AFRICAN CENTRE OF EXCELLENCE IN DATA SCIENCE

COLLEGE OF BUSINESS & ECONOMICS

TRANSFER LEARNING FOR CLASSIFICATION OF BEAN PLANT LEAF DISEASES IN PRECISION AGRICULTURE

A Dissertation submitted in partial fulfillment of the
requirements of the degree of Master of Science in Data Science
in Data Mining at the University of Rwanda

By

Millicent OCHIENG

Registration Number: 220000223

Supervisor: Prof. Weiwei PAN

Kigali, September 2022

DECLARATION

I declare that this dissertation entitled **Transfer Learning for Classification of Bean Plant Leaf Diseases in Precision Agriculture** is the result of my own work and has not been submitted for any other degree at the University of Rwanda or any other institution.

Name: Millicent OCHIENG

Student Reg. No: 220000223

Signature:

Date: 21st September, 2022

APPROVAL SHEET

This dissertation entitled **Transfer Learning for Classification of Bean Plant Leaf Diseases in Precision Agriculture** written and submitted by **Millicent OCHIENG,** in partial fulfilment of the requirements for the degree of **Masters of Science in Data Science - Data Mining Specialization,** is hereby accepted and approved. The rate of plagiarism tested using Turnitin is **15%** which is less than 20% accepted by ACE-DS.

Supervisor: Prof. Weiwei PAN

Head of Training: Dr. Ignace Kabano

DEDICATION

Dedicated to my family especially my mom and dad, who have been a constant source of support and encouragement during the entire period of conducting this research

ACKNOWLEDGEMENTS

I owe much to the wonderful people who inspired, assisted and sup- ported me from the start up to successful completion of this study. Working on this dissertation has been a fascinating journey that has not only helped me acquire new skills and experience in the field of deep learning, but it has also marked the starting point to exploring more in my career.

This research is focused on transfer learning for classification of bean plant leaf disease in precision agriculture. First and foremost, my appreciation goes to the Makerere AI lab team in collaboration with the National Crops Resources Research Institute (NaCRRI) for establishing and making the bean dataset available for this research work.

A special gratitude to my three supervisors; Professor Weiwei Pan and Professor Pavlos Protopapas from Harvard University and Dr. Melanie Fernandez from Microsoft Research for enabling me make clear and meaningful research progress throughout the entire period of conducting this research. They all monitored my work very closely, advised and guided me appropriately to ensure effective research. In addition, they helped me understand clearly the state-of-the-art approach implemented in study which is transfer learning technique. Their encouragement, patience and support for me was incredible and is very much appreciated.

At the African Center of Excellence in Data Science (ACE-DS) - University of Rwanda, Many thanks to the director Dr. Charles Ruranga, head of training Dr. Ignace Kabano and the entire administration for their humble time and support. I am also grateful to all my colleagues at the center for their support and time especially Guyo Jarso who inspired me with his great passion for Data Science.

At the school of Sciences and Aerospace Studies - Moi University where I did my BSc. in Applied Statistics with computing, many thanks to my former dean professor Ambrose Kiprop and my former lecturer Mr. Charles Mutai who believed in me, advised and motivated me to take my MSc. in Data Science at the ACE-DS.

I am grateful to the Almighty God for giving me strength, intelligence and wisdom I needed to conduct this study. Last but not least, special thanks to my family for their belief in me and continual encouragement that helped me to stay focused during the entire time of this study.

ABSTRACT

Bean plant is one of the most valuable cash crops worldwide. However, it is susceptible to serious diseases hindering its production and therefore, early detection and diagnosis is crucial. This study used a dataset contain- ing healthy and infected bean leaf images. It had **1035**, **133** and **128** images on train, validation and test sets respectively. Image data augmentation was performed on the dataset to artificially increase the training set, by creating different versions of leaves. This was done to expose the classier to a massive amount of data during training and ensure model robustness even in case of leaf angle/ light variations. The study adopted transfer learning approach using pre-trained MobileNet model and obtained **4** models from its architecture; MobileNet model without/ with data augmentation and fine-tuned MobileNet model without/ with data augmentation. The models were com- pared and fine-tuned MobileNet model with data augmentation produced the best generalization accuracy of **95%** on validation and **94%** on test sets. To validate the results obtained from the best classifier, sensitivity analysis was performed by **i.** training our best model on **10** random samples from the data. One-sample Student T - Test was used to examine the statistical difference between sample mean accuracy and our known best ac- curacy. At **5%** level of significance, this difference was found to be statistically insignificant. **ii.** Training our best classier using different random seeds and hyper-parameters (learning rates, mini-batch sizes & epochs). Similarly, there was a small difference in accuracies obtained, proving model robust- ness. A **95%** confidence interval was constructed to provide a range of values within which our overall mean could lie. The interval was **[94.94, 96.19]**, im- plying that there was a **95%** chance that the confidence interval contained the true overall mean accuracy. The robust network recorded a very low misclassification count of **7** on validation set and **8** on test set. Model interpretability results revealed that, there was a misclassification only when the model was not paying attention to the leaf or due to odd cases like presence of a hole on the leaf otherwise, there was a correct prediction. Therefore, towards precision agriculture, the robust system developed can be successfully employed in bean disease diagnosis even in case of leaf angle/ light variations.

Keywords: bean plant disease; computer vision; transfer learning; MobileNet, Precision Agriculture.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	**A**rtificial **I**ntelligence
ML	**M**achine **L**earning
DL	**D**eep **L**earning
NN	**N**eural **N**etwork
CNN	**C**onvoluntional **N**eural **N**etwork
TL	**T**ransfer **L**earning
GPU	**G**raphics **P**rocessing **U**nit
CPU	**C**entral **P**rocessing **U**nit
GPS	**G**lobal **P**ositioning **S**ystem
NaCRRI	**N**astional **C**rops **R**esources **R**esearch **I**nstitute

CHAPTER 1: INTRODUCTION

1.1 Background

Phaseolus vulgaris L. (Fabaceae) also known as the common bean or French bean is a herbaceous annual plant, grown worldwide for its edible beans, used both as dry seed and as unripe fruits. The leaf is occasionally used as a vegetable, and the straw is used for fodder (Kuete, 2014). Common bean is consumed in developed countriess as well as developing nations of the world. It is third in importance after soybean and peanut. It is widely known for its contribution to protein intake in human diet (Harlen and Jati, 2018). Apart from proteins (20–38%) and complex carbohydrates (50–60%), beans are rich in minerals, vitamins, and polyunsaturated free fatty acids (Rehman, Salariya, and Zafar, 2001).

As stated by Castro-Guerrero et al. (2016), common bean was domesticated about 8000 years ago in central Mexico and South America. In Africa, it was introduced about 400 years ago and has become a major crop underpinning nutritional and economic security for smallholder farmers in Eastern, Southern, and Central Africa, as well as a major commercial crop in these regions (Worrall et al., 2015).

In East Africa, beans are grown as a cash crop by large scale and smallholder farmers however, in most parts of the region it is grown by women for local market and for subsistence as it is a source of human dietary protein and the “meat” especially for the local people. In these regions, the cultiva- tion of beans takes place twice a year from March to April and September to October as the sowing seasons. It is cultivated in warm climatic conditions within a range of temperatures of 17.5 - 27 degrees Celsius (Source).

During its growth, bean plant is affected by a number of diseases on its roots as well as leaves hindering its yield. According to Gavhale (2014), fungi, bacteria and viruses are the three main pathogenic agents causing the crop diseases hence, diagnosis of these diseases is crucial. Misdiagnosis of these bean plant pathogenic diseases impacting the crop can lead to misuse of chemicals leading to the emergence of resistant pathogen strains, more outbreaks with significant economic loss, increased input cost and above all environmental impacts.

Current plant disease diagnosis in East African region is based on human scouting which is not only difficult, subjective and inefficient but it is also very expensive even for well-experienced experts. It is therefore important to manage these pathogens causing diseases using highly efficient and effective method with minimum damage to the environment in precision agriculture (Abed-Ashtiani et al., 2017). The availability of cheap smart-phones, digital cameras and the explosive growth on the internet have made the diagnosis relatively less complex with the availability of tools and information about the disease online (Mohanty, Hughes, and Salathé, 2016).

In other studies, different researches have used different computer vision-based models for crop disease diagnosis. These deep learning models have shown the promise to increase efficiency, although great variance in angle and light conditions of leaf images have decreased the accuracy of detec- tion. In addition, deep learning algorithms may take long hours to train and require large dataset for an accurate classification task however, acquiring these image datasets and labelling for supervised learning task is quite time- consuming.

For this reason, this research will utilize a transfer learning technique in the classification of the bean plant leaf diseases in precision agriculture us- ing images of these plant leaves. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task (Source). Transfer learning goes beyond solv- ing a specific tasks and domains. It tries to see how to leverage knowledge (Features, Weights etc.) from pre-trained models (MobileNet, MobileNetV2, VGG16, AlexNet) to solve new problems that is, training newer models and even tackling problems like less data. This research will employ the transfer learning approach using pre-trained MobileNet algorithm. More about this technique has been elaborated in Chapter 2 at the algorithm definition Sec- tion and how we adopted it has been described in Chapter 4 at the method of classification Section.

1.2 Problem Statement

Plant or crop diseases are a major threat to food security as they are important items in the reduction of quality and quantity in agriculture (Coulibaly et al., 2019). As stated earlier, beans are one of the most planted cash crops worldwide and a major source of protein especially for the East African communities. However, it is susceptible to serious diseases such as bean rust and angular leaf spot just to mention a few, which hinder its production causing significant losses hence, early detection and diagnosis is very crucial. The greatest challenge is finding a fast and accurate algorithm that detects bean crop leaf diseases and remains effective even in case of variation in light and angle of the leaf images as these conditions occur many times in real life.

This study aims at filling this gap by using bean plant leaf images (healthy/ infected), to train a robust deep transfer learning-based model (pre-trained MobileNet) to accurately classify a given bean leaf image (unseen) into different infected or healthy category. In addition, this study will also address depth perception—angle rotation and light alteration effect of the leaf images on model performance and lastly, analyze the model predictions on both misclassified and correctly predicted cases and explain the reasons for model confusion during learning the misclassified cases and model attention for the correctly predicted cases.

1.3 Justification of the Study

Plant disease diagnosis is an area of study that has been conducted by many researches worldwide focusing on different plant diseases using various traditional machine learning and deep learning techniques. However, in East Africa, no study has been done before on bean plant leaf disease recognition using the approach of transfer learning while addressing model performance in case of image angle rotation and light alteration as these variations occur in real life situations. In addition, analyzing the model predictions on both misclassified and correctly predicted cases while explaining the model confusion for misclassification cases and model attention for correct predictions cases.

1.4 Significance of the Study

This study is crucial since it aims at building a robust classifier that can accurately discriminate between the various healthy and disease categories for bean plant leaves. The system developed could empower farmers who are the primary target users, analysts, plant pathologists, researchers and agricultural extension officers from the Ministry of Agriculture who carry out various crop disease analysis by providing appropriate information for prevention and control of bean plant pathogenic diseases, reduce time for dis- ease diagnosis and increasing yield. Other benefits include; decreased chemicals use, minimized environmental degradation, decreased input cost and improved quality and quantity of agricultural products in East Africa.

Towards precision agriculture, the end users of the automated system built will make use of their smart phones to capture the infected bean leaf picture as well as its geographical location using the global positioning system (GPS). They will then upload the picture to the web application to find the correct diagnosis for the leaf. As a result, instead of spraying their farms at a certain point in time after plantation, farmers will spray the exact plants that are infected with correct herbicides.

1.5 Research Questions

This study is guided by these research questions:

a) Can we employ transfer learning approaches e.g., using pre-trained MobileNet to train a robust classifier that can accurately discriminate between healthy and infected bean leaves?

b) If we fine-tune the trained network, what will be the change in its performance?

c) How robust will the classifier be? If we change the input by leaf image angle rotation and light alteration, what will be the effect on the model performance?

d) Similarly, if we fine-tune the network trained after changing input data, how will it perform?

e) Can we evaluate and compare performance of the trained networks before/ after fine-tuning and with/ without changing input data to identify the best classifier?

f) Can we perform sensitivity analysis on the best model to prove its robustness in the presence of other uncertainties?

g) Finally, will the best classifier have any misclassifications? What will be the reasons for model confusion in classifying these classes? Can we trust the classifier? More broadly, can we explain how the network arrived at its decisions/ conclusions?

1.6 Research Objectives

1.6.1 Main Objective

To train a robust deep transfer learning model (pre-trained MobileNet) to accurately discriminate between healthy and infected bean plant leaves even in case of variation in angle and light conditions of a leaf.

1.6.2 Specific Objectives

We have five research objectives:

i. train and fine-tune pre-trained MobileNet model.

ii. test the effect of leaf image angle rotation as well as light alteration on the model performance before and after model fine-tuning.

iii. evaluate and compare the performance of the classifiers trained before/ after model fine- tuning and with/ without changing the input data to investigate which model works well.

iv. perform sensitivity analysis on the best classifier. Lastly,

v. visualizes the model attention on both correct & misclassified classes and explain how the model arrived at its decisions or conclusions.

The rest of this section basically gives an overview of how we achieved the above-mentioned objectives. However, Chapter 4 fully describes more about the methodology adopted to achieve each and every objective.

For the first objective, we considered two approaches and as a result we obtained two models. First, we trained MobileNet by freezing the pre-trained base layers and only adjusting and training the head layer of the network. This was our first classifier. Secondly, we fine-tuned the network by consid- ering the second approach whereby instead of adjusting and training only the classifier/ head layer, we also adjusted and trained the last layer just be- fore the classifier. Keeping the rest of the base layers freezed, we trained the network again and got our second model.

As mentioned before, one of the main aims of this research is to train a robust network capable of detecting bean plant leaf diseases even in case of angle as well as light variations. Data Augmentation is a technique used to artificially create modified versions of images in a training dataset. Thus, for the second objective, we considered changing the input data by performing data augmentation using two augmentation techniques. Random rotation to create leaf images with different angle variations and random brightness to create leaf images with different light alteration. We then trained and fine-tuned the network after augmenting data and obtained another two classifiers.

For the third objective, we used classification accuracy, confusion matrix and classification report evaluation metrics to evaluate the performance of the classifiers. We summarized the results on a table and compared the four classifiers based on these metrics.

After obtaining our best classifier from objective three, we achieved sensitivity analysis which is our fourth objective by considering two approaches. First, we randomly re-sampled ten samples from the dataset and trained to observe the difference in their accuracy as compared to our best classifier. We averaged sample accuracies and used one-sample Student T - Test to examine whether the difference in mean performance and our known best accuracy is statistically significant.

In the second approach, we considered training the classifier again using different random seeds. This enabled us achieve reproducibility through different initializations. We trained the network ten times each time with different random seeds. We then averaged the accuracies obtained and provided a range of values within which the accuracy could possibly lie by calculating a 95% confidence interval.

Finally for the fifth objective, we visualized the model attention to ex- plain how the network arrived at its decisions or conclusions. We achieved this by obtained saliency maps for the images in both correct and misclassified classes using Gradient-Weighted Class Activation Mapping (Grad-Cam) Algorithm.

1.7 Limitations of the Study

There are quite a number of different bean plant diseases. However, the dataset used had only 2 classes of bean disease (bean rust & angular leaf spot) and a healthy class limiting the study on only these 2 diseases.

1.8 Outline of the thesis

The rest of this thesis is structured as follows: Chapter 2 presents the technical background of this study, related works are presented in Chapter 3. In Chapter 4, the methodology adopted is described. Chapter 5 presents the detailed experimental context and lastly, Chapter 6 presents the discussions, conclusions and future work.

CHAPTER 2: TECHNICAL BACKGROUND

2.1 Definition of models

2.1.1 Neural Networks

This research implements deep learning which is simply many layers of hidden units in a neural network. Neural networks also artificial neural net- works (ANN) is an algorithm inspired by the structure and the function of human brain hence has the goal of having machines mimic how the human brain works. Unlike other models, ANN are flexible/ complex functions built by composing simple functions. For this reason, they are fast to train, they do not take too much space, they are simple to manipulate and they can approximate complex non-linear trends in a numerically stable way.

The building blocks of neural networks are called perceptrons. It can be single if the network is made up of only an input and an output layer as in Figure 2.1. On the other hand, when data is huge having many inputs that a simple perceptron cannot handle, then a third layer is added called hidden layer and the network becomes a multi-layer perceptron (MLP). An example of MLP with an input, output and two hidden layers is as presented in Figure 2.2. The input layer takes in input data into the network while the output layer gives out the output based on the input data. The hidden layer can be one or many in a network and are responsible for learning the map- ping between input and output through a series of matrix multiplications and mathematical transformations.

Every layer in a network has a basic unit called a neuron or a number of neurons where each neuron has its own weight and bias. A neuron performs both summation and activation functions. In the summation, all features are multiplied by their weights and bias are summed up. The summed function is then applied over an activation function (Source). A neural network model learns from these weights whereby training a neural networks, means updating the values of the weights until the network can make correct predictions based on the actual input. Two processes are involved; forward propagation and backward propagation.

In forward propagation, input data is fed into the network at the input layer and goes to the first hidden layer without any operations. The first hidden layer takes in the output of the input layer as its input and applies multiplication, addition and activation operation and passes this value to the next layer. The same procedure repeats for next hidden layers to the final layer where the output value is obtained.

Backward propagation process takes place after getting the predicted value from the forward propagation. The goal of a neural network is to make good predictions with minimal error. To compute the error made during forward propagation, predicted and actual values are compared and a loss function is used to calculate the error value. In back propagation, chain rule is used to compute the derivative/ gradient of error value with respect to weights starting from the last layer going backwards until all the gradients for each weight in the entire network is obtained. The derivative/ gradient value is then subtracted from the weight value to reduce the error value or loss.

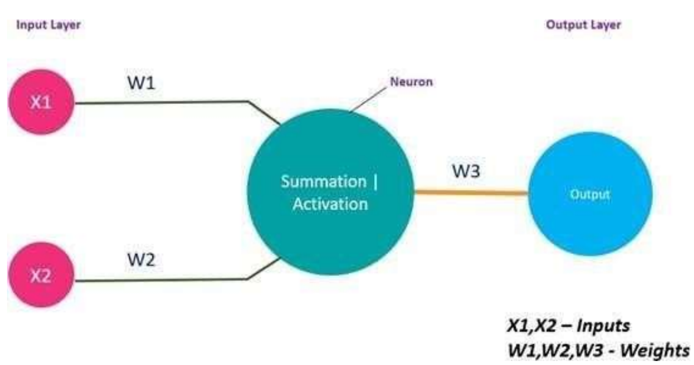


FIGURE 2.1: Single perceptron (Source)

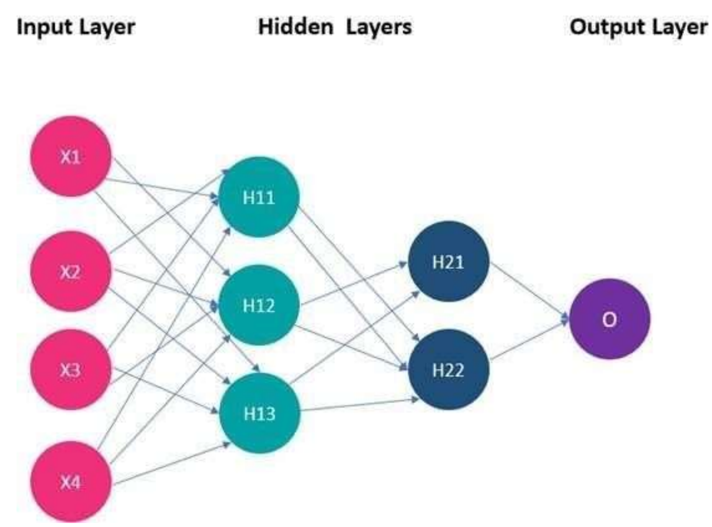


FIGURE 2.2: Multi-layer perceptron (MLP) with two hidden layers (Source)

2.1.2 Convolutional Neural Networks (CNN)

This research involves classification of images which are very high dimensional. When it comes to image processing, the amount of trainable weights suddenly becomes unmanageable for MLPs since they are designed to use one perceptron for each input. This may cause difficulties during training and probably over-fitting. Another drawback of MLPs is the fact that they are not translation invariant. Meaning, they react differently to an image in- put and its shifted versions since they assume that a particular image learnt should always appear on a given area of a frame. In image processing, nodes closer to each other help define features of an image and when we flatten an image into MLPs, spatial information is lost and this is a major problem of MLPs (Source). Convolutional neural networks (CNNs) help solve all these problems as they try to leverage spatial correlation of an image pixels or features no matter where the image may appear in a frame, they are capable of scaling appropriately for images and they can handle various image translations. For these reasons, CNNs are the most suitable class of deep neural network for computer vision. Hence, this study adopted an algorithm based on CNNs for effective image processing.

A CNN refers to a deep learning model that takes in an image input, assigns trainable weights and biases to objects in an image so as to differentiate one image from the other. CNN consists of layers that are not fully connected but have filters that introduce translation invariance and parameter sharing. Filters are sets of cube-shaped weights applied on image and their size and number are pre-determined by the user. The values for these filters are randomly chosen at first then during network training, they automatically up- date themselves. For every filter, a feature map is extracted which is taken to an activation function to add non-linearity.

As shown in Figure 2.3, a CNN consists of three types of layers; **i.** convolutional layer, **ii.** pooling layer and **iii.** fully connected layer, each of which performs different tasks. In the convolutional layers, we define network parameters by specifying the number and sizes of filters and apply them to the original image or the extracted feature maps. At the pooling layers, the dimensionality reduction takes place by either taking the maximum value in a given filter area (max-pooling) or taking the average value in a filter area (average-pooling). Finally, flattening of resulting extracted features takes place at the last layer or the fully connected layer (Source).
Deep CNN layers basically learn filters of increasing complexity. The first layers learn filters that detect basic features in an image like edge or corners, the middle layers learn filters that can recognize parts of objects and the last layers which have higher representations learn filters that detect full objects, in different shapes and positions. The more the number of filters the more features a convolutional neural network will learn.

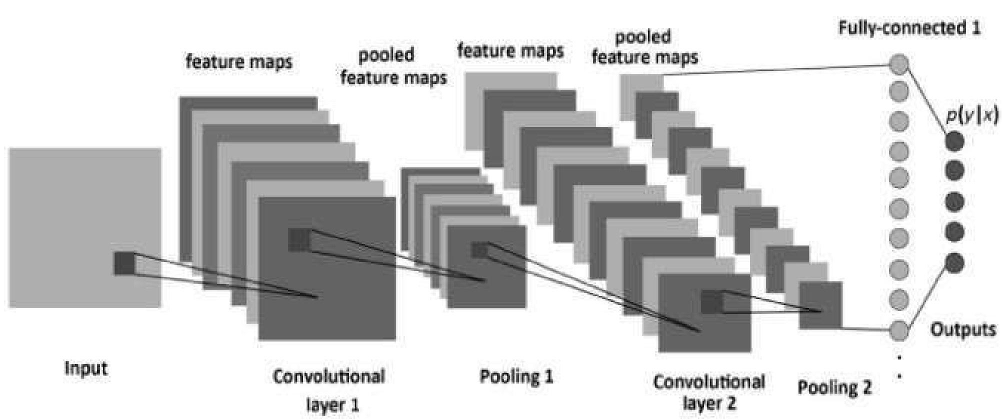


FIGURE 2.3: A CNN example with two convolutional and max pooling layers (Source)

2.1.3 Pre-trained MobileNet Model

As mentioned before, CNNs are the best for image processing as opposed to MLPs however, to produce better generalization accuracy, we need a large dataset to train. An important challenge is acquiring and labelling image data, which might be time consuming. In addition, training deep CNNs for better accuracy might take longer durations. For these reasons, this research proposed a light-weight pre-trained model based on CNN called MobileNet.

MobileNet presented by Howard et al., 2017 and open-sourced by Google is an efficient convolutional neural network that seeks to perform well on mobile and embedded vision applications with low computational power. It is TensorFlow's first mobile computer vision model. The light-weight model of 16MB has been trained on the huge ImageNet dataset with 1000 labels and 4,253,864 parameters and has shown excellent performance compared to other popular models by achieving top-5 accuracy of 89.5%. It takes less training time on CPU and can perform well even on very little data.

MobileNet is based on a streamlined architecture as shown in Figure 2.4 that uses depth-wise separable convolutions, which reduces the number of trainable parameters resulting to light weight deep neural networks as compared to a network with standard convolutions with the same depth in the network. The difference between a standard convolution and a depth-wise separable convolution is as illustrated in Figure 2.5.

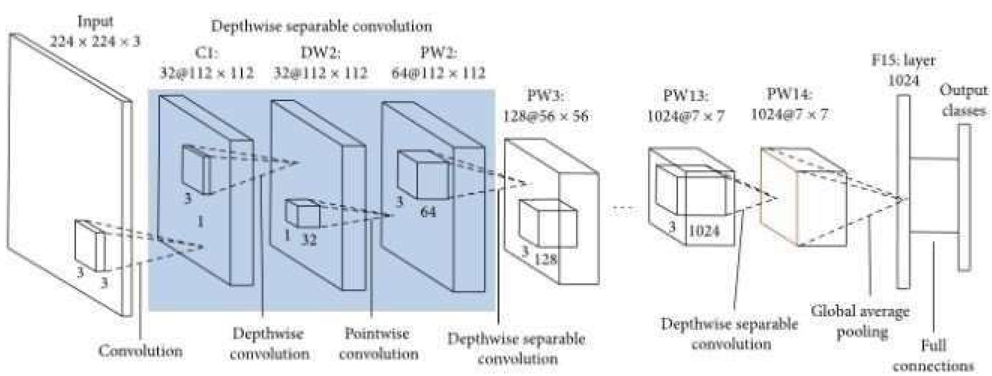


FIGURE 2.4: The architecture of MobileNet (Source)

(A) *StandardConvolution* (B) *DepthwiseConvolution*

FIGURE 2.5: A standard Vs a depth-wise separable convolution (Source)

Difference between a standard convolution in CNN and a depth-wise separable convolution in MobileNet

A traditional CNN uses a standard convolution which only deal with spatial dimensions. For example, as seen in Figure 2.5a, there is a single 3x3 convolution followed by batch normalization and ReLU activation. MobileNet on the other hand, uses a depth-wise separable convolution. This convolu- tion does not only deal with the spatial dimensions but also with the depth dimension. It is made from two operations which involve splitting a kernel into 2 separate kernels to do the two convolutions i.e depth-wise convolution and a 1x1 point-wise convolution. For example, as shown in Figure 2.5b, MobileNet separates the convolution into a 3x3 depth-wise convolution, batch normalization, ReLU activation and a 1x1 point-wise convolution followed by batch normalization then ReLU activation. The process of separating a convolution into two, reduces the number of trainable parameters making MobileNet a light-weight deep neural network (Source).

2.2 Definition of algorithms

As mentioned in the previous section, this research used pre-trained model (MobileNet) for image processing. In this section, we present how we implemented the algorithm using transfer learning.

2.2.1 The art of Transfer Learning

Transfer Learning (TL) according to Wikipedia, is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. It is an approach where a model trained on one task is re-purposed on another similar task. As shown in Figure 2.6 this technique involves training model M (source model) on one big dataset D (source dataset) for task T, storing the knowledge and ability learnt and using model M for a smaller dataset d (target dataset) for the same task T while only adjusting the weights to obtain a new model m (target model).

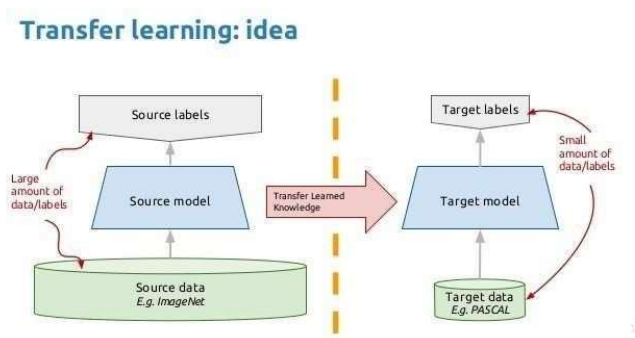


FIGURE 2.6: The idea of transfer learning (Source)

TL goes beyond solving a specific tasks and domains, tries to see how to leverage knowledge (Features, Weights etc.) from the state of the art (SOTA) models to solve new problems like training newer models and even tackling problems like less data. In this sense, learning process can be faster, more accurate and/or need less training data. These pre-trained models are available at Keras Applications together with their pre-trained weights. They are deep learning models already trained on the ImageNet dataset consisting of 1000 categories and 1.2 million images. These SOTA models include AlexNet, VGG16, ResNet50, MobileNet, MobileNetV2 etc. For the same reasons dis- cussed earlier, we employed transfer learning approach using pre-trained MobileNet model.

2.2.2 Why Transfer Learning for this Study

It is easy to implement transfer learning using pre-trained MobileNet for this study due to the following two main reasons; **a) Task similarity** It is easier to utilize TL methodology by using MobileNet for this research work since the pre-trained model (source model) was trained for classification of images which is the same task as this research.

b) Small dataset Transfer learning using the pre-trained networks is mostly applied when the dataset is small and by using pre-trained MobileNet a higher performance can be achieved compared to using deep learning mod- els like CNN. This is because deep learning algorithms requires large dataset for an accurate classification task however, acquiring these image datasets and labelling for supervised learning task is quite time consuming.

2.3 Definition of terms

In this section, all the terminologies used in this study have been defined. Machine Learning: is a part of artificial intelligence that involves the study of computer algorithms which improve automatically through experience and by the use of data. Deep Learning: is part of a broader family of ma- chine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised. Neural Networks: also, Artificial neural networks (ANN), are computing systems vaguely inspired by the structure and function of animal brains thus has the goal of making machines mimic how animal brain functions. Convo- lutional Neural Network: in deep learning, a convolutional neural network (CNN) is a class of deep neural network, most commonly applied to analyze visual imagery.

Transfer Learning: is a technique in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. **Computer vision**: is an interdisciplinary sci- entific field that deals with how computers can gain high-level understanding from digital images or videos. **Keras**: is an open-source software library that provides a Python interface for artificial neural networks.

CHAPTER 3: LITERATURE REVIEW

Several works were selected, reviewed and evaluated mainly focusing on different traditional machine learning algorithms as well as deep learning methods employed and their performance in the classification of plant leaf diseases. In addition, this chapter presents drawbacks of some of the approaches used to solve this problem and lastly, the contribution of this work that other researches did not address.

3.1 Related Works

Rangarajan, Purushothaman, and Ramesh (2018) employed transfer learn- ing approach using pre-trained AlexNet and VGG16 net pre-trained models for classifying 6 different diseases and a healthy class of the tomato crop from the image dataset of 13,262 images. They analyzed the role of number of im- ages in a dataset and the significance of hyperparameters that is; minibatch size, weight and bias learning rate in the classification accuracy and execution time. The results after comparing the two pre-trained models shown that AlexNet performed better that VGG16 net with classification accuracies of 97.49% and 97.23% for AlexNet and VGG16 net respectively.

AlexNet and VGG16 net pre-trained models used in this research are very heavy models (over 533MB), they have large weights and they are slow to train. Another drawback is that these models are very computationally expensive for small datasets and they are prone to over-fitting especially for small data. For our research, we implement pre-trained MobileNet which is a light- weight model (16MB), fast to train and very computationally inexpensive for small data.

In the recognition of novel rice blast using 2906 positive and 2902 negative samples, Liang et al. (2019) compared the features extracted and classified by CNN with features extracted by local binary patterns histograms (LBPH) and Haar-WT (Wavelet Transform) and later fed into the Support Vector Machine

(SVM) for classification. The researchers evaluated results and inferred that the high-level features extracted by CNN were more discriminative and effective than traditional handcrafted features including LBPH and Haar-WT with classification accuracies of 95.83%, 82.59%, 83.85% for CNN, LBPH + SVM and Haar-WT + SVM respectively.

The researchers used a traditional approach for feature extraction and a CNN for classification. They also used a CNN for feature extraction and a traditional machine learning technique for classification. This is quite time-consuming and computationally expensive. For our research, the same MobileNet architecture has been used for both feature extraction as well as classification. This reduces time for computation and minimizes resources.

Mishra, Sachan, and Rajpal (2020), used real time deep learning based model for classification of major corn diseases without the requirement of internet using pre-trained deep learning CNN model. After deploying the model, an accuracy of 88.46% was achieved during recognition of corn leaf diseases demonstrating the feasibility of this method.

This work used a deep CNN model which is expensive to train. This is because to achieve better generalization accuracy, it requires a large dataset and longer hours or even days to train. In our research on the other hand, MobileNet pre-trained model based on CNN has been implemented. This network trains faster due to its light-weight. In addition, the network has high feature extraction ability as a result of being originally trained from a huge ImageNet dataset with 1000 labels. Fitting the network in a relatively smaller dataset results into a better generalization accuracy.

In other studies, Krishnaswamy, Rangarajan and Raja (2020) presented Visual Geometry Group 16 (VGG16) architecture in classifying 5 major eggplant diseases and evaluated performance after converting the RGB images obtained in laboratory and field conditions into different color spaces namely Hue Saturation Value (HSV), YCbCr and grayscale. Results shown that the dataset created with RGB and YCbCr images in field condition was promising with a classification accuracy of 99.4%. The researches also used VGG16 as feature extractor from 8th convolution layer and fed these features for classification in Multi-Class Support Vector Machine (MSVM). The analysis depicted an equivalent or in some cases produced better accuracy.

Like in the previous work, VGG16 net which is a very heavy model has been used. This model easily over-fits and it is very slow to train. It is also computationally expensive to use a pre-trained model for feature extraction and a traditional machine learning technique for classification. Our research on the other hand, not only uses a light-weight network but we also have the advantage of minimizing resources by using the same MobileNet model for both feature extraction and classification.

Amara, Bouaziz, and Algergawy (2017) employed LeNet architecture as a convolutional neural network to develop a system that automates the pro- cess of classifying banana leaves diseases. The results demonstrate the effectiveness of the proposed approach even under challenging conditions such as illumination, complex background, different resolution, size, pose, and orientation of real scene images with an accuracy of 98.61% with color images and 94.44% for gray images.

This study only addressed the effectiveness of their approach under challenging conditions. In contrast, our study not only addresses the robustness of our model in challenging conditions that occur in real life such as light alterations and angle variations during prediction, but we have also explained how the model arrived at its decisions or conclusions.

3.2 Novel Contributions of this Work

This research has implemented transfer learning approach to train a robust deep learning model (pre-trained MobileNet), to accurately classify a given bean leaf image (unseen) into different infected or healthy category. In addition, this study has addressed the effectiveness of the classifier in challenging real-life situations such as angle rotation and light alteration of a leaf.

Lastly, we have analyzed the model predictions on both misclassified and correctly predicted cases and explained the reasons for model confusion during learning the misclassified cases and model attention for the correctly predicted cases.

CHAPTER 4: MATERIALS AND METHODS

4.1 System Architecture

The hardware configuration used for conducting this research was Windows 10 operating system, Graphics processing unit (GPU) was NVIDIA Tesla K80 GPU and 4GB Random Access Memory(RAM).

The software used was python programming language which is an interpreted high-level and general-purpose programming language for all data manipulation and analysis. Google Colab was used as the computational environment and TensorFlow as the deep learning framework.

4.2 Data Description

4.2.1 Dataset

This research utilized a dataset from Uganda to generalize for the East African communities (Kenya, Uganda, Tanzania and Rwanda). The dataset con- tained bean leaf images that are healthy, leaf images affected by angular leaf spot which is a bacterial disease caused by *Pseudomonas Syringae pv. Lachrymans* mostly affecting crops grown in warm areas. Lastly, leaf images affected by bean rust which is a fungal disease mostly attacking the crop in wet weather. Bean rust is easily spread by wind from one plant to another and can reduce yield up to 30% (Source).

These leaf images were taken in fields from different districts in Uganda by the Makerere AI lab in collaboration with the National Crops Resources Re- search Institute (NaCRRI), the national body in charge of research in agriculture in Uganda. For more information see the dataset here.

4.2.3 Summary Statistics of the Data

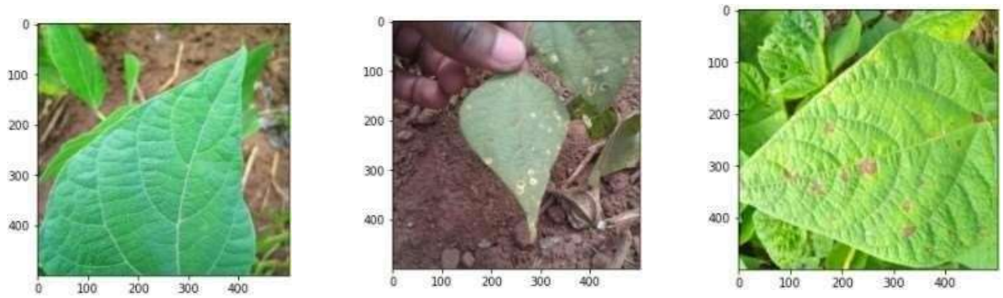
There are 3 folders provided in the dataset. **i.** Train set containing 1035 images which was used to fit the models, **ii.** validation set containing 133 images used to estimate prediction error for model selection and **iii.** test containing 128 images used for assessment of the generalization error of the final chosen model. All the sets contained the three categories healthy leaf images, leaf images affected by angular leaf spot and leaf images affected by bean rust as shown in Table 4.1. The dataset is well-balanced hence a classifier built on it have minimal chances of being biased towards a specific class.

Categories	**Train**	**Validation**	**Test**	**Total**
Healthy	342	44	42	428
Angular Leaf Spot	345	44	43	432
Bean Rust	348	45	43	436
Total	**1035**	**133**	**128**	**1296**

TABLE 4.1: Summary statistics of the 3 classes in train, validation and test sets
From this summary, different categories of images are almost equally distributed across the 3 sets. The dataset is well balanced

4.2.4 Data Visualization

Samples of bean leaf images used in this study are as shown in Figure 4.1 whereby Figure 4.1a is sample of a healthy bean leaf. Figure 4.1b represents sample of a bean leaf affected by bean rust. Lastly, figure 4.1c is sample of a bean leaf image that has been affected by angular leaf spot disease.



(A) *Healthy* (B) *BeanRust* (C) *AngularLea f Spot*

FIGURE 4.1: Visualization of the 3 categories of leaves
Visually, distinguishing bean rust from angular leaf spot is not easy due to presence of spots in both the leaves. However, distinguishing a healthy leaf from the two leaves is easy since it does not have any spot.

4.3 Data Exploration

4.3.1 Data Distribution Visualization

This is a crucial initial step in any data analysis since it helps uncover important trends in the data and points of interest. In this study, bar charts were used to visualize the distribution of image data across train, validation and test sets. This visualization was important as it helped in checking for class imbalance.

4.3.2 Image Data Augmentation

This technique creates modified versions of images in the dataset with a goal of artificially expanding the size of a training dataset. As mentioned before, this research aims at building a robust classifier that can remain effective even in the case of angle rotation and light alteration. For this reason, image data augmentation was performed to not only increase the number of images to train but also to create image variations so as to improve the generalizability of the trained classifier. This was achieved using ImageDataGenerator class available in keras deep learning library which provides the ability to augment the data automatically during training. The 2 augmentation techniques performed were random rotation and random brightness augmentations.

a) Random Rotation

This augmentation is important since images will not always be in exact angle. It involves rotating images through any degree between 0 and 360 using ImageDataGenrator via a rotation_range argument. During rotation, some pixels are moved out of the image frame leaving regions of the frame empty that is; with no pixel data which must be filled in and by de- fault fill_mode argument fills in this region using the nearest pixel value.

b) Random Brightness

This technique is also very useful because most of the times images will not be under perfect lighting condition. It involves randomly changing the brightness of an image by either darkening images, brightening images, or both. Using ImageDataGenrator class via a brightness range argument, a list of two float values representing minimum and maximum range is specified whereby values less than 1.0 darken the image whereas values larger than 1.0 brighten the image and 1.0 has no effect on brightness.

4.4 Data Processing

This research deals with analysis of image data. Normally images are very high dimensional and reducing them into a manageable size for further analysis is a crucial step. Image dimensionality reduction helps to save on computation time and storage.

Bean leaf images used in this study were of size **500 x 500**. We reduced their dimensions by transforming them into size **224 x 224** taking into account the **3** color channels to have input shape as (**224,224,3**). The input images were converted into arrays, normalized and expanded the dimensions so that the input shape is (**1, 224, 224, 3**).

4.5 Classification Method

To accurately classify the three distinct classes (healthy, angular leaf spot and bean rust), this section describes the two architectures resulting into four different models used in the study. The hyper-parameters used in each case have also been presented. Lastly, model training process have been described.

4.5.1 MobileNet Architecture and Transfer Learning Strategies

Figure 4.2 presents the body of MobileNet architecture that was used to derive two distinct architectures, using two strategies described below.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1 Conv / s1	$3 \times 3 \times 512$ dw $1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$ $14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

FIGURE 4.2: MobileNet body architecture (Source)
This is the architecture of MobileNet used to classify ImageNet dataset with 1000 categories.

c) Transfer Learning Strategy 1: Transferring the base of MobileNet on a new head

In this TL strategy, a new architecture was created by transfering the base of MobileNet on a new head in this manner; all the convolutional and the global average pooling layers of MobileNet model also termed here as the base was obtained, leaving out the fully connected layer or head that was meant to classify 1000 different categories of ImageNet dataset. These layers can be observed from Figure 4.2. A new head or classifier was then added to the pre-trained base, with the number of neurons of the new classifier equal to the number of groups (classes) of the new dataset. During training, the base was frozen and only the adjusted classifier was re-trained. In other words, this is transferring the ability of MobileNet to extract features from the new data.

The weights of MobileNet were designed to perform classification task on a huge dataset implying that fitting the model on a relatively small dataset could possibly result into over-fitting. To handle this, L1 regularization technique was employed. Therefore, while initializing kernel and bias weights at the fully connected or dense layer, they were also regularized using kernel and bias regularizers respectively. L1 regularization was specified as 0.005 which was found to be the optimal value in both cases. This prevented the network from over-fitting.

Using this strategy, 2 models were trained; MobileNet model without and with data augmentation.

FIGURE 4.3: Transfer learning using pre-trained MobileNet strategy 1

Input data is fed into the network to get predictions. The pre-trained base (convolutional layers) in the white rectangular box is frozen and not trained. Only the head/ classifier in the blue box is trained

d) Transfer Learning Strategy 2: Transferring the base of MobileNet on a new head by fine-tuning the base

Another architecture was also created with a slight difference from the first strategy. Here, the pre-trained MobileNet base was transferred on a new head but this time, unfreezing the last convolutional layer of the network just before the global average pooling and fully connected layer as observed in Figure 4.2. This technique of adjusting weights slightly to fit the new data without changing them very much is what we refer here as fine-tuning. In this sense, the ability of MobileNet to learn more patterns or extract more features in the data is increased without destroying the whole architecture.

Fine-tuning pre-trained MobileNet, increases the number of trainable parameters. This makes it even more expressive which would lead to over- fitting on a relatively smaller data. Hence like in the first strategy, the kernel and bias weights of the network were regularized. L1 regularization was specified as

0.005 which was found to be the optimal value. The fine-tuned layer together with the classifier or the head were re-trained with smaller learning rate due to increased number of trainable parameters.

Using this strategy, 2 models were trained; fine-tuned MobileNet model without and with data augmentation.

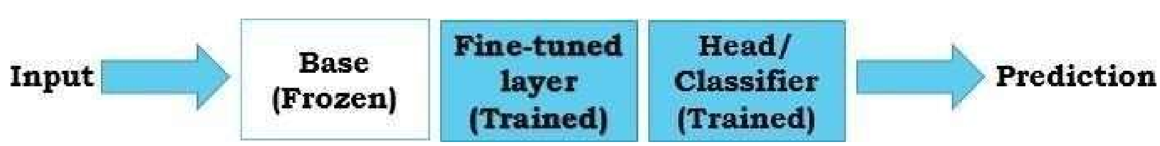


FIGURE 4.4: Transfer learning using pre-trained MobileNet strategy 2

Input data is fed into the network to get predictions. The rest of the pre-trained base (convolutional layers) in the white rectangular box is frozen and not trained. Only the last layer together with the head/ classifier in the blue rectangular boxes are trained

4.5.2 Hyper-parameter Settings

During model building, different hyper-parameters were used. While train- ing the models, we monitored the trace plots and classification accuracy at each run. Based on the best trace plots and highest accuracy obtained, the best hype-parameters were chosen. During this training time, we also fixed a random seed such that at every run, same results were obtained. In this sub- section, we present the optimal hyper-parameters used to train the models before/ after fine-tuning and without/ with data augmentation.

As mentioned in the previous section, not all the layers of MobileNet architecture were trained. We transferred the feature extraction ability of the convolutional layers (base) and only trained the dense layer/ fully connected layer. At this layer, softmax activation function was used to obtain the predictions. Softmax was used because this study involves classification of three classes (multi-classification).

For model optimization, Adam optimizer with a specified learning rate was used. A learning rate of **0.003** was found to be optimal for training models before fine-tuning. However, after fine-tuning the network, the number of trainable parameters was increased hence smaller learning rate of **0.0001** was found to be optimal. Again, due to classification of three distinct categories, loss function employed was Categorical Cross-Entropy which is the loss function used for multi-class classification and metric used was accuracy. The models were trained on train data and validated on validation data with a minibatch size or batch size = **16**. Training duration was longer for the networks with than without data augmentation. For training without data augmentation, an optimal value of **100** epochs and **25** epochs were used before and after model fine-tuning respectively.

For training with data augmentation, we increased the training duration to an optimal value of **150** epochs and **50** epochs before and after model fine- tuning respectively. As mentioned earlier, augmenting input data not only increased the data points but it also introduced different versions of data which were hard to classify. Hence when the training duration was in- creased, the models classification performance on these data points was also increased. Tables 4.2 & 4.3 gives a simplified summary of the hyper-parameters employed in this study to train our four different classifiers. These classifiers are **i.** MobileNet model without data augmentation, **ii.** fine-tuned MobileNet model without data augmentation, **iii.** MobileNet model with data augmentation, **iv.** fine-tuned MobileNet model with data augmentation.

Without Data Augmentation		
Hyper-parameter	**Before fine-tuning**	**After fine-tuning**
Learning rate	0.003	0.0001
Epoch	100	25
Batch size	16	16
Kernel weight	0.005	0.005
Bias weight	0.005	0.005

TABLE 4.2: Hyper-parameter settings for the networks without data augmentation

The hyper-parameters used were same before and after fine-tuning except learning rate and epochs. Smaller learning rates was used after fine-tuning due to increased number of trainable parameters. In addition due to fine-tuning, the network was more expressive and a shorter duration was needed to learn.

With Data Augmentation		
Hyper-parameter	**Before fine-tuning**	**After fine-tuning**
Learning rate	0.003	0.0001
Epoch	150	30
Batch size	16	16
Kernel weight	0.005	0.005
Bias weight	0.005	0.005

TABLE 4.3: Hyper-parameter settings for the network with data augmentation

Similarly for training with data augmentation, a smaller learning rate was used after model fine-tuning due to increased number of parameters. In addition, as a result of fine-tuning, the network was more expressive and a shorter duration was needed to learn.

4.5.3 Model Training Process

All the four models followed the same training process. As mentioned in Chapter 2, training a neural network involves two stages; forward and backward propagation. Figure 4.5, has been used to illustrate this procedure and explain how the training process occurred.

In forward propagation, bean leaf images were fed into the network as input data $\boldsymbol{x}$. MobileNet pre-trained model was used to learn a mapping function $\boldsymbol{f(x)}$ to give a correct output $\boldsymbol{y}$ the target. This network as seen in Figure 4.2, consists of several convolutional and pooling layers to extract as many features as possible. In between them is the ReLu activation function applied elementwise that is ***f (x)*** = ***max (0, x)*** with a threshold set at zero. Feature map sizes are then reduced and more complex features are extracted

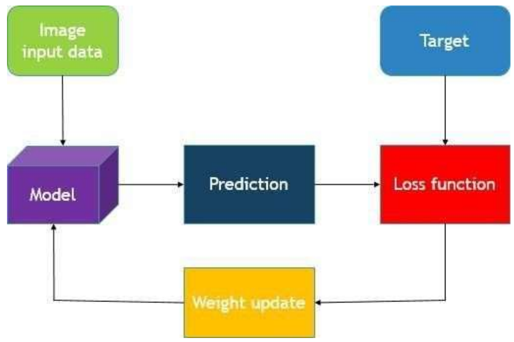


FIGURE 4.5: Training process

after several convolutional and pooling layers.
The trained network makes predictions at the last layer (fully connected layer) with the help of softmax activation function given by:

$$\textbf{Softmax}(\mathbf{x_i}) = \frac{\exp(\mathbf{x_i})}{\sum_j \exp(\mathbf{x_j})}$$

Where x_i is the input vector, exp (x_i) is the standard exponential function for input vector and exp (x_j) is the standard exponential function for output vector. Softmax activation function normalizes the output of a network to a probability distribution over predicted output classes in this case **3**. The fully connected layer is also the loss layer that is used to specify how the network training penalizes the deviation between the predicted and true labels.
Therefore, for every ***i*** predictions, the predicted network outputs and their corresponding target values, the error or loss ***L*** made during the forward propagation is computed. Again, due to classification of 3 classes, categorical cross-entropy loss function was employed. It is given by:

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

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. In backward propagation, the weights are changed according to the gradient descent direction of an error or loss surface ***L***. Learning is achieved by adjusting the weights such that $\hat{y}_i$ (model output/ prediction) is as close as possible or equals to corresponding y_i (target).

4.6 Model Evaluation Methods

Model evaluation is a very crucial part of building an effective machine learning model since it gives an explanation on the model performance. To evaluate a model, evaluation metrics are employed to discriminate among the model results. This research adopted 3 model evaluation metrics that is; classification accuracy, confusion matrix and classification report.

4.6.1 Classification Accuracy

Accuracy measures how close measurements are to the specific value. This metrics works by comparing the predictions of a classification model with the known labels. It is then calculated as the number of correct predictions divided by the total number of predictions.

It is given by:

Accuracy = Correct Predictions / Total Predictions

This research employed this metrics to understand the overall proportion of correct predictions to total predictions for the four models trained.

4.6.2 Confusion Matrix

Evaluation of a classification model should not depend on classification accuracy alone since this metrics can be misleading in these cases: **i.** when the number of classes in a dataset exceed two like in this study we have three classes and if we get a high accuracy, we would not know whether it is be- cause all the three classes are being predicted equally well or if only one or two classes are being neglected by the classifier. **ii.** though not the case in this study, when the number of observations in each class are unequal be- cause the classifier can be highly biased towards the majority class. This research thereby calculated classification matrix to give a better understand- ing of what the classifier was getting right as well as what types of errors it was making during classification. A confusion matrix refers to a table used to summarize by each class, the number of correct and incorrect predictions of a classifier with count values.

		Actual Values		
		Positive	Negative	Total
Predicted Values	Positive	*TP*	*FP*	$TP + FN$
	Negative	*FN*	*TN*	$FN + TN$
	Total	$TP + FN$	$FP + TN$	N

TABLE 4.4: Confusion matrix for binary classification

From table 4.4, true positive (**TP**) are the actual positives which the classifier correctly predicted as positive. True negative (**TN**) are the actual negative that the classifier correctly predicted as negative. **TP** and **TN** are both termed as correct predictions for the positive and the negative classes respectively. False positive (**FP**) are the actual negative but the classifier wrongly predicted as positive. We also call this type I error. Lastly, false negative (**FN**) are the actual positive but the classier wrongly predicted as negative. This is also known as type II error. **FP** and **FN** are both called the incorrect/ misclassified predictions for the positive and the negative classes respectively.

4.6.3 Classification Report

Like confusion matrix, this classification report also uses true positives, false positives, true negatives and false negatives to measure how many predictions are true and how many are false. It measures the quality of predictions from a classifier. First from the classification report, this study used **precision = TP / (TP + FP)** to determine the reliability of the model. It tells the proportion of the correctly predicted cases that actually turned out to be positive. Secondly, this study considered **recall = TP / (TP + FN)** which tells the proportion of the positive cases that were successfully predicted by the classifier. Lastly, we considered **F1 Score = 2*(Recall * Precision) / (Recall + Precision)** which is the weighted harmonic mean of precision and recall. F1 score value of 1.0 indicates best score while a value of 0.0 signifies worst performance.

4.7 Sensitivity Analysis Methods

Sensitivity analysis is a technique used to evaluate the robustness of a machine learning model prediction, when the input data is changed. Out of the four trained models, sensitivity analysis was performed on the best model to check how robust it is using the following described two approaches; **i.** random re-sampling and **ii.** different initialization using different random seeds.

In random re-sampling approach, ten samples were obtained from the dataset and trained with fine-tuned MobileNet model with data augmentation to observe the difference in their accuracy as compared to the best classifier. The sample accuracies were averaged and one-sample Student T - Test was employed to examine whether the difference in the sample mean and our known best accuracy is statistically significant.

In the second approach, the goal was to achieve reproducibility of the best network through different initializations. Hence, the classifier was trained again using ten different random seeds. The ten accuracies obtained were averaged and we provided a range of values within which we were sure the accuracy could possibly lie by calculating a 95% confidence interval.

4.8 Model Interpretation Method

Machine learning (ML) models especially neural networks used in this research can be incredibly good at achieving high accuracy as they are com- plex and very expressive. However, they often don't give an explanation on which input features were used to draw conclusions to the output and / or which neuron played what role during the classification. Hence, we call neural network models "black boxes". Model interpretation is therefore crucial, so as to understand what the classifier was paying attention to during classification. In this sense we are can trust the model. To visualize the model attention, we adopted Gradient-Weighted ClassActivation Mapping (Grad- Cam) Algorithm to obtain saliency maps for the classified images.

Grad-Cam is a local back-propagation based interpretation method for CNNs. This algorithm is applied after training a network and obtaining the classifications. A classified image is then fed back into the network to calculate the Grad-Cam heat-map for that image for a chosen class of interest. It works by finding the last convolutional layer in the network and examining the derivative or gradient information flowing into that layer. The output of Grad-Cam is a heatmap visualization for a given class label.

4.9 The Conceptual Framework

The whole procedure employed from data acquisition is as presented in Figure 4.6. Data was acquired and explored. After pre-processing, data was trained while in another experiment data was augmented first before training. After model evaluation from validation and test results, the best model was obtained. Sensitivity analysis was then performed on this best classifier and finally, model interpretation.

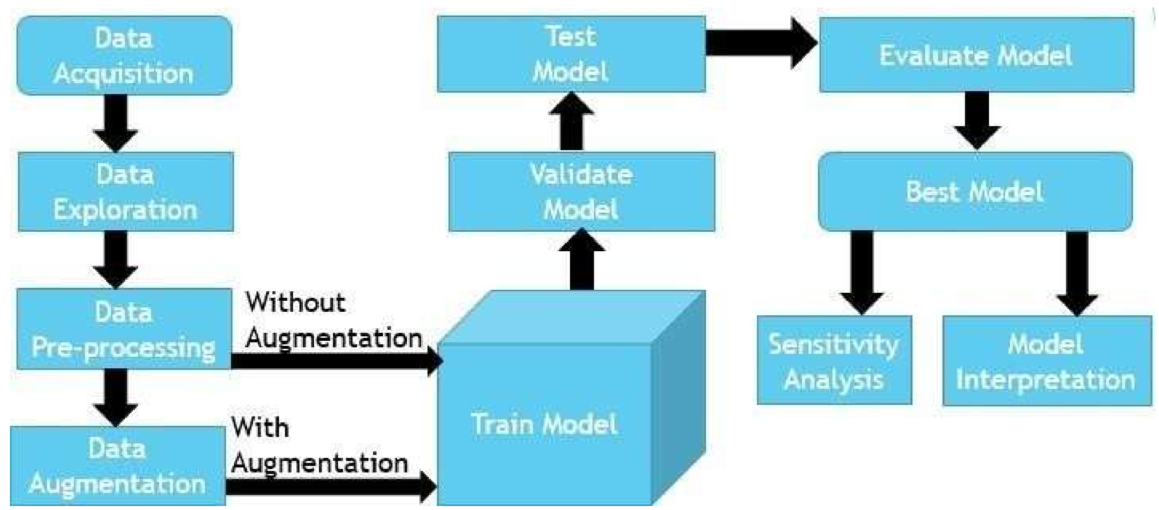


FIGURE 4.6: Conceptual framework

CHAPTER 5: EXPERIMENTS AND RESULTS

This chapter entails various experiments performed and results of research presented as per the objectives. Prior to that, experiments and results obtained on exploratory data analysis are presented. These include data visualization and data augmentation.

5.1 Data Visualization

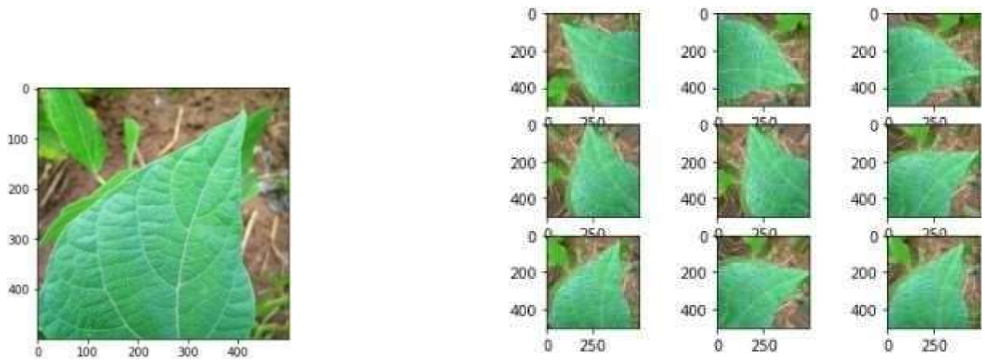
Bar graph was used to visualize the distribution of images in different classes as seen in Figure 5.1.

FIGURE 5.1: Distribution of images in train, validation and test sets

As shown by the length of bars, the dataset is well balanced across the sets

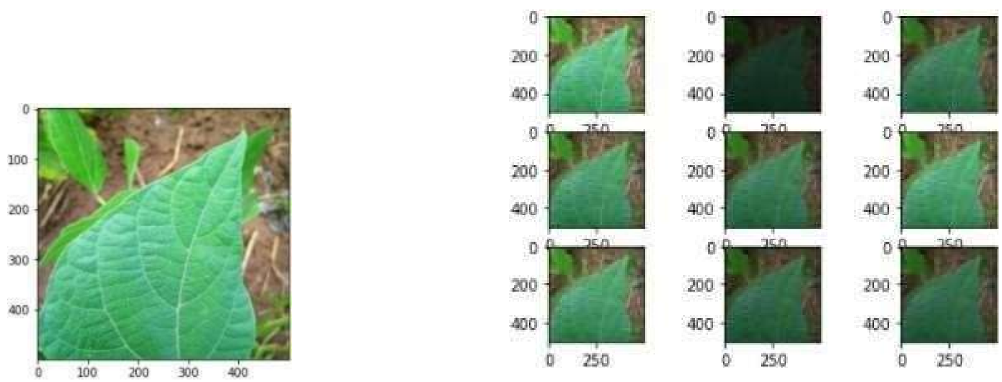
5.2 Data Augmentation

In this study, random rotation was achieved with rotation range argument specified as 45 degrees. Figure 5.2 presents the original image and how the image was transformed after random rotation. On the other hand, random brightness was performed with brightness range argument set between 0.2



(A) *OriginalLea f Image* (B) *RotatedLea f Images*

FIGURE 5.2: Original vs. images generated through random rotation
Original image looks easy to categorize but categorization of different rotated versions might not be easy



(A) *Original Image* (B) *Lea f ImagesWithLightAltered*

FIGURE 5.3: Original vs. images generated with light altered
Original image looks easy to categorize although categorization of different versions created by light alteration might not be easy

and 1.2. Figure 5.3 shows the original image and how the image was transformed after random brightness.

This visualization of the transformed images was important as we were able to see the different versions of the leaf images that we were to train our model with.

5.3 Model Training Without Data Augmentation

With already discussed architecture in Chapter 4, we trained 4 models. In this section we present two models trained without data augmentation. The results in Table 5.1 shows different accuracies obtained before and after fine- tuning the model.

Model Type	Train Accuracy	Validation Accuracy	Test Accuracy
MobileNet Model without Augmentation	97%	93%	88%
Tuned MobileNet Model without Augmentation	99%	95%	92%

TABLE 5.1: Classification accuracy for the 2 models trained without data augmentation

Both classifiers have good generalization performance although fine-tuning the classifier increased accuracy across all sets

5.4 Model Training With Data Augmentation

On the other hand, this section presents two models trained with data augmentation. The results as in Table 5.2 shows different accuracies obtained before and after fine-tuning.

Model Type	Train Accuracy	Validation Accuracy	Test Accuracy
MobileNet Model with Augmentation	91%	92%	88%
Tuned MobileNet Model with Augmentation	99%	95%	94%

TABLE 5.2: Classification accuracy for the 2 models trained with data augmentation

Both classifiers have good generalization performance although like the previous case, fine-tuning the classifier increased accuracy across all sets

5.5 Model Evaluation and Comparison of Classifiers Performance

Model evaluation was done using classification accuracy, confusion matrix and F1 score from the classification report for all the models trained with/ without data augmentation. The confusion matrix and classification report for the models are as in appendix B. The four models classification accuracy together with the number of misclassified cases for each model have been summarized and compared in Figure 5.4.

Model Type	Train Accuracy	Validation Accuracy	Test Accuracy	No. of Val Misclassifications	No. of Test Misclassifications
MobileNet model Without Augmentation	97%	93%	88%	9	15
Tuned MobileNet model without Augmentation	99%	95%	92%	7	10
MobileNet model with Data Augmentation	91%	92%	88%	11	15
Tuned MobileNet model With Data Augmentation	99%	95%	94%	7	8

FIGURE 5.4: Comparison of the 4 models trained

All the models have good generalization ability. However, accuracy is lesser before fine-tuning especially for the case of classification with data augmentation. This may be due to introduction of different version of leaves which are harder to classify.

Fine tuning the classifier, gave us the best model out of the four with way less cases of misclassifications in both validation and test sets

5.6 Sensitivity Analysis

After obtaining our best classifier, we validated model robustness by performing sensitivity analysis using two approaches; **i.** random re-sampling and **ii.** different initialization using different random seeds. Hence, all the plots in this section corresponds to validation of our best classifier.

a) Random Re-sampling

We randomly obtained ten samples from the dataset and trained them using fine-tuned MobileNet model with data augmentation which was our best classifier. The goal was to examine whether there is a difference between the sample mean accuracy and our best classifier and if the difference is statistically significant at **5%** level of significance. As indicated by the box plot in Figure 5.5, the accuracy values were in the range between **94.5%** and **96.9%** with an average of **95.4%.**

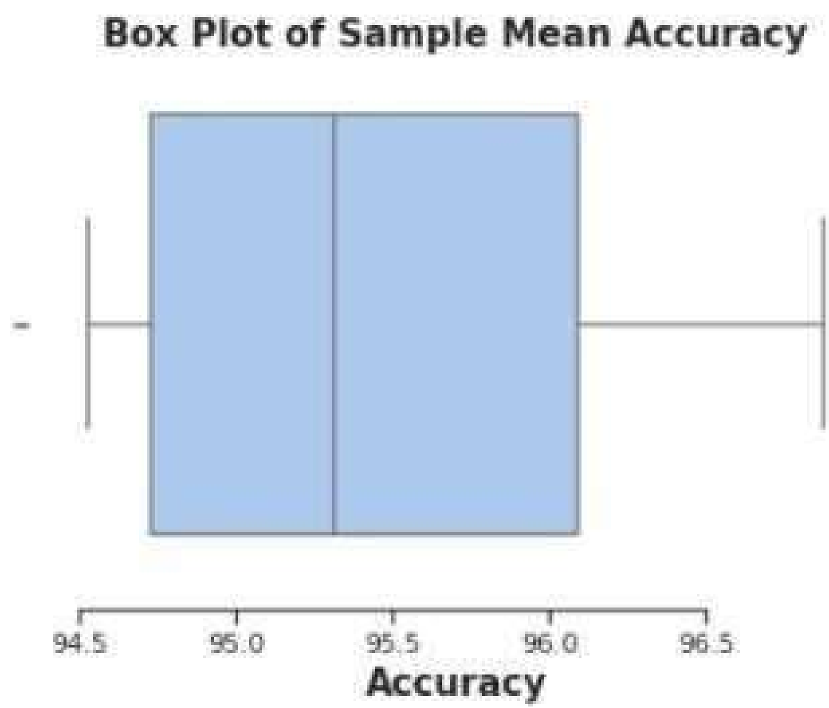


FIGURE 5.5: Box plot showing the sample accuracy range
From the box plot, the highest accuracy produced by the samples was ***96.9%*** *and the lowest was* ***94.5%****. It is also important to note that the accuracy was centered around* ***95.4%***

The sample mean accuracy was **95.47%**, approximately (**95.5%**) and our best accuracy **95%**. At **5%** level of significance, we performed one-sample Student T - Test to examine whether the difference in the sample mean and our known best accuracy is statistically significant.

Under **H0:** There is no statistical difference between the sample mean and our known best accuracy.

Vs **H1:** There is statistical difference between the sample mean and our known best accuracy.

We reported a **P-value = 0.0994** and failed to reject **H0** since the P-value obtained was greater than the **0.05**.

We then concluded that at **5%** level of significance, the difference between the sample mean and our known best accuracy is not statistically significant.

Further robustness check was performed using this approach by training the model under different minibatch sizes, learning rates and number of epochs each time recording the overall validation accuracy and mean validation accuracy of the ten samples.

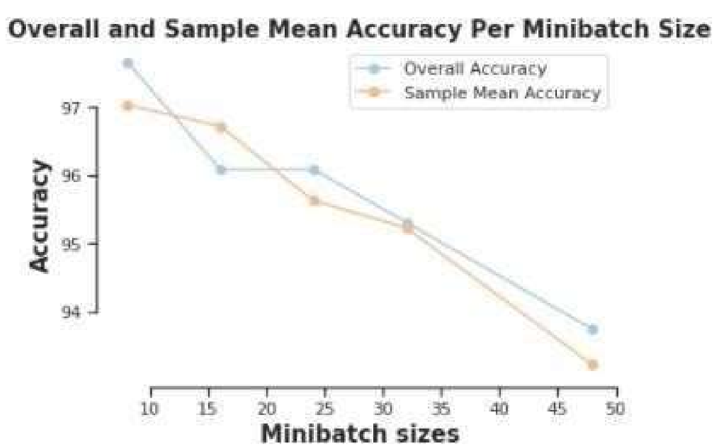


FIGURE 5.6: Line plot showing the overall and sample mean accuracy at different minibatch sizes

Accuracy decreased as the minibatch sizes increases. However a minibatch size of ***16*** *was optimum for this study. In all the batches the difference between the overall and sample mean accuracy is very minimal.*

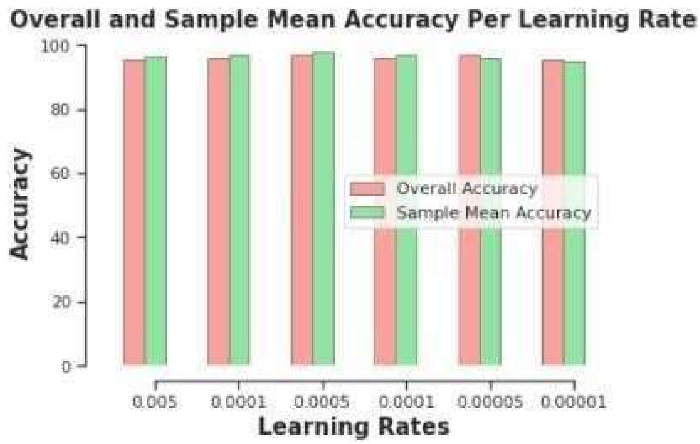


FIGURE 5.7: Bar plot showing Overall and sample mean accuracy at different learning rates

From the length of the bars at every learning rate, the difference between the overall and sample mean accuracy is minimal.

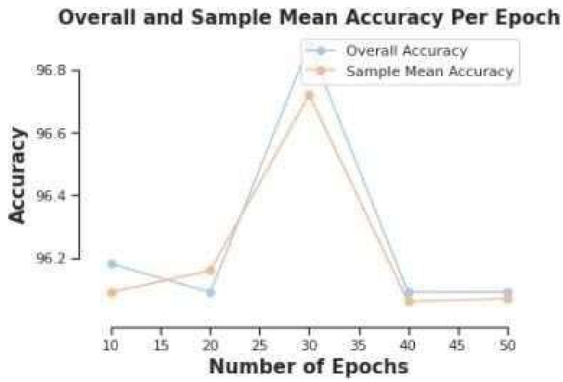


FIGURE 5.8: Line plot showing overall and sample mean accuracy at different number of epochs

*Accuracy is highest at **30** which was the optimal number epochs. Accuracy decreased and nearly the same for other epochs. In all the epochs the difference between the overall and sample mean accuracy is very minimal*

b) Random Initialization

To achieve reproducibility through different initialization, we trained our best classifier (fine-tuned MobileNet model with data augmentation) ten times each time with different random seeds. We obtained the overall accuracy as well as the mean accuracy for different samples in each run. The results are as summarized in Table 5.3.

Random Seed	**Overall Accuracy**	**Sample Mean Accuracy**
0	96.09%	96.72%
123	95.31%	95.70%
420	95.0%	95.31%
233	94.88%	94.33%
911	96.09%	96.09%
42	96.88%	97.03%
456	94.45%	94.66%
1000	95.31%	96.41%
100	96.88%	96.48%
500	94.75%	94.48%

TABLE 5.3: Overall and sample mean accuracy at different random seed

From the table, the difference between the overall and sample mean accuracy is minimal

By training using different initializations, the goal was to test whether there is a huge difference between the overall accuracy obtained from different random seeds and sample mean accuracy at each run. The length of bars showing the difference is as observed in Figure 5.9.

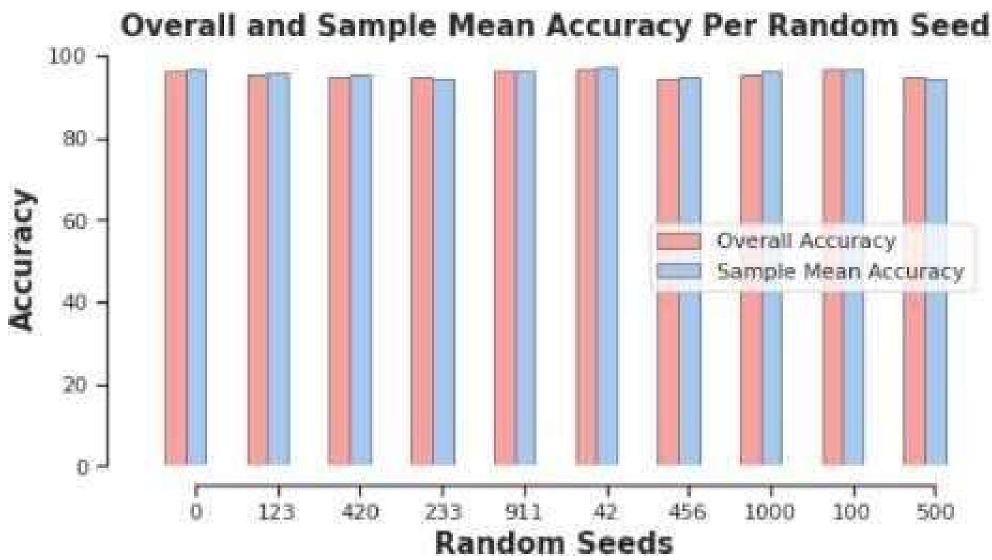


FIGURE 5.9: Bar plot showing overall and sample mean accuracy at different random seeds

From the length of the bars for both overall and sample mean accuracy, the accuracy difference is minimal at every random seed

Therefore, to conclude the sensitivity analysis, we averaged the overall accuracies and provided a range of values within which the accuracy could possibly lie by calculating a 95% confidence interval.

95% confidence interval = (94.94, 96.19)

We concluded that there is a **95%** chance that the confidence interval of **[94.94, 96.19]** contains the true overall mean accuracy of disease classification. In other words, there is only **5%** chance that the true overall mean accuracy of classification is less than **94.94%** or greater than **96.19%**.

5.7 Visualization of Model Attention

The misclassified and correctly predicted cases from the 3 categories (healthy, angular leaf spot and bean rust) were extracted from the last convolutional layer of the network to obtain saliency maps. Using Gradient-Weighted Class Activation Mapping (Grad-Cam) Algorithm, saliency maps for the retrieved cases were obtained to visualize the model attention and to explain how the model arrived at its decisions or conclusions. The red heatmap color on the saliency maps depict the part of the image contributing more to the model prediction.

a) Saliency maps for the misclassified predictions

Figures 5.10, 5.11 and 5.12 represent the original images for each class with the corresponding saliency maps.

b) Saliency maps for the correct predictions

Figures 5.13, 5.14 and 5.15 represent the original images for each class with the corresponding saliency maps.

(A) *Healthy* (B) *SaliencyMap*

FIGURE 5.10: Original vs. Saliency map of the image misclassi- fied as bean rust
Model not focusing on the leaf hence misclassification

(A) *AngularLea f Spot* (B) *SaliencyMap*

FIGURE 5.11: Original vs. Saliency map of the image misclassified as bean rust

Model not focusing on the leaf hence misclassification

(A) *BeanRust* (B) *SaliencyMap*

FIGURE 5.12: Original vs. Saliency map of the image misclassified as angular leaf spot

Model focusing on the leaf although presence of holes on the leaf might have led to misclassification

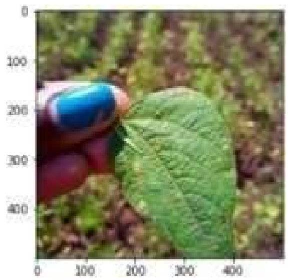
(A) *Healthy* (B) *SaliencyMap*

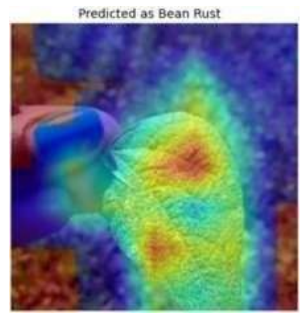
FIGURE 5.13: Original vs. Saliency map of the image correctly classified as Healthy

Model paying attention to the leaf hence correct prediction

(A) *AngularLea f Spot* (B) *SaliencyMap*

FIGURE 5.14: Original vs. Saliency map of the image correctly classified as angular leaf spot
Model paying attention to the leaf hence correct prediction





(A) *BeanRust* (B) *SaliencyMap*

FIGURE 5.15: Original vs. Saliency map of the image correctly classified as bean rust
Model paying attention to the leaf hence correct prediction

CHAPTER 6: DISCUSSION, CONCLUSION AND FUTURE WORK

This section summarizes of all experimental findings as per the objectives of the study. Conclusions and future works are also presented.

6.1 Discussion of Experimental Results

To train and fine-tune a pre-trained MobileNet model was our first objective. Using transfer learning, we successfully trained and fine-tuned the network on our bean dataset collected from various fields in districts of Uganda, East Africa. Prior to model training, we visualized the distribution of data on the train, validation and test sets provided. The dataset was well balanced, therefore a classifier built on it cannot be biased towards a particular class. It contained three distinct categories; healthy leaves, leaves affected by angular leaf spot and bean rust diseases. We viewed a sample of leaf images from the three categories and realized that visually, distinguishing bean rust from angular leaf spot was not easy due to presence of spots in both leaves. However, distinguishing a healthy leaf from the two leaves was easy due to absence of spots on the leaf.

We achieved our first objective by freezing the base of MobileNet and training only the dense layer of the network. The model obtained performed well with **97%**, **93%** and **88%** accuracy on the train, validation and test sets respectively. We fine-tuned this classifier by not only training the dense layer but also adjusting and training the last convolutional layer just before the fully connected layer. We obtained a model with increased generalization accuracy, having **99%** on train, **95%** on validation and **92%** on test set. Both the generalization accuracies before and after fine-tuning were good, demonstrating the feasibility of the approach. In addition, we realized that fine-tuning the network without destroying its whole architecture, increased the feature extraction ability on our dataset hence high accuracy of prediction.

To test the effect of leaf image rotation as well as light alteration on the model performance before and after model fine-tuning, was our second objective. We performed image data augmentation which not only increased the number of our input data, but it also created different versions of input images through random rotation and random brightness. The findings after training the network on augmented data shown that, the performance was not quite good before fine-tuning. The accuracies reported were **91%, 92% & 88%** on train, validation and test sets respectively. This decline in the accuracy of detection could have possibly been due to creation of different versions of leaves that were harder to classify thus confusing the model. However, fine-tuning the network increased the generalization ability and reduced misclassification cases on the validation and test sets. Using this classifier, we obtained **99%** accuracy on the train set, **95%** & **94%** accuracy on validation and test sets respectively. It is worth noting that fine-tuning the network increased the number of trainable parameters. Hence, the network was able to extract more patterns from the data and by training for longer duration, we achieved the best classifier.

We obtained 4 models from the architecture of pre-trained MobileNet model that was implemented in this research. These classifiers were: **i.** MobileNet model without data augmentation, **ii.** fine-tuned MobileNet model with- out data augmentation, **iii.** MobileNet model with data augmentation and finally, **iv.** fine-tuned MobileNet model with data augmentation. To evaluate and compare the performance of these classifiers so as to identify which classifier works well, was our third objective. Analyzing this objective, we found that, fine-tuned MobileNet model with data augmentation had the best generalization ability on validation and test sets with the least number of misclassifications out of the four trained classifiers.

We performed sensitivity analysis on this best classifier. This was our fourth objective and the findings illustrated that the model was robust. Firstly, this was shown by small difference between our known best accuracy and mean sample accuracy obtained from training the model using different random samples. This small difference was found to be not statistically significant at **5%** level of significance. Secondly, small difference between the overall and sample mean accuracy when model was trained using different learning rates, minibatch sizes and epochs. It is also worth mentioning that the model performance was maximum when the learning rate was **0.0001**, minibatch size of **16** and epoch of **30** which were our optimal hyper-parameters. Lastly with regards to this objective, reproducibility of the robust network was ensured by training the network using different random seeds. Similar results of minimum difference between the overall and sample mean accu- racy was obtained. From this analysis, we constructed a **95%** confidence in- terval within which the accuracy could possibly lie. The confidence interval was **[94.94, 96.19]**. We finally concluded that there was a **95%** chance that the confidence interval contains the true overall mean accuracy of bean disease classification.

Fine-tuned network with data augmentation had the best accuracy of detection however, this model still had **7** and **8** misclassifications on validation and test sets respectively. It is also worth mentioning that out of the **15** misclassifications, there was only **1** misclassified healthy leaf as bean rust. The rest of misclassifications occurred in distinguishing between angular leaf spot and bean rust classes. For this reason, we visualized the model attention on both correct & misclassified classes to explain how the model arrived at its decisions or conclusions which was our fifth objective. Research findings obtained here suggested that, whenever the model was paying attention to the leaf image, there was a correct prediction. However, misclassifications occurred every-time the model was not paying attention to the leaf or due to odd cases like presence of a hole on the leaf.

Based on this, a future line of work could be to combine this work with a ML model to perform segmentation of the leaf in the image. If we know what part of the image corresponds to the leaf, we expect the classier to perform even much better with very few or no misclassifications.

6.2 Conclusion

Transfer learning approach using pre-trained deep learning architecture namely MobileNet, has been successfully employed in training a robust classifier. The network can accurately detect bean leaf diseases even in case of angle variations and light alteration of a leaf with a classification accuracy of **95%** on validation set and **94%** on another unseen set namely test set.

The network has recorded very low misclassification count of **7** and **8** on validation and test sets respectively. With regards to model interpretability, we concluded that whenever the model was paying attention to the leaf image, there was a correct prediction. However, misclassifications occurred every-time the model was not paying attention to the leaf or due to odd cases like presence of a hole on the leaf.

6.3 Future Work

Our approach of transfer learning using MobileNet model for classification of bean crop leaf diseases achieved satisfactory results. However, further research is needed to improve accuracy and reliability in bean plant disease detection.

To be specific in further studies, we plan to collect more data from other East African nations so as to expand the dataset for bean disease, build and deploy a comprehensive tool for bean disease diagnosis. This will include performing object detection of bean leaf in the image to increase the accuracy of disease detection and minimize misclassifications.

In addition, we plan to test if the built system can work for other crops like tomato plant, potato etc. by taking advantage of transfer learning technique. The goal here is to develop and integrate systems into one robust system that can perform disease detection for more than one crop. In real life, this can be beneficial to farmers since in most cases they grow and control various plant diseases in their fields.

Appendix A

Model Evaluation metrics results

Confusion Matrix and Classification Report for Models

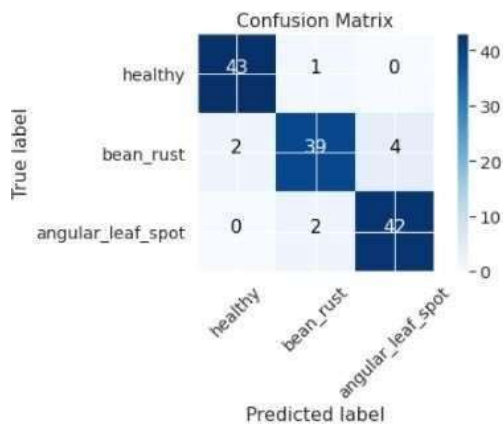
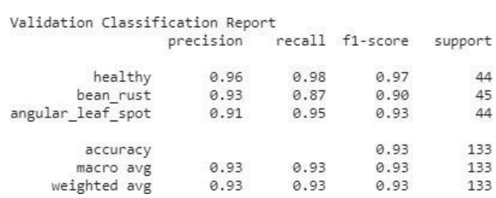


FIGURE A.1: Validation set confusion matrix for the MobileNet model before fine-tuning. Without data augmentation



Validation Classification Report

	precision	recall	f1-score	support
healthy	0.96	0.98	0.97	44
bean_rust	0.93	0.87	0.90	45
angular_leaf_spot	0.91	0.95	0.93	44
accuracy			0.93	133
macro avg	0.93	0.93	0.93	133
weighted avg	0.93	0.93	0.93	133

FIGURE A.2: Validation set classification report for the MobileNet model before fine-tuning. Without data augmentation

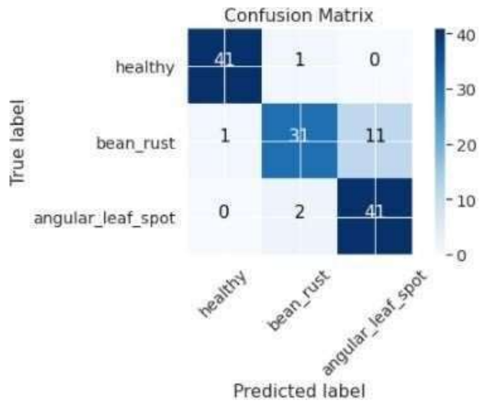


FIGURE A.3: Test set Confusion Matrix for the mobileNet model before fine-tuning. Without data augmentation

Test Classification Report

	precision	recall	f1-score	support
healthy	0.98	0.98	0.98	42
bean_rust	0.91	0.72	0.81	43
angular_leaf_spot	0.79	0.95	0.86	43
accuracy			0.88	128
macro avg	0.89	0.88	0.88	128
weighted avg	0.89	0.88	0.88	128

FIGURE A.4: Test set classification report for the mobileNet model before fine-tuning. Without data augmentation

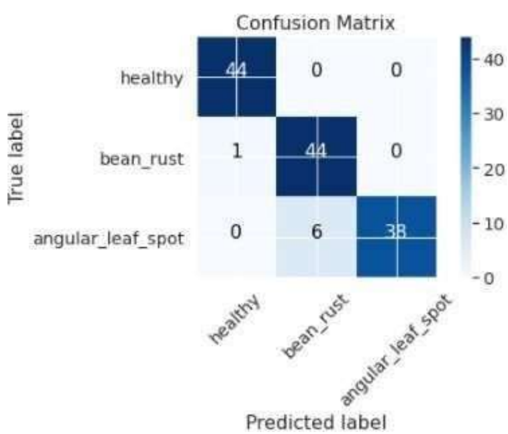


FIGURE A.5: Validation set confusion matrix for the mobileNet model after fine-tuning. Without data augmentation

Validation Classification Report

	precision	recall	f1-score	support
healthy	0.98	1.00	0.99	44
bean_rust	0.88	0.98	0.93	45
angular_leaf_spot	1.00	0.86	0.93	44
accuracy			0.95	133
macro avg	0.95	0.95	0.95	133
weighted avg	0.95	0.95	0.95	133

FIGURE A.6: Validation set classification report for the mo- bileNet model after fine-tuning. Without data augmentation

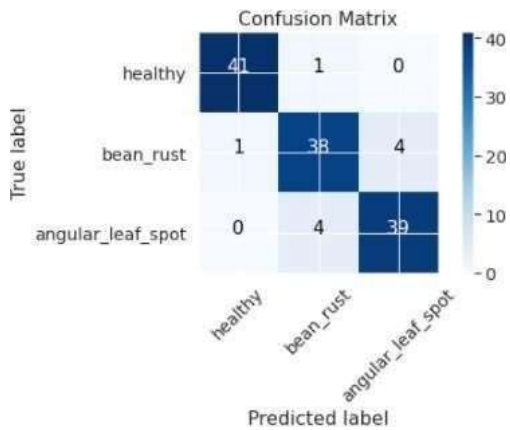


FIGURE A.7: Test set confusion matrix for the mobileNet model after fine-tuning. Without data augmentation

Test Classification Report

	precision	recall	f1-score	support
healthy	0.98	0.98	0.98	42
bean_rust	0.88	0.88	0.88	43
angular_leaf_spot	0.91	0.91	0.91	43
accuracy			0.92	128
macro avg	0.92	0.92	0.92	128
weighted avg	0.92	0.92	0.92	128

FIGURE A.8: Test set classification report for the mobileNet model before fine-tuning. Without data augmentation

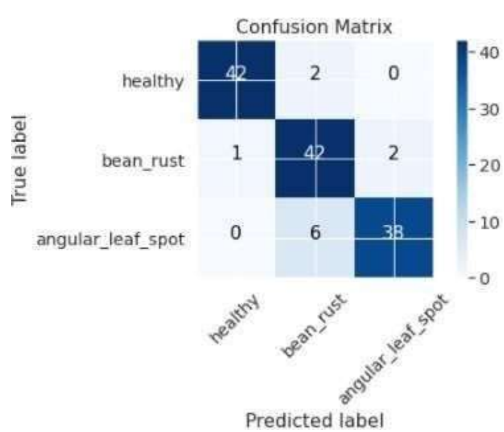


FIGURE A.9: Validation set confusion matrix for the mobileNet model before fine-tuning. With data augmentation

Validation Classification Report

	precision	recall	f1-score	support
healthy	0.98	0.95	0.97	44
bean_rust	0.84	0.93	0.88	45
angular_leaf_spot	0.95	0.86	0.90	44
accuracy			0.92	133
macro avg	0.92	0.92	0.92	133
weighted avg	0.92	0.92	0.92	133

FIGURE A.10: Validation set classification report for the mo- bileNet model before fine-tuning.With data augmentation

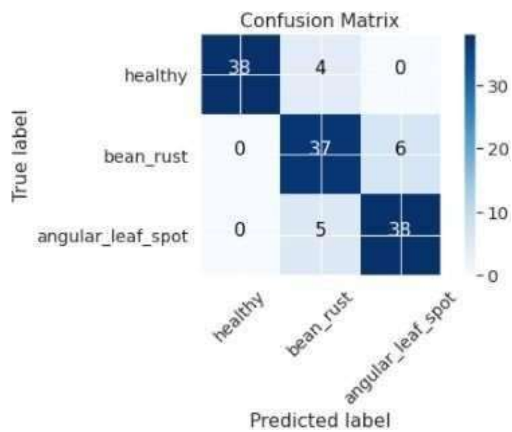


FIGURE A.11: Test set confusion matrix for the mobileNet model before fine-tuning.With data augmentation

Test Classification Report

	precision	recall	f1-score	support
healthy	1.00	0.90	0.95	42
bean_rust	0.80	0.86	0.83	43
angular_leaf_spot	0.86	0.88	0.87	43
accuracy			0.88	128
macro avg	0.89	0.88	0.89	128
weighted avg	0.89	0.88	0.88	128

FIGURE A.12: Test set classification report for the mobileNet model before fine-tuning.With data augmentation

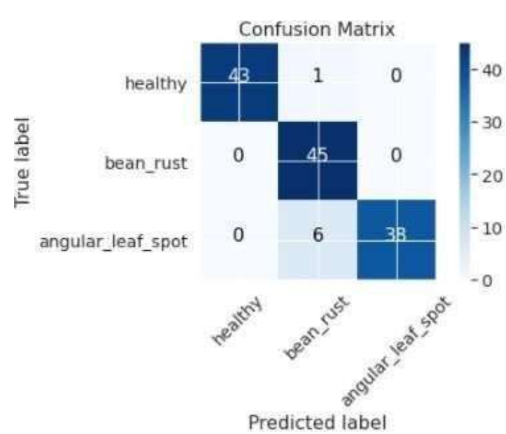


FIGURE A.13: Validation set confusion matrix for the mo- bileNet model after fine-tuning. With data augmentation

Validation Classification Report

	precision	recall	f1-score	support
healthy	1.00	0.98	0.99	44
bean_rust	0.87	1.00	0.93	45
angular_leaf_spot	1.00	0.86	0.93	44
accuracy			0.95	133
macro avg	0.96	0.95	0.95	133
weighted avg	0.95	0.95	0.95	133

FIGURE A.14: Validation set classification report for the mo- bileNet model after fine-tuning. With data augmentation

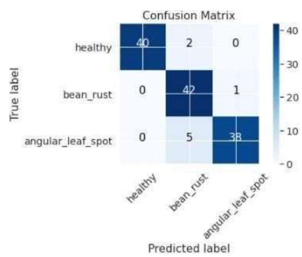


FIGURE A.15: Test set confusion matrix for the mobileNet model before fine-tuning. With data augmentation

Test Classification Report

	precision	recall	f1-score	support
healthy	1.00	0.95	0.98	42
bean_rust	0.86	0.98	0.91	43
angular_leaf_spot	0.97	0.88	0.93	43
accuracy			0.94	128
macro avg	0.94	0.94	0.94	128
weighted avg	0.94	0.94	0.94	128

FIGURE A.16: Test set classification report for the mobileNet model before fine-tuning

References

Abed-Ashtiani, Farnaz et al. (Nov. 2017). "Plant tonic, a plant-derived bioactive natural product, exhibits antifungal activity against rice blast dis- ease". In: *Industrial Crops and Products* 112, pp. 105–112. DOI: 10.1016/j.indcrop.2017.11.013.

Amara, Jihen, Bassem Bouaziz, and Alsayed Algergawy (2017). "A Deep Learning-based Approach for Banana Leaf Diseases Classification". In: *Datenbanksysteme für Business, Technologie und Web (BTW 2017) - Workshop- band*. Ed. by Bernhard Mitschang et al. Bonn: Gesellschaft für Informatik e.V., pp. 79–88.

Castro-Guerrero, Norma A. et al. (2016). "Common Bean: A Legume Model on the Rise for Unraveling Responses and Adaptations to Iron, Zinc, and Phosphate Deficiencies". In: *Frontiers in Plant Science* 7, p. 600. ISSN: 1664- 462X. DOI: 10.3389/fpls.2016.00600. URL: https://www.frontiersin.org/article/10.3389/fpls.2016.00600.

Coulibaly, Solemane et al. (2019). "Deep neural networks with transfer learning in millet crop images". In: *Computers in Industry* 108, pp. 115–120. DOI: 10 . 1016 / j . compind . 2019 . 02 . 003. URL: https : / / hal . archives - ouvertes.fr/hal-02104287.

Gavhale K., Gawande U. (2014). "An Overview of the Research on Plant Leaves Disease Detection Using Image Processing Techniques". In:

Harlen, Winda Christina and Ignasius Radix A.P. Jati (2018). "Chapter 8 - Antioxidant Activity of Anthocyanins in Common Legume Grains". In: *Polyphenols: Mechanisms of Action in Human Health and Disease (Second Edi- tion)*. Ed. by Ronald Ross Watson, Victor R. Preedy, and Sherma Zibadi. Second Edition. Academic Press, pp. 81–92. ISBN: 978-0-12-813006-3. DOI: https://doi.org/10.1016/B978-0-12-813006-3.00008-8. URL: https://www.sciencedirect.com/science/article/pii/B9780128130063000088.

Howard, Andrew G. et al. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. arXiv: 1704.04861 [cs.CV].

Krishnaswamy Rangarajan, Aravind and P. Raja (Feb. 2020). "Disease Classification in Eggplant Using Pre-trained VGG16 and MSVM". In: *Scientific Reports* 10, p. 2322. DOI: 10.1038/s41598-020-59108-x.

Kuete, Victor (2014). "22 - Physical, Hematological, and Histopathological Signs of Toxicity Induced by African Medicinal Plants". In: *Toxicological*

Survey of African Medicinal Plants. Ed. by Victor Kuete. Elsevier, pp. 635– 657. ISBN: 978-0-12-800018-2. DOI: https://doi.org/10.1016/B978-0-12-800018-2.00022-4. URL: https://www.sciencedirect.com/science/article/pii/B9780128000182000224.

Liang, Wan-jie et al. (2019). "Rice Blast Disease Recognition Using a Deep Convolutional Neural Network". In: *Scientific Reports*.

Mishra, Sumita, Rishabh Sachan, and Diksha Rajpal (Jan. 2020). "Deep Convolutional Neural Network based Detection System for Real-time Corn Plant Disease Recognition". In: *Procedia Computer Science* 167, pp. 2003– 2010. DOI: 10.1016/j.procs.2020.03.236.

Mohanty, Sharada P., David P. Hughes, and Marcel Salathé (2016). "Using Deep Learning for Image-Based Plant Disease Detection". In: *Frontiers in Plant Science* 7, p. 1419. ISSN: 1664-462X. DOI: 10.3389/fpls.2016.01419. URL: https://www.frontiersin.org/article/10.3389/fpls.2016.01419.

Rangarajan, Aravind Krishnaswamy, Raja Purushothaman, and Aniirudh Ramesh (2018). "Tomato crop disease classification using pre-trained deep learning algorithm". In: *Procedia Computer Science* 133. International Confer- ence on Robotics and Smart Manufacturing (RoSMa2018), pp. 1040–1047. ISSN: 1877-0509. DOI: https:// doi.org/ 10 .1016 / j.procs.2018 .07 .070. URL: https://www .sciencedirect.com /science/article/pii/S1877050918310159.

Rehman, Zia-ur, A.M Salariya, and S.I Zafar (May 2001). "Effect of processing on available carbohydrate content and starch digestibility of kidney beans (Phaseolus vulgaris L.)" In: *Food Chemistry* 73, pp. 351–355. DOI: 10.1016/ S0308-8146(00)00311-3.

Worrall, Elizabeth A. et al. (2015). "Chapter One - Bean Common Mosaic Virus and Bean Common Mosaic Necrosis Virus: Relationships, Biology, and Prospects for Control". In: ed. by Margaret Kielian, Karl Maramorosch, and Thomas C. Mettenleiter. Vol. 93. Advances in Virus Research. Academic Press, pp. 1–46. DOI: https://doi.org/10.1016/bs.aivir.2015.04.002. URL: https://www.sciencedirect.com/science/article/pii/S0065352715000779.

Turnitin Report

Millicent Turnitin Report

ORIGINALITY REPORT

17%	13%	11%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	towardsdatascience.com Internet Source	1%
2	www.nature.com Internet Source	1%
3	www.ncbi.nlm.nih.gov Internet Source	1%
4	mafiadoc.com Internet Source	<1%

<1%

<1%

<1%

<
1
%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		