this reason, there is a huge possibility of a deficit of potatoes and an increase in the price of potato cost. If the machine can initially detect this disease, people will quickly get rid of this potato shortage. In that case, farmers will also get benefit from it. We collected image samples of the inside and outside of the potatoes to detect the diseases and then divide the samples into different categories based on fresh and infected potatoes. In this case, here research has worked on two diseases of potato, one Blackheart and the other Common scab. When potatoes are infected with Common Scab, brown blisters appear on their skin. When black heart disease occurs in potatoes, then inside on the body show black spots and holes. As a result, the machine can easily identify the samples of potato images accurately. As a result, the machine can make a difference between healthy and sick potatoes correctly. If researchers collect potatoes directly from the field and detect the disease, the time will increase, and accuracy will decrease, and also the researchers do research from our observation, then their working time will reduce, and the accuracy of the working process will increase. For detecting potato diseases, here used a deep learning approach that is part of computer vision technology. We collected the image data, divided it into three different classes, then trained 85% of the image data from our data set, 10% of data is used for validation, and tested 5%

of the data from our dataset. Then applied this data set into CNN algorithms for getting the expected outcome. Then the system has shown different types of charts for the visualization of the expected outcome. From the CNN algorithm, which model achieves the high accuracy, took that accuracy as the final result.

## **1.2 Motivation**

Potato diseases are a vital issue for growing potatoes. Lots of potatoes have been wasted for different types of diseases. Potato collectors face problems finding out about the diseases. Sometimes they do not know the type of diseases of potatoes. For this reason, they cannot take the initial step to reduce the diseases. That is why we want to create a system that identifies diseases by taking pictures. After finding diseases, the system will give some solutions to reduce diseases. Increase the production of potatoes that will impact our national economy.

When we were thinking about our final year project, we deeply observed that deep learning-related work had been much less in Bangladesh. But compared to that, a lot of machine learning-related work has been done in Bangladesh. In Bangladesh, there was no work to detect potato diseases digitally. Then we decided to create a system that can easily find out the potato disease digitally. The farmers and researchers from agricultural universities can quickly identify potato diseases. Both farmers and researchers will easily take the step to protect the potatoes by finding the potato diseases digitally. At the same time, the nutritional value of potatoes and the quality of potatoes will be easily increased. When potato diseases find out easily, the possibility of loss to the farmer will be reduced and increase potatoes production. As a result, the rate of export of potatoes in Bangladesh will increase manifold.

### **1.3 Problem Statement**

To detect the potato diseases manually, the farmers face many problems, and many times, they cannot easily detect potato diseases correctly. Digital identification of potatoes is an accurate and effective method to overcome this problem, and this method is much less costly than manual detection. Through the digital method, the rate of accurate identification of potato diseases is much higher than the manual method. That is why we have chosen the deep learning (Convolutional Neural Network) method for identifying potato diseases.

### **1.4 Working Procedure**

- Topic selection
- Topic analysis
- Go to the field to collect data
- Take interviews from potato cultivators
- Dataset Collection
- Google Colab as implementation tools
- Data preprocessing for reshaping image.
- Apply algorithm CNN
- Apply models are MobileNet, Xception, Inception V3, VGG16, ResNet50, VGG19
- Reach the final Result
- Result in visualization as the confusion matrix

### **1.5 Objective**

We are going to detect potato diseases using deep learning. First of all, we have to collect data. Then, we will use a convolutional neural network (CNN) algorithm in the deep learning field. CNN algorithm helps to detect the images of potato diseases that we will collect. As a tool and language, we will use Python.

- Interact our dataset to the program for identifying diseases.



- To help cultivators for finding disease potatoes.
- To know about the importance of fresh potatoes.
- To know about the importance of deep learning.
- To know how to implement image data in the CNN algorithm.

## 1.6 Expected Outcome

Digital methods help diagnose potato diseases in a very short time. Our main target is increasing the accuracy rate than the previous potato disease detection of other researchers. Now we want to complete our work perfectly with high accuracy. That is why we apply the best deep learning algorithm, the Convolutional neural network. In this CNN algorithm, we will apply the six best architecture:

- MobileNet
- Xception
- Inception V3
- VGG16
- ResNet50
- VGG19

After training the dataset over MobileNet, Xception, Inception V3, VGG16, ResNet50, VGG19. From these six different architectures, our expected accuracy will be up 99%. From the comparison of six different architectures, we expect that the best architecture will be MobileNet. By the CNN algorithm, cultivators can identify potatoes that are infected by the disease.

## **CHAPTER 2**

### **RELATED WORK**

#### **2.1 Related Work**

For a long time, the Convolutional neural network (CNN) is a popular algorithm for detecting diseases and classifying image data. So, there are lots of researchers who use CNN to identify the image data. Wenxue Tan et al. in [1] proposed a method to identify the disease of fruit-melon by a video sensor using CNN. After using CNN, the accuracy was up to 97.5%.

Other researchers by Inkyu Sa et al. in [2] proposed a method to detect different kinds of fruit images by using the R-CNN Algorithm. A pre-trained ImageNet model is used to fit the dataset, and the image color method was RGB and NIR.

By using the ANN algorithm, in 2016, a group of researchers Cubero, S. et al. in [3] proposed a way to detect the citrus fruits and get the expected result around 98% accuracy.

In LSVRC2010, a group of researchers, Alex Krizhevsky et al. in [4] won the contest to classify a large number of data by using the CNN model with an error rate of 37.5% for the top 1 and 17% for the top 5. On that paper, researchers used 1000 classes, including high-resolution images, to classify the images.

In 2019 Dor Oppenheim et al. [5] used the convolutional neural network to classify four types of potato diseases from 2465 images, and the accuracy is 97.1%.

Monzurul Islam, Anh Dinh, Khan Wahid, Pankaj Bhowmik [7] worked with the support vector machine (SVM) to detect potato leaf diseases. The types of diseases are Late blight, early blight, and the accuracy was 95%. From 300 images, 60% of images were selected for training purposes, and 40% of images were chosen for testing purposes.

To detect the different types of potato diseases, in 2017, D. Oppenheim et al. [8] offered a method to detect the five types of potato diseases by using the CNN algorithm. In that research, they got an accuracy between 90% to 95%.

Priyadarshini Patil, Nagaratna Yaligar, Meena S M [9], in their research paper, proposed a method to identify the potato leaf diseases by using ANN with an accuracy of 92%, SVM with an accuracy of 84%, RF with the accuracy of 79%. From 892 images where 462 images were fresh, and infected images were 430 images were infected.

Tianmei Guo, Jiwen Dong, Henjian LiYunxing Gao, in their research paper they applied Convolution Neural Network(CNN) Algorithm on image classification to classify their dataset with the architecture of CNN, such as LeNet-5, Alexnet, VGGNet, ZFNet, and GoogleNet [10].

## **CHAPTER 3**

### **BACKGROUND STUDY**

#### **3.1 Introduction**

We had to learn many topics to complete our research which we have described here. Here we will present some important topics required for image processing, such as Convolutional Neural Network Rainforest and Deep Learning. Here we will also discuss the various Architectures of the Convolutional Neural Network.

#### **3.2 Computer Vision**

Computer Vision is a major part of Artificial Intelligence. With computer vision, the computer can easily collect information from any digital pictures or videos and give the right solution from the collected information. Like the human brain can solve a problem by looking at pictures or videos, a computer can do that with computer vision. When we classify the images using a machine or computer, the image classification is done by computer vision.

#### **3.3 Machine Learning**

Donald Hebb established machine learning in 1949. machine learning is one of the most important applications of artificial intelligence, and It is supervised learning. For working machine learning, it takes the inputs, then analyses the data and shows the output it is the fifth-generation application. The main vision of machine learning is to learn and improve the experience and develop the computer brain. It can train the computer program automatically. By machine learning, a machine can think similarly to a human brain. We can improve social services, customer support, detect email spam, specific malware filtering, and Image processing by machine learning techniques. Currently, robots rely on machine learning to run their activities.

### **3.4 Deep learning**

In 1986 Rina Dechter was introduced deep learning to us. Deep learning is also a part of artificial intelligence and machine learning. Deep learning is mostly working with images. In this field, image classification is variously applied. Our potato disease detection is a part of Deep Learning. It is the Algorithm of machine learning. In the deep learning field, there are lots of algorithms applied. The algorithms are the artificial neural network, convolutional neural network, random forest support vector machine. In these algorithms, there is much architecture applied. We have applied the Algorithm convolutional neural network for our project, which is part of the deep learning algorithm.

### **3.5 Image Classification**

Image classifications are applied by artificial intelligence. With the help of artificial intelligence and machine learning, a machine can classify images. With image classification, we can diagnose different diseases, classify different fruits, face detection, identify gender, increase office security and identify thieves. Image classification is a part of computer vision. We can classified images by using different types of processes.

### **3.6 Artificial neural network(ANN)**

Work on the first artificial neural network(ANN) model began in 1943. The names of the inventors of Artificial Neural Networks are Warren Mcculloch and Walter Pitts. Artificial Neural Network is a subpart of Deep Learning. Artificial neural networks are connected by neurons that act as processing units and that indicated by the node. One neuron is connected to another neuron, and thus many neurons are connected to make a neural network. The artificial neural network has three types of layers. The first layer is the input layer, where it receives input, and the second is the hidden layer and output layer. It works like the biological brain, where input is given outside, and it calculates the accurate result. An artificial neural network is one of the best algorithms for image classification, speech recognition, text classification, Natural language processing, binary classification.

### **3.7 Random Forest**

Random forest is a supervised learning algorithm, and Breiman first proposed this Algorithm in 2001, and Breiman discovered the regression tree in 1984. This Algorithm is a part of Deep Learning. Currently, a random forest algorithm is mostly used in image classification. This Algorithm is very easy to use. Also, with this Algorithm, we can achieve good results.

Random forest works in four steps:

- At first, need to the selected given dataset from a random forest sample.
- then have to make a decision tree one by one for each sample,
- then have to make each predicted result for a vote and

then select the layer that has the best results

### **3.8 Support vector machine(SVM)**

Support vector machine used for data classification or detection. However, support vector machines are mostly used in image classification. It is a supervised learning method, and in 1963 it was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis. SVMs have a unique method for machine learning algorithms. The kernel is one where the kernel has a trick to take low dimensional input and create a high dimensional space. It represents different classes in hyperplane and finds the highest marginal hyperplane. Support Vectors, Hyperplane, Margin, are the important concepts in Support vector machine.

### **3.9 Convolutional neural network (CNN)**

The convolutional neural network is an algorithm that is based on Deep learning. This Algorithm is a part of a neural network. This is the best Algorithm for image detection. It comes from the artificial neural network. This album is mostly used for image detection. We use this Algorithm for our project. Because CNN has a lot of popular architecture, the image can be detected accurately by that architecture.

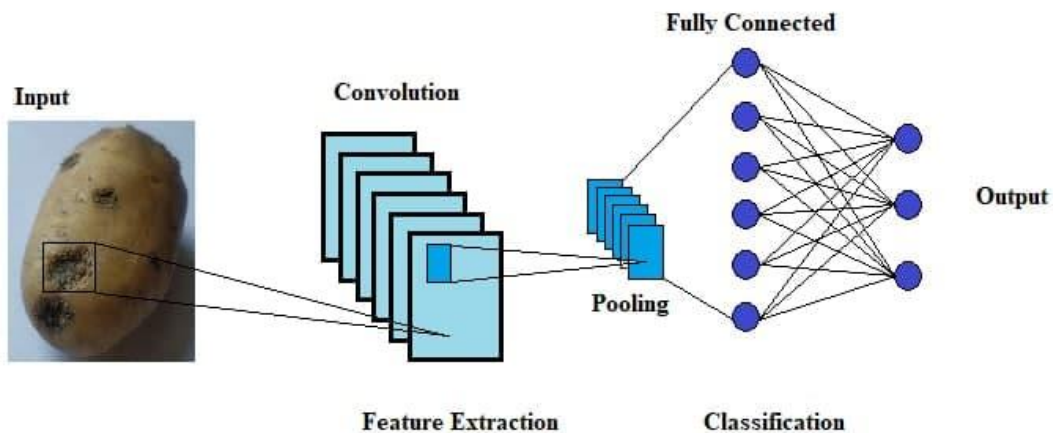


Figure 3.1: Basic CNN architecture

### 3.9.1 CNN Architectures

CNN comes for overcoming the limitations of the image section in ANN. In CNN, it has some layers like Input Layer, Convo Layer, Pooling Layer, Fully Connected Layer, Softmax Layer, Output Layer. It has some best pre-trained architectures that help generate the best accuracy.

## CHAPTER 4

### RESEARCH METHODOLOGY

#### Research Methodology

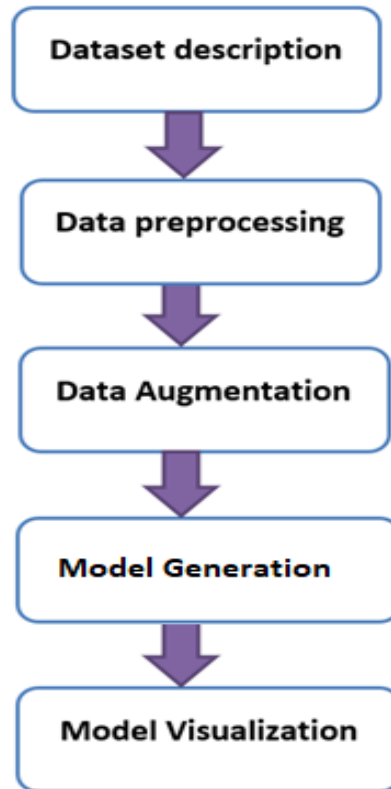


Figure 4.1: Diagram of data dataset preparation

#### 4.1 Dataset description

In our dataset, we have taken a total of 5781 images of data. There are 2639 pictures of diseased potatoes and another 3142 pictures of non-diseases. These pictures are captured from various distances and angles. We have taken all images from our devices. We do not



collect our images from different mediums and sites. Some devices like Samsung, Symphony, and One plus capture almost all photos. The whole dataset is divided into three different types. These are Common scab disease that consists of 1763 images, Blackheart disease that consists of 340 photos, and Fresh Potatoes consists of 3142 pictures.

## **4.2 Data preprocessing**

Before applying the dataset to the CNN model, the images are converted into standard shapes resized by 224 x 224 for applying in the CNN model. Image pixels are divided by 255 for rescaling the image where images are scaled between 0 to 1. Here RGB color images are used for data processing. We have used data augmentation for only blackheart disease.

## **4.3 Data Processing**

Some images of Blackheart are converted into a different shape by data augmentation. All data are separated into three categories test set, validation set, and training set. Each set has three different classes, which are Common scab, Blackheart, and Fresh Potatoes. 85% of the data is stored in the training set. In the validation set, 10% of the data is stored, and 5% of the data is stored in the test set. By using the Image Data Generator class of Keras, we loaded all data from the directory. For the 5781 images dataset, we applied batch size 32 for step-by-step training, where the first 32 images are trained to step by step. After completing the first 32 images, it started training the second 32 images. This process continued to the last stage of the batch size. Batch size one is applied for the testing dataset. Since we have more than two classes of our dataset, therefore categorical class mode we have used.

## **4.4 Data Augmentation**

For preparing the dataset, we have augmented some data. For Blackheart disease, we have augmented pictures. It increases our data in different shapes. By one picture, it creates lots of different pictures.

## 4.5 Model Visualization

When we get the output from our dataset, the output can be shown at different angles. This is the visualization of the model. For our model, we have shown the accuracy of our training and testing output. We have also shown the test data loss, training data loss, training accuracy, and testing accuracy by graphically. For deep visualization, We have shown the confusion matrix that shows the accuracy and error of data.

## 4.6 Algorithm

For detecting two types of diseases, common scab, Blackheart, and healthy potatoes, we applied the most useful Algorithm named Convolutional neural network (CNN). Generally, a Convolutional neural network (CNN) is commonly used in image classification, and it is also used in image segmentation. In the CNN algorithm, it has single layers as well as two or more layers. For the image classification of these diseases, we have applied multiple layers of CNN.

## 4.7 CNN Model

We have used six different CNN models for training our dataset. These models are given below :

- VGG16
- Inception V3
- ResNet50
- VGG19
- MobileNet
- Xception

### 4.7.1 VGG16

It is one of the best architectures of the Convolutional neural network. K. Simonyan and A. Zisserman developed this Convolutional neural network. Vgg 16 model won ILSVRC-2014 competition In 2014. This model accrued 92.7% for the top 5 test accuracy. It has 3\*3 multiple kernel sizes. Fifteen million data were taken for analyzing Vgg 16. Multiple layers were used for the VGG16 model. It has 16 layers, and the layers are convolution layer and relu, max pooling layer, fully connected layer, softmax activation layer. For the conv1 layer, the size of the image always will be 224\*224.

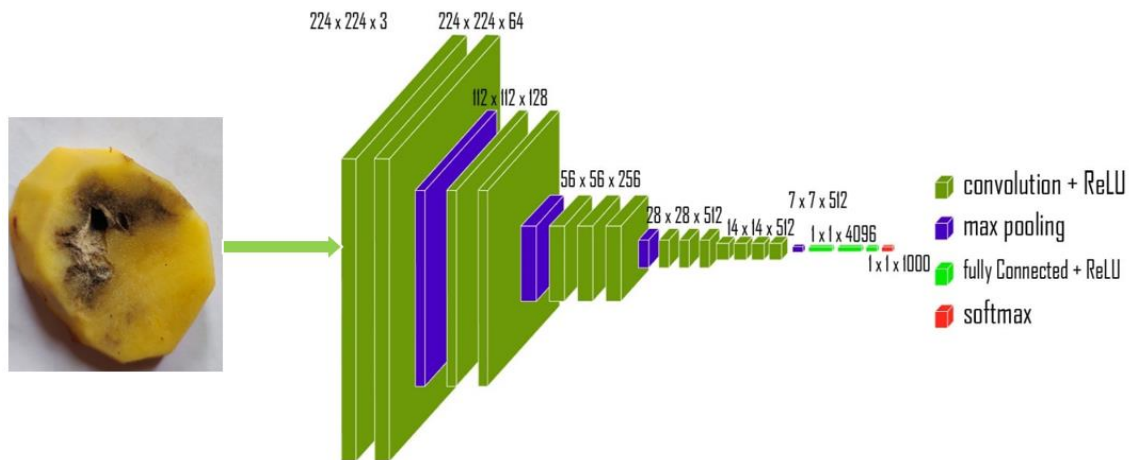


Figure 4.2: Vgg 16 Architecture

### 4.7.2 Inception v3

It is another architecture of the Convolutional neural network (CNN). This model is used for image classification and so on. It comes to overcome the previous version of the inception model. It has 24 million parameters. Some layers are applied In this model. It has 48 layers, and the layers are Convolution, AvgPooling, MaxPooling, Concat, Fully connected, Softmax, and Dropout. The image size of the image is 299\*299\*3. Inception v3 was trained by 1 million images and 1000 classes with the ImageNet dataset. By this model can strongly predict the images with high accuracy and gives a better output.

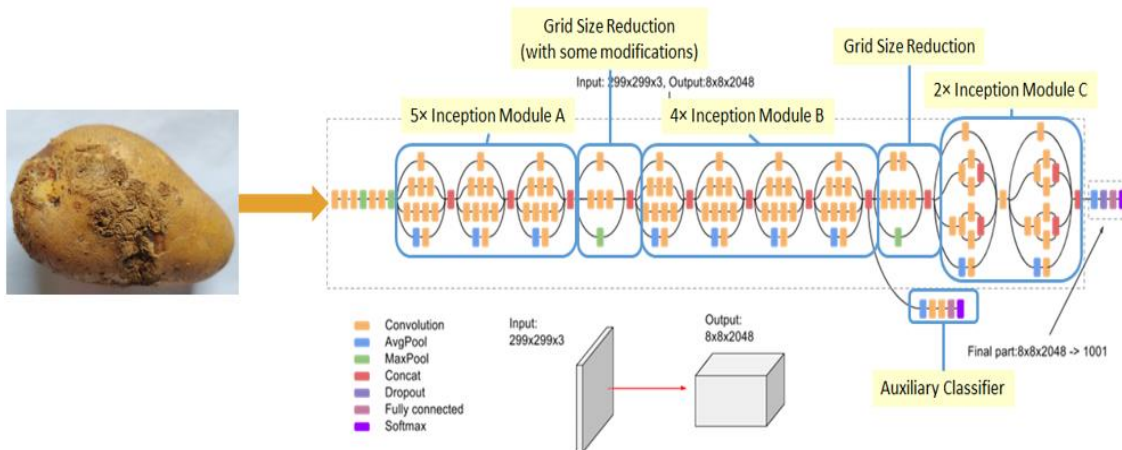


Figure 4.3: Inception V3 Architecture

### 4.7.3 Resnet 50

This model is a part of deep learning, and it is the architecture of the Convolutional neural network (CNN) algorithm. The model is the best and popular among all the architectures of the CNN algorithm. Till now, this model has given high accuracy. It has 50 layers. The model is proposed by the researchers in 2015 and won the competition of ImageNet in 2015. The image size of this model is 224\*224\*3. The model has about 23 million trainable parameters. It works with RGB images. By this model, images are perfectly detected and give high accuracy among all architectures.

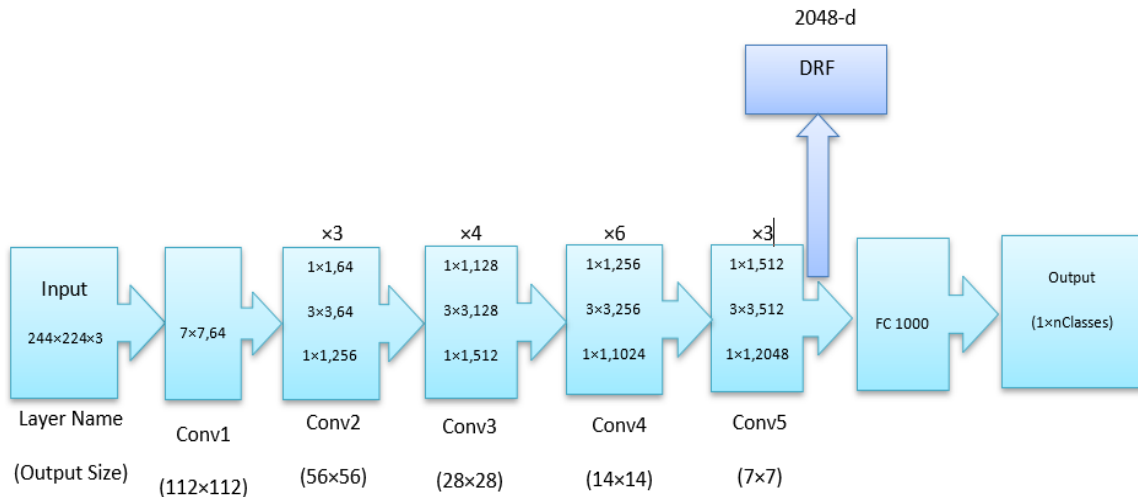


Figure 4.4: Resnet 50 Architecture

#### 4.7.4 MobileNet

MobileNet was developed by Andrew G. Howard in 2017, but it was proposed before publishing Xception in 2016. It is a CNN model. It uses deeply separable convolutional neural networks by choice of standard convolutions to reduce computation and model size. It can be used to create light deep neural networks for mobile, smartphone, and embedded applications. It is popular for its model and calculation speed. For this reason, this is suitable for deployment.

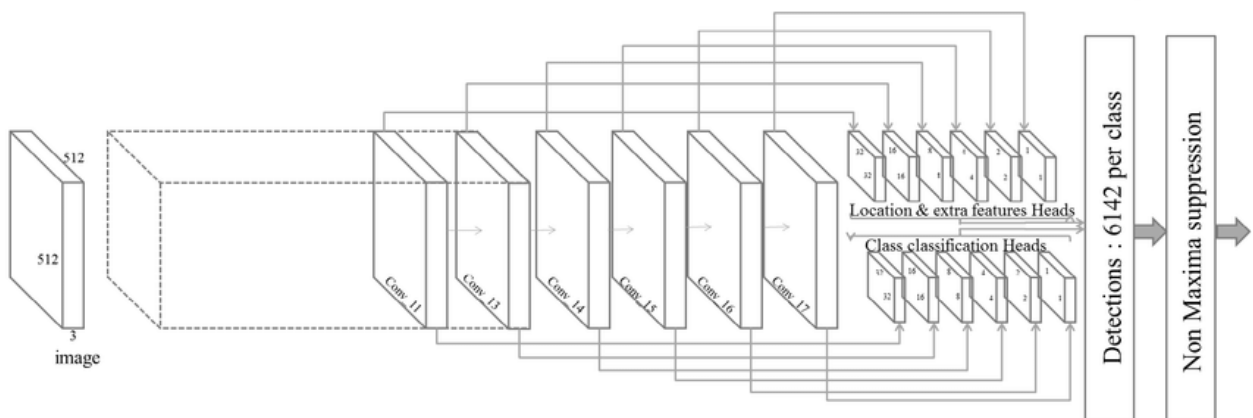


Figure 4.5: MobileNet Architecture

### **4.7.5 VGG 19**

VGG-19 is an updated model of VGG-16. It is one of the best architectures of CNN. From the University of Oxford, VGG-19 was published in 2014 by a group of Visual Geometry Group. This model achieved second place from the competition of the ImageNet Classification challenge. It has 19 layers, and the layers contain sixteen convolution layers, three fully connected layers, five max-pooling layers, and one softmax layer. The image size of this model is 224\*224 for RGB.

### **4.7.6 Xception**

Xception architecture was proposed by Francois Chollet. This model is a part of deep learning, and It is the architecture of the Convolutional neural network (CNN) algorithm. With the help of this architecture, we achieved outstanding results. Xception architecture is an extension of the other architecture of Inception v3. This model is used for image classification. It has 71 layers. The Image size of Xception architecture is 299\*299.

## CHAPTER 5

### RESULT ANALYSIS AND DISCUSSION

#### 5.1 Introduction

In this section, we will describe the result of each CNN model. For detecting the potato disease correctly, we applied five CNN models. In each model, we applied a total of 5781 images. We have classified three classes for our dataset, which are fresh potato, common scab, Blackheart. We captured 3678 fresh potato images and 1763 images for common scab disease, and 340 images for blackheart disease. For the training model, we stored 85% of the data in the training classes. The rest 10% of image data are stored in the validation classes, and 5% data are stored in the test classes.

In each CNN model, we have applied batch size 32 for the training data set, and one batch size is applied for the test data set. Hence our classes are multiple, so we have applied the categorical class mode for each model. For the five CNN architectures, a maximum of 20 epochs is applied to generate the accuracy and loss for the dataset. To show the results subtly and accurately, we have shown the test accuracy, confusion matrix, and diagram of training accuracy, training loss, validation accuracy, and validation loss of each CNN model.

#### 5.2 Dataset summary

Table 5.1: Data summary

<b>Train</b>	<b>Validation</b>	<b>Test</b>	<b>Total</b>
4,931	566	284	5781

Table 5.2: Dataset of fresh Photo


Potato disease	Train	Validation	Test	total	Data view
Fresh potato	3,142	356	80	3,678	

Table 5.3: Dataset of Common Scab Photo disease



Potato disease	Train	Validation	Test	total	Data view
Common Scab	1,500	176	87	1,763	



Table 5.4: Dataset of Black Heart Photo disease

Potato disease	Train	Validation	Test	total	Data view
Black Heart	289	34	17	340	

### 5.3 Result analysis for VGG-19 model

In the VGG-19 model, 85% of image data is used for training purposes. 10% data is used for validation purposes which are 356 images. 5% of data is used for testing purposes. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In the training set, we have uses three classes for our dataset. The name of the classes are fresh potato, common scab, Blackheart. In the validation set, we have uses three classes for our dataset. The name of the classes are fresh potato, Common Scab, Blackheart. In the test set, we have uses three classes for our dataset. The name of the classes are fresh potato, Common Scab, Blackheart. We have used 3,142 fresh potato images for the training, 1500 images are for Common Scab, 289 images are used for Blackheart. For validation, we have used 256 fresh potato images, 176 images are used for common scab, and 34 images are used for Blackheart. For testing, we have used 180 fresh potato images, 87 images are used for common scab, and 17 images are used for Blackheart.

In this model, total parameters are 20,552,771 and trainable parameters are 528,387 and Non-trainable parameters are 20,024,384.

### 5.3.1 Training and validation Accuracy for VGG-19

We run the program for 20 epochs. Per epoch shows the training loss, validation loss, training accuracy, validation accuracy. Each epoch shows the different accuracy of training and testing, which are shown in the figure.

```
Epoch 1/20
155/155 [=====] - 2413s 15s/step - loss: 0.6522 - accuracy: 0.7350 - val_loss: 0.3921 - val_accuracy: 0.8889
Epoch 2/20
155/155 [=====] - 683s 4s/step - loss: 0.3198 - accuracy: 0.8812 - val_loss: 0.1744 - val_accuracy: 0.9444
Epoch 3/20
155/155 [=====] - 691s 4s/step - loss: 0.2533 - accuracy: 0.9034 - val_loss: 0.0620 - val_accuracy: 1.0000
Epoch 4/20
155/155 [=====] - 681s 4s/step - loss: 0.1974 - accuracy: 0.9311 - val_loss: 0.2100 - val_accuracy: 0.8889
Epoch 5/20
155/155 [=====] - 685s 4s/step - loss: 0.1612 - accuracy: 0.9420 - val_loss: 0.1321 - val_accuracy: 0.9444
Epoch 6/20
155/155 [=====] - 683s 4s/step - loss: 0.1417 - accuracy: 0.9475 - val_loss: 0.3090 - val_accuracy: 0.9444
Epoch 7/20
155/155 [=====] - 679s 4s/step - loss: 0.1385 - accuracy: 0.9458 - val_loss: 0.4241 - val_accuracy: 0.9444
Epoch 8/20
155/155 [=====] - 676s 4s/step - loss: 0.1290 - accuracy: 0.9531 - val_loss: 0.1213 - val_accuracy: 0.9444
Epoch 9/20
155/155 [=====] - 685s 4s/step - loss: 0.1100 - accuracy: 0.9611 - val_loss: 0.0264 - val_accuracy: 1.0000
Epoch 10/20
155/155 [=====] - 683s 4s/step - loss: 0.1144 - accuracy: 0.9568 - val_loss: 0.0160 - val_accuracy: 1.0000
```

```

Epoch 10/20
155/155 [=====] - 683s 4s/step - loss: 0.1144 - accuracy: 0.9568 - val_loss: 0.0160 - val_accuracy: 1.0000
Epoch 11/20
155/155 [=====] - 684s 4s/step - loss: 0.0957 - accuracy: 0.9664 - val_loss: 0.0273 - val_accuracy: 1.0000
Epoch 12/20
155/155 [=====] - 686s 4s/step - loss: 0.1016 - accuracy: 0.9663 - val_loss: 0.0090 - val_accuracy: 1.0000
Epoch 13/20
155/155 [=====] - 684s 4s/step - loss: 0.0997 - accuracy: 0.9635 - val_loss: 0.0133 - val_accuracy: 1.0000
Epoch 14/20
155/155 [=====] - 678s 4s/step - loss: 0.0863 - accuracy: 0.9745 - val_loss: 0.1403 - val_accuracy: 0.9444
Epoch 15/20
155/155 [=====] - 683s 4s/step - loss: 0.0869 - accuracy: 0.9692 - val_loss: 0.0113 - val_accuracy: 1.0000
Epoch 16/20
155/155 [=====] - 678s 4s/step - loss: 0.0729 - accuracy: 0.9726 - val_loss: 0.0206 - val_accuracy: 1.0000
Epoch 17/20
155/155 [=====] - 682s 4s/step - loss: 0.0786 - accuracy: 0.9712 - val_loss: 0.0171 - val_accuracy: 1.0000
Epoch 18/20
155/155 [=====] - 681s 4s/step - loss: 0.0770 - accuracy: 0.9684 - val_loss: 0.0073 - val_accuracy: 1.0000
Epoch 19/20
155/155 [=====] - 684s 4s/step - loss: 0.0689 - accuracy: 0.9782 - val_loss: 0.0397 - val_accuracy: 1.0000
Epoch 20/20
155/155 [=====] - 686s 4s/step - loss: 0.0687 - accuracy: 0.9777 - val_loss: 0.0627 - val_accuracy: 1.0000

```

Figure 5.1: Training and validation accuracy of VGG-19

### 5.3.2 Test Accuracy for VGG-19

After completing our training dataset training, we have applied our test dataset to get the actual accuracy. We got the 98% test accuracy that shows in the figure.

```

test_loss, test_acc = model.evaluate(test_set, verbose=2)
print('\nTest accuracy:', test_acc)

```

```

284/284 - 67s - loss: 0.0498 - accuracy: 0.9824

```

```

Test accuracy: 0.9823943376541138

```

Figure 5.2: Test accuracy of VGG-19

### 5.3.3 Confusion Matrix for VGG-19

Here is accurately identified Blackheart and fresh potatoes. There are no error data in these two classes. The common scab data of the test set predicted 82 images out of 87.

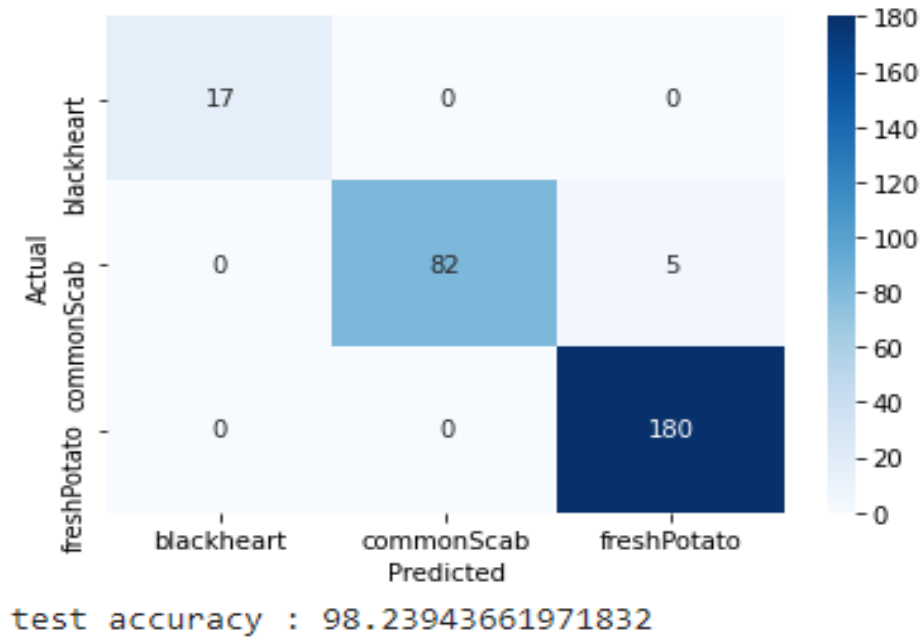


Figure 5.3: Confusion Matrix of VGG-19

### 5.3.4 Diagram of training and validation accuracy for VGG-19

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training accuracy, and the orange line shows the validation accuracy.

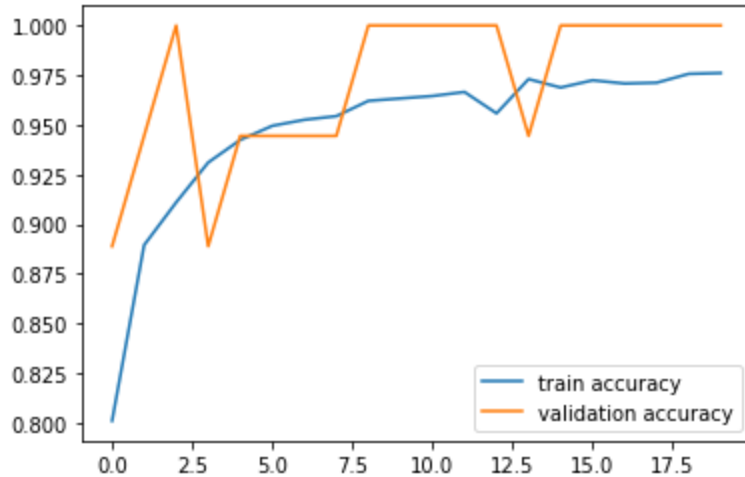


Figure 5.4: Diagram of training and validation accuracy for VGG-19

### 5.3.5 Diagram of training and validation Loss for VGG-19

In twenty epochs for our dataset, each epoch shows the training loss and validation loss shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.

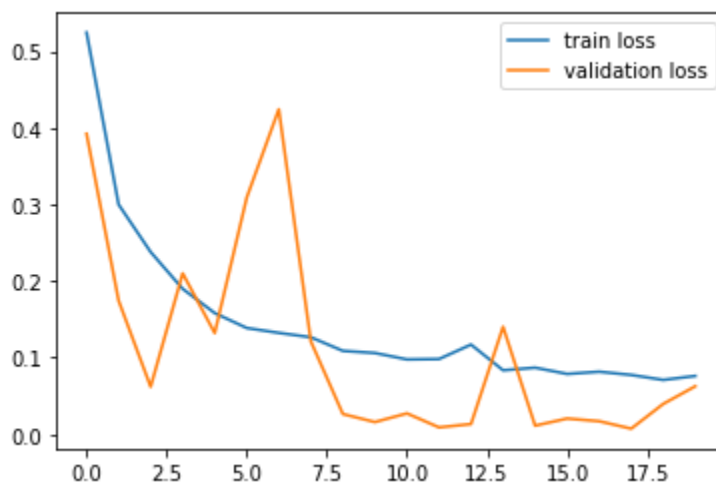


Figure 5.5: Diagram of training and validation loss for VGG-19

## 5.4 Result analysis for MobileNet model

In the MobileNet model, 85% of image data is used for training purposes. 10% data is used for validation purposes which are 356 images. 5% of data is used for testing purposes. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In this model, total parameters are 4,281,539 and trainable parameters are 1,052,675 and Non-trainable parameters are 3,228,864.

### 5.4.1 Training and validation Accuracy for MobileNet:

We run the program for 20 epochs. Per epoch shows the training loss, validation loss, training accuracy, validation accuracy. Each epoch shows the different accuracy of training and testing, which are shown in the figure.

```
Epoch 1/20
155/155 [=====] - 2372s 15s/step - loss: 0.3423 - accuracy: 0.8912 - val_loss: 0.0230 - val_accuracy: 1.0000
Epoch 2/20
155/155 [=====] - 675s 4s/step - loss: 0.0355 - accuracy: 0.9883 - val_loss: 0.0014 - val_accuracy: 1.0000
Epoch 3/20
155/155 [=====] - 671s 4s/step - loss: 0.0228 - accuracy: 0.9925 - val_loss: 0.0011 - val_accuracy: 1.0000
Epoch 4/20
155/155 [=====] - 677s 4s/step - loss: 0.0130 - accuracy: 0.9957 - val_loss: 6.8506e-04 - val_accuracy: 1.0000
Epoch 5/20
155/155 [=====] - 674s 4s/step - loss: 0.0460 - accuracy: 0.9838 - val_loss: 0.6505 - val_accuracy: 0.9444
Epoch 6/20
155/155 [=====] - 673s 4s/step - loss: 0.0078 - accuracy: 0.9974 - val_loss: 0.0007 - val_accuracy: 0.9444
Epoch 7/20
155/155 [=====] - 672s 4s/step - loss: 0.0045 - accuracy: 0.9991 - val_loss: 0.0039 - val_accuracy: 1.0000
Epoch 8/20
155/155 [=====] - 672s 4s/step - loss: 0.0088 - accuracy: 0.9974 - val_loss: 0.0188 - val_accuracy: 1.0000
Epoch 9/20
155/155 [=====] - 669s 4s/step - loss: 0.0021 - accuracy: 0.9996 - val_loss: 0.0033 - val_accuracy: 1.0000
Epoch 10/20
155/155 [=====] - 671s 4s/step - loss: 0.0145 - accuracy: 0.9943 - val_loss: 5.4504e-06 - val_accuracy: 1.0000
```

```

Epoch 11/20
155/155 [=====] - 681s 4s/step - loss: 0.0144 - accuracy: 0.9939 - val_loss: 0.0029 - val_accuracy: 1.0000
Epoch 12/20
155/155 [=====] - 670s 4s/step - loss: 0.0186 - accuracy: 0.9953 - val_loss: 0.1482 - val_accuracy: 0.9444
Epoch 13/20
155/155 [=====] - 676s 4s/step - loss: 0.0031 - accuracy: 0.9990 - val_loss: 6.0690e-05 - val_accuracy: 1.0000
Epoch 14/20
155/155 [=====] - 674s 4s/step - loss: 0.0032 - accuracy: 0.9993 - val_loss: 0.0010 - val_accuracy: 1.0000
Epoch 15/20
155/155 [=====] - 672s 4s/step - loss: 0.0015 - accuracy: 0.9997 - val_loss: 2.2451e-06 - val_accuracy: 1.0000
Epoch 16/20
155/155 [=====] - 673s 4s/step - loss: 0.0025 - accuracy: 0.9993 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 17/20
155/155 [=====] - 671s 4s/step - loss: 0.0132 - accuracy: 0.9961 - val_loss: 7.7554e-05 - val_accuracy: 1.0000
Epoch 18/20
155/155 [=====] - 675s 4s/step - loss: 0.0069 - accuracy: 0.9976 - val_loss: 0.0475 - val_accuracy: 0.9444
Epoch 19/20
155/155 [=====] - 673s 4s/step - loss: 0.0646 - accuracy: 0.9839 - val_loss: 0.0107 - val_accuracy: 1.0000
Epoch 20/20
155/155 [=====] - 676s 4s/step - loss: 0.0176 - accuracy: 0.9954 - val_loss: 5.1070e-05 - val_accuracy: 1.0000

```

Figure 5.6: Training and validation accuracy for MobileNet

### 5.4.2 Test Accuracy for MobileNet

After completing our training dataset training, we have applied our test dataset for getting the test accuracy. We got the average of 99% test accuracy that shows in the figure.

```

test_loss, test_acc = model.evaluate(test_set, verbose=2)
print('\nTest accuracy:', test_acc)

```

```

284/284 - 62s - loss: 0.0265 - accuracy: 0.9930

```

```

Test accuracy: 0.9929577708244324

```

Figure 5.7: Test accuracy for MobileNet

### 5.4.3 Confusion Matrix of MobileNet

Here is accurately identified Blackheart and fresh potatoes. There are no error data in these two classes. The common scab data of the test set predicted 86 images out of 87.

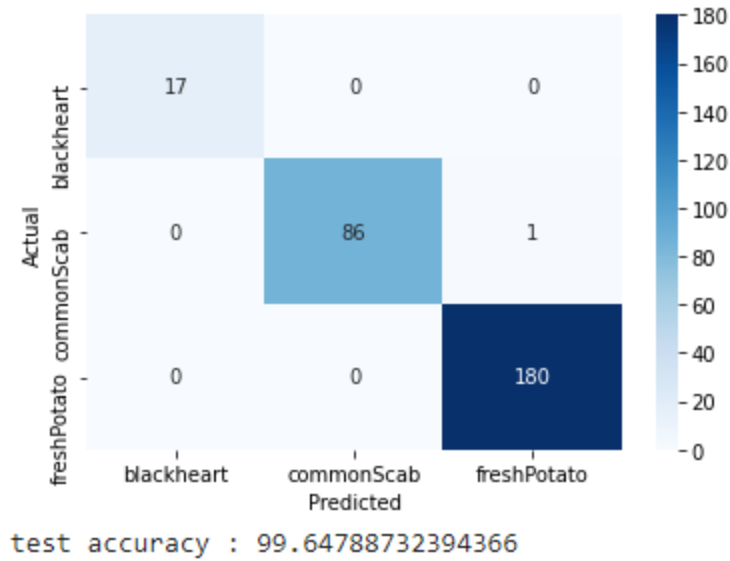


Figure 5.8: Confusion Matrix of MobileNet

#### 5.4.4 Diagram of training and validation accuracy for MobileNet

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training accuracy, and the orange line shows the validation accuracy.

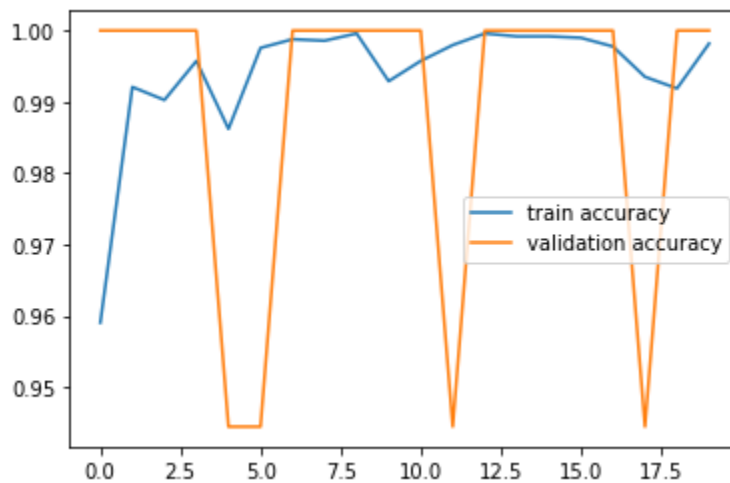


Figure 5.9: Diagram of training and validation accuracy for MobileNet



### 5.4.5 Diagram of training and validation Loss for MobileNet:

In twenty epochs for our dataset, each epoch shows the training loss and validation loss that are shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.

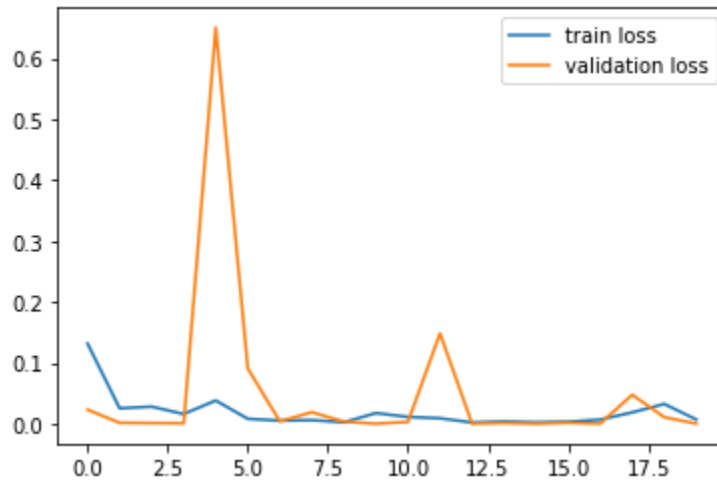


Figure 5.10: Diagram of training and validation loss for MobileNet

### 5.5 Result analysis for InceptionV3 Model:

In the InceptionV3 model, a total of 85% of image data is used for training, and 15% data is used for validation purposes. For testing, 5% of data is used. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In InceptionV3 model, total parameters are 23,904,035 and trainable parameters are 2,101,251 and Non-trainable parameters are 21,802,784.

## 5.5.1 Training and validation Accuracy

```
Epoch 1/20
155/155 [=====] - 2252s 14s/step - loss: 1.0468 - accuracy: 0.8222 - val_loss: 0.0076 - val_accuracy: 1.0000
Epoch 2/20
155/155 [=====] - 674s 4s/step - loss: 0.1143 - accuracy: 0.9618 - val_loss: 0.0842 - val_accuracy: 0.9444
Epoch 3/20
155/155 [=====] - 673s 4s/step - loss: 0.0753 - accuracy: 0.9709 - val_loss: 0.0076 - val_accuracy: 1.0000
Epoch 4/20
155/155 [=====] - 677s 4s/step - loss: 0.0365 - accuracy: 0.9903 - val_loss: 0.0167 - val_accuracy: 1.0000
Epoch 5/20
155/155 [=====] - 680s 4s/step - loss: 0.0618 - accuracy: 0.9737 - val_loss: 0.0054 - val_accuracy: 1.0000
Epoch 6/20
155/155 [=====] - 681s 4s/step - loss: 0.0486 - accuracy: 0.9824 - val_loss: 0.1451 - val_accuracy: 0.8889
Epoch 7/20
155/155 [=====] - 681s 4s/step - loss: 0.0337 - accuracy: 0.9884 - val_loss: 0.2353 - val_accuracy: 0.9444
Epoch 8/20
155/155 [=====] - 679s 4s/step - loss: 0.0228 - accuracy: 0.9916 - val_loss: 0.0011 - val_accuracy: 1.0000
Epoch 9/20
155/155 [=====] - 685s 4s/step - loss: 0.0247 - accuracy: 0.9911 - val_loss: 0.0366 - val_accuracy: 1.0000
Epoch 10/20
155/155 [=====] - 680s 4s/step - loss: 0.0288 - accuracy: 0.9884 - val_loss: 0.1230 - val_accuracy: 0.9444

Epoch 11/20
155/155 [=====] - 682s 4s/step - loss: 0.0297 - accuracy: 0.9900 - val_loss: 0.0035 - val_accuracy: 1.0000
Epoch 12/20
155/155 [=====] - 685s 4s/step - loss: 0.0228 - accuracy: 0.9925 - val_loss: 3.0934e-04 - val_accuracy: 1.0000
Epoch 13/20
155/155 [=====] - 684s 4s/step - loss: 0.0185 - accuracy: 0.9925 - val_loss: 0.0192 - val_accuracy: 1.0000
Epoch 14/20
155/155 [=====] - 689s 4s/step - loss: 0.0178 - accuracy: 0.9926 - val_loss: 0.0018 - val_accuracy: 1.0000
Epoch 15/20
155/155 [=====] - 688s 4s/step - loss: 0.0187 - accuracy: 0.9928 - val_loss: 4.3157e-04 - val_accuracy: 1.0000
Epoch 16/20
155/155 [=====] - 687s 4s/step - loss: 0.0235 - accuracy: 0.9912 - val_loss: 0.0087 - val_accuracy: 1.0000
Epoch 17/20
155/155 [=====] - 687s 4s/step - loss: 0.0443 - accuracy: 0.9819 - val_loss: 0.6913 - val_accuracy: 0.8889
Epoch 18/20
155/155 [=====] - 684s 4s/step - loss: 0.0149 - accuracy: 0.9936 - val_loss: 5.2277e-04 - val_accuracy: 1.0000
Epoch 19/20
155/155 [=====] - 691s 4s/step - loss: 0.0089 - accuracy: 0.9969 - val_loss: 0.0043 - val_accuracy: 1.0000
Epoch 20/20
155/155 [=====] - 687s 4s/step - loss: 0.0197 - accuracy: 0.9942 - val_loss: 1.5304e-04 - val_accuracy: 1.0000
```

Figure 5.11: Training and validation accuracy of Inception V3

### 5.5.2 Test Accuracy for Inception V3

After completing the training for training dataset then we have applied our test dataset for getting the actual accuracy. We got the 97% test accuracy that shows in the figure.

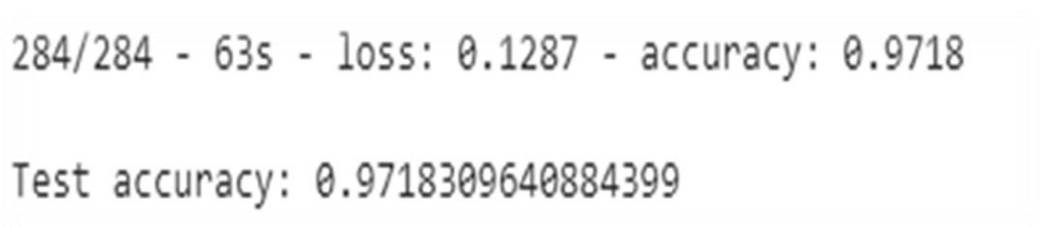


Figure 5.12: Test accuracy of Inception V3

### 5.5.3 Confusion Matrix for Inception V3

Here is accurately identified Blackheart and fresh potatoes. There are no error data in fresh potato classes. The test set is blackheart data, it predicted 14 images in total 17, and the common scab data of the test set predicted 81 images out of 87.

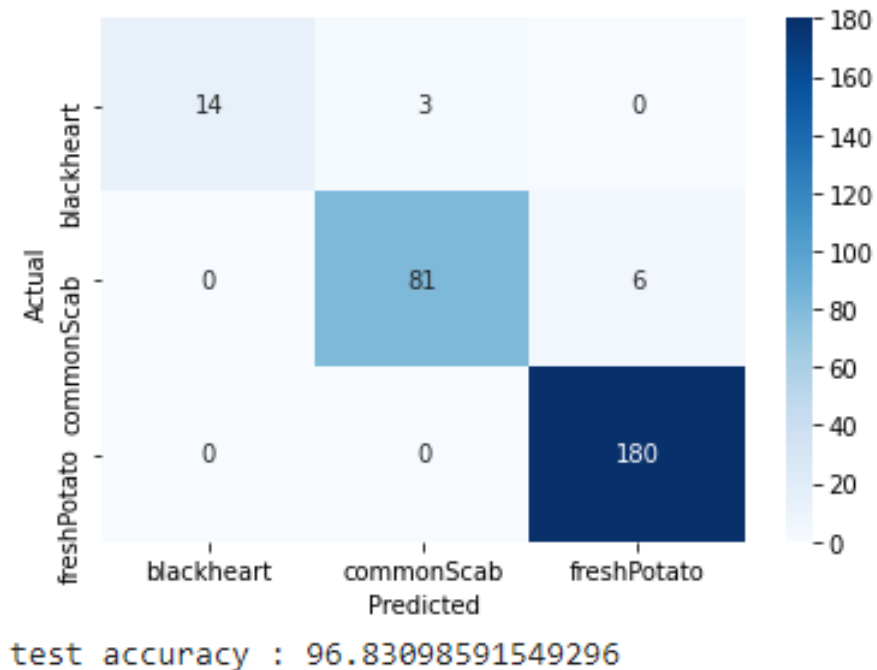


Figure 5.13: Confusion Matrix of Inception V3

### 5.5.4 Diagram of training and validation accuracy for Inception V3

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training accuracy, and the orange line shows the validation accuracy.

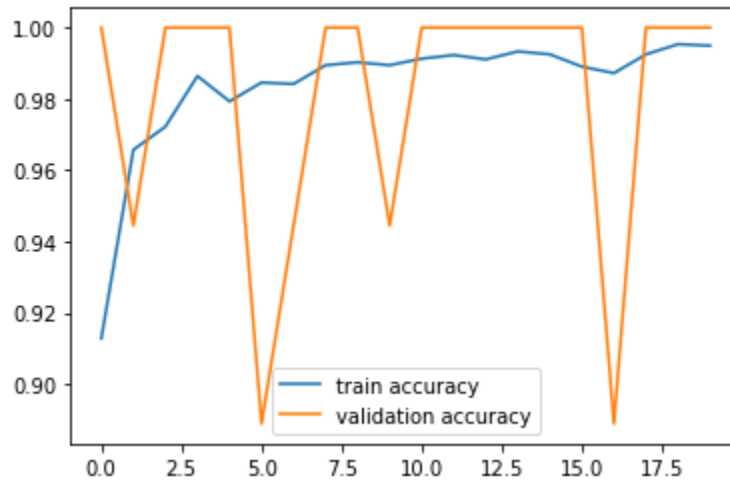


Figure 5.14: Diagram of training and validation accuracy for Inception V3

### 5.5.5 Diagram of training and validation Loss for Inception V3

In twenty epochs for our dataset, each epoch shows the training loss and validation loss shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.

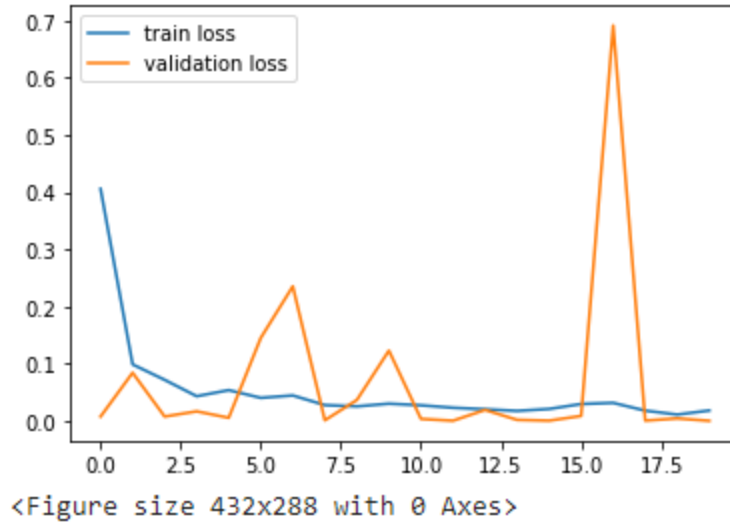


Figure 5.15: Diagram of training and validation loss for Inception V3

## 5.6 Result analysis for ResNet50 model

In the ResNet50 model, 85% of image data is used for training purposes. 10% data is used for validation purposes which are 356 images. 5% of data is used for testing purposes. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In this model, total parameters are 25,688,963 and trainable parameters are 2,101,251 and Non-trainable parameters are 23,587,71

### 5.6.1 Training and validation Accuracy for ResNet50

We run the program for 20 epochs. Per epoch shows the training loss, validation loss, training accuracy, validation accuracy. Each epoch shows the different accuracy of training and testing, which are shown in the figure.

```

Epoch 1/20
155/155 [=====] - 2072s 13s/step - loss: 0.9498 - accuracy: 0.5780 - val_loss: 0.9821 - val_accuracy: 0.5000
Epoch 2/20
155/155 [=====] - 685s 4s/step - loss: 0.8034 - accuracy: 0.6363 - val_loss: 0.9007 - val_accuracy: 0.5556
Epoch 3/20
155/155 [=====] - 684s 4s/step - loss: 0.7455 - accuracy: 0.6712 - val_loss: 0.6739 - val_accuracy: 0.7778
Epoch 4/20
155/155 [=====] - 684s 4s/step - loss: 0.7492 - accuracy: 0.6678 - val_loss: 0.7612 - val_accuracy: 0.7222
Epoch 5/20
155/155 [=====] - 684s 4s/step - loss: 0.7067 - accuracy: 0.6812 - val_loss: 0.6388 - val_accuracy: 0.6667
Epoch 6/20
155/155 [=====] - 684s 4s/step - loss: 0.6864 - accuracy: 0.7017 - val_loss: 0.8040 - val_accuracy: 0.5556
Epoch 7/20
155/155 [=====] - 680s 4s/step - loss: 0.6831 - accuracy: 0.7033 - val_loss: 0.6154 - val_accuracy: 0.7222
Epoch 8/20
155/155 [=====] - 681s 4s/step - loss: 0.6667 - accuracy: 0.7168 - val_loss: 0.5026 - val_accuracy: 0.7778
Epoch 9/20
155/155 [=====] - 684s 4s/step - loss: 0.6980 - accuracy: 0.6942 - val_loss: 0.6115 - val_accuracy: 0.7222
Epoch 10/20
155/155 [=====] - 682s 4s/step - loss: 0.6457 - accuracy: 0.7285 - val_loss: 0.5325 - val_accuracy: 0.8333
- . . . .-

Epoch 11/20
155/155 [=====] - 681s 4s/step - loss: 0.6658 - accuracy: 0.7131 - val_loss: 0.6274 - val_accuracy: 0.7778
Epoch 12/20
155/155 [=====] - 679s 4s/step - loss: 0.6138 - accuracy: 0.7397 - val_loss: 0.8121 - val_accuracy: 0.6667
Epoch 13/20
155/155 [=====] - 681s 4s/step - loss: 0.6390 - accuracy: 0.7241 - val_loss: 0.6389 - val_accuracy: 0.7222
Epoch 14/20
155/155 [=====] - 680s 4s/step - loss: 0.6404 - accuracy: 0.7298 - val_loss: 0.4211 - val_accuracy: 0.8333
Epoch 15/20
155/155 [=====] - 679s 4s/step - loss: 0.6170 - accuracy: 0.7465 - val_loss: 0.5165 - val_accuracy: 0.7778
Epoch 16/20
155/155 [=====] - 679s 4s/step - loss: 0.6435 - accuracy: 0.7174 - val_loss: 0.4782 - val_accuracy: 0.7778
Epoch 17/20
155/155 [=====] - 683s 4s/step - loss: 0.5961 - accuracy: 0.7532 - val_loss: 0.3882 - val_accuracy: 0.7778
Epoch 18/20
155/155 [=====] - 682s 4s/step - loss: 0.5898 - accuracy: 0.7547 - val_loss: 0.3738 - val_accuracy: 0.8333
Epoch 19/20
155/155 [=====] - 681s 4s/step - loss: 0.6185 - accuracy: 0.7429 - val_loss: 0.4883 - val_accuracy: 0.7222
Epoch 20/20
155/155 [=====] - 680s 4s/step - loss: 0.5878 - accuracy: 0.7560 - val_loss: 0.5327 - val_accuracy: 0.7778

```

Figure 5.16: Training and validation accuracy for ResNet50

### 5.6.2 Test Accuracy for ResNet50

After completing our training dataset training, we have applied our test dataset for getting the test accuracy. We got the average of 84% test accuracy that shows in the figure.

```
284/284 - 59s - loss: 0.5315 - accuracy: 0.8451
```

```
Test accuracy: 0.8450704216957092
```

Figure 5.17: Test accuracy for ResNet50

### 5.6.3 Confusion Matrix of ResNet50

The confusion matrix shows some errors of fresh potato, common scab, Blackheart. The blackheart data of the test set predicted four images out of 17. The common scab data of the test set predicted 68 images out of 87. The fresh potato data of the test set predicted 159 images out of 180.

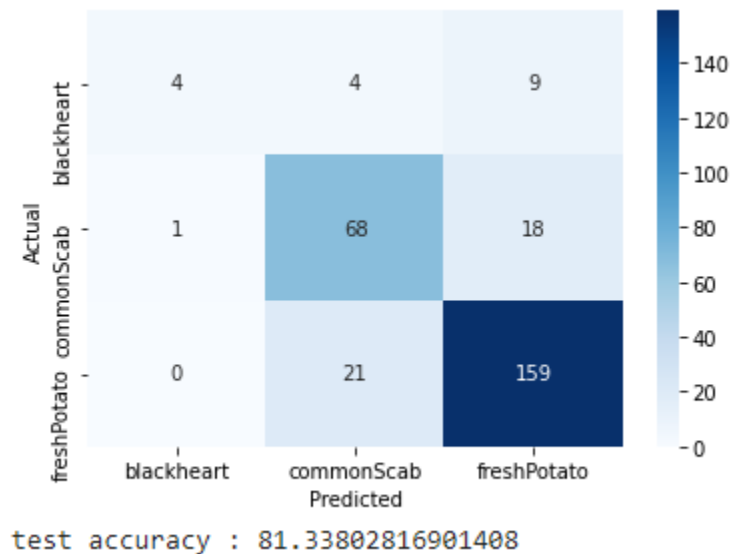


Figure 5.18: Confusion Matrix of ResNet50

### 5.6.4 Diagram of training and validation accuracy for ResNet50

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training, and the orange line shows the validation accuracy.

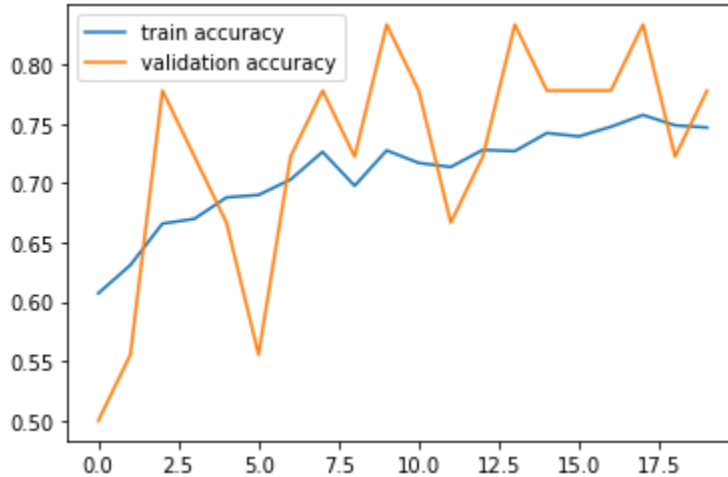


Figure 5.19: Diagram of training and validation accuracy for ResNet50

### 5.6.5 Diagram of training and validation Loss for ResNet50

In twenty epochs for our dataset, each epoch shows the training loss and validation loss shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.



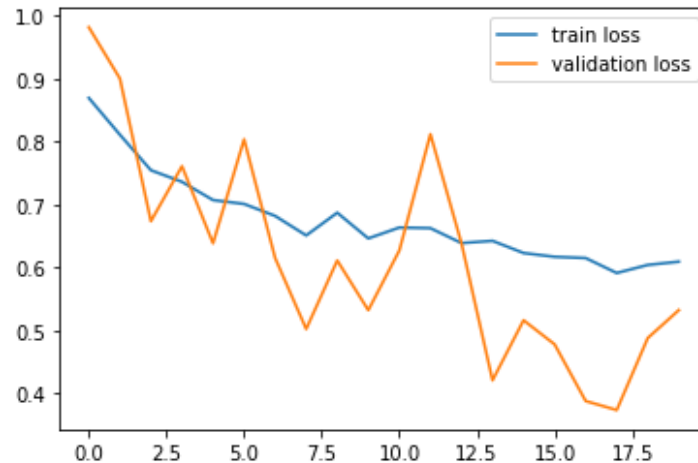


Figure 5.20: Diagram of training and validation loss for ResNet50

## 5.7 Result analysis for Xception model

In the Xception model, 85% of image data is used for training purposes. 10% data is used for validation purposes which are 356 images. 5% of data is used testing purposes. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In this model, total parameters are 22,962,731 and trainable parameters are 2,101,251 and Non-trainable parameters are 20,861,480

### 5.7.1 Training and validation Accuracy for Xception

We run the program for 15 epochs. Per epoch shows the training loss, validation loss, training accuracy, validation accuracy. Each epoch shows the different accuracy of training and testing, which are shown in the figure.

```

Epoch 1/15
155/155 [=====] - 2208s 14s/step - loss: 0.4087 - accuracy: 0.8781 - val_loss: 0.0288 - val_accuracy: 1.0000
Epoch 2/15
155/155 [=====] - 695s 4s/step - loss: 0.0462 - accuracy: 0.9831 - val_loss: 0.0190 - val_accuracy: 1.0000
Epoch 3/15
155/155 [=====] - 693s 4s/step - loss: 0.0724 - accuracy: 0.9771 - val_loss: 0.0097 - val_accuracy: 1.0000
Epoch 4/15
155/155 [=====] - 698s 5s/step - loss: 0.0309 - accuracy: 0.9907 - val_loss: 0.0284 - val_accuracy: 1.0000
Epoch 5/15
155/155 [=====] - 694s 4s/step - loss: 0.0256 - accuracy: 0.9910 - val_loss: 0.0046 - val_accuracy: 1.0000
Epoch 6/15
155/155 [=====] - 693s 4s/step - loss: 0.0267 - accuracy: 0.9868 - val_loss: 0.0024 - val_accuracy: 1.0000
Epoch 7/15
155/155 [=====] - 692s 4s/step - loss: 0.0245 - accuracy: 0.9914 - val_loss: 0.1750 - val_accuracy: 0.8889
Epoch 8/15
155/155 [=====] - 693s 4s/step - loss: 0.0132 - accuracy: 0.9947 - val_loss: 0.0011 - val_accuracy: 1.0000
Epoch 9/15
155/155 [=====] - 692s 4s/step - loss: 0.0131 - accuracy: 0.9955 - val_loss: 0.0265 - val_accuracy: 1.0000
Epoch 10/15
155/155 [=====] - 693s 4s/step - loss: 0.0308 - accuracy: 0.9906 - val_loss: 2.1591e-04 - val_accuracy: 1.0000

Epoch 11/15
155/155 [=====] - 690s 4s/step - loss: 0.0195 - accuracy: 0.9925 - val_loss: 0.4455 - val_accuracy: 0.9444
Epoch 12/15
155/155 [=====] - 693s 4s/step - loss: 0.0132 - accuracy: 0.9958 - val_loss: 0.0051 - val_accuracy: 1.0000
Epoch 13/15
155/155 [=====] - 690s 4s/step - loss: 0.0148 - accuracy: 0.9941 - val_loss: 9.5977e-05 - val_accuracy: 1.0000
Epoch 14/15
155/155 [=====] - 690s 4s/step - loss: 0.0096 - accuracy: 0.9973 - val_loss: 1.3755e-05 - val_accuracy: 1.0000
Epoch 15/15
155/155 [=====] - 689s 4s/step - loss: 0.0098 - accuracy: 0.9965 - val_loss: 0.0084 - val_accuracy: 1.0000

```

Figure 5.21: Training and validation accuracy for Xception

## 5.7.2 Test Accuracy for Xception

After completing our training dataset training, we have applied our test dataset for getting the test accuracy. We got the average of 99% test accuracy that shows in the figure.

```

284/284 - 40s - loss: 0.0111 - accuracy: 0.9930

Test accuracy: 0.9929577708244324

```

Figure 5.22: Test accuracy for Xception

### 5.7.3 Confusion Matrix of Xception

Here is accurately identified Blackheart and fresh potatoes. There are no error data in these two classes. In the common scab data of the test set, It predicted 85 images out of 87.

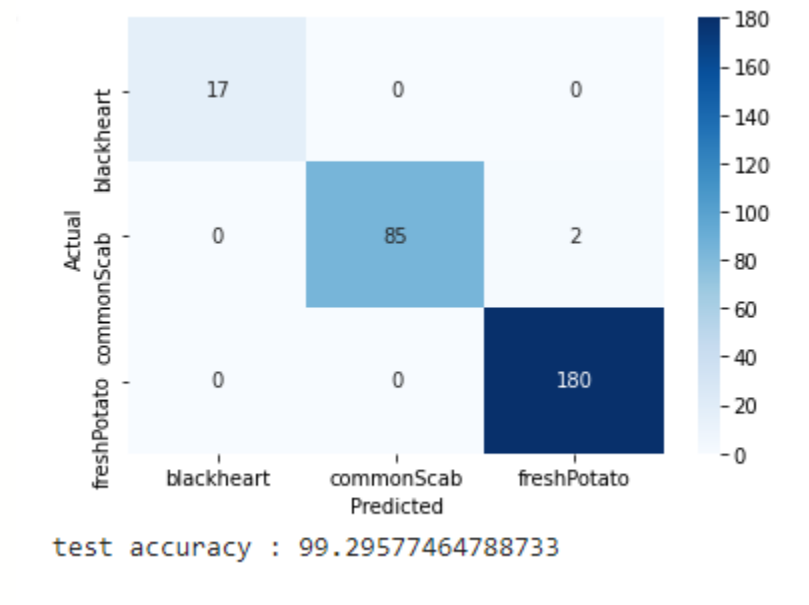


Figure 5.23: Confusion Matrix of Xception

### 5.7.4 Diagram of training and validation accuracy for Xception

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training, and the orange line shows the validation accuracy.

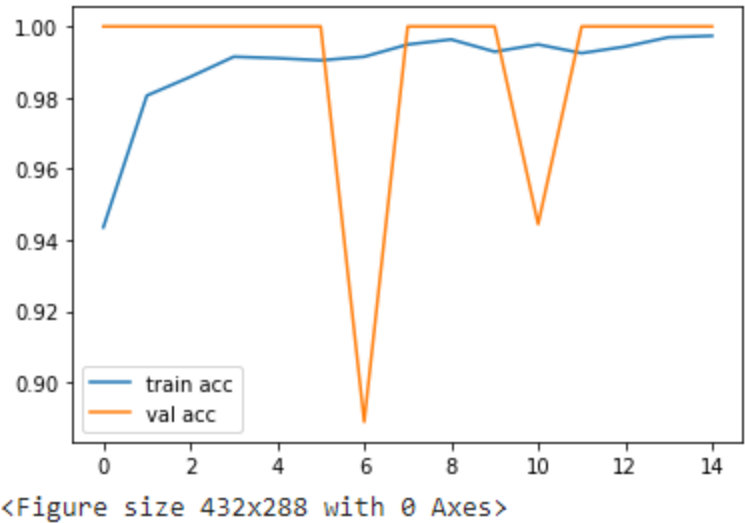


Figure 5.24: Diagram of training and validation accuracy for Xception

### 5.7.5 Diagram of training and validation Loss for Xception

In twenty epochs for our dataset, each epoch shows the training loss and validation loss shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.

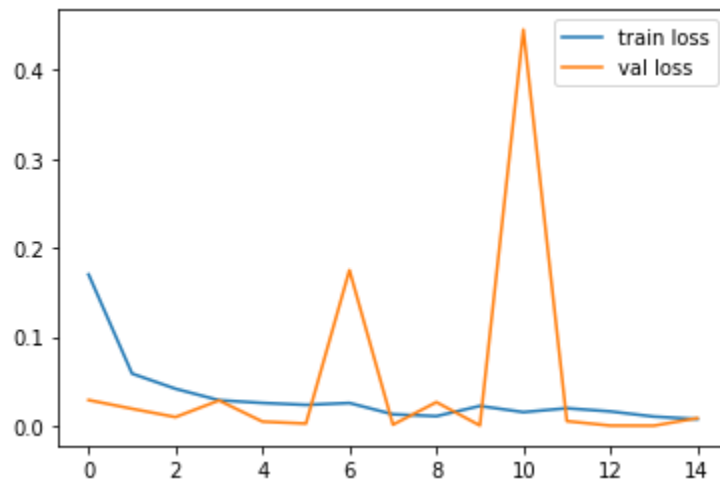


Figure 5.25: Diagram of training and validation loss for Xception

## 5.8 Result analysis for VGG-16 model

In the VGG-16 model, 85% of image data is used for training purposes. 10% data is used for validation purposes which are 356 images. 5% of data is used testing purposes. In total, 5781 images are used where 4,931 images are used for training, 566 images are used for validation, and 284 images are used for testing.

In this model, total parameters are 15,243,075 and trainable parameters are 528,387 and Non-trainable parameters are 14,714,688

### 5.8.1 Training and validation Accuracy for VGG-16

We run the program for 15 epochs. Per epoch shows the training loss, validation loss, training accuracy, validation accuracy. Each epoch shows the different accuracy of training and testing which are shown in the figure.

```
Epoch 1/15
155/155 [=====] - 2152s 14s/step - loss: 0.6435 - accuracy: 0.7496 - val_loss: 0.2282 - val_accuracy: 0.9444
Epoch 2/15
155/155 [=====] - 690s 4s/step - loss: 0.2724 - accuracy: 0.8992 - val_loss: 0.1904 - val_accuracy: 0.9444
Epoch 3/15
155/155 [=====] - 691s 4s/step - loss: 0.1803 - accuracy: 0.9383 - val_loss: 0.1542 - val_accuracy: 0.8889
Epoch 4/15
155/155 [=====] - 686s 4s/step - loss: 0.1552 - accuracy: 0.9485 - val_loss: 0.0485 - val_accuracy: 1.0000
Epoch 5/15
155/155 [=====] - 684s 4s/step - loss: 0.1345 - accuracy: 0.9525 - val_loss: 0.1834 - val_accuracy: 0.9444
Epoch 6/15
155/155 [=====] - 683s 4s/step - loss: 0.1050 - accuracy: 0.9664 - val_loss: 0.0754 - val_accuracy: 0.9444
Epoch 7/15
155/155 [=====] - 683s 4s/step - loss: 0.1113 - accuracy: 0.9566 - val_loss: 0.0213 - val_accuracy: 1.0000
Epoch 8/15
155/155 [=====] - 683s 4s/step - loss: 0.0730 - accuracy: 0.9781 - val_loss: 0.0046 - val_accuracy: 1.0000
Epoch 9/15
155/155 [=====] - 685s 4s/step - loss: 0.0852 - accuracy: 0.9702 - val_loss: 0.2271 - val_accuracy: 0.8889
Epoch 10/15
155/155 [=====] - 684s 4s/step - loss: 0.0778 - accuracy: 0.9715 - val_loss: 0.0365 - val_accuracy: 1.0000
```

```

Epoch 11/15
155/155 [=====] - 684s 4s/step - loss: 0.0786 - accuracy: 0.9707 - val_loss: 0.1561 - val_accuracy: 0.9444
Epoch 12/15
155/155 [=====] - 687s 4s/step - loss: 0.0870 - accuracy: 0.9707 - val_loss: 0.0120 - val_accuracy: 1.0000
Epoch 13/15
155/155 [=====] - 685s 4s/step - loss: 0.0777 - accuracy: 0.9716 - val_loss: 0.0071 - val_accuracy: 1.0000
Epoch 14/15
155/155 [=====] - 684s 4s/step - loss: 0.0709 - accuracy: 0.9739 - val_loss: 0.0103 - val_accuracy: 1.0000
Epoch 15/15
155/155 [=====] - 691s 4s/step - loss: 0.0640 - accuracy: 0.9771 - val_loss: 0.0063 - val_accuracy: 1.0000

```

Figure 5.26: Training and validation accuracy for VGG-16

## 5.8.2 Test Accuracy for VGG-16

After completing our training dataset training, we have applied our test dataset for getting the test accuracy. We got the average of 98% test accuracy that shows in the figure.

```

284/284 - 69s - loss: 0.0591 - accuracy: 0.9859

Test accuracy: 0.98591548204422

```

Figure 5.27: Test accuracy for VGG-16

## 5.8.3 Confusion Matrix of VGG-16

The confusion matrix shows some errors of fresh potato, common scab, Blackheart. The blackheart data of the test set predicted 16 images out of 17. The common scab data of the test set predicted 86 images out of 87. The fresh potato data of the test set predicted 177 images out of 180.

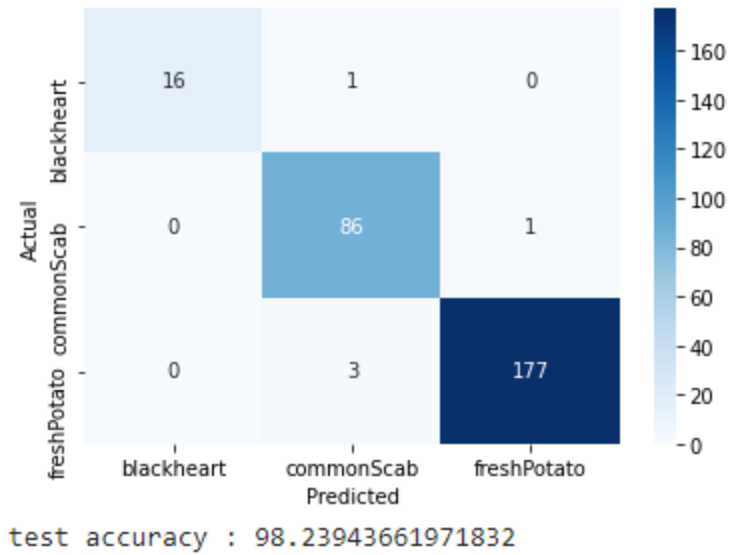


Figure 5.28: Confusion Matrix of VGG-16

### 5.8.4 Diagram of training and validation accuracy for VGG-16

In twenty epochs for our dataset, each epoch shows the training accuracy and validation accuracy shown in the given diagram. Sky blue line shows the training, and the orange line shows the validation accuracy.

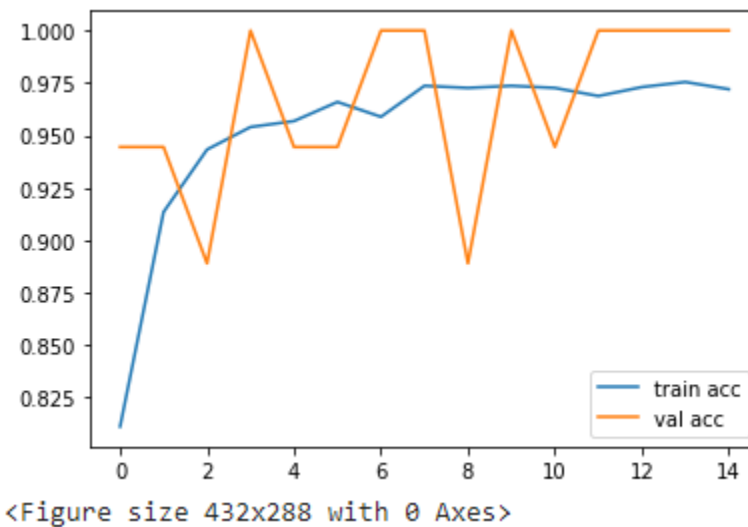


Figure 5.29: Diagram of training and validation accuracy for VGG-16

### 5.8.5 Diagram of training and validation Loss for VGG-16

In twenty epochs for our dataset, each epoch shows the training loss and validation loss shown in the given diagram. Sky blue line shows the training loss, and the orange line shows the validation loss.

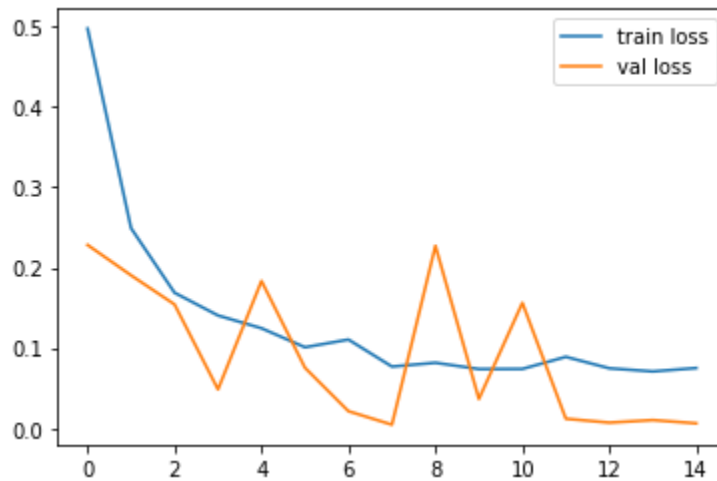


Figure 5.30: Diagram of training and validation loss for VGG-16

## 5.9 Discussion

We observed the results and obtained that after increasing the epochs in the model, we gained more training and validation accuracy. We have applied 20th epochs VGG19, Resnet 50, Mobilenet, Inception v3 models and applied 15th epochs VGG19 and Xception models.

From analyzing the six models, the MobileNet has predicted most of the images. Here find the fewer error data from all of the other models. We find the lowest accuracy from the Resnet 50 model. The accuracy from the Resnet 50 model is 84%. We get the best accuracy from the MobileNet model. It achieved an accuracy average of 99%.



## 5.10 Performance Comparison

Performance comparison of six CNN models is given below. In our dataset, MobileNet architecture gives the best accuracy. MobileNet achieved 99% accuracy.

Table 5.5: Accuracy of our CNN models

Sl. No	Model Name	Epochs	Accuracy (%)	Training Image	Test Image	Validation Image	Total Images
1	VGG-19	20	0.98	85%	5%	10%	5781
2	MobileNet	20	0.99	85%	5%	10%	5781
3	Inception V3	20	0.97	85%	5%	10%	5781
4	ResNet50	20	0.84	85%	5%	10%	5781
5	Xception	15	0.99	85%	5%	10%	5781
6	VGG16	15	0.98	85%	5%	10%	5781

## CHAPTER 6

### CONCLUSION, RECOMMENDATION AND FUTURE WORK

#### 6.1 Conclusion

There is a shortage of food day by day, and we can overcome this problem of food shortages by using modern technology. We can use the machine to detect problems of foods. We can use the artificial brain of a computer here. Machine learning is one such medium. Today machine learning is an important part of computer science, and Convulsive neural networks are a part of machine learning.

Following this, we have set up a Convolutional Neural Network model to predict the diagnose of potatoes. Where using six several models and got our desired results, for this process, we created the necessary dataset of potatoes. We have described details about them inside this paper. Among them, Mobile Net was the best.

We hoped that this work would further encourage researchers. This type of work will help in overcome the upcoming food shortage.

#### 6.2 Recommendation

we discussed six architecture in our Research paper. MobileNet architecture has made the best predictions among these six architectures because MobileNet architecture has predicted all the images except one image. Of this architecture test accuracy quantity 99.29 % out of 100 % on our dataset. So we recommend MobileNet architectures.

#### 6.3 Future work

Now we have captured the two types of diseases and a fresh potato which is why there are three classes in our dataset. In the future, we can increase more than three classes of the dataset by taking more diseases of the potatoes. We can get more accuracy for using more

classes. We can increase the pictures without augmentation and without taking images from any website. We can apply our dataset to more than five algorithms and compare the accuracy among five algorithms. We can apply this deep learning process to other datasets to classify the images.

## References

- [1] B. Zhang, W. Huang, J. Li, C. Zhao, S. Fan, J. Wu, and C. Liu, "Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review," *Food Research International*, vol. 62, pp. 326–343, 2014.
- [2] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "Deepfruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, 2016.
- [3] S. Cubero, W. S. Lee, N. Aleixos, F. Albert, and J. Blasco, "Automated systems based on machine vision for inspecting citrus fruits from the field to postharvest—a review," *Food and Bioprocess Technology*, vol. 9, no. 10, pp. 1623–1639, 2016.
- [4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [5] D. Oppenheim, G. Shani, O. Erlich, and L. Tsrur, "Using deep learning for image-based potato tuber disease detection," *Phytopathology*, vol. 109, no. 6, pp. 1083–1087, 2019.
- [6] A. A. Elsharif, I. M. Dheir, A. S. A. Mettleq, and S. S. Abu-Naser, "Potato classification using deep learning," 2020.
- [7] M. Islam, A. Dinh, K. Wahid, and P. Bhowmik, "Detection of potato diseases using image segmentation and multiclass support vector machine," in *2017 IEEE 30th Canadian conference on electrical and computer engineering (CCECE)*. IEEE, 2017, pp. 1–4.
- [8] D. Oppenheim and G. Shani, "Potato disease classification using convolution neural networks," *Advances in Animal Biosciences*, vol. 8, no. 2, p. 244, 2017.
- [9] P. Patil, N. Yaligar, and S. Meena, "Comparision of performance of classifiers-svm, rf and ann in potato blight disease detection using leaf images," in *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. IEEE, 2017, pp. 1–5.
- [10] T. Guo, J. Dong, H. Li, and Y. Gao, "Simple convolutional neuralnetwork on image classification," in *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*(. IEEE, 2017, pp. 721–724.

# Detection of Potato Diseases using Deep Learning

---

## ORIGINALITY REPORT

---

23%

SIMILARITY INDEX

16%

INTERNET SOURCES

15%

PUBLICATIONS

12%

STUDENT PAPERS

---

## PRIMARY SOURCES

---

1

[dspace.daffodilvarsity.edu.bd:8080](https://dspace.daffodilvarsity.edu.bd:8080)

Internet Source

5%

2

Submitted to Daffodil International University

Student Paper

4%

3

Sharvari Adiga, DV Vaishnavi, Suchitra Saxena, Shikha Tripathi. "Multimodal Emotion Recognition for Human Robot Interaction", 2020 7th International Conference on Soft Computing & Machine Intelligence (ISCM), 2020

Publication

2%

4

Anu Paulson, S Ravishankar. "AI Based Indigenous Medicinal Plant Identification", 2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA), 2020

Publication

1%

5

[export.arxiv.org](https://export.arxiv.org)

Internet Source

1%

6

[dokumen.pub](https://dokumen.pub)

Internet Source

1%

---

7	"Proceedings of International Conference on Computational Intelligence, Data Science and Cloud Computing", Springer Science and Business Media LLC, 2021 Publication	1 %
8	Submitted to Delhi Technological University Student Paper	1 %
9	"Cybernetics, Cognition and Machine Learning Applications", Springer Science and Business Media LLC, 2021 Publication	1 %
10	<a href="https://www.insightsociety.org">insightsociety.org</a> Internet Source	<1 %
11	Nikos Petrellis. "Plant Disease Diagnosis with Color Normalization", 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCASST), 2019 Publication	<1 %
12	<a href="https://journal.uad.ac.id">journal.uad.ac.id</a> Internet Source	<1 %
13	S. Nithya Tanvi Nishitha, Shahana Bano, G. Greeshmanth Reddy, Pujitha Arja, Gorsa Lakshmi Niharika. "Stock Price Prognosticator using Machine Learning Techniques", 2020 4th International Conference on Electronics,	<1 %

---

# Communication and Aerospace Technology (ICECA), 2020

Publication

14

Yashwant Kurmi, Suchi Gangwar, Dheeraj Agrawal, Satrughan Kumar, Deepratna Saxena, Moly Saxena, Harishanker Shrivastava. "An Algorithm for Various Crop Diseases Detection and Classification using Leaves Images", 2nd International Conference on Data, Engineering and Applications (IDEA), 2020

Publication

<1 %

15

[arxiv.org](https://arxiv.org)  
Internet Source

<1 %

16

[repositorio.ufscar.br](https://repositorio.ufscar.br)  
Internet Source

<1 %

17

[www.hindawi.com](https://www.hindawi.com)  
Internet Source

<1 %

18

"Computer Vision and Image Processing", Springer Science and Business Media LLC, 2021

Publication

<1 %

19

[spj.sciencemag.org](https://spj.sciencemag.org)  
Internet Source

<1 %

20

[ocs.unud.ac.id](https://ocs.unud.ac.id)  
Internet Source

<1 %

21

[www.ijrte.org](http://www.ijrte.org)

Internet Source

<1 %

22

"Machine Intelligence and Smart Systems",  
Springer Science and Business Media LLC,  
2021

Publication

<1 %

23

Daniela Onita, Adriana Birlutiu, Liviu P. Dinu.  
"Towards Mapping Images to Text Using  
Deep-Learning Architectures", Mathematics,  
2020

Publication

<1 %

24

"Proceedings of International Joint Conference  
on Computational Intelligence", Springer  
Science and Business Media LLC, 2020

Publication

<1 %

25

Submitted to Ashesi University

Student Paper

<1 %

26

Sofia Marino, Pierre Beausery, André  
Smolarz. "Weakly-supervised learning  
approach for potato defects segmentation",  
Engineering Applications of Artificial  
Intelligence, 2019

Publication

<1 %

27

"Proceeding of the International Conference  
on Computer Networks, Big Data and IoT  
(ICCBI - 2019)", Springer Science and Business  
Media LLC, 2020

<1 %



28 Submitted to TechKnowledge <1 %  
Student Paper

---

29 Submitted to Rochester Institute of Technology <1 %  
Student Paper

---

30 Submitted to Liverpool John Moores University <1 %  
Student Paper

---

31 ceur-ws.org <1 %  
Internet Source

---

32 Nadiya Shvai, Antoine Meicler, Abul Hasnat, Edouard Machover, Paul Maarek, Stephane Loquet, Amir Nakib. "Optimal Ensemble Classifiers Based Classification for Automatic Vehicle Type Recognition", 2018 IEEE Congress on Evolutionary Computation (CEC), 2018 <1 %  
Publication

---

33 greenworkforce.in <1 %  
Internet Source

---

34 Md. Arifur Rahman, Md. Mukitul Islam, G M Shahir Mahdee, Md. Wasi Ul Kabir. "Improved Segmentation Approach for Plant Disease Detection", 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), 2019 <1 %

35

Utpal Barman, Ridip Dev Choudhury, Diganto Sahu, Golap Gunjan Barman. "Comparison of convolution neural networks for smartphone image based real time classification of citrus leaf disease", Computers and Electronics in Agriculture, 2020

Publication

<1 %

---

36

[dsl.serc.iisc.ernet.in](http://dsl.serc.iisc.ernet.in)

Internet Source

<1 %

---

37

[ijsab.com](http://ijsab.com)

Internet Source

<1 %

---

38

[juniperpublishers.com](http://juniperpublishers.com)

Internet Source

<1 %

---

39

[www.ijsta.com](http://www.ijsta.com)

Internet Source

<1 %

---

40

[www.mdpi.com](http://www.mdpi.com)

Internet Source

<1 %

---

41

Rajasekaran Thangaraj, Pandiyan P, Vishnu Kumar Kaliappan, Anandamurugan S, Indupriya P. "Potato Leaf Disease Classification using Transfer Learning based Modified Xception Model", Innovations in Information and Communication Technology Series, 2020

Publication

<1 %

---

42	<a href="https://dspace.cuni.cz">dspace.cuni.cz</a> Internet Source	<1 %
43	<a href="https://houseofbots.com">houseofbots.com</a> Internet Source	<1 %
44	<a href="https://monarch.qucosa.de">monarch.qucosa.de</a> Internet Source	<1 %
45	<a href="https://publications.waset.org">publications.waset.org</a> Internet Source	<1 %
46	"Intelligent Systems and Applications", Springer Science and Business Media LLC, 2021 Publication	<1 %
47	Lecture Notes in Computer Science, 2015. Publication	<1 %

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off