

# Cassava Leaf Disease Classification using Deep Learning and Convolutional Neural Network Ensemble

by

Hasan Shahriar

20301476

Protick Sarker Shuvo

16301078

Md. Saidul Haque Fahim

16301053

Md Sobuj Sordar

16201032

Md Esadul Haque

17301148

A thesis submitted to the Department of Computer Science and Engineering  
in partial fulfillment of the requirements for the degree of  
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering  
Brac University  
January 2022

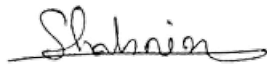
© 2022. Brac University  
All rights reserved.

# Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**



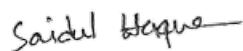
---

Hasan Shahriar  
20301476



---

Protick Sarker Shuvo  
16301078



---

Md. Saidul Haque Fahim  
16301053



---

Md Sobuj Sordar  
16201032



---

Md Esadul Haque  
17301148

# Approval

The thesis/project titled “Cassava Leaf Disease Classification using Deep Learning and Convolutional Neural Network Ensemble” submitted by

1. Hasan Shahriar(20301476)
2. Protick Sarker Shuvo(16301078)
3. Md. Saidul Haque Fahim(16301053)
4. Md Sobuj Sordar(16201032)
5. Md Esadul Haque(17301148)

Of Fall, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 16, 2022.

## Examining Committee:

Supervisor:  
(Member)



---

Amitabha Chakrabarty, PhD  
Associate Professor  
Department of Computer Science and Engineering  
Brac University

Program Coordinator:  
(Member)

---

Dr. Golam Rabiul Alam  
Associate Professor  
Department of Computer Science and Engineering  
BRAC University

Head of Department:  
(Chair)

---

Sadia Hamid Kazi, PhD  
Chairperson and Associate Professor  
Department of Computer Science and Engineering  
Brac University

## **Ethics Statement (Optional)**

This is optional, if you don't have an ethics statement then omit this page

# Abstract

Cassava is a high-protein and nutrient-dense plant, notably inside the leaves. Cassava is often used as a rice alternative. Pests, viruses, bacteria, and fungus may cause a variety of illnesses on cassava leaves. This study consists of four main diseases that commonly affect cassava leaves: Cassava Bacterial Blight (CBB), Cassava Brown Streak Disease (CBSD), Cassava Green Mite (CGM), and Cassava Mosaic Disease (CMD) and we took these four diseases as labels in our research. Furthermore, we took 22000 infected images from Kaggle and we have transformed our dataset into four different image transformation to ensure the accuracy of our model. These four different augmentations are Random Crop Augmentation, Random Flip Augmentation, Random Rotation Augmentation and Random Contrast Augmentation. Finally, we used six algorithms to detect the diseases of cassava leaves. These six algorithms are Xception, EfficientNetB0 Resnet50, VGG16 Densenet121, InceptionV3. While we operated these algorithms on our trained dataset, it gave diverse precision. For the Xception, it gave 91.3% accuracy, EfficientNetB0:91.1%, ResNet50: 85.0 %, VGG16: 68.0 %, DenseNet121: 87.0 % and for the InceptionV3, it gave 86.4 % precision respectively. Here, not every one of the algorithms performed well. Xception and EfficientNetB0 have the most noteworthy accuracy among these.

**Keywords:** Machine learning; Deep Learning; Cassava leaf; Prediction; Decision tree; Xception, Neural Network, EfficientNet B0, Resnet 50, VGG16, Inception V3, DenseNet 121

## Dedication (Optional)

A dedication is the expression of friendly connection or thanks by the author towards another person. It can occupy one or multiple lines depending on its importance. You can remove this page if you want.

## Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis has been completed without any major interruption. Secondly, to our advisor Amitabha Chakrabarty, PhD sir for his kind support and advice in our work. He helped us whenever we needed help. And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

# Table of Contents

Declaration	i
Approval	ii
Ethics Statement	iii
Abstract	iv
Dedication	v
Acknowledgment	vi
Table of Contents	vii
List of Figures	ix
List of Tables	x
Nomenclature	xii
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Research Objective . . . . .	3
1.3 Research Problem . . . . .	4
1.4 Contribution and Impact . . . . .	4
1.5 Scopes and Limitations . . . . .	5
1.6 Documentation Outline . . . . .	5
<b>2 Literature Review and Related Work</b>	<b>6</b>
<b>3 Data Processing</b>	<b>10</b>
3.1 Data Structure . . . . .	10
3.2 Data Pre-processing . . . . .	13
3.2.1 Resized Image . . . . .	13
3.2.2 Divided into Batches . . . . .	14
3.2.3 Image Transformation . . . . .	15
3.2.4 Random Crop Augmentation . . . . .	15
3.2.5 Random Flip Augmentation . . . . .	15
3.2.6 Random Rotation Augmentation . . . . .	15
3.2.7 Random Contrast Augmentation . . . . .	15



<b>4</b>	<b>Methodology</b>	<b>17</b>
4.1	Xception . . . . .	17
4.2	VGG16 . . . . .	18
4.3	InceptionV3 . . . . .	19
4.4	ResNet-50 . . . . .	21
4.5	EfficientNetB0 . . . . .	22
4.6	DenseNet-121 . . . . .	23
4.7	Model Compilation . . . . .	23
4.7.1	Loss Function . . . . .	24
4.7.2	Metric . . . . .	24
4.7.3	Optimizer . . . . .	24
<b>5</b>	<b>Result and Analysis</b>	<b>25</b>
5.1	Confusion Matrix . . . . .	30
<b>6</b>	<b>Conclusion and Future Scope</b>	<b>31</b>
6.0.1	Future Work . . . . .	31
	<b>Bibliography</b>	<b>35</b>
	<b>Appendix A Adam optimization pseudocode</b>	<b>36</b>

# List of Figures

3.1	Cassava Bacterial Blight (CBB)	11
3.2	Cassava Brown Streak Disease (CBSD)	12
3.3	Cassava Green Mottle (CGM)	12
3.4	Cassava Mosaic Disease (CMD)	13
3.5	Cassava Healthy Leaf	14
4.1	Xception architecture	17
4.2	Model accuracy and Model loss of Xception	18
4.3	VGG layers	19
4.4	VGG 16	19
4.5	Model accuracy and Model loss of VGG-16	20
4.6	Replacing one 5x5 convolution with two 3x3 convolutions	20
4.7	Schematic diagram of Inception-v3	21
4.8	Schematic diagram of ResNet-50 architecture	21
4.9	Model accuracy and Model loss of ResNet50	22
4.10	Basic block diagram of EfficientNet model	23
4.11	Model accuracy and Model loss of ResNet50	23
5.1	Model accuracy of Xception	27
5.2	Model accuracy of EfficientNet B0	27
5.3	Model accuracy of Resnet 50	28
5.4	Model accuracy of VGG16	28
5.5	Model accuracy of Densenet 121	29
5.6	Model accuracy of Inception V3	29
5.7	Comparison of validation accuracies of all models	30
5.8	Confusion Matrix of Xception	30
6.1	Adam optimization pseudocode	36

# List of Tables

2.1	Comparison of Related Work . . . . .	9
5.1	Proposed model vs previous state of art models. . . . .	25
5.2	Model parameters and Accuracy across different folds. . . . .	26
5.3	Training and Validation Accuracy of all the models. . . . .	26

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

*ACMV* African Cassava Mosaic Virus

*AdaGrad* Adaptive Gradient algorithm

*Adam* Adaptive Moment Estimation

*CBB* Cassava Bacterial Blight

*CBSD(genn)* Cassava Brown Streak Disease GPU-enhanced Neural Network

*CBSD* Cassava Brown Streak Disease

*CGM* Cassava Green Mottle

*CMD* Cassava Mosaic Disease

*CNN* Convolutional Neural Network

*COCO* Common Objects in Context

*DCNN* Deep Convolutional Neural Network

*DHT11* Digital Temperature and Humidity Sensor

*DNA* Deoxyribonucleic Acid

*EACMV* East African Cassava Mosaic Virus

*FAO* Food and Agriculture Organization

*GAN* Generative Adversarial Network

*GPDCNN* Global Pooling Dilated Convolutional Neural Network

*GUI* Graphical User Interface

*ILSVRC* ImageNet Large Scale Visual Recognition Challenge

*ILSVRC* The ImageNet Large Scale Visual Recognition Challenge

*PSO* Particle Swarm Optimization

*RSMProp* Root Mean Square Propagation algorithm

*SACMV* South African Cassava Mosaic Virus

*SoC* System on a Chip

*SVM* Support Vector Machine

*UCBSV* Ugandan Cassava Brown Streak Disease

*UV* Ultraviolet

*ValLoss* Validation Error

*VGGNET* Visual Geometry Group net

# Chapter 1

## Introduction

### 1.1 Background

Cassava could be a lasting plant with expansive, nearly palm-like leaves. Of the comparing castor-oil plant, but more profoundly separated into five to nine lobes. It is cultivated as an annual crop in the tropical area and as well as in the subtropical areas of the earth because of its tubers and the leaves that are considered as byproducts. The Food and Agriculture Organization (FAO) (2020), in their article, mentioned that Nigeria is right now the biggest maker of cassava within the world with an yearly yield of over 34 million tons of tuberous roots, almost 30% of the world's production [8]. Cassava is mostly cultivated by poor farmers who have lived in fragile environments particularly on erosion prone, acid and infertile land. To to grow cassava, it requires low fertile land along with cheap instruments and cost [16]. This ability to grow in infertile land where many plants would fail to grow has given cassava an undeserved reputation. Cassava leaves have a high protein, mineral, and vitamin content. However, the leaves of cassava can not be taken raw due to antinutrients and cyanogens in the leaves. Antinutrients and cyanogens prevent cassava as human food. Additionally, crude cassava has cyanide in it which is poisonous to ingest. Subsequently, it is basic to get ready accurately sometime eating. In various parts of the world people eat cassava in various ways such as by backing and boiling them, by drying them under the sun. Among them boiling and backing is the common method that is followed by millions. There are different types of foods which are made from cassava. In Bangladesh cassava has been produced for almost a decade. In our country, the food, textile and pharmaceutical industries need a huge amount of starch. Bangladesh imports 90% of it [44]. In this case, cassava offers an increasingly attractive crop option. In contrast, as the demand for cassava is rising, the disease of cassava is rising as well. In our thesis, we will classify cassava leaf disease using machine learning and image processing.

Cassava is a kind of plant that's tall in protein and vitamins, particularly within the leaves. It is frequently utilized as a rice elective in numerous districts of the world. Cassava production comes second with a total production of 5,323.00 tons after rice plants with a total production of 29,583.68 tons, according to statistics from Bandar Lampung City's Central Statistics Agency for the years 2015-2017 [1]. Cassava production, on the other hand, fell by 34.5 percent in 2018 compared to the previous year (Central Statistics Agency of Bandar Lampung City, 2019) [1]. In

many tropical regions of the world cassava production has been decreased due to its chronic diseases.. Pests, viruses, bacteria, and fungi cause a variety of illnesses on cassava leaves. [38]. Cassava Bacterial Blight (CBB), Cassava Brown Streak Disease (CBSD), Cassava Green Mite (CGM), and Cassava Mosaic Disease are the four illnesses that often affect cassava leaves in this research (CMD). Since takes off are a basic component of the plant and serve as a location for photosynthesis, edit yields will be influenced in case they are assaulted by the sickness. Photosynthesis Comes about are transported to all other segments of the plant by the phloem tissue. The Development of the stems and tubers will be immaculate on the off chance that the plant's clears out are sound and the photosynthetic preparation is accurately carried out. When infection assaults the takes off and disturbs the photosynthetic prepare, the development of stems and tubers is additionally disturbed, coming about in low-quality trim yields. To distinguish a malady in cassava takes off, a research facility test or help from a plant master is commonly utilized. However, as it is a very costly method to detect the disease in the laboratory and is time consuming, it would be the reason why farmers will not be able to respond to the disease accurately. As a result, we require an cleverly framework which is competent of understanding these issues within the way of an master, with an intelligently show based on Python GUI program for less demanding operation.

CBB is a disease caused by the bacteria *Xanthomonas manihotis* that is most commonly seen in humid environments [40]. Starting signs and indications incorporate calculated injuries, tissue passing at the contamination location, and leads to death. *Bemisia tabaci*, a whitefly virus, causes CBSD (Genn). There are two types of leaf symptoms: yellow chlorosis on secondary and tertiary leaf bones, and chlorotic patches[41]. When we find the arrangement of dry sepia to brown, foamy, and necrotic bruises within the tuber tissue, it confirms that these are normal signs of this viral disease. The bug *Mononychellus tanajoa* causes CGM by chewing the undersides of youthful takes off. Chlorophyll is drained from the cells, resulting in chlorosis and the death of the leaves. CMD is a virus belonging to the genus *Begomovirus* that causes the disease. On the takes off, a mixture of yellow and white chlorotic patches appear, which, depending on their concentration, impede photosynthesis and prevent the plant from progressing. As a result, the quantity of generation decreases.

We have used several models in order to classify the diseases. The first model we have used to classify is the Xception . Xception is a seventy-one-layer convolutional neural network. We have used a pre-trained version of the setup that has been trained on over a million images from the ImageNet database. The pre-trained system can classify images into 1000 different inquiry categories, including console, mouse, pencil, and a variety of animals. As a result, the system has learned a variety of rich highlight representations for a variety of images. Furthermore, this show has becoming increasingly popular for leaf infection detection We have got the greatest accuracy utilizing the Xception demonstrate which is 91.3. Several investigate has been done to classify the leaf infection utilizing Xception. In this manner, we have chosen this demonstrate in our investigation.

The second model we have used in classification is VGG16. VGG16 is a convolutional neural network named after the Oxford-based Visual Geometry Bunch. In

2014, it was used to win the ILSVRC (ImageNet) competition. The VGG16 demonstrated is prevalent for plant infection discovery. As of late, individuals are utilizing this show for leaf illness discovery. A few inquiries about papers have been made for leaf illness location utilizing the vgg16 demonstration. Utilizing this show we have got a critical yield. We got the most reduced precision utilizing the VGG16 demonstration. The accuracy of VGG16 is 68.

Thirdly we used the EfficientNet-b0 model for the classification. EfficientNet-b0 is a convolutional neural network built using over a million images in the ImageNet database. In EfficientNet-B0, there are a total of 237 layers. The system can sort images into 1000 different question categories, including console, mouse, pencil, and a variety of animals. This model provided us with the second highest level of accuracy. The accuracy is 91.1.

The fourth show that we have utilized in classification is densenet. A DenseNet is a kind of convolutional neural architecture that employs thick linkages between layers through Thick Pieces, in which all layers (with corresponding featuremap sizes) are directly interfaced with one another. DenseNet was created particularly to progress the declined exactness caused by the vanishing slope in high-level neural systems. In less complex terms, due to the longer way between the input layer and the yield layer, the data vanishes some time recently coming to its goal. The accuracy we have got is 87.

Furthermore, we have used resnet-50 for the classification. ResNet-50 is also a convolutional neural network that is 50 layers deep. You'll be able to stack a pre-trained adaptation of the organization prepared on more than a million pictures from the ImageNet database. The pretrained arrangement can classify pictures into 1000 question categories, such as console, mouse, pencil, and numerous creatures. ResNet50 could be a variation of the ResNet show which has 48 Convolution layers at the side 1 MaxPool and 1 Normal Pool layer. We got an average accuracy from this model and the accuracy is 85.

Finally, we have used inception V3. Inception v3 is a convolutional neural network that was developed as a Googlenet module to aid with picture processing and object recognition. It's the third version of Google's Inception Convolutional Neural Network, which was first shown off at the ImageNet Recognition Challenge. Inception-v3 is a 48-layer deep convolutional neural network. You may use the ImageNet database to load a pre-trained version of the network that has been trained on over a million photos. The network can categorize photos into thousand different object categories, including keyboards, mice, pencils, and a variety of animals. The accuracy we got through the Inception V3 model is 86.4.

## 1.2 Research Objective

Our essential reason for this research is to classify the cassava leaf infection utilizing six diverse models. The image used as input might be from inside or outside the dataset with which we developed and tested our models. It will be able to classify the sickness if we provide a photo of a cassava infected leaf outside of the dataset. A comparison of the precision of the five models in diagnosing the leaf infection is



also one of our goals. In addition, the way different highlights appear when used in models differs in exactness degree, which has been investigated as part of our investigation. On both models, all of the highlights have been used individually as well as inside the shapes of stackings described, to see how accurate they are in relation to the models. In addition, as said above, cassava is exceptionally new in Bangladesh and getting to be well known day by day, we need to form an impression on cassava so that a little research can take place in the future.

### 1.3 Research Problem

In our thesis we were working on betel leaf disease detection. While working , we faced serious trouble in collecting the dataset. As the pandemic has spreaded all over the world, there was a strict lockdown going on in our country. As a result, We could not go out directly to the field to collect the dataset. Therefore, we had to rely on a secondary source of information for the dataset hence we have changed our topic to cassava leaf disease classification. Furthermore, we had met trouble while training our dataset. It happened due to the high gpu that was used for it. Moreover, model implementation was not an easy job for us. We had to fix several bugs while implementing our models. Therefore, It took more than 4 weeks to implement all the models for classification successfully. In addition, as we know The cassava cultivation in Bangladesh is very little and very limited so far. It was really difficult for us to collect the dataset with infected leaves. As a result, we took our dataset for cassava leaf disease classification from kaggle.

### 1.4 Contribution and Impact

In our research, we aimed to bring up the fact that which model shows the maximum accuracy while implementing them for classification of cassava leaf disease. On all of the models, we adjusted the dataset, using both augmented and original datasets, to ensure that the comparison is conducted on equal footing. After that, we have seen that all the models can classify the diseases accurately. On the other hand, the accuracy of the models for classification was not the same. In our research, we also attempted to contribute to the discussion on the impact of various characteristics on model classification accuracy. We have proven that various classifiers, because to their differences in characteristics, display varied accuracy for each of the models when using numerous features independently. Among the five models, Xception got the highest accuracy which is 91.3 and it is the maximum accuracy that anyone has got in this field so far. After that, 87.0 for DenseNet121, inception V3 got 86.4, 85.0 for ResNet50 and VGG16 got the lowest accuracy which is 68. And in research where comparisons have been shown, it does not fully depict the fact that which model is better in classifying leaf disease in comparison to the other models. According to our knowledge, the features that we have employed singly as well as stacked have not been done in the same way as ours. To summarize, unsupervised learning research can be carried out with more experiments and analysis.

## 1.5 Scopes and Limitations

As we have mentioned earlier, we have taken our dataset from the kaggle. Due to the pandemic we could not collect the raw dataset and eventually we had to choose kaggle for the dataset. The dataset we have worked on is a standard dataset. Apart from this, it is also possible to deal with newly developed image datasets. Furthermore, we have used supervised learning for both our models and scopes, but unsupervised learning may also be used to train and test the model. The resources we received or had on hand for our thesis research were not the most efficient, and it was irritating at times since We had to rely on our own resources at home, which slowed us down. We could have overcome some of the limits and expanded the scope of our study if we had been given the right resources at the right time.

## 1.6 Documentation Outline

The following is how our research paper is structured. The literature review section 2 of our research paper highlights prior similar works in the subject of plant leaves disease detection and classification. Following that, in part 3, we went over the data structure and data processing in further detail, including information on the data set we used and Data Pre-processing. The methodology is discussed in section 4 of the paper. Here, we go over the models we utilized in our research and provide information on model compilation. In part 5, we discuss the results and analyze them, demonstrating why such results are received and what impact they have.

## Chapter 2

# Literature Review and Related Work

This chapter discusses several Deep Learning and Convolutional Neural Network Ensemble techniques for detecting illness in plant leaves. We are implementing effective strategies for identifying sick leaves that aid in crop loss reduction and productivity gains. This section discusses various deep neural network strategies for detecting plant diseases that are currently available.

Yoon et al (2020) [34] utilized the generative adversarial network technique for image processing and subsequently the DCNN for enhanced plant disease detection performance. It was able to enhance even more as the sample image generated by GAN had limited and relevant attributes. The authors found that employing synthetic samples generated using GAN architecture improved (+5.2 percent) compared to (+0.8 percent) lengthen using traditional augmentation techniques.

Nilay Ganatra and A. Patel (2020) [32] have provided picture based plant disease classification using multiple CNN models that were fine-tuned. VGG16, Inception V4, ResNet 50, and ResNet101 are among the frameworks evaluated and compared. ResNet50 and ResNet101 have 99.70 percent and 99.73 percent evaluation precision, sequentially.

Similarly, LAKSHMI THUSHARA and MAHABOOB RASOOL (2020) [31] presented a one kind of a method for swiftly recognizing and analyzing plant illnesses from leaf images using a deep learning strategy. The authors' recommended technique distinguished active and 4 opposed diseased leaves. The dataset included of actual collected photos (infected and healthy leaf) with a 96 percent accuracy rate.

According to Chaitanya and Yasudha (2020) [35], the use of tracking control and executive mechanisms is quickly spreading due to the creative progress. Disease's vast spread causes the majority of horticultural crop mortality. The model accuracy is 98.84 percent.

For cucumber leaf disease detection, Yun Shi and others (2019) [29] introduced a Novel Deep Learning mechanism named GPDCNN. The authors replaced entirely

connected layers with global pooling layers to improve the convoluted receptive field without increasing complexity. According to the author, the dataset contains six illness images of cucumber disease, with an accuracy rate of 94.65%.

In addition, For identifying the fungal illness Anthracnose in mango leaves, Pratap Singh and others (2019) [24] developed a Multilayer CNN model. The picture of the dataset was captured at jammu and kashmir in India. There are photographs of both contaminated and uncontaminated leaves in the data set. Using this model, we were able to attain 97.13 percent accuracy.

In addition, V. Singh (2019) [25] developed an image segmentation approach in light of Particle Swarm Optimization to identify proof of diseases in Sunflower plant leaves that efficiently perceived and coordinated infections. It does not require any prior knowledge or the viewing of many sections, as is the case with other existing approaches. The proposed algorithm's typical exactness or accuracy of categorization is 98 percent.

Sachan and others (2019) [33] also developed a deep learning model for disease recognition in Corn leaves. Disease detection in Corn leaves was obtained with an accuracy of 88.66 percent.

Ghai et al. (2019) [36] studied an abnormal response to the problem using fragmented image details to create the CNN model. In implementation rose from 42.3 percent to 98.6 percent due to model execution. Furthermore, a quantitative study of self arrangement assurance revealed a significant improvement, with 82 percent of test data sets indicating an increase in conviction.

Gupta and others (2019) [30] also developed a CNN based model for disease identification in tomato leaves. The authors employed a three convolution neural network with max-pooling layers. The data was gathered from the Plant Village dataset, which includes 9 illness classes and 1 healthy class. This paper had an average accuracy of 91.2 percent, ranging from 76 to 100 percent in four categories.

Similarly, Khamparia et al. (2019) [22] developed a deep convolution encoder network model. This model helped detect disease in crop leaves. The maize leaves in this data set were healthy and diseased, taken from the Plant Village collection. The identification of corn leaves condition is accurate to the tune of 97.50 percent.

By studying the leaves of a specific plant, Honakeri and others (2019) [27] proposed a Deep Learning technique for identifying and classifying plant illnesses. The classification was done in phases to eliminate possibilities at each level, resulting in greater forecast accuracy.

K Rasadurai (2019) [23] used K-Means, Otsu Segmentation approach, CNN, and SVM classifier algorithms to detect and adapt each change in the plant in the DHT11 Sensor, Soil PH Sensor, Soil Moisture Sensor, UV Sensor-based plant growth monitoring, and control microcontroller. The author got results with remarkable precision when compared to the multilayer Perceptron method; the author's approach pro-

duces legitimate findings. The architectural precision of the system is 11.06 percent higher than that of the current system.

Ahmad and others (2018) [19] created a deep-learning system for identifying wheat illnesses from images taken real-time by camera devices of various conditions. There were 4 types of wheat disease in the database: routine, stem rust, Powderly and yellow rust . There was a total of 2,207 photos in per category. The authors employed a CNN to train a classifier. One of the most significant advantages of CNN is the ability to extract features quickly by directly processing picture data. Farmers can use the model to protect their wheat crops against the illnesses indicated because it was 84.54 percent reliable.

Arsenovic et al. (2016) [11] were interested in a novel plant disease detection model. The latest convolution neural network generation has seen promising image classification results using a Deep convolution network. The constructed model detected 13 different types of diseases and which ones are healthy. Berkley Vision and Learning Centre developed a deep learning architecture to train the Caffe, DCNN. The authors' precision ranged from 91 percent to 98 percent, with an average of 96.3 percent.

Furthermore, to monitor numerous insects in coconut trees, Abraham Chandy [21] utilized the NVIDIA Tegra Machine on Chip (SoC) with a camera interfaced robot, a smart farming technology. The author used a deep learning system to collect photographs and interpret data from the drone flying over the coconut field to classify the sick and insect-affected trees. The deep learning technique employed a collection of datasets of model pests. The data was transferred immediately to the farmer's mobile using internet. This aids in the timely management of bug infested trees and the growth of tree output.

Also, according to Kumar and Khanna [37], the RESNET-152 deep CNN based model was built, and the research was carried out utilizing live images of the dragon fruit at various stages. Unlike the VGGNET, whose consistency declined as the network deepened and the number of epochs increased, the findings discovered shows a better accuracy in preparation and research even with greater amounts of epochs.

Such excellent research has aided us by providing several insights into various environmental datasets and achieving high effectiveness in cassava leaf disease classification.

A Comparison and summary of Literature work is given in the Table 2.1

Topic Name	Reference and Year	Objective	Data Set	Technique Used	Output	Advantage	Disadvantage
Unsupervised image translation using adversarial networks for improved plant disease recognition	Yoon and others (2020) [34]	Leaf disease identification.	2780 tomato plant disease images	Generative Adversarial Network And Deep CNN	Accuracy = 86.1%	Improved evidence of details appropriation (more sharp images).  GAN may prepare any form of source architecture.	Preparation is difficult, and the training process is insecure.  To get satisfactory outcomes, you'll need to follow a lot of rules.  Issue with Mode Collapse.
Cucumber leaf disease identification with global pooling dilated Convolutional Neural Network.	Yun Shi and others (2019)[29]	Leaf disease identification	Acquisition of 600 cucumber ill leaves of six healthy cucumber leaf infected	GPDCNN	Accuracy = 94.65%	GPDCNN outperforms other algorithms in terms of robustness.	Because the utterly related layer contains many parameters, it slows down the preparation (training) process and effectively causes over-fitting.
Multilayer Convolution Neural Network for the classification of Mango leaves infected by anthracnose disease	Pratap Singh and others (2019) [24]	Classification of the Mango leaves contaminated by the Anthracnose contagious sickness.	Captured images at SMVDU	Multilayer convolutional neural network (MCNN)	Accuracy = 97.13%	The fundamental difference between MCNN and its models is that it recognizes the crucial elements without the need for human interaction.	If the PC does not have a good CPU and MCNN doesn't have more layers, the training process will take a long time.
Sunflower leaf diseases detection using image segmentation based on particle swarm optimization	V. Singh (2019) [25]	Detection of Sunflower leaf disease(6 diseases)	Capture Sunflowers leaves.	Particle Swarm Optimization Algorithm.	Accuracy = 98%	PSO has the advantage of being simple to implement and having few boundaries to adjust.  When it comes to computing efficiency, the PSO outperforms the GA.	PSO is a well-known method, however due of the straightforward qualities, its application for the problem is not muddled.
Deep convolutional neural network-based detection system for real-time corn plant disease recognition	Sachan and others (2019) [33]	Lleaf disease identification	Plant Village dataset.	DCNN	Accuracy = 88.46%	This algorithm takes less human work because it does not rely on pre-processing. It's a self-learner, which makes the pre-processing step go more smoothly.	To analyze and train the neural structure, a large dataset is required.
Performance analysis of deep learning CNN models for disease detection in plants using image segmentation	Ghai and others (2019) [36]	Leaf disease identification	Tomato healthy and infected leaves images	CNN	Accuracy = 98.6%	Perhaps CNN's most advantageous position is the automated extraction of highlights from basic images.	CNN's lack of ordered outlines is an essential part of human creativity.
Tomato Leaf Disease Detection using Convolution Neural Network.	Gupta and others (2019) [30]	Tomato leaf disease detection.	Plant Village dataset.	CNN	Accuracy = 76% - 100%  Average Accuracy for disease =91.2%	The suggested model required around 1.5MB of disk space, whereas pre-prepared models required nearly 100MB of extra space.	CNN is effectively slower due to an activity such as pooling
Seasonal Crops Disease Prediction and classification Using Deep Convolutional Encoder Network	Khamparia et al. (2019) [22]	Leaf disease identification	Plant Village Dataset	Deep Convolution Encoder Network	Accuracy = 97.50%	The output layer employs the Soft-Max classifier. If a multi-order model is encountered, it returns the probabilities of each class, with the target class having a high likelihood.	A strategy for mapping deep layer feature maps to input dimensions is absent from this method.
Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification	Arsenovic and others (2016) [11]	Perceive 13 unique sorts of plant infections out of healthy leaves	agricultural experts captured images.	Deep Convolution Neural Network	Accuracy = 96.3%	DCNNs were used to classify images and objects, detect faces, and segment images.  DCNN has more hidden layers, notably over 5, which improves accuracy.	The location and orientation of an object are not encoded by CNN.  Lack of capacity to be physically consistent in response to incoming data.

Table 2.1: Comparison of Related Work

# Chapter 3

## Data Processing

### 3.1 Data Structure

The major focus of deep learning research is the collection of big datasets, and obtaining data with good comprehension from reputable sources is critical. To compensate for the complexities of actual life, we need models trained on real-world pictures. This finding inspired us to develop a dataset for reliable plant disease identification in the farm environment by collecting photos from Kaggle competition. The dataset for cassava leaf disease classification includes 21,367 labeled images with a resolution of 512x512x3, which we later converted to 224x224x3. The majority of the images were gathered from farmers who took photos of their gardens and analyzed by professionals from the National Crops Resources Research Institute (NaCRRI) in partnership with Makerere University's artificial intelligence lab. It was a five-output category multi-label classification task, with four diseases and one healthy leaf, and their label to disease mapping is shown below.

- 0: Cassava Bacterial Blight (CBB)
- 1: Cassava Brown Streak Disease (CBSD)
- 2: Cassava Green Mottle (CGM)
- 3: Cassava Mosaic Disease (CMD)
- 4: Healthy

The figure 3.1-3.4 below illustrates four of the most common cassava illnesses. Each illness has its own set of signs and symptoms, which may be used to visually distinguish and categorize infections, as well as automatically by deep learning algorithms.

Cassava Bacterial Blight (CBB): Cassava Bacterial Blight initially found in Brazil in 1992, the malady has taken after the development of cassava over the world. Among diseases which beset cassava around the world, Bacterial Blight causes the biggest misfortunes in terms of yield [18] . In figure 3.1 we can see there are several symptoms shown in this disease which are blight, wilting, dieback and vascular necrosis. A more demonstrative indication distinct in cassava with *X. axonopodis* contamination are precise necrotic marking of the leaves—often with a chlorotic ring encompassing the spots [7] .



Figure 3.1: Cassava Bacterial Blight (CBB)

Cassava Brown Streak Disease (CBSD): Cassava brown streak disease (CBSD) may be a harming illness of cassava plants, and is particularly troublesome in East Africa [4]. It was to begin with recognized in 1936 in Tanzania, and has widened to other littoral zones of East Africa, from Kenya to Mozambique. Two viruses are responsible for this disease: cassava brown streak infection (CBSV) and Ugandan cassava brown streak infection (UCBSV). CBSD is seen by extreme chlorosis and necrosis on contaminated leaves, and then a yellowish, blotched impression. Figure 3.2 shows symptoms of CBSD.

Cassava Green Mottle (CGM): Cassava green mottle disease was known from Solomon Islands. It was to begin with found on Choiseul within the 1970s; more as of late (2010), comparable indications were seen on Malaita. The record of the infection in Australia requires affirmation. The record of the infection in Australia requires affirmation. Cassava green mottle disease shows several symptoms. A sample of this disease is shown in figure 3.3. Youthful leaves are puckered with black out to particular yellow spots, green patterns (mosaics), and bent edges. More often than not, the shoots recuperate from indications and show up solid. Once in a while, plants gotten to be seriously hindered, eatable roots are truant or, on the off chance that display, they are little and woody when cooked.

Cassava Mosaic Disease (CMD): Cassava mosaic disease may be a plant infection that decreases cassava production. The infection causes twisting of leaves and the arrangement of white-greenish or yellow-greenish patches inside the green leaf which is known as chlorosis or mosaic [2]. A sample of this disease is shown in figure 3.4. African cassava mosaic infection (ACMV), East African cassava mosaic infection (EACMV), and South African cassava mosaic infection (SACMV) are particular





Figure 3.2: Cassava Brown Streak Disease (CBSD)



Figure 3.3: Cassava Green Mottle (CGM)



species of circular single-stranded DNA infections which are transferred by whiteflies and basically contaminate cassava plants [10] . Cassava Healthy Leaves: Cassava



Figure 3.4: Cassava Mosaic Disease (CMD)

Healthy leaves doesn't have any spot. It'd be a dark green color. A sample of healthy cassava leaf can be shown in figure 3.5.

## 3.2 Data Pre-processing

Data preprocessing are the means taken to arrange images before they are used by model preparing and induction [3]. This includes, yet isn't restricted to, resizing, arranging, and shading rectifications. Basically, pre-processing is needed to clean image information for our model data. Image pre-processing may likewise decrease model training time and speed up. Assuming the images are especially large, reducing the size of these images will drastically further develop model training time without altogether decreasing model execution. Augmentation makes new preparing models out of existing preparing data. It's difficult to genuinely catch a image that records for each true situation a model might include. Changing existing preparing data to sum up to different circumstances permits the model to gain from a more extensive cluster of circumstances. Hence, we resized the images first then divided these these images into batches and lastly transformed the images.

### 3.2.1 Resized Image

Changing the size of an image sounds wierd, however there are contemplations to consider. Modifying an image to be a square calls for either extending its aspects



Figure 3.5: Cassava Healthy Leaf

to fit to be a square or keeping its angle proportion consistent and filling in recently made "dead space" with new pixels [15]. In addition, input pictures might be different sizes, and some might be more modest than the ideal image size. However, we did not focus on preserving scale because it is not necessary all the time. Instead of this, we focused on filling dead pixels along with reflected image content and also downsampling large images to smaller images. The images get size of  $224 \times 224$  respectively. The main advantage of resizing images is to get the result too fast. However, there are some disadvantages as well. Chances of decreasing resolution is one of these.

### 3.2.2 Divided into Batches

Batch size is a hyperparameter that characterizes the quantity of sample work for working the internal model parameters first [14]. We think of a batch as a for loop. While the for loop iterates over one or multiple samples by which we can make predictions. After that we can compare our predictions with the desired outputs at the very last moment of the batch. At that time we calculate an error. This error improves the model by using updated algorithm. The learning algorithm is batch gradient descent when all of the training data are combined into one batch. When the batch size is comparable to one sample, we refer to the learning algorithm as stochastic gradient descent. We refer to a learning algorithm as small batch gradient descent when the batch size is more than one sample and not exactly the size of the learned dataset.

### 3.2.3 Image Transformation

An image is gotten in spatial directions (x, y) or (x, y, z). There are many benefits assuming the spatial domain image is changed into another domain. In which arrangement of any issue can be found without any problem. As our all work is about cassava, so we had to transform each and every image to get the accurate result.

### 3.2.4 Random Crop Augmentation

It is a data augmentation method wherein we make a random subset of the main image. This assists our model with summing up on the grounds that the objects of interest we need our models to learn are not in every case completely noticeable in the image or similar scale in our training data. To be mentioned, our resized image is 224x224. As we are creating such a model that recognize diseases of cassava leaf, we may have not such a leaf that is affected or affected in a very small area. In this case, random crop is a better augmentation technique for this model.

### 3.2.5 Random Flip Augmentation

Flip of an image does mean switching the column or row of pixels on account of a vertical or horizontal flip individually. For our model, this augmentation makes sense because, when it flips, it does flip horizontally but vertically does not.

### 3.2.6 Random Rotation Augmentation

Random Rotation Augmentation spins the image clockwise by a specified number of degrees between 0 and 360 at random. To execute this for our model, we have to pass the argument `rotation_range` in the `ImageDataGenerator`. The `rotation_range` argument accepts an integer value in between 0 to 360. It is a very common widely used data augmentation technique. Random rotation is a valuable augmentation specifically in light of the fact that it changes the points that objects show up in our dataset during training. Maybe, during the image collection process, images were just gathered with an object on horizontal, yet underway, the object could be slanted one or the other way. Random rotation can further develop our model without us gathering and label more data. Considering our model, if a user wants to detect a disease using this model, he/she may not have their phone perfectly perpendicular, in this case random rotation is a great augmentation technique to give the best user experience of our model.

### 3.2.7 Random Contrast Augmentation

Randomly change the contrast of an image when a contrast range along with seeds is given [45]. In lower and upper interval it disjointedly chose the range of contrast of an individual image. Basically, in this method we will prepare the system with diverse colour escalated images, so that they can generalize the pictures which are not prepared indeed on distinctive lightening conditions. Depending on the argument we pass to the `ImageGenerator` class constructor, the model can make the images a little brighter, darker or both. We accomplish this through the parameter `Brightness_range`. We pass the max and min values as a float which treats that

esteem as a rate to apply to the image. As the brightness and darkness depend on the boundary value, in our case the boundary value is 0.2. In case it gives the value less than 0.2, at that point it will darken the image. If the value is more than 0.2, then the image will be brighter.

# Chapter 4

## Methodology

### 4.1 Xception

An Xception, which stands for extreme inception, is first proposed by Ref. [12]. Xception is a CNN architecture solely made up of depth-wise separable convolution layers, which has 36 layers of convolutional neural networks. The 14 modules make up the 36 convolutional layers. Except for the first and last, all modules contain linear residual connections. Exceptional architecture is composed of complex layers that can be separated in depth. The distinction between depth-wise and extreme-beginnings convolution can be distinguished by doing wise spatial convolution first and 1x1 convolution. In the meantime, the inception executes 1x1 convolution first. Then can view the architecture of Xception in Fig.4.1

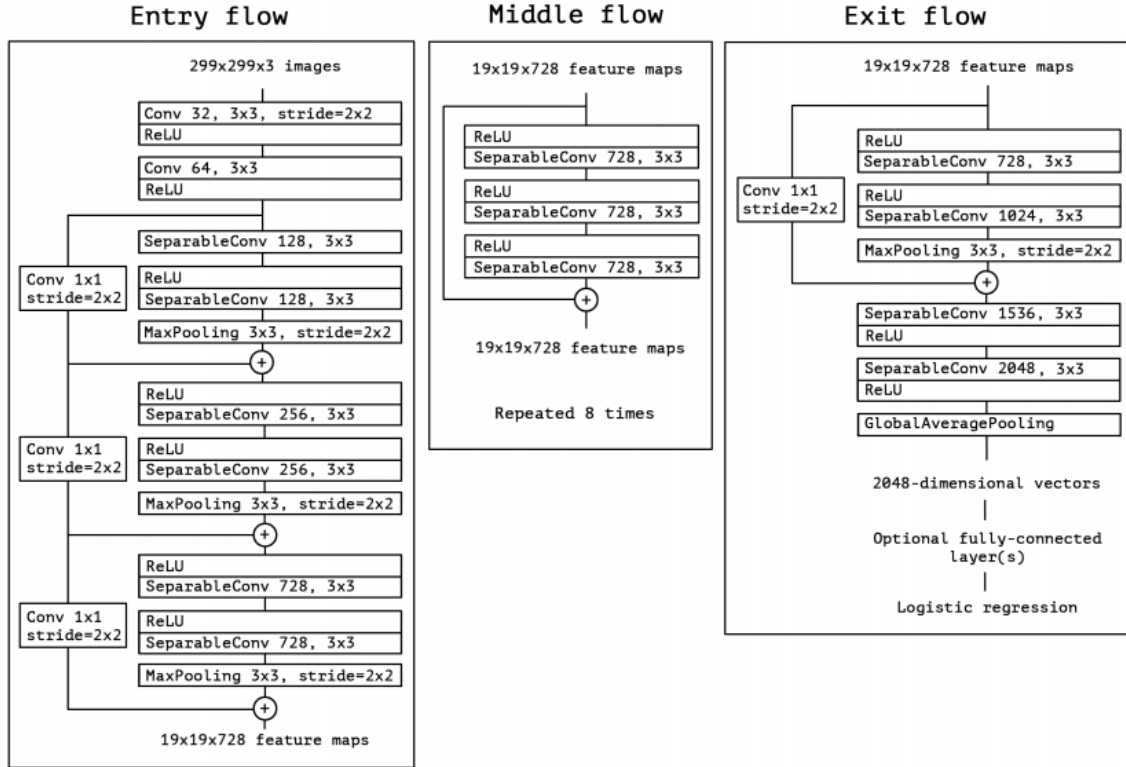


Figure 4.1: Xception architecture

We only used two epochs to train this model. The model accuracy and loss of



Xception in the Y-axis and the epochs in the X-axis are shown in the figure 4.2. The period that corresponds to the whole dataset during one cycle is called an epoch. The model's validation error (Val loss) continues to decrease. In one epoch, the entire dataset is passed forward and backwards through the model [17]. Because an epoch is too large to feed through the model all at once, we divided the dataset into 32 batches. We kept the batch size at 32 because small can help our model generalize better. Because the batch size is set to 32, the model will take 32 samples for training and repeat the process in the same order until all pieces have been propagated through the model.

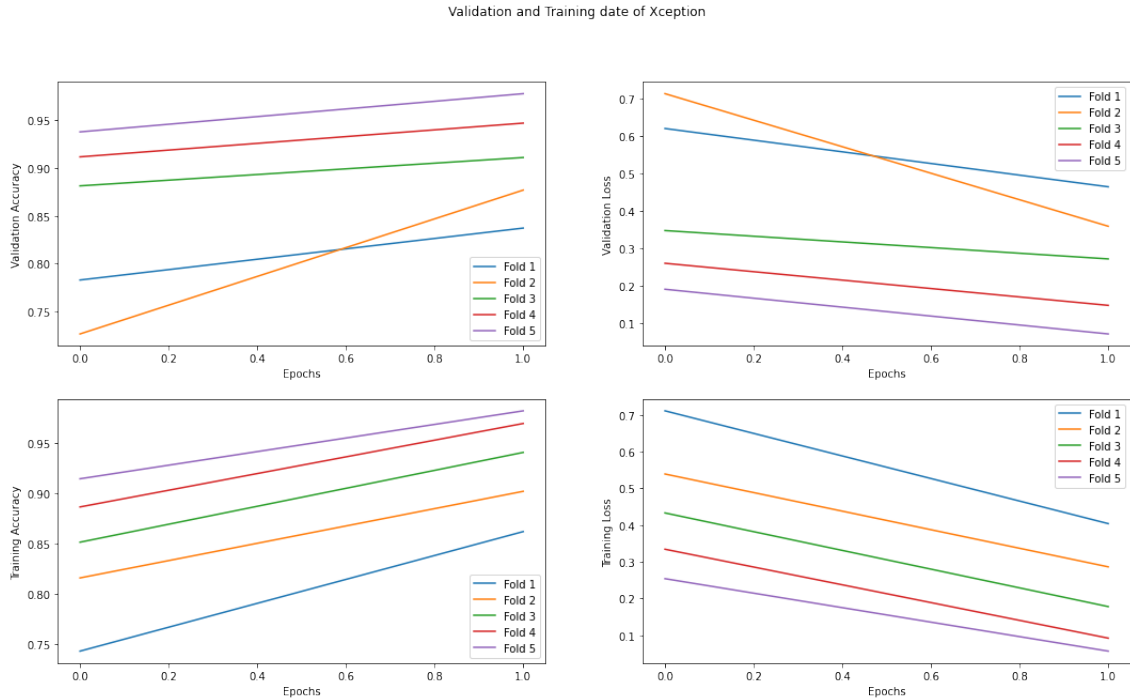


Figure 4.2: Model accuracy and Model loss of Xception

In Figure 4.2 of the model loss graph, the Loss is plotted on the Y-axis and the no. of epochs is plotted on the X-axis. It is shown that as the no. of epochs increases, so does the model's loss decreases steadily.

## 4.2 VGG16

VGG16 is a CNN model introduced by [5] K. Simonyan and A. Zisserman of the University of Oxford in their paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model outperforms 92.7 % top-5 test accuracy in ImageNet, a dataset of over 14 million pictures categorized into 1000 classes. It was among the most well-known models entered in the 2014 ILSVRC. It surpasses AlexNet by successively replacing large kernel-size filters (11 and 5 in the 1st and 2nd convolutional layers, accordingly) with multiple 3x3 kernel-size filters. VGGNet-16 comprises 16 convolutional layers [Fig.4.4] and a very uniform design, making it highly attractive. It has just 3x3 convolutions but several filters, comparable to AlexNet. It may be taught for two to three weeks on 4 GPUs. It is presently the most used approach for retrieving attributes from photographs in the industry.

The VGGNet weight setting is free source and has been used as a baseline feature extractor in a variety of different applications and challenges. The load arrangement of the VGGNet is accessible to the general public and regularly modified.

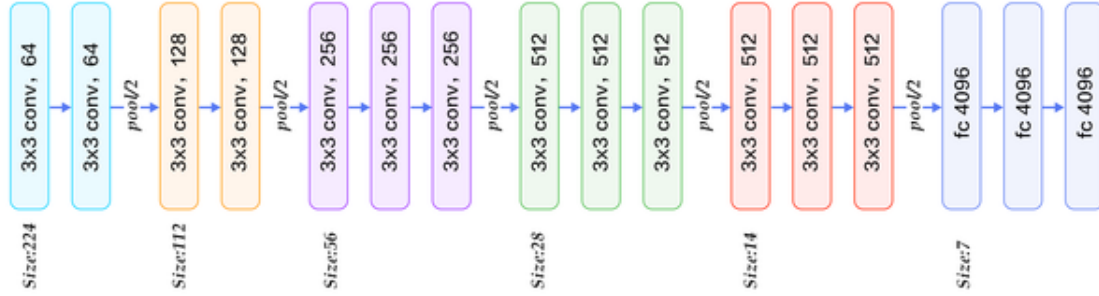


Figure 4.3: VGG layers

VGGNet, on the other hand, contains 138 million parameters, which might be difficult to handle. Transfer Learning may assist you in achieving VGG. The model is pre trained on a dataset, the parameters are tuned for greater accuracy, and the values of the parameters may be utilized.

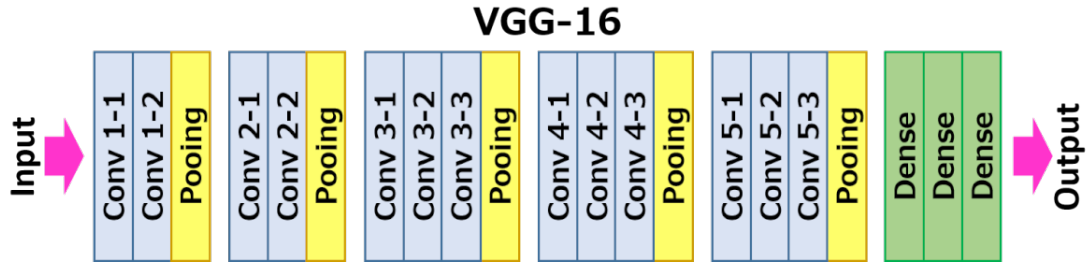


Figure 4.4: VGG 16

We only used two epochs to train this model. The model accuracy and loss of VGG-16 in the Y-axis and the epochs in the X-axis are shown in the figure [4.5]. The period that corresponds to the whole dataset during one cycle is called an epoch. The model's validation error (Val loss) continues to decrease. In one epoch, the entire dataset is passed forward and backwards through the model [17]. Because an epoch is too large to feed through the model all at once, we divided the dataset into 32 batches. We kept the batch size at 32 because small can help our model generalize better. Because the batch size is set to 32, the model will take 32 samples for training and repeat the process in the same order until all pieces have been propagated through the model.

In Figure 4.5 of the model loss graph, the Loss is plotted on the Y-axis and the no. of epochs is plotted on the X-axis. It is shown that as the no. of epochs increases, so does the model's loss decreases steadily.

### 4.3 InceptionV3

The InceptionV3 [6] model is the third iteration of the Inception micro-architecture, initially described by Szegedy and other Googlers in their work Going Deeper with Convolutions, widely known as GoogLeNet, in 2014. The Inception component's



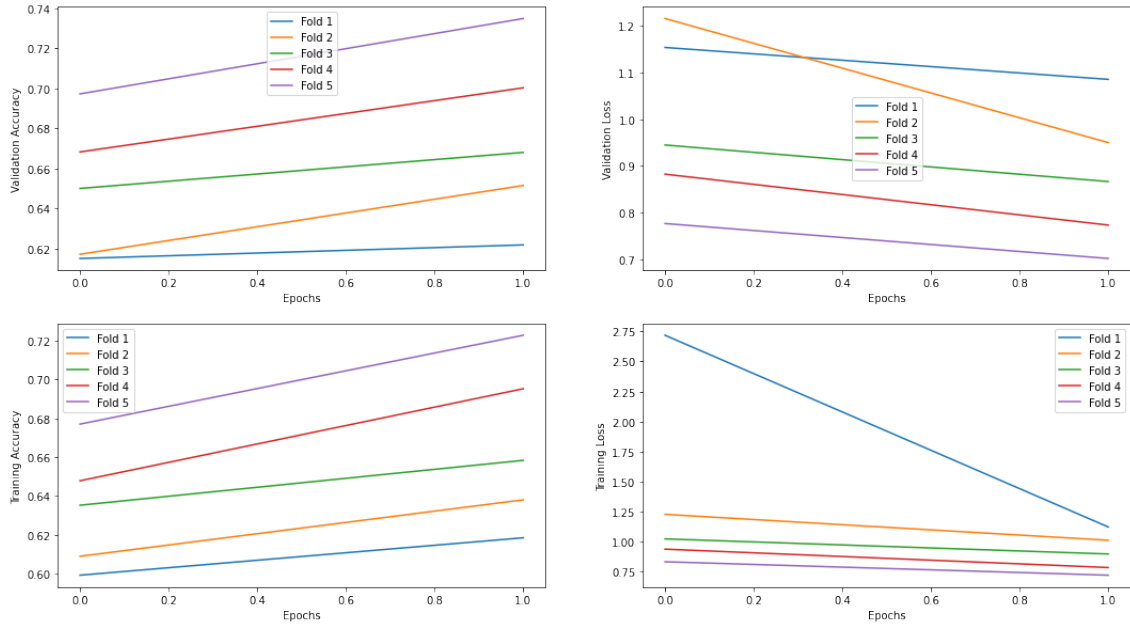
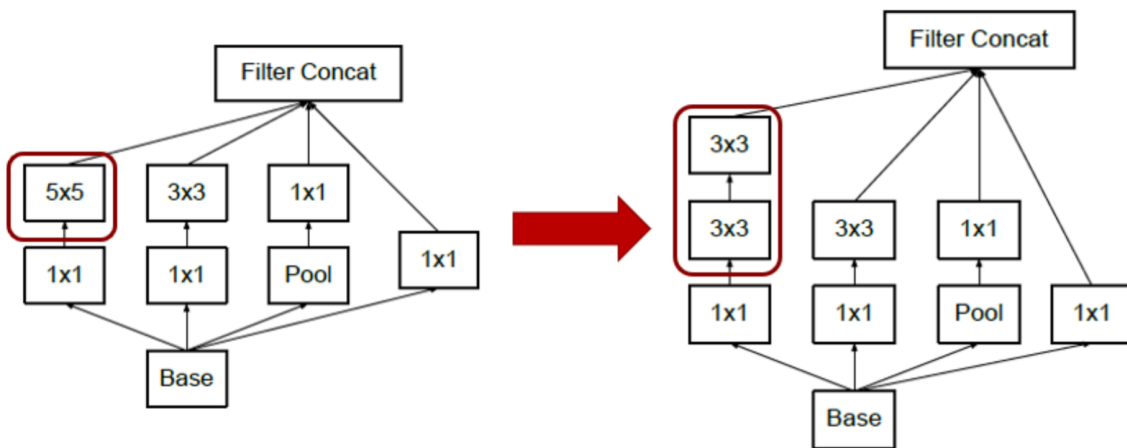


Figure 4.5: Model accuracy and Model loss of VGG-16

objective is to boost the model's resource usage efficiency while decreasing computational complexity by dividing high output sizes into shorter convolutions and intensive normalization through label flattening. For instance, a  $5 \times 5$  convolution filter in Figure 4.6 is  $25/9=2.78$  times more computationally costly than a layer of  $3 \times 3$  convolution. Hence, a two-layer of  $3 \times 3$  filters (with  $3 \times 3 + 3 \times 3 = 18$  parameters) may lower the number of factors by 28 percent compared to a layer of  $5 \times 5$  filters (with  $5 \times 5 = 25$  parameters)

Figure 4.6: Replacing one  $5 \times 5$  convolution with two  $3 \times 3$  convolutions

As demonstrated in Figure 4.7, the whole Inception-v3 model is made up of symmetric and asymmetric building elements such as convolutions, average pooling, max pooling, concerts, dropouts, and fully linked layers. The results of these unique

techniques beat the state-of-the-art performance on the ILSVRC 2012 classification challenge, with 21.2 percent top-1 and 5.6 percent top-5 error rates while using 2.5 times fewer computing resources.

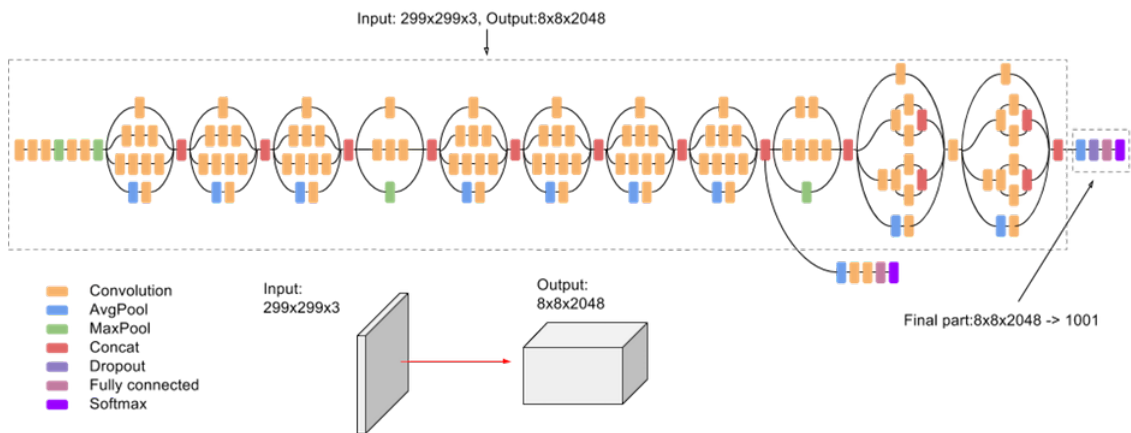


Figure 4.7: Schematic diagram of Inception-v3

## 4.4 ResNet-50

Residual neural network (ResNet) was published in 2015 by [9]. It has received over 65k citations and has placed first in several image detection competitions, including the ILSVRC 2015 classification task, ImageNet detection, ImageNet localization, COCO image detection, and COCO segmentation. In addition, ResNet suggested a revolutionary way to build a deep neural network model by stacking several residual blocks without adding extra parameters or computational complexity. Figure 4.8 shows a ResNet-50 schematic design with solid arrows representing skip links and dotted arrows showing a particular skip link for constraint building pieces across various dimension input and output layers. This wonderful strategy significantly assists in the prevention of disappearing slopes including the erosion issue. Nevertheless, as the network depth rises, the transnational may get saturated.

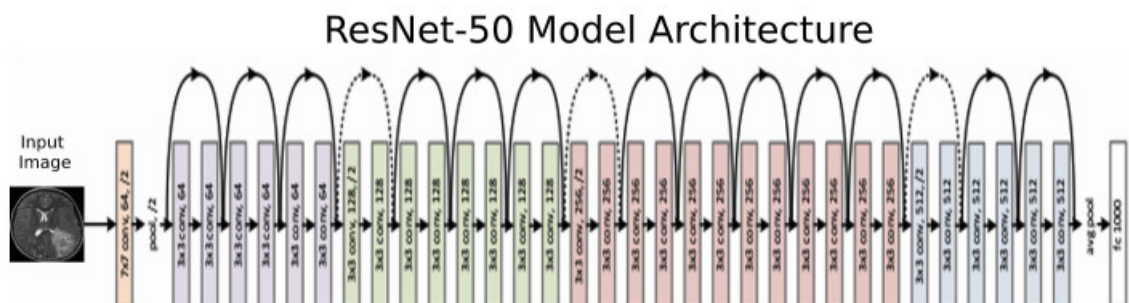


Figure 4.8: Schematic diagram of ResNet-50 architecture

We only used two epochs to train this model. The model accuracy and loss of ResNet50 in the Y-axis and the epochs in the X-axis are shown in the figure [4.9].

The period that corresponds to the whole dataset during one cycle is called an epoch. The model's validation error (Val loss) continues to decrease. In one epoch, the entire dataset is passed forward and backwards through the model [17]. Because an epoch is too large to feed through the model all at once, we divided the dataset into 32 batches. We kept the batch size at 32 because small can help our model generalize better. Because the batch size is set to 32, the model will take 32 samples for training and repeat the process in the same order until all pieces have been propagated through the model.

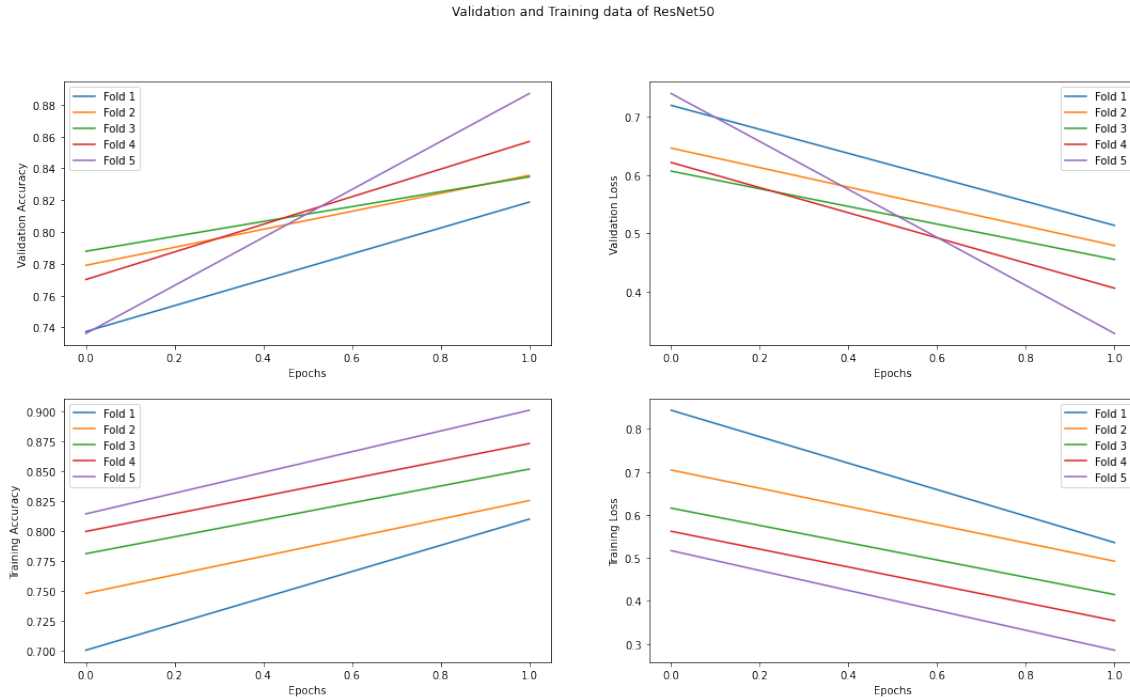


Figure 4.9: Model accuracy and Model loss of ResNet50

In Figure 4.9 of the model loss graph, the Loss is plotted on the Y-axis and the no. of epochs is plotted on the X-axis. It is shown that as the no. of epochs increases, so does the model's loss decreases steadily.

## 4.5 EfficientNetB0

A deep-learning architecture strives to improve performance accuracy and efficiency with smaller models. The EfficientNet architecture, unlike other state-of-the-art deep-learning models, is a compound scaling strategy that uses a compound coefficient to equally scale network breadth, depth, and resolution [26]. EfficientNet is made up of eight different models ranging from B0 to B7. EfficientNet utilizes a new function called swish activation instead of the ReLU activation function. Inverted bottleneck convolution, which was first presented in the MobileNetV2 model and consisted of a layer that first grows the network and then compresses the channels [20], is used by EfficientNet. Compared to standard convolution, this architecture reduces computation by a factor of  $f^2$ , where  $f$  is the filter size. EfficientNetB0 is the most basic network of all eight models, according to the authors of [26], and it

has fewer parameters. As a result, we used EfficientNetB0 to evaluate performance in our experiment. EfficientNetB0's basic block diagram is shown in Figure 5.



Figure 4.10: Basic block diagram of EfficientNet model

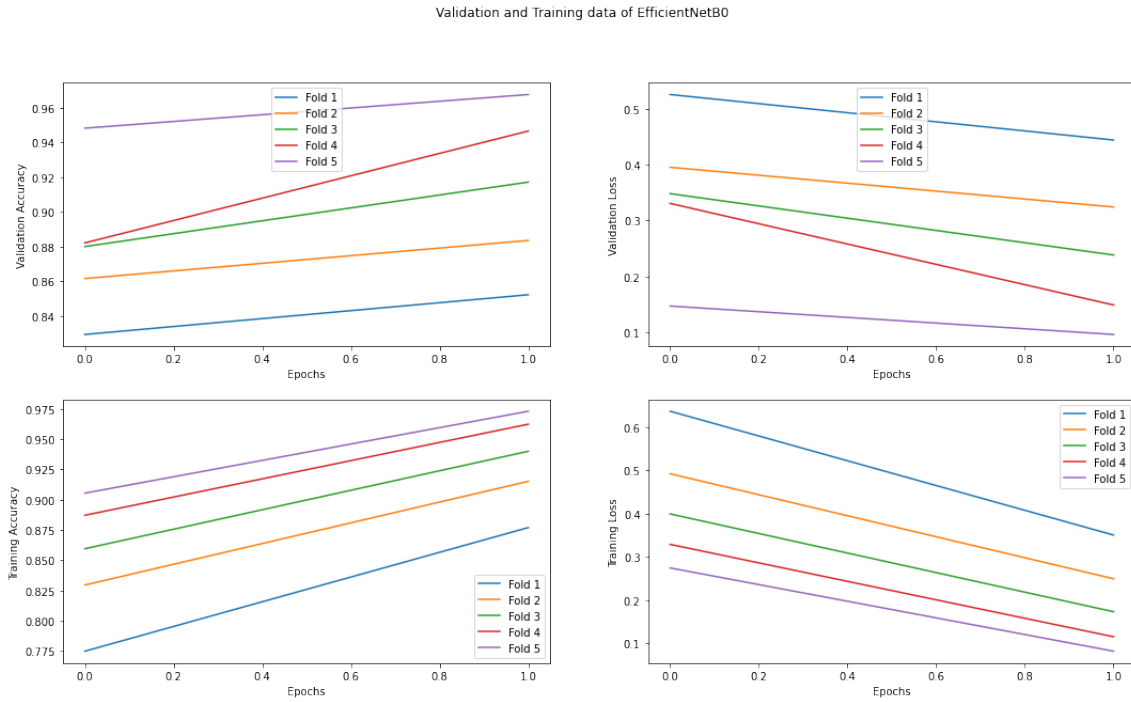


Figure 4.11: Model accuracy and Model loss of ResNet50

## 4.6 DenseNet-121

The DenseNet-121 [13] model is a deep network with modest parameters. The architecture comprises dense blocks with different numbers of filters but the same feature maps. Each block has access to the feature maps that came before it, and each layer only adds new feature maps. Any extra feature map is removed. The DenseNet-121 model performs the convolutions in this manner, resulting in a model with a modest parameter size.

## 4.7 Model Compilation

To construct our models, we use the three parameters listed below:

### 4.7.1 Loss Function

Because of the growing number of classes in our models that impact the classification of a particular label, we used Categorical Crossentropy as the loss function in all of our models. The Categorical Crossentropy is often used to evaluate the distributions of predictions (activations of output layers) and correct distributions, where valid class is 1 and the remaining of the classes are 0. A lower score indicates that the model is doing better. The definition of Categorical Crossentropy [42] is:

$$Loss = - \sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i \quad (4.1)$$

where  $\hat{y}_i$  is the  $i$ -th scalar value in the model output,  $y_i$  is the corresponding target value, and output size is the number of scalar values in the model output.

### 4.7.2 Metric

In our models, the accuracy metric is expressed as a percentage of the model's predictions. Model training allowed us to observe the validation data accuracy score. The following equation from [43] is used to calculate accuracy:

$$Accuracy = \frac{No.ofcorrectpredictions}{No.ofTotalPredictions} \quad (4.2)$$

### 4.7.3 Optimizer

We utilized Adam Optimizer [28] 2019 to determine individual parameter's effective learning rate. Adam is a deep learning model training approach that uses stochastic gradient descent instead of stochastic gradient descent. Adam combines the most delicate aspects of the AdaGrad and RMSProp techniques to provide an optimization solution for noisy problems with patchy gradients. With each parameter, the adaptive learning rate models may identify distinct learning phases. The pseudo-code of Adam optimizer is provided in the appendix.

# Chapter 5

## Result and Analysis

The results we obtained using different transfer learning models are promising given that some of the images are taken from a distance and at varying angles. The dataset contains total images of 21396 including some noisy images but using Xception, we were able to reach an accuracy of 91.3 which is a state-of-the-art performance for cassava leaf classification. On the other hand, we did not use augmentation to increase the number of samples therefore the accuracy is obtained on the samples from the dataset. In addition, EfficientNetB0 has also resulted in an exceptional accuracy given that it has less trainable parameters compared to Xception. Therefore, EfficientNetB0 is the most optimal model in our research with respect to training time, number of parameters and accuracy. The transfer learning models we used are Xception, EfficientNetB0, ResNet50, VGG16, DenseNet121 and InceptionV3. The training and validation accuracy reached by each model is given in Table 5.3. Whereas Table 5.1 demonstrates our performance compared to all the available research on the cassava leaves classification and we can clearly see that the accuracy reached using the methodologies we used have overpowered the previously proposed implementation on this domain. In our methodology we have used cross validation which is a technique that helps the model to train and test on the whole dataset. The images are first loaded into training and validation data generators with a target size of 224 and a batch size of 32. Therefore, the generator automatically brings all the images to a fixed size before using as an input into the mentioned models. On the other hand, the batch size of 32 helps to improve the training speed because not all the images are loaded into the ram and then into the model.

In Table 5.2, we can see the models with respect to the number of parameters and the

Model	Classification Accuracy(%)
<b>EfficientNet B4 [39]</b>	81.48
<b>MobileNetV2 [38]</b>	65.6
<b>Xception</b>	91.3
<b>EfficientNetB0</b>	91.1
<b>ResNet50</b>	85.0
<b>VGG16</b>	68.0
<b>DenseNet121</b>	87.0
<b>InceptionV3</b>	86.4

Table 5.1: Proposed model vs previous state of art models.

Model Name	Parameters	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Xception	22,910,480	0.8371	0.8766	0.9105	0.9462	0.9769
EfficientNetB0	5,330,571	0.8521	0.8834	0.9170	0.9465	0.9675
ResNet50	25,636,712	0.8187	0.8355	0.8345	0.8567	0.8869
VGG16	138,357,544	0.6217	0.6514	0.6679	0.7002	0.7348
DenseNet121	8,062,504	0.8430	0.8631	0.8668	0.8876	0.9037
InceptionV3	23,851,784	0.8222	0.8449	0.8486	0.8689	0.8946

Table 5.2: Model parameters and Accuracy across different folds.

Model Name	Training Accuracy(%)	Validation Accuracy(%)
Xception	98.25	91
EfficientNetB0	97.30	91.33
ResNet50	90.06	85
VGG16	72.28	68
DenseNet121	91.64	87
nceptionV3	89.80	86

Table 5.3: Training and Validation Accuracy of all the models.

accuracy across different folds. In cross validation the dataset is first divided into 5 folds, each fold containing a training and testing set. Therefore, the model is able to generalize and validate on the whole dataset. In Table 5.2, we can see that VGG16 has the highest number of parameters and therefore takes the maximum time to train but the average accuracy across all the folds is very low. This is because a lot of pretrained weights here do not generalize well with the cassava dataset which diminishes the performance of the model. However, EfficientNetB0 performs best across all folds because the less number of pretrained weights generalizes well with the matrix value of the images of the cassava leaves.

Lastly, Table 5.3 shows the training and validation accuracy of all the models that we used where we can see that Xception and EfficientNetB0 fits on the samples points really and the difference between the validation accuracy is not much therefore, we can conclude that there is no overfitting in our state-of-the-art models (Xception and EfficientNetB0) for this domain.

In figure 5.1, we can see the maximum accuracy is reached after fold 5 for Xception model this is because the model is trained on different sets of the same dataset with 2 epochs on each set. Therefore, the model is trained a total of 10 times in total with the different sets of dataset. This helps the imagenet weight to generalize well on the dataset.

In figure 5.2, we also achieved a accuracy of 91.1% which is second highest accuracy. every model in here is trained with various sets of the same dataset, each with two epochs. In figure 4.11, Resnet50 we have achieved an accuracy of 0.85%. In figure 5.4, VGG 16 we found the the least accuracy which is around 68.0%. Densenet121 produced a decent result with an accuracy of 87.0% in Figure 5.5. Similarly, on Inception V3, Figure 5.6 we obtained an accuracy of 86.4%

Finally, in Figure 5.7, we can see a comparison of the results of the six classification models we utilized. This figure was illustrated as a bar chart so that it was clear which model produced the greatest results.

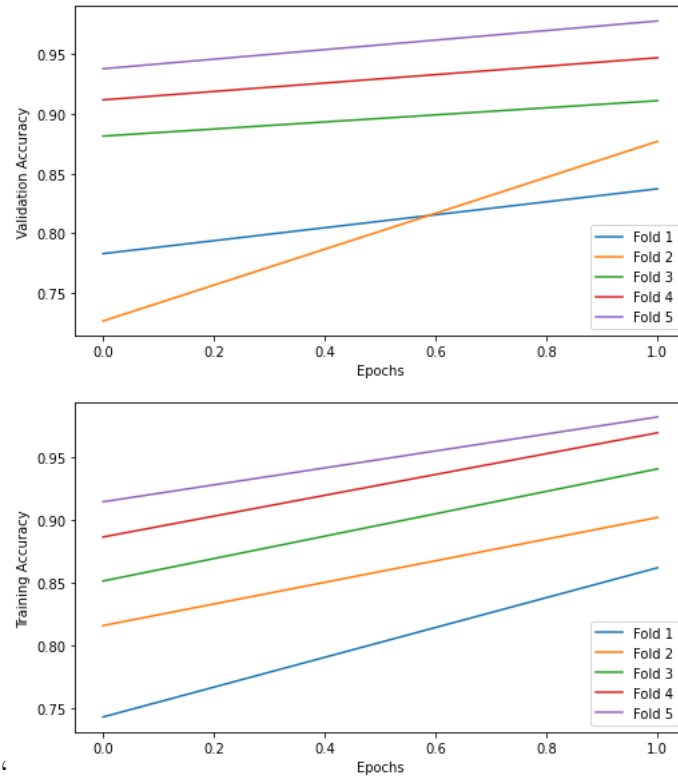


Figure 5.1: Model accuracy of Xception

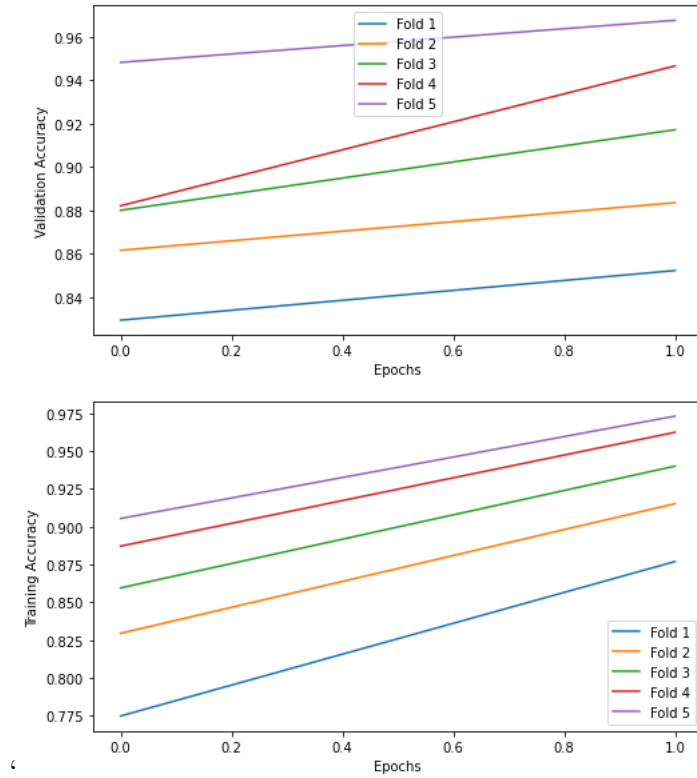


Figure 5.2: Model accuracy of EfficientNet B0



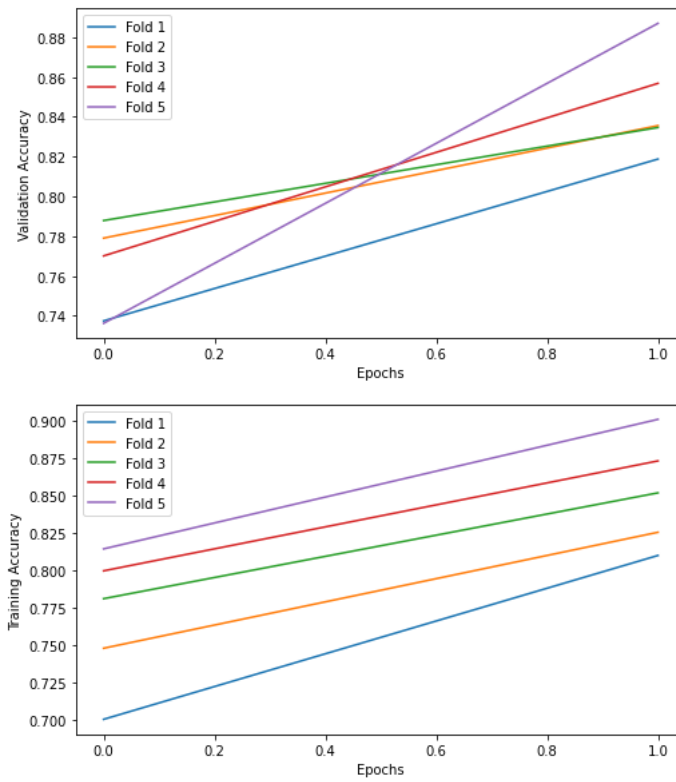


Figure 5.3: Model accuracy of Resnet 50

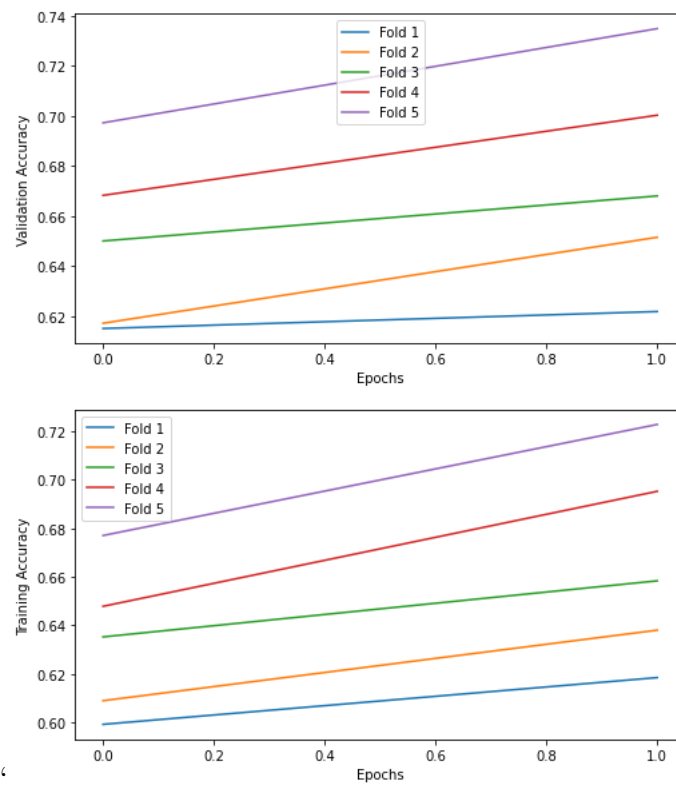


Figure 5.4: Model accuracy of VGG16

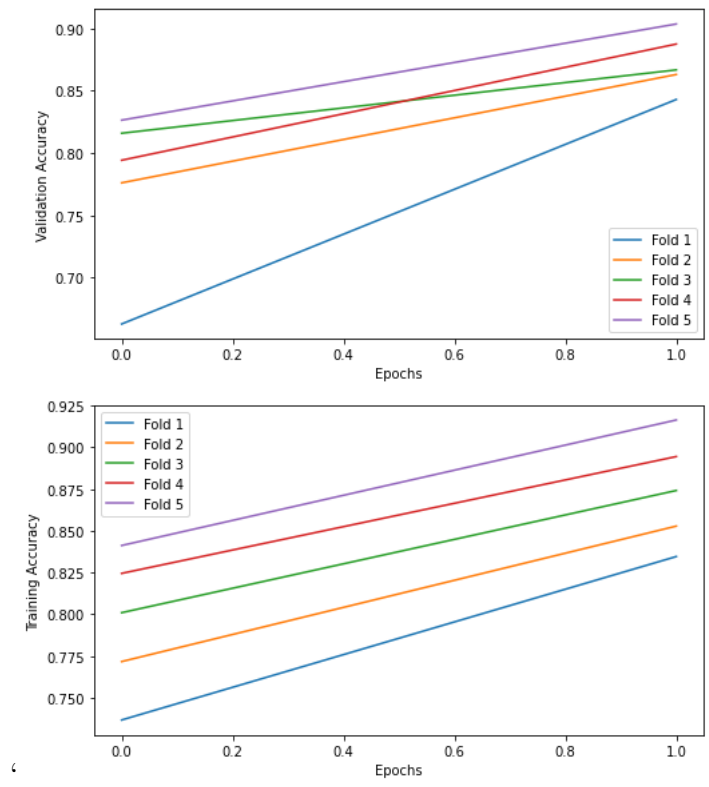


Figure 5.5: Model accuracy of Densenet 121

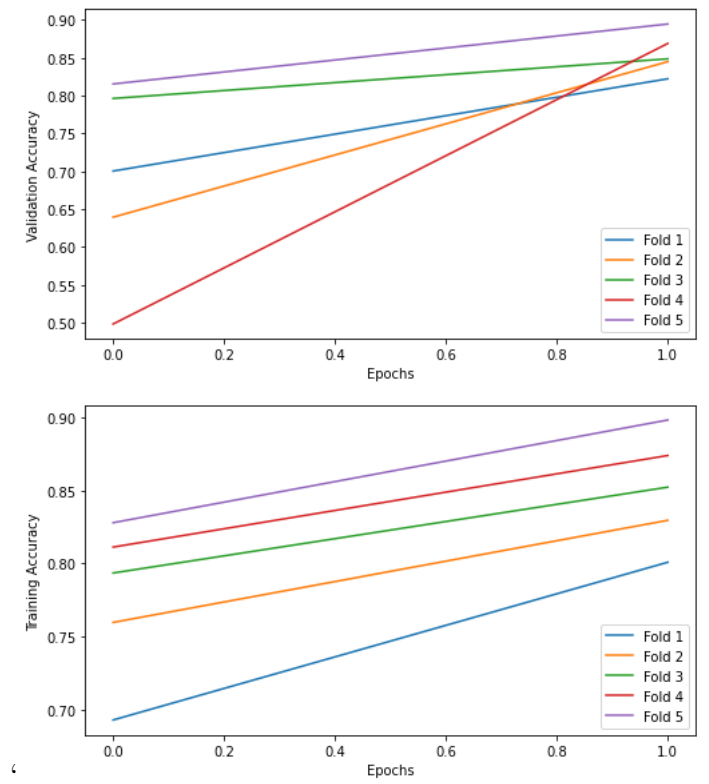


Figure 5.6: Model accuracy of Inception V3

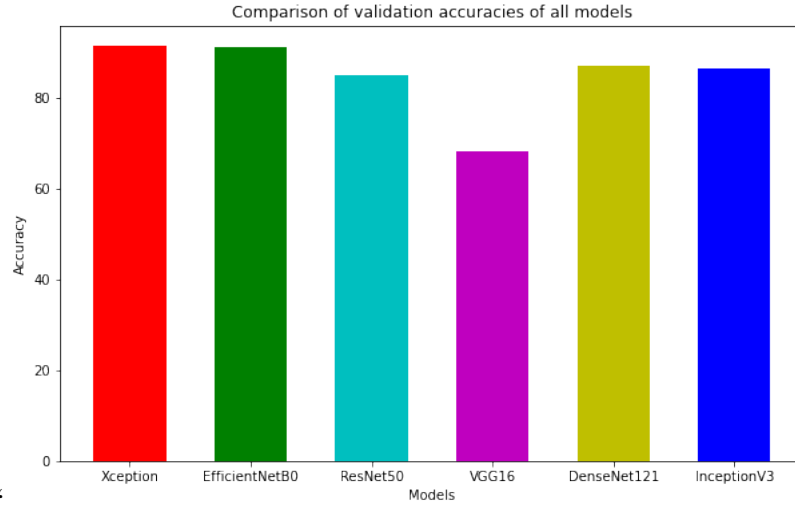


Figure 5.7: Comparison of validation accuracies of all models

## 5.1 Confusion Matrix

Figure 5.8 depicts the confusion matrix of our highest scoring technique, Xception. We claim our technique to be futuristic in Cassava classification since it outperformed all prior models. According to the confusion matrix, our model erroneously identified just 71 testing samples out of 4279 of enhanced testing data, yielding an accuracy of 98.35 percent. We can see from the figure that our strategy works well in recognizing distinct groupings (CBB,CBSD,CGM,CMD). Furthermore, since our dataset includes noise, we can see that healthy leaf is misclassified as CBB. Overall, our approach performs well across all the classes in Cassava Leaves Disease Classification.

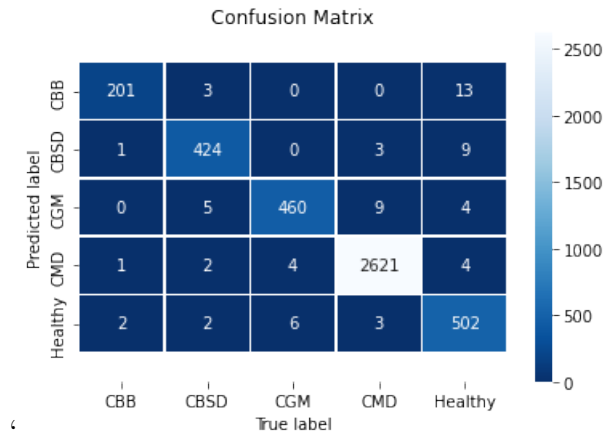


Figure 5.8: Confuison Matrix of Xception

# Chapter 6

## Conclusion and Future Scope

Our main concern was to develop a system which can detect the diseases from the cassava leaf and give the smooth and pleasure user experience to the farmers who cultivate cassava in a large amount. Intending to do that, we have to make sure that our system works absolutely fine. The hard work behind this makes it happen. Firstly, we searched out various dataset online, but at the end we stayed with one that can satisfy our requirements. Thus, this dataset was not properly augmented. As a result, we had to do augmentation for each of the images was also a very important task to do for our project. Besides, there are some datasets that were prepossessed or well augmented. Hence, it took less time to get all the trained data than we expected. Data validation and testing was also an important task for the project. After all the work done of datasets, we had to think about the algorithms that can be used for our system. We have used total 6 algorithms. Some of them are widely used and some of them are newly used in this field. These algorithms are Xception, EfficientNetB0, Resnet50, VGG16, Densenet121, InceptionV3. While we used these algorithms on our trained dataset, it gave different accuracy. For the Xception, it gave 91.3% accuracy, EfficientNetB0:91.1%, ResNet50: 85.0%, VGG16: 68.0%, DenseNet121: 87.0% and for the InceptionV3, it gave 86.4% accuracy respectively. We can see, not all the algorithms performed well. Xception and EfficientNetB0 have the highest accuracy among these. However, there's lot of things to improve our system by keep implementing our algorithms to see whether it gives same results or it shows some improvement. In addition, we don't think this is the better version of it, rather we think about future. We will keep trying to implement new algorithms on our dataset for the improvement and to decrease the time complexity of it. If we talk about the limitations, the pandemic situation is the only limitation that we've faced. For this reason, the main problem we have ever faced is to get the dataset.

### 6.0.1 Future Work

Finally, we believe that, our research is a very small step to make it tremendous one day. Our future goal is to connect computer vision with disease detection approaches for plants. In the future, we wish to enhance Xception's performance and the performance of the Convolution Neural Network and detect various diseases. We also believe that, in near future new enthusiasts will benefit overwhelmingly from our exploration as they will have an astute considered what's the expectation from a system like our own.

# Bibliography

- [1] N. Nagib Nassar and A. Marques, “Cassava leaves as a source of protein,” *Journal of Food, Agriculture and Environment*, vol. 4, Jan. 2006.
- [2] O. J. Alabi, P. L. Kumar, and R. A. Naidu, “Multiplex pcr for the detection of african cassava mosaic virus and east african cassava mosaic cameroon virus in cassava,” *Journal of Virological Methods*, vol. 154, no. 1-2, pp. 111–120, 2008. DOI: 10.1016/j.jviromet.2008.08.008.
- [3] S. Bhattacharyya, “A brief survey of color image preprocessing and segmentation techniques,” *Journal of Pattern Recognition Research*, vol. 1, no. 1, pp. 120–129, 2011.
- [4] I. U. Mohammed, M. M. Abarshi, B. Muli, R. J. Hillocks, and M. N. Maruthi, “The symptom and genetic diversity of cassava brown streak viruses infecting cassava in east africa,” *Advances in Virology*, vol. 2012, pp. 1–10, 2012. DOI: 10.1155/2012/795697.
- [5] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014. eprint: arXiv:1409.1556.
- [6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the inception architecture for computer vision*, 2015. eprint: arXiv:1512.00567.
- [7] I. Wonni, L. Ouedraogo, S. Dao, C. Tekete, O. Koita, G. Taghouti, P. Portier, B. Szurek, and V. Verdier, “First report of cassava bacterial blight caused by *xanthomonas axonopodis* pv. *manihotis* in burkina faso,” *Plant Disease*, vol. 99, no. 4, pp. 551–551, 2015. DOI: 10.1094/pdis-03-14-0302-pdn.
- [8] F. Faostat and A. C. Production, “Food and agriculture organization of the united nations, 2010,” *Roma, Italy*, 2016.
- [9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2016. DOI: 10.1109/cvpr.2016.90. [Online]. Available: <https://doi.org/10.1109/cvpr.2016.90>.
- [10] T. Kathurima, A. Nyende, S. Kiarie, and E. Ateka, “Genetic diversity and distribution of cassava brown streak virus and ugandan cassava brown streak virus in major cassava-growing regions in kenya,” *Annual Research Review in Biology*, vol. 10, no. 5, pp. 1–9, 2016. DOI: 10.9734/arrb/2016/26879.
- [11] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, “Deep neural networks based recognition of plant diseases by leaf image classification,” *Computational Intelligence and Neuroscience*, vol. 2016, pp. 1–11, 2016. DOI: 10.1155/2016/3289801. [Online]. Available: <https://doi.org/10.1155/2016/3289801>.

- [12] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jul. 2017. DOI: 10.1109/cvpr.2017.195. [Online]. Available: <https://doi.org/10.1109/cvpr.2017.195>.
- [13] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jul. 2017. DOI: 10.1109/cvpr.2017.243. [Online]. Available: <https://doi.org/10.1109/cvpr.2017.243>.
- [14] C.-M. Huo, H. Zheng, H.-Y. Su, Z.-L. Sun, Y.-J. Cai, and Y.-F. Xu, "Tongue shape classification integrating image preprocessing and convolution neural network," in *2017 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, IEEE, 2017, pp. 42–46.
- [15] D. A. Pitaloka, A. Wulandari, T. Basaruddin, and D. Y. Liliana, "Enhancing cnn with preprocessing stage in automatic emotion recognition," *Procedia computer science*, vol. 116, pp. 523–529, 2017.
- [16] A. Ramcharan, K. Baranowski, P. McCloskey, B. Ahmed, J. Legg, and D. P. Hughes, "Deep learning for image-based cassava disease detection," *Frontiers in Plant Science*, vol. 8, 2017. DOI: 10.3389/fpls.2017.01852.
- [17] S. SHARMA, *Epoch vs batch size vs iterations*, Sep. 2017. [Online]. Available: <https://developers.google.com/machine-learning/crash-course/classification/accuracy>.
- [18] A. A. Fanou, V. A. Zinsou, and K. Wydra, "Cassava bacterial blight: A devastating disease of cassava," *Cassava*, 2018. DOI: 10.5772/intechopen.71527.
- [19] A. Hussain, M. Ahmad, I. Mughal, and H. Ali, "Automatic disease detection in wheat crop using convolution neural network," Dec. 2018. DOI: 10.13140/RG.2.2.14191.46244.
- [20] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.
- [21] A. Chandy, "PEST INFESTATION IDENTIFICATION IN COCONUT TREES USING DEEP LEARNING," *Journal of Artificial Intelligence and Capsule Networks*, vol. 01, no. 01, pp. 10–18, Sep. 2019. DOI: 10.36548/jaicn.2019.1.002. [Online]. Available: <https://doi.org/10.36548/jaicn.2019.1.002>.
- [22] A. Khamparia, G. Saini, D. Gupta, A. Khanna, S. Tiwari, and V. H. C. de Albuquerque, "Seasonal crops disease prediction and classification using deep convolutional encoder network," *Circuits, Systems, and Signal Processing*, vol. 39, no. 2, pp. 818–836, Jan. 2019. DOI: 10.1007/s00034-019-01041-0. [Online]. Available: <https://doi.org/10.1007/s00034-019-01041-0>.
- [23] C. Sampoorina and D. K. Rasadurai, "Tomato plantleaf disease detection using k-means and svm classifier," vol. 5, 6 2019, ISSN: 2349-5162. [Online]. Available: [www.ijisrt.com](http://www.ijisrt.com).

- [24] U. P. Singh, S. S. Chouhan, S. Jain, and S. Jain, "Multilayer convolution neural network for the classification of mango leaves infected by anthracnose disease," *IEEE Access*, vol. 7, pp. 43 721–43 729, 2019. DOI: 10.1109/access.2019.2907383. [Online]. Available: <https://doi.org/10.1109/access.2019.2907383>.
- [25] V. Singh, "Sunflower leaf diseases detection using image segmentation based on particle swarm optimization," *Artificial Intelligence in Agriculture*, vol. 3, pp. 62–68, Sep. 2019. DOI: 10.1016/j.aiia.2019.09.002. [Online]. Available: <https://doi.org/10.1016/j.aiia.2019.09.002>.
- [26] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 6105–6114. [Online]. Available: <https://proceedings.mlr.press/v97/tan19a.html>.
- [27] A. Venkataramanan and P. Agarwal, "Plant disease detection and classification using deep neural networks," Aug. 2019.
- [28] J. Yuan and Y. Tian, "An intelligent fault diagnosis method using GRU neural network towards sequential data in dynamic processes," *Processes*, vol. 7, no. 3, p. 152, Mar. 2019. DOI: 10.3390/pr7030152. [Online]. Available: <https://doi.org/10.3390/pr7030152>.
- [29] S. Zhang, S. Zhang, C. Zhang, X. Wang, and Y. Shi, "Cucumber leaf disease identification with global pooling dilated convolutional neural network," *Computers and Electronics in Agriculture*, vol. 162, pp. 422–430, Jul. 2019. DOI: 10.1016/j.compag.2019.03.012. [Online]. Available: <https://doi.org/10.1016/j.compag.2019.03.012>.
- [30] M. Agarwal, A. Singh, S. Arjaria, A. Sinha, and S. Gupta, "ToLeD: Tomato leaf disease detection using convolution neural network," *Procedia Computer Science*, vol. 167, pp. 293–301, 2020. DOI: 10.1016/j.procs.2020.03.225. [Online]. Available: <https://doi.org/10.1016/j.procs.2020.03.225>.
- [31] T. M. BATTALA LAKSHMI THUSHARA, "Analysis of plant diseases using expectation maximization detection with bp-ann classification," vol. 13, Aug. 2020. [Online]. Available: <http://www.jctjournal.com/Volume-13-Issue-8-2020>.
- [32] N. Ganatra and A. Patel, "Performance analysis of fine-tuned convolutional neural network models for plant disease classification," vol. 13, pp. 293–305, Jan. 2020.
- [33] S. Mishra, R. Sachan, and D. Rajpal, "Deep convolutional neural network based detection system for real-time corn plant disease recognition," *Procedia Computer Science*, vol. 167, pp. 2003–2010, 2020. DOI: 10.1016/j.procs.2020.03.236. [Online]. Available: <https://doi.org/10.1016/j.procs.2020.03.236>.
- [34] H. Nazki, S. Yoon, A. Fuentes, and D. S. Park, "Unsupervised image translation using adversarial networks for improved plant disease recognition," *Computers and Electronics in Agriculture*, vol. 168, p. 105 117, 2020. DOI: 10.1016/j.compag.2019.105117.

- [35] K. Y. P. Krishna Chaitanya, “Image based plant disease detection using convolution neural networks algorithm,” vol. 5, pp. 331–334, May 2020, ISSN: 2456-2165. [Online]. Available: [www.ijisrt.com](http://www.ijisrt.com).
- [36] P. Sharma, Y. P. S. Berwal, and W. Ghai, “Performance analysis of deep learning CNN models for disease detection in plants using image segmentation,” *Information Processing in Agriculture*, vol. 7, no. 4, pp. 566–574, Dec. 2020. DOI: 10.1016/j.inpa.2019.11.001. [Online]. Available: <https://doi.org/10.1016/j.inpa.2019.11.001>.
- [37] D. V. T. and M. V. R., “Mellowness detection of dragon fruit using deep learning strategy,” *Journal of Innovative Image Processing*, vol. 2, no. 1, pp. 35–43, Mar. 2020. DOI: 10.36548/jiip.2020.1.004. [Online]. Available: <https://doi.org/10.36548/jiip.2020.1.004>.
- [38] H. R. Ayu, A. Surtono, and D. K. Apriyanto, “Deep learning for detection cassava leaf disease,” *Journal of Physics: Conference Series*, vol. 1751, no. 1, p. 012072, 2021. DOI: 10.1088/1742-6596/1751/1/012072.
- [39] A. Maryum, M. U. Akram, and A. A. Salam, “Cassava leaf disease classification using deep neural networks,” *2021 IEEE 18th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET)*, 2021. DOI: 10.1109/honet53078.2021.9615488.
- [40] C. A. Zárate-Chaves, D. Gómez de la Cruz, V. Verdier, C. E. López, A. Bernal, and B. Szurek, “Cassava diseases caused by xanthomonas phaseoli pv. manihoti and xanthomonas cassavae,” *Molecular Plant Pathology*, vol. 22, pp. 1520–1537, 2021. DOI: 10.1111/mpp.13094.
- [41] E. K. Anderson, A. G. Hager, T. B. Voigt, and D. Lee, “\* graduate research assistant, associate professor, associate professor, and assistant professor, department of crop sciences, university of illinois at urbana-champaign, urbana, il 61801. corresponding author’s e-mail: Leedk@ illinois. edu,” *STUDY OF AGRONOMIC FACTORS IMPORTANT TO THE ESTABLISHMENT OF VARIOUS PERENNIAL GRASS BIOENERGY FEEDSTOCK SPECIES*, vol. 1001, p. 46,
- [42] *Categorical crossentropy loss function: Peltarion platform*. [Online]. Available: <https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy>.
- [43] *Classification: Accuracy — machine learning crash course*. [Online]. Available: <https://developers.google.com/machine-learning/crash-course/classification/accuracy>.
- [44] F. Djibodé-Favi, *Winrock international - cassava production in bangladesh*. [Online]. Available: [https://winrock.org/volunteer\\_blog/cassava-production-in-bangladesh/](https://winrock.org/volunteer_blog/cassava-production-in-bangladesh/).
- [45] C. Shorten and T. Khoshgoftaar, *A survey on image data augmentation for deep learning. j. big data 6 (1), 1–48 (2019)*.



# Adam Optimizer Pseudo-code

---

---

●  $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \eta = 10^{-8}$  (Defaults)

$m_0 \leftarrow 0$  (Initialize 1<sup>st</sup> moment vector)

$v_0 \leftarrow 0$  (Initialize 2<sup>nd</sup> moment vector)

$i \leftarrow 0$  (Initialize step)

**while**  $\Theta_i$  not converged **do**

$i \leftarrow i + 1$

$g_i \leftarrow \nabla_{\Theta} f_i(\Theta_{i-1})$  (Get gradients at step  $i$ )

$m_i \leftarrow \beta_1 \cdot m_{i-1} + (1 - \beta_1) \cdot g_i$  (Update biased first moment estimate)

$v_i \leftarrow \beta_2 \cdot v_{i-1} + (1 - \beta_2) \cdot g_i^2$  (Update biased second raw moment estimate)

$\hat{m}_i \leftarrow m_i / (1 - \beta_1^i)$  (Compute bias-corrected first moment estimate)

$\hat{v}_i \leftarrow v_i / (1 - \beta_2^i)$  (Compute bias-corrected second raw moment estimate)

$\Theta_i \leftarrow \Theta_{i-1} - \alpha \cdot \hat{m}_i / (\sqrt{\hat{v}_i} + \eta)$  (Update parameters)

**end while**

**return**  $\Theta_i$  (resulting parameters)

---

---

Figure 6.1: Adam optimization pseudocode