DEEP LEARNING FOR DETECTION OF MANGO LEAF DISEASE: A COMPARATIVE STUDY USING CONVOLUTIONAL NEURAL NETWORK MODELS

BY

MUNSHI OMAR FARUQUE RABBI
ID: 201-15-13907

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

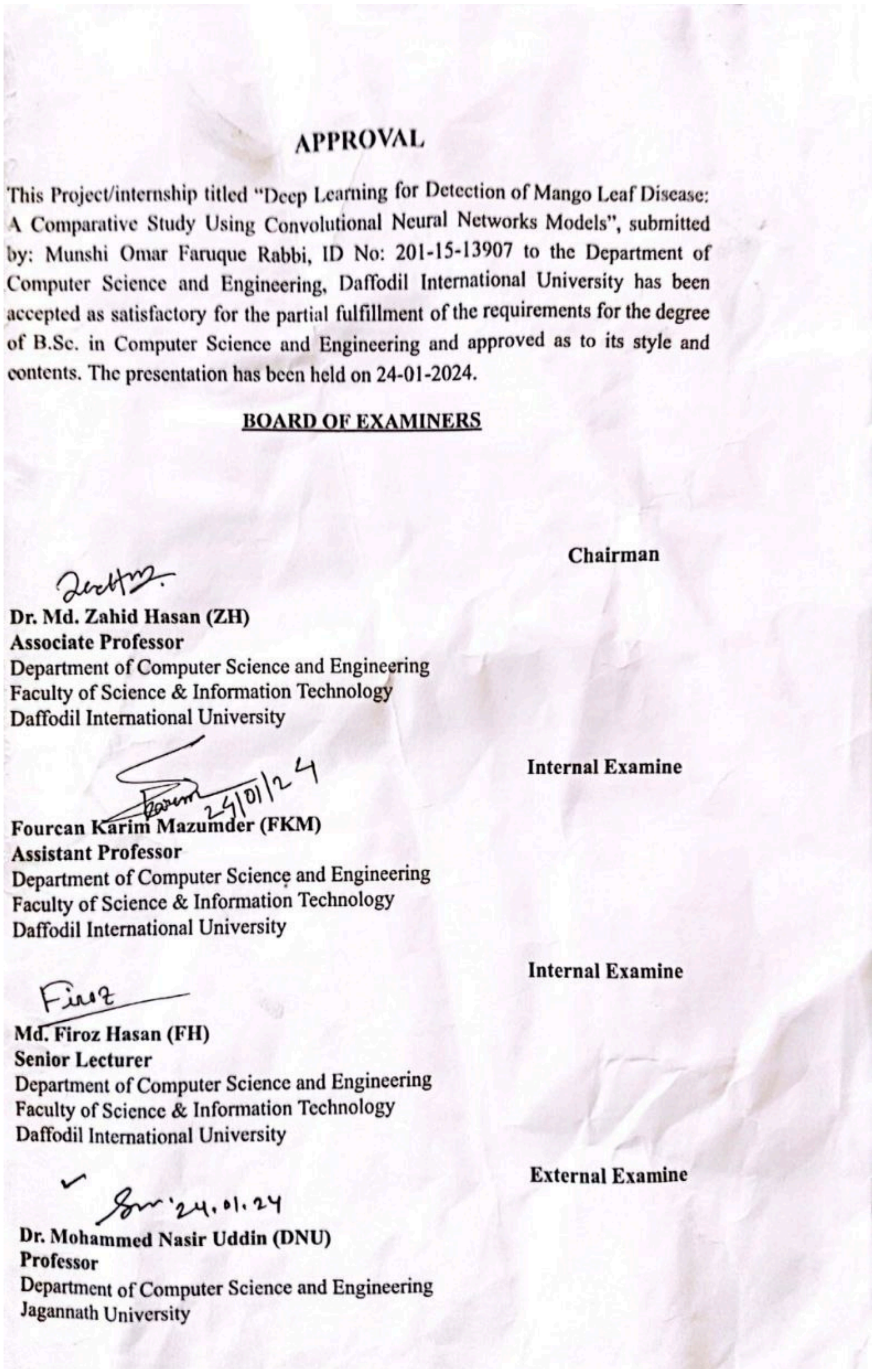
Supervised By

ISRAT JAHAN
Lecturer (Senior Scale)
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

JANUARY 2024



APPROVAL

This Project/internship titled "Deep Learning for Detection of Mango Leaf Disease: A Comparative Study Using Convolutional Neural Networks Models", submitted by: Munshi Omar Faruque Rabbi, ID No: 201-15-13907 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 24-01-2024.

DECLARATION

I hereby declare that this project has been done by us under the supervision of **Israt Jahan, Lecturer (Senior Scale), Department of CSE** Daffodil International University. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

[illegible]

Israt Jahan
Lecturer (Senior Scale)
Department of CSE
Daffodil International University

Submitted by:

Omar Faruque Rabbi

Munshi Omar Faruque Rabbi
ID: 201-15-13907
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

First, I offer my heartfelt appreciation and gratitude to almighty God for His divine grace, which has enabled me to successfully finish the final thesis.

I am grateful and wish to express my profound indebtedness to **Israt Jahan, Lecturer(Senior Scale),** Department of CSE, Daffodil International University, Dhaka. deep knowledge and the keen interest of my supervisor in the field of "Deep Learning" to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, and reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

I want to express my gratitude to **Dr. Sheak Rashed Haider Noori**, Professor and Head of the Department of CSE at Daffodil International University, as well as the other academics and staff members of the CSE department for their kind support in seeing my project through to completion

I want to express my gratitude to all of my Daffodil International University students who took part in this conversation while still in class.

Last but not the least, I want to thank my parents for their unfaithful love and understanding.

ABSTRACT

The diagnosis of diseases in mango leaves presents a complex challenge, compounded by the diversity of crop types, variability in agricultural disease indicators, and a multitude of environmental factors. Early detection of these diseases is particularly difficult, as existing methods often rely on data limited to specific geographic areas, thus restricting their efficacy. The timely identification and management of such diseases are crucial in averting substantial financial losses for farmers. This research introduces an innovative approach, leveraging image processing technology to detect diseases in mango leaves. The study employs a Convolutional Neural Network (CNN) model, specifically the Proposed model, which has shown remarkable accuracy of 97.92% in this context. This model was rigorously tested on both normal and diseased mango leaves, showing its efficacy in differentiating between healthy and unhealthy foliage. The application of this algorithm to leaf images facilitates the categorization of mango tree leaves into healthy or diseased categories, offering a timely and efficient solution for early disease detection. This novel approach not only promises to enhance disease management in mango cultivation but also sets a precedent for applying similar techniques in broader agricultural contexts, potentially revolutionizing plant disease diagnosis and prevention.

Keywords — Convolutional Neural Networks (CNNs), Mango Leaf Disease Detection, Image Processing in Agriculture, Deep Learning for Plant Health

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

In order to effectively manage diseases and prevent significant financial losses in mango production, it is vital to identify infections in mango leaves early on. But because of the wide range of weather factors, it is hard to find diseases early on in mango leaves[1]. Mango leaf diseases can now be correctly identified with the help of several deep-learning systems. These days, a lot of the methods I use have problems that make them less useful. As an example, they are only judged based on pictures of plant leaves from certain areas[2]. Utilizing both image processing and a CNN model, I present a fresh way to precisely find sick mango leaf groups. Convolutional neural networks (CNNs) will be used to build a sequence model for sickness prediction in mango leaves. This will help farmers get a quick and accurate diagnosis[3].
Finding diseases quickly in mango leaves is very important. Unfortunately, mangoes are an important food source that could lose a lot of output if diseases aren't controlled properly. Agricultural areas can also be quickly destroyed by disease breakouts. It is important to find diseases early and use effective disease control methods if I want to lessen the effect they have on mango production. It has always been up to trained eyes to look at the leaves and figure out if they have mango leaf disease. To be honest, it is hard, takes a long time, and mistakes happen often[4]. New developments in deep learning methods have made more and more people interested in making automatic systems that can find plant diseases[5].
I can now find out if a plant is sick by looking at its leaves using image processing and analysis. Looking at digital pictures of plant leaves and figuring out what the important parts are can help you tell the difference between healthy and sick leaves. A big set of labeled pictures can be used to teach deep learning algorithms, like convolutional neural networks (CNNs), how to understand the complex patterns and traits linked to different diseases[6].To figure out what diseases might hurt mango leaves, this study uses convolutional neural networks (CNNs)[7]. Another type of deep learning model that was created to work with visual data is the convolutional neural network (CNN). A lot of computer vision jobs, like identifying objects and

putting pictures into groups, have been done very well by them. I can accurately predict illness with convolutional neural networks (CNNs) because they are hierarchical. This means that the model can learn from the pictures that are fed into it and automatically pull out relevant information[8].

To test how well our suggested CNN-based sequential model worked, I used a set of pictures of mango leaves. So that changes in shape can be tracked as the sickness gets worse, the collection has both healthy and sick mango leaves. The high accuracy rate of 97.92% in our model means it might be useful for finding diseases in mango leaves[9]. The model's accuracy was all over the place, though, when tested on mango leaves that had wrong parts. So I need to do more work to make the model better at finding illnesses that have signs that aren't typical[10]. There is a reliable way for farmers to tell the difference between healthy and sick mango leaves using a convolutional neural network (CNN) sequence model that I describe[11]. Crop rotation, early identification, focused chemical treatment, and other similar steps can help farmers lower the number and intensity of illnesses that affect their crops[12].

Image processing and a convolutional neural network-based sequence model are used in this study to show a new way to predict disease in mango leaves[13]. With these results, i can see that the suggested method might work to find sick mango leaf groups. To make the model better at dealing with broken leaves, more study is needed, but our work helps move automatic systems forward that can keep mango output going and keep farms from losing a lot of money[14].

1.2 Motivation

Bangladesh, a horticultural nation, is thriving with a variety of tropical and subtropical organic products, including mango, banana, jackfruit, pineapple, papaya, and litchi, which are created on 79 percent of the collected zone of crops. If i can cultivate it in a proper way that is disease-free, it will be beneficial to our economy. The disease in those fruits almost comes from their leaves. If somehow one leaf turns into stains or disease, this disease will increase day by day in each leaf. Thus, the fruits will also suffer from stains, causing them to die. To address this issue in mango leaves, I am using modern technology to develop an automated system for disease prediction. The study utilizes deep learning techniques, specifically image processing and Convolutional Neural Networks (CNNs), in order to enhance the efficiency and accuracy of disease detection. This will provide producers with the ability to proactively prevent disease outbreaks and mitigate the subsequent negative effects. This innovative approach will help ensure long-term food security for Bangladesh's diverse population and contribute to the country's economic development.

1.3 Rational of the Study

Leaf diseases are often studied because there is strong proof that shows how important and useful it is to understand these things. The following is an example of how this study could be used: Because many people think the Alphonso mango is the best type of mango, it's often called the "king of mangoes." There are a lot of them in many places around the world. Unfortunately, growing mangoes can be hard because they are prone to many diseases that can kill mango trees or make the fruit not very good. The continued growth and economic success of mango farms rest on our ability to find and treat these diseases. It is important to learn about the diseases that hurt mango leaves for a number of reasons. Before starting, it's a good idea to learn about the most common problems that mango trees have and how they affect the quality and amount of fruit they produce. Healthcare workers must carefully study these illnesses' signs, causes, and development in order to effectively avoid, discover, and treat them. The main goal of studying mango leaf diseases is to find ways to increase mango production while reducing the amount of money lost to these diseases. To keep fruit losses to a minimum and get the best peaches, farmers need to come up with good ways to deal with diseases.

1.4 Research Questions

1. What kinds of diseases could deep learning systems find in leaves?
2. What is the difference between deep learning-based disease identification methods for mango leaves and more standard methods?
3. How well do different pre-trained models (like Proposed model, DenseNet121, VGG19, ResNet50, InceptionV3, AlexNet, EfficientNetV2, and MobileNetV2) find leaf diseases?
4. What traits can the skilled models tell from the pictures of the sick mango leaves?
5. How does the system deal with pictures of leaves that are healthy? Is it true that

they don't have any diseases?

6. What kind of picture quality or sharpness does the system need to be able to properly find leaf diseases?

7. How does the system work when pictures are taken from different angles or in different lighting?

8. How can the system be set up to find diseases in real time?

9. How does using different optimizers, like Adam, change how accurate the model is?

10. What can be done to make it easier for the system to find leaf diseases?

11. How can the system be connected to other plans for managing farms so that farmers have the whole package?

12. How can the system be changed so that it can spot diseases in different kinds of leaves?

1.5 Expected Output

The goal of this project is to create a CNN-based sequence model for figuring out if mango leaves are sick. The computer will be very good at figuring out what diseases people have. The program should work better than standard eye examinations, making it a more accurate and useful way to find illnesses early on. The model should be able to correctly find a lot of different diseases that hurt mango leaves. These steps will help farmers fight diseases and keep them from losing a lot of money. The study also wants to see how well the model works with both healthy and misshapen mango leaves to find out how well it can deal with differences in leaf structure and rare sickness signs. It can also test how well the model works in different places, which means it can be used in a wide range of farming settings. The main goal of this study is to make a CNN-based linear model that is flexible and reliable. People think that this method would make it much easier to find sick oranges and keep them healthy and active.

1.6 Project Management and Finance

Project Management

The main goal of the project is to create a deep learning tool that can use transfer learning to look for diseases on leaves. Models that have already been trained will be used to test how well the system finds diseases. Proposed model, ReSNet50, Inception V3, AlexNet, EfficientNet V2, and MobileNet V2 are some of these models. I need to get data first, then build the system, test it, and finally use it. It will take a while. There will be a plan for each step.There will be farmers on the team, as well as data scientists (who will collect and study data), software engineers (who will set up systems), and machine learning engineers (who will make models). It's possible that i don't have enough training data, that i can't find illnesses correctly, or that the technology will make it hard to put the system together. For each danger, a plan will be made to lower it. Quality Assurance: It will be checked and tested a lot to make sure it works well. It should be easy to change the method to keep up with new sickness trends and work well.

Finance

Making a plan for money By making a system, testing it thoroughly, and then using it, I can get a good idea of how much it will cost. In addition, it will include the costs of the team's tools and salaries. Government agencies, groups that help farmers, or people who have a personal stake in farming technology may give money to my project. Which choice is better: better control of diseases or maybe more food production? Specifically, the costs of setting it up and keeping it up will be taken into account. The technology changed the fast food industry's rate of growth and cut down on food waste due to illness by a large amount. I will look at the following things to figure out the return on investment (ROI). If you don't have enough money or your actual costs are higher than expected, you could lose money. This event is less likely to happen if people plan ahead and are careful with their money. How much long-term help the project gets will depend on how well the system keeps getting better. This could mean that the group has the chance to make sales or get more money if it happens.

1.7 Report Layout

This study consists of a total of six chapters. Below is an overview of the six chapters:

- In the first chapter, I gave a general introduction to the work, including the research question and its purpose, the motivation behind it, the project's management and funding, the report format, and the expected results.

- The second chapter looks at the history of the project, including the preparation work, related works, comparative assessments, and problems that were faced.
- Data collection, proposed technique, statistical rationale, and execution, The third chapter covers all the requirements in detail.
- Chapter 4 talks in detail about the research, setting, and results of the experiment.
- In Chapter 5, the effects on society, the environment, and moral issues were all looked at in detail.
- The overview, conclusion, and future directions of the study were all in Chapter 6.

CHAPTER 2
BACKGROUND

2.1 Introduction

Background research is crucial for understanding the etiology, clinical manifestations, and incidence of mango leaf disease. This data can help develop effective detection techniques and management strategies. The results of background research can guide future research and ensure the relevance and applicability of findings. One approach to mango leaf disease detection is image processing, which entails gathering, cleaning, and evaluating pictures according to standards including accuracy, precision, data collection, and preprocessing.

Present work, related work, research summary, scope of study, goal, and problems encountered are all detailed in this chapter. The chapter also delves into the many methods that were examined in the research.

2.2 Related Works

In recent years, significant advancements have been made in the detection and recognition of plant diseases using various image processing and deep learning techniques. Tete and Kamlu (2017) demonstrated the efficacy of using KNN and ANN algorithms for disease detection with accurate results in less computing time[1]. Similarly, Mohapatra et al. (2022) employed CSUBW and CNN, achieving an accuracy of 91.2%, highlighting the role of enhanced feature extraction[2]. Jiang et al. (2019b) developed a system using the INAR-SSD algorithm on the Apple Leaf Disease Dataset (ALDD), consisting of 26,377 images, which achieved a mean Average Precision (mAP) of 78.80%. This system showed real-time detection capability[3]. Pham et al. (2020) utilized an ANN with an Adaptive Particle-Grey Wolf metaheuristic on a dataset of 450 real images, reaching an accuracy of 89.41%, indicating faster performance suitable for practical applications[4]. Chouhan et al. (2018) introduced a Bacterial Foraging Optimization based Radial Basis Function Neural Network (BRBFNN) for the identification and classification of plant diseases. They used datasets of 6 and 270 images from real images and crowdAI.org, respectively, achieving high performance in terms of identification and classification[5]. Wu et al. (2018) explored datasets related to depression and ADHD, utilizing techniques like Greedy Deep Weighted Dictionary Learning (GDWDL) with l2-norm regularization constraint, FDDL, and DFDL. Their methods enhanced convergence and classification accuracy with rates of 90.67%, 77.64%, and 68.82%, respectively[6]. Srivastava et al. (2021) employed a Deep Convolutional Neural Network (CNN) on the Kaggle Plant Village dataset, achieving an 88% accuracy[7]. Anitha and Saranya (2022) used a simple CNN on the Cassava dataset from Kaggle, demonstrating the successful application of data augmentation, with accuracy improving from 54% (original dataset) to 90% (augmented dataset)[8]. Gining et al. (2021) detected and classified mango leaf diseases using a CNN on the Harumanis Mango Leaf Dataset of 146 images, achieving an accuracy of 68.89%[9]. Fuentes et al. (2017) provided a real-time solution using the Tomato Diseases and Pests Dataset consisting of 5000 images and employing algorithms like Faster R-CNN, SSD, and R-FCN, with accuracies of 83%, 82.53%, and 85.98% respectively[10]. Frederick et al. (2023) demonstrated the effectiveness of CNNs on a dataset of 525 images of Valencia oranges, achieving accuracies of 94.9% and 90.2% using SVM on PCA and SVM on LDA, respectively[11]. Yohannes Agegnehu Bezabih (2024) used a Hybrid CNN model (ResNet & LeNet) on a collection of 1000 images, resulting in a superior performance of 96.21%[12]. Kusrini et al. (2020) examined the effectiveness of data augmentation on the Real Image dataset, employing the VGG 16 model and noting a significant impact on model performance[13]. Argüeso et al. (2020) demonstrated the benefits of CNN for small datasets, using a collected dataset of 5507 images and achieving 94% accuracy[14]. Sujatha et al. (2021) compared the performance of Deep Learning (DL) methods against traditional Machine Learning methods on the Real Image dataset, consisting of 551 images. Their findings indicated that DL methods, particularly VGG-16, outperformed others with an accuracy of up to 89.5%[15]. Fiona et al. (2019) highlighted the use of ANN for disease classification in their study on 350 images from the Real Image dataset, achieving an accuracy of 90%[16]. Tian et al. (2023) employed a two-band ratio algorithm coupled with IWSA on hyperspectral reflectance images, leading to high recognition rates of 92.5%[17]. Özgüven and Adem (2019) achieved a high classification rate of 95.48% using a modified Faster R-CNN on 155 real images[18]. Barbedo (2019c) utilized GoogleNet on a dataset of 1567 images, showing that localized images provide improved results with an accuracy of 94%[19]. Hu et al. (2019) enhanced the performance of the original CIFAR10 model using modified techniques on 144 real images, resulting in an accuracy of 92.5%[20]. Zhang et al. (2019) applied Three-Channel Convolutional Neural Networks (TCCNN) for improving recognition rates compared to traditional methods, achieving 91.16% accuracy on the PlantVillage dataset and 94.15% on real cucumber images[21]. Sladojević et al. (2016) achieved a high accuracy of 96.3% using CaffeNet (AlexNet) on a dataset of 33469 real augmented images[22]. Ramesh and Vydeki (2020) explored the Recognition and Classification of Paddy Leaf Diseases using an Optimized Deep Neural Network with Jaya Optimization Algorithm (DNN_JOA) on 650 real images, achieving a noteworthy improvement in accuracy over traditional methods such as ANN, DAE, and DNN, with an overall accuracy of 94.25%[23]. Pantazi et al. (2019) developed a One-class SVM classifier for disease detection, demonstrating a high generalization capability with a success rate of 95%. Their method used the GrabCut algorithm for segmentation and RGB to HSV conversion on a dataset of 94 real images[24]. Rangarajan et al. (2018) employed pre-trained deep learning architectures, AlexNet and VGG16, on the PlantVillage Dataset consisting of 13,262 images. Their study showcased the effectiveness of these pre-trained models, achieving high accuracy rates of 96.19% and 95.81% respectively[25]. These studies collectively represent significant advancements in the field of plant disease detection and recognition, utilizing a variety of datasets, preprocessing techniques, and deep learning models. The results not only demonstrate high accuracy rates but also underline the potential of deep learning in revolutionizing agricultural practices through early and reliable disease detection.

2.3 Comparative Analysis & Summary

Here, is a comparison analysis and a table where my thesis "Deep Learning for Detection of Mango Leaf Disease: A Comparative Study Using Convolutional Neural Networks Models" is compared with the reviewed papers, highlighting where MY models are best suited:

My proposed model (a combination of InceptionV3 & DenseNet121) shows the highest accuracy (97.92%) among all reviewed studies, indicating its superiority in classifying mango leaf diseases. The usage of advanced deep learning architectures like DenseNet121 and InceptionV3 in my study might have contributed to the higher accuracy, surpassing traditional models like KNN and ANN used in other studies. My study's focus on mango leaf disease detection with a high accuracy suggests a strong applicability and relevance in this specific domain, compared to other studies that covered a broader range of plant diseases but with lower accuracy. The effective combination of InceptionV3 & DenseNet121 in my study demonstrates the potential of using hybrid deep learning models for enhanced performance in specific problem domains, as opposed to using single, pre-trained models like VGG16 or AlexNet. The comparison of related work shows in below table 2.1.

Table 2.1: Comparison between my work & others work

Problem Domain	**Dataset**	**Models**	**Accuracy**	**References**
Detection	26,377 images	INAR-SSD	78.80% mAP	Jiang et al. (2019b)
Detection	450 images	ANN using Adaptive Particle-Grey Wolf metaheuristic	89.41%	Pham et al. (2020)
Detection	155 images	Faster R-CNN (modified)	95.48%	Özgüven and Adem (2019)
Classification	30 datasets	GDWDL, FDDL, DFDL	90.67%, 77.64%, 68.82%	Wu et al. (2018)
Detection	144 images	Modified CIFAR10	92.5%	Hu et al. (2019)
Classification	Kaggle Dataset	Deep CNN	88%	Srivastava et al. (2021)
Detection	1698/3958 images	Simple CNN	54% for original & 90% for augmented	Anitha and Saranya (2022b)

Detection & Classification	146 images	CNN	68.89%	Gining et al. (2021)
Detection	1000 images	Hybrid CNN model (ResNet & LeNet)	96.21%	Yohannes Agegnehu Bezabih (2024)
Classification	330469 images	CaffeNet (AlexNet)	96.3%	Sladojević et al. (2016)
Detection	510/46500/ 62,047 images	VGG16	67%, 68%, 74%	Kusrini et al. (2020)
Detection	5507 images	CNN	94%	Argüeso et al. (2020)
Classification	13,262 images	VGG16, AlexNet	96.19%, 95.81%	Rangarajan et al. (2018)
Detection	94 images	One-class SVM classifier	95%	Pantazi et al. (2019)
Detection	1011 images	DenseNet121, VGG19, ResNet50, InceptionV3, AlexNet, MobileNetV2, EfficientNetV2, Proposed model (combination of InceptionV3 & DenseNet121)	95.31%, 95.07%, 93.60%, 91.13%, 91%, 89%, 82.27%, 97.92%	My Study

Overall, my thesis presents a significant contribution to the domain of mango leaf disease detection using convolutional neural networks. The high accuracy of my proposed model, coupled with the specific focus on mango leaf diseases, positions my study as a benchmark in this area of research. It demonstrates the effectiveness of using a combination of advanced deep learning models to achieve superior results compared to traditional single-model approaches.

2.4 Scope of the problem

This research aims to identify illnesses in mango leaves at the early stages. The complexity of the issue arises from the wide range of crop types, symptoms of agricultural illnesses, and ecological factors that might affect the diagnosis of diseases. The problem is the challenges faced by farmers in accurately identifying illnesses in their early stages. This is due to the fact that a delayed or inaccurate diagnosis might result in substantial financial repercussions. The research focuses on the use of deep learning techniques, namely image processing and convolutional neural network (CNN) models, to develop an efficient method for detecting illnesses in mango leaf populations. This research aims to assess the performance of the model on both normal and abnormal mango leaves. Furthermore, the research seeks to evaluate the applicability of the model in different geographical areas. The study aims to provide a holistic approach that may improve disease control procedures in mango cultivation by addressing these issues.

2.5 Challenges

▪ Achieving a model accuracy of over 95%

▪ Implementing with high precision

▪ Conducting a comprehensive literature review

▪ Identifying research gaps in other works

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

This section provides a concise summary of the study's objectives and the methodology used to achieve them, with an emphasis on a comprehensive and relevant approach. The dependability of our data collection methods and the categorization of diseases that affect mango plants will be discussed. The research uses a convolutional neural network (CNN) to categorize 2D photos in a manner that facilitates the identification of mango leaf illnesses. This method enables the effective treatment or removal of illnesses from images of leaves. Subsequent research on photo processing and sorting have used the findings from this particular study. The data collection procedures used in this research are reliable due to their meticulousness and efficacy.

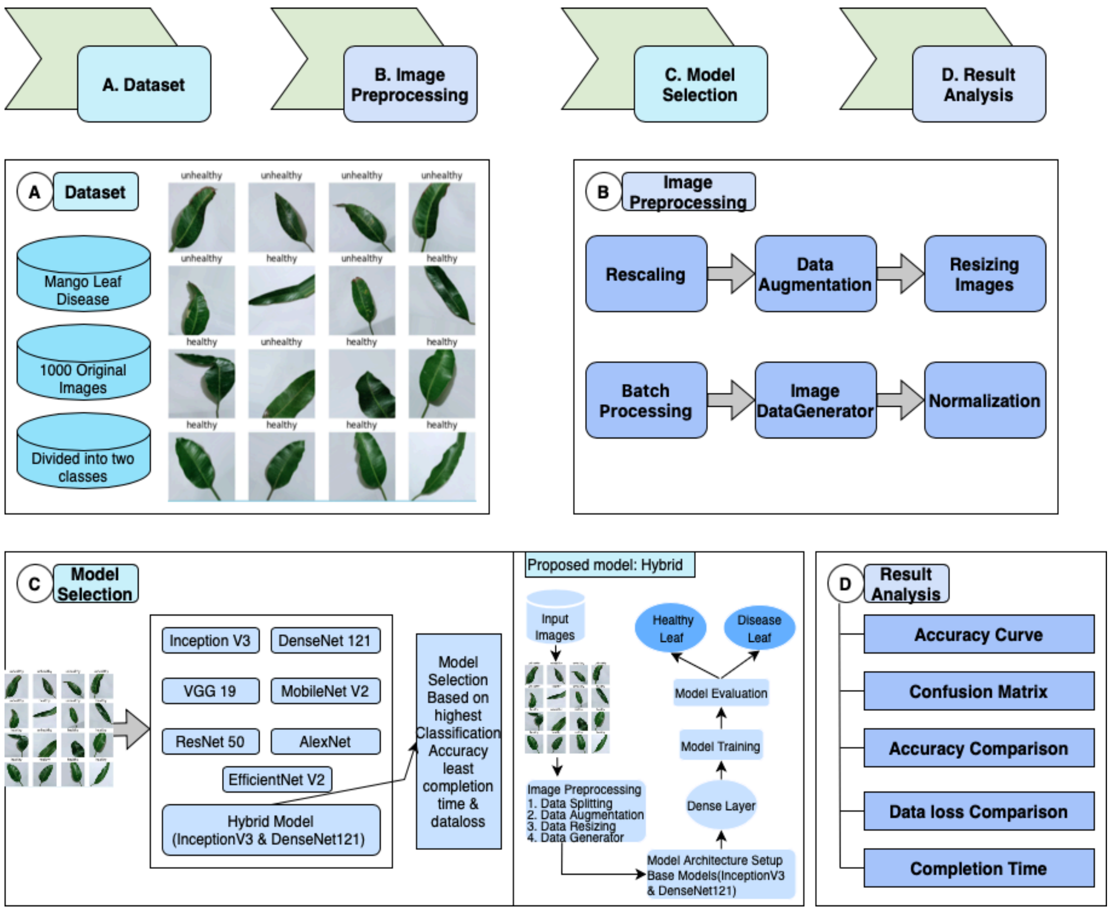


Figure 3.1: Working Procedure.

3.2 Research Subject and Instrumentation

My research subject is "Deep Learning for Detection of Mango Leaf Disease: A Comparative Study Using Convolutional Neural Networks Models" where the model will detect whether the leaf is disease-free or not while the comparison will be on CNN models & Hybrid or proposed model.

Instruments:

For this project, I used some of the suggested models along with real photos of unhealthy and healthy leaves to figure out if the picture was of a disease or not. You should have a fast PC, GPU, and other parts to work with a deep learning model. If you don't have these, it will be hard to set up. As well as these are some of the methods and tools that will be needed to complete the study:

Hardware Requirements:

- Processor core i5
- Ram 8 GB
- Rom 256 GB SSD
- Hard Disk Space usable

Software & Libraries Requirements:

- Python version (3.10.4)
- Google Colab
- Operating system: Macbook Pro
- Tensorflow 2.8.0
- Numpy
- Pandas
- Internet Connection

3.3 Data Collection Procedure

Getting and figuring out things that are connected to data is what data collection is all about. Even though a lot of experts get their information from places like Kaggle, Mendeley, and more. I learn things from the yard and crop field, though. At the same time, I only get a little info from the web to learn how to get real data and change it. When I get the data, I sort it into groups and give each one a name. To run my tests, I get three kinds of information. There is a training dataset, a confirmation dataset, and a tested dataset.

Most of the shots in my collection are of nature. First, I work on the picture that was taken. I turn it into numbers in the end. I made seven plans to help me reach my goal after that.

3.4 Statistical Analysis

Approximately 800 field pictures were captured for the purpose of my experiment. In the end, I ended up with a dataset of 1011 photos after augmenting a few of my own. I've divided them into two categories.I trained using 808 data points (about 80%) and tested with 203 data points (20%) out of a total of 1011 data points. I have divided everything into two categories: Unhealthy & Healthy.

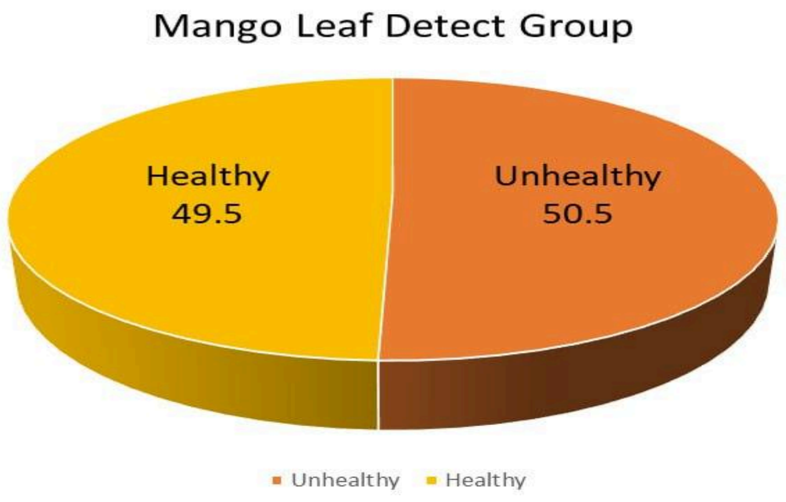


Figure 3.2: Categories of Leaf

This is the mango leaf data group. The percentages of dangerous and healthy mango leaves are 49.5% & unhealthy leaves are 50.5%. I put together information in my collection about different diseases or new groups. The following graph shows how the data is split into groups of people with diseases and people who don't have diseases.

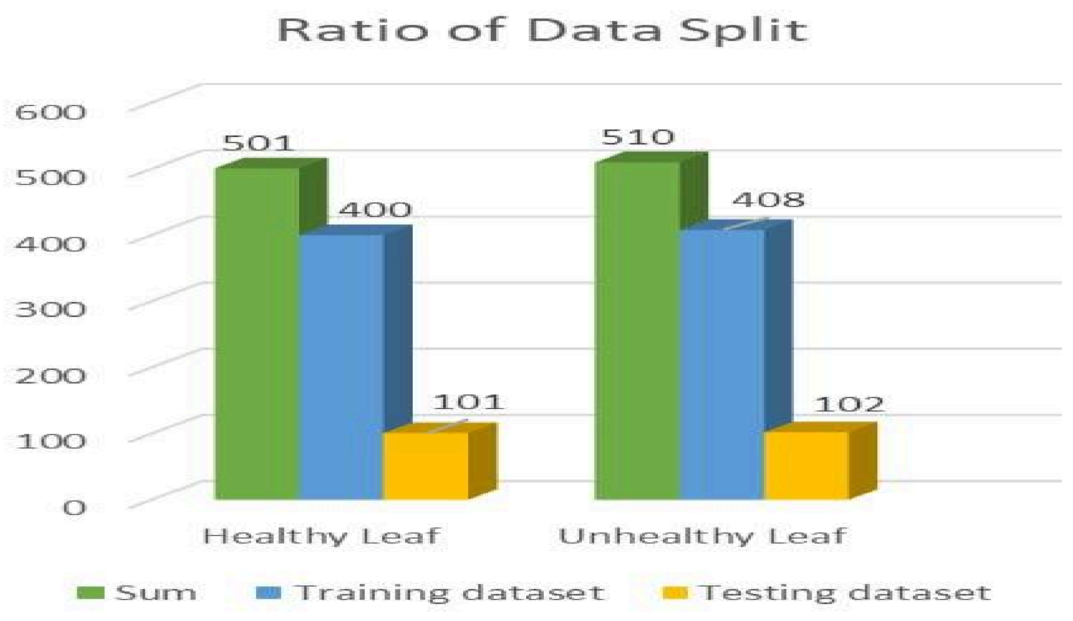


Figure 3.3: Ratio of Data Split

This is the percentage of all the info that was collected. Here, mostly Fresh leaves and Diseased leaves are shown.This 3.3 figure shows their train and test data as well as the sum of the data.

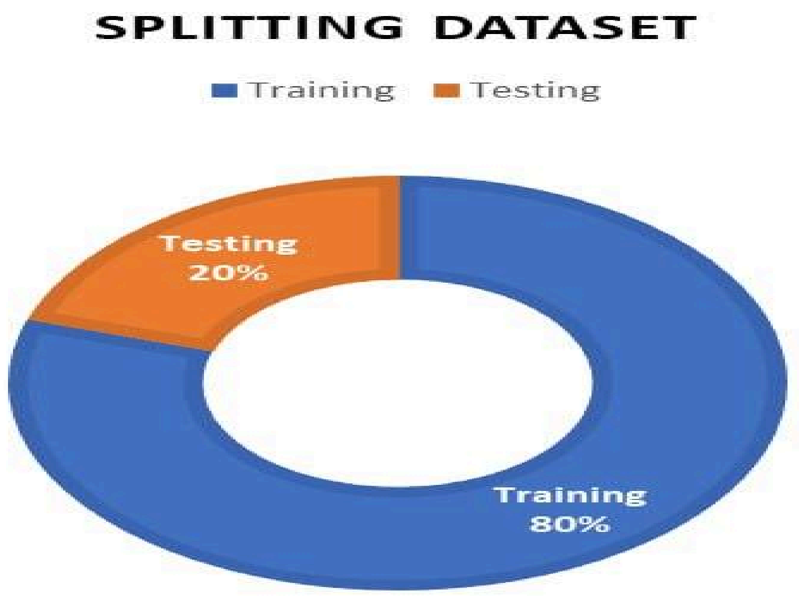


Figure 3.4: Splitting Datasets between Train & Test

By the figure 3.4, I can easily tell from my 1011 dataset that I split it into Train 80% and Test 20% datasets.

3.5 Data Pre-Processing

I utilize two important tactics to enhance and enrich our training dataset as part of our workflow for data preparation at our company. ('IMG-20230421-WA0058.jpg') is an example of an image that is exposed to a number of modifications via the use of an ImageDataGenerator called DataGen. The first method includes individual picture augmentation. This generator can apply features like a rotation of up to forty degrees, shifts of twenty percent of the image's dimensions in both the horizontal and vertical directions, shearing at a maximum angle of two degrees, zooming by twenty percent, and horizontal flipping. To fill in any gaps that may appear during the augmentation process, the 'nearest' fill method is used. The picture is loaded, then transformed into a NumPy array by using the img_to_array function, and lastly, it is rearranged to include a batch dimension. Additionally, the datagen.flow method is used to do augmentation on the picture as it is being processed inside a loop. In accordance with the actual filename of the picture, the enhanced photographs are stored in a certain directory ('/content/drive/MyDrive/data/train_data/healthy/') with a prefix of 'leaf_58' in the filename. In order to demonstrate the many visual modifications that were made to this particular picture, the loop is programmed to run for a maximum of five to twenty-five iterations. This is appropriate for some data in order to equate healthy data with unhealthy data. Now i will move on to the second technique, which involves leveraging the train_datagen to do batch-level augmentation on the full training dataset. Additionally, the flow_from_directory method is utilized to generate batches of augmented images from the specified training directory ('/content/drive/MyDrive/data/train'). This image data generator is configured in a manner that is comparable to that of an individual image generator. This includes rescaling the pixel values to the range of [0, 1], rotation, horizontal and vertical shifts, shearing, zooming, and horizontal flipping. In order to scale the photos to the dimensions (IMG_HEIGHT and IMG_WIDTH), which are specified individually for each model, such as 150x150 pixels or 224x224 pixels, the dimensions are not the same. Additionally, they are classified using the 'categorical' class mode, which is acceptable for numerous classes, and divided into batches of an indeterminate size (BATCH_SIZE), such as 16, 25, or 32, where the size is not defined. By injecting variety at the level of individual images as well as across the whole training set, these twin techniques provide an all-encompassing approach to data augmentation. By guaranteeing that the model can successfully generalize to a variety of circumstances and variances that are present in real-world data, this diversification is an essential component in the training process for a robust image classification model.

3.6 MODELS

CNNs are a class of deep neural networks, most commonly applied to analyzing visual imagery. The architecture typically consists of an input layer followed by multiple convolutional layers, pooling layers, fully connected layers, and an output layer that makes a prediction.

The diagram shows the flow from the input images to the predicted class. It starts with the input images going through a convolutional layer, which is designed to automatically and adaptively learn spatial hierarchies of features from input images. Then, the pooling layer reduces the spatial size of the representation, reducing the number of parameters and computation in the network. This process repeats through another set of convolutional and pooling layers.

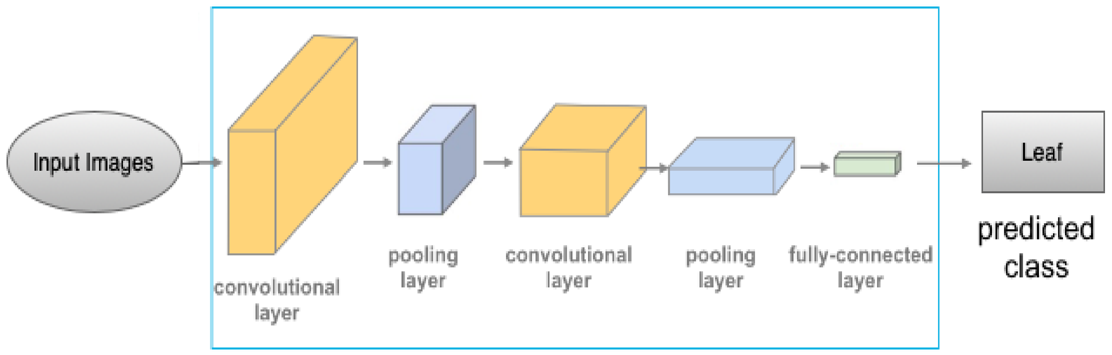


Figure 3.5: CNN Model Workflow

After the last pooling layer, the representation is flattened into a vector and fed through fully connected layers, where deeper representations are learned. The final layer is typically a softmax or logistic regression layer that outputs a probability distribution over the classes. The output 'Leaf' suggests that the CNN is used for classifying images of leaves, which could be part of a plant recognition system or a scientific study for classifying species of plants.

In my research, I am using some of the CNN architecture for detecting leaves.

3.6.1 Proposed Model (InceptionV3 with DenseNet121):

Finding a disease that affects mango leaves may be done in a variety of different methods. Here, I implement my proposed model which is a Hybrid model or combination of (InceptionV3 & Densenet121) models. To implement the code for detection there might be some procedure. These steps of code following as below:

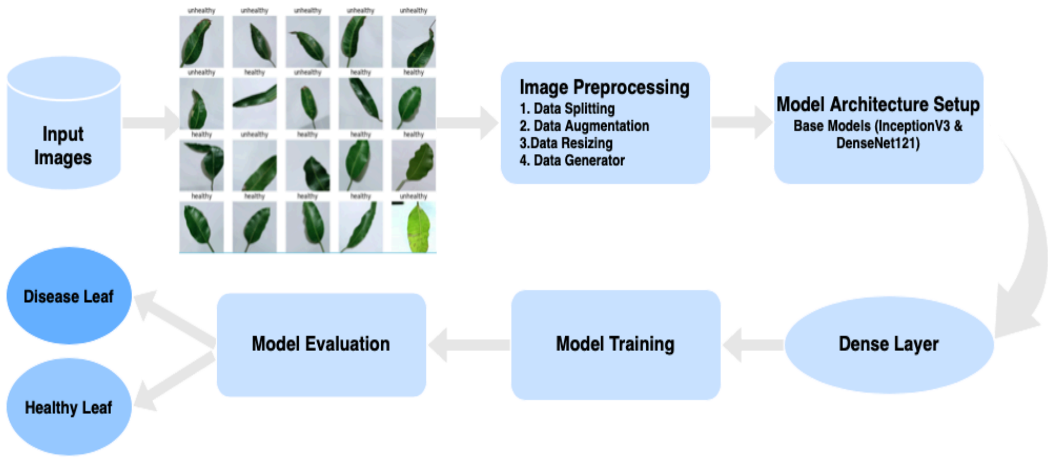


Figure 3.6: Proposed Model Workflow

Environment Setup and Data Loading

Begin by setting up the environment, which includes installing necessary libraries (tensorflow, keras, numpy, etc.) and mounting the Google Colab drive for data access. Load the mango leaf dataset, typically stored on the drive, segregating it into distinct folders for healthy and unhealthy leaves.

Data Preprocessing and Augmentation

Perform exploratory data analysis to understand the dataset's structure, size, and characteristics. Utilize ImageDataGenerator from Keras for preprocessing and augmenting the images. This includes rescaling the images and applying transformations like rotation, zoom, width and height shifts, shear, and horizontal flip to increase dataset variability.

Model Selection and Customization

Choose a base CNN model appropriate for image classification tasks, such as Proposed model (InceptionV3 with DenseNet121), InceptionV3, DenseNet, or MobileNet. Load the pre-trained weights (from ImageNet) and exclude the top layer to customize for the specific task. Add custom layers (like Global Average Pooling and Dense layers) to the base model, tailoring it for binary classification (healthy vs. unhealthy mango leaves).

Model Training and Fine-Tuning

Configure the model for training, setting trainable layers, optimizer (like Adam), loss function (categorical crossentropy), and metrics (accuracy, precision, recall). Train the model using the training dataset with specified batch size and steps per epoch. Implement callbacks like Early Stopping to prevent overfitting. Fine-tune the model by selectively training some of the top unfreeze layers of the base model along with the added custom layers.

Model Evaluation and Visualization:

Evaluate the model's performance on a separate test or validation dataset, focusing on key metrics such as loss, precision, recall, and accuracy. Plot graphs to visualize the model's training and validation accuracy and loss over epochs.

Prediction and Analysis:

Use the trained model for predicting the class (healthy or unhealthy) of new mango leaf images. Optionally, create a confusion matrix and a classification report to assess the model's performance in more detail. Implement functions to facilitate individual image predictions, displaying the image alongside its predicted class for verification.

Finally, after the presentation of Figure 3.6, the block design of the proposed model technique for this investigation is shown up with all steps.

3.6.2 Inception V3:

Inception V3 is an improved version of the original, just like its predecessor, Inception V1. It improved its network efficiency by using many methods, which made the Inception V3 model more flexible. It's very effective. As compared to the Inception V1 and V2 types, it has a bigger network and faster connections. It's also cheaper. The general mistake rate of the structure, which has 42 layers, is much lower than that of its predecessors. These changes were made to the Inception V3 model.

Factorization into Smaller Convolutions: The Inception V1 model's capacity to drastically lower dimensions was one of its most notable and beneficial characteristics. The model was able to increase its efficiency by factorizing the bigger Convolutions into smaller ones.

Spatial Factorization into Asymmetric Convolutions: Although the process of factorizing the bigger convolutions results in smaller convolutions. If i were able to further factorize, for example, into a 2x2 convolution, you may be wondering what the outcome would be. However, asymmetric convolutions proved to be a more efficient option that improved the model's effectiveness.

Utility of Auxiliary Classifiers: To help ultra-deep neural networks agree, an extra filter is used. The main goal of the extra classifier is to fix the problem of the gradient going down in very deep networks.

Efficient Grid Size Reduction: In the past, max pooling and average pooling were used to make the squares of feature maps smaller. By making the network filters' activation measure bigger, the Inception V3 model can shrink the grid size more effectively. When all the optimizations are finished, the final Inception V3 model looks like this:

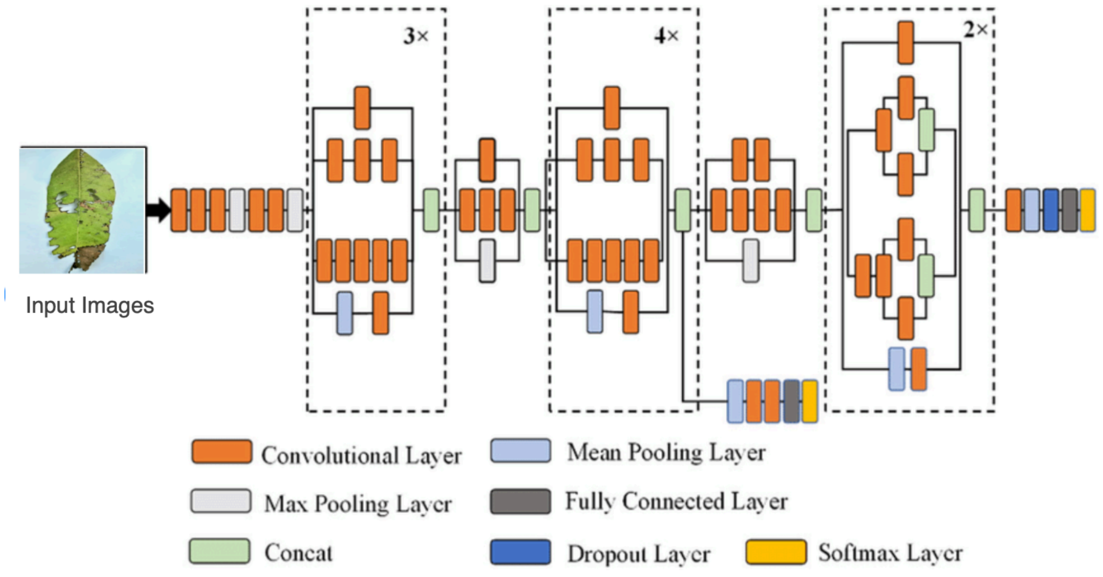


Figure 3.7: Inception V3 model architecture

3.6.3 DenseNet121:

DenseNet is one of many architectures for feed-forward convolutional neural networks (CNNs). All layers are linked together. By lowering the number of parameters and improving the gradient flow during training, the network may improve its learning potential by refining features. The architecture of DenseNet is made up of transition layers and dense nodes. Each convolutional layer inside a dense block is linked to every other layer.

At first, It starts with an input image, which then passes through multiple dense blocks where each layer receives feature maps from all preceding layers, promoting feature reuse. Transition layers between dense blocks perform convolution and pooling to reduce dimensionality. Finally, the network ends with a softmax layer that outputs a probability distribution over the classes, with the highest probability indicating the predicted class. DenseNet's unique connectivity pattern enhances learning efficiency and accuracy with fewer parameters compared to other deep network architectures.

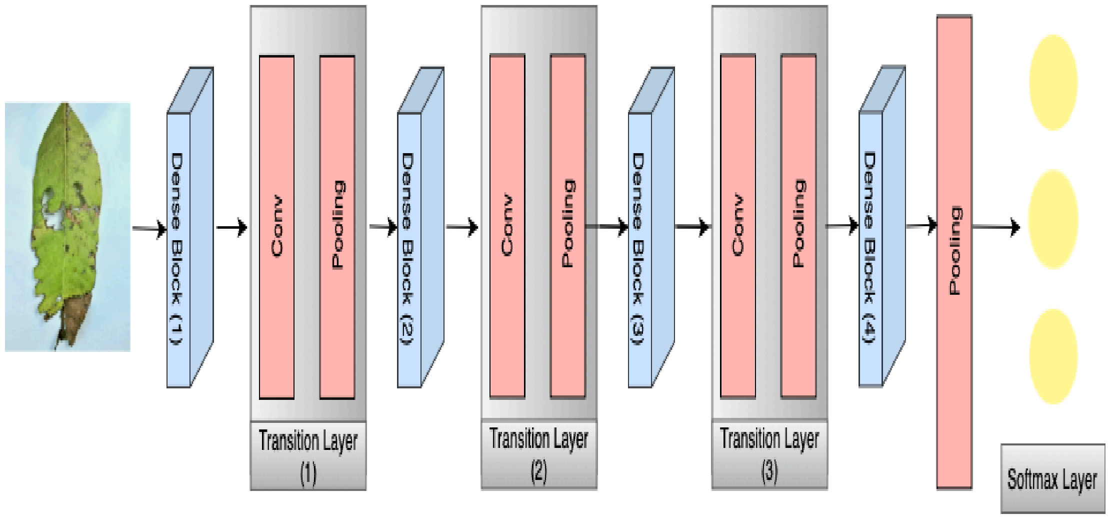


Figure 3.8: DenseNet 121 model architecture

3.6.4 ResNet50:

The ResNet50 architecture is built on five conv2D layers. Convolutional and MaxPooling layers are integrated in this way because it corresponds to the normal behavior of residual networks.Gradients often lose strength as they travel through additional layers of a deep neural network. This makes changing the weights of previous steps difficult. The network maintains a strong signal and speeds up learning by using residual connections, which allow gradients to avoid specific layers.

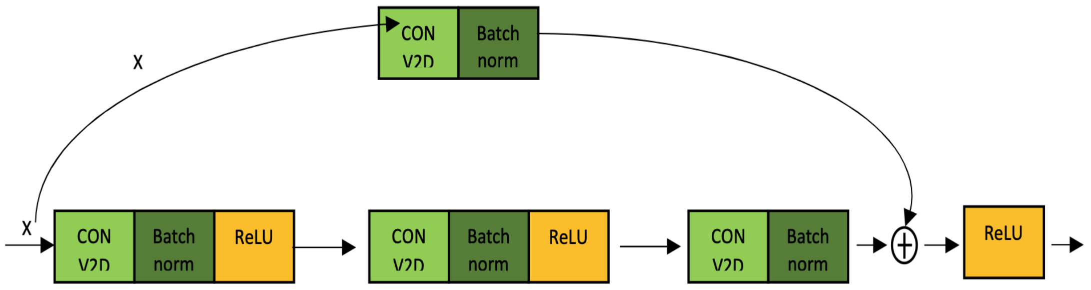


Figure 3.9: ResNet 50 model Bypass layer

The center of the 64-by-2 stride convolutional layer is 7 by 7 after traversal. A max-pooling layer with a stride of two follows the convolutional layers. A 3x3 kernel and a 64x64 filter are used in the first layer. The next layer makes use of a 256x256 filter and a 64x64 kernel. The MaxPooling layer is made up of a three-times-repeated sequence of two operations.

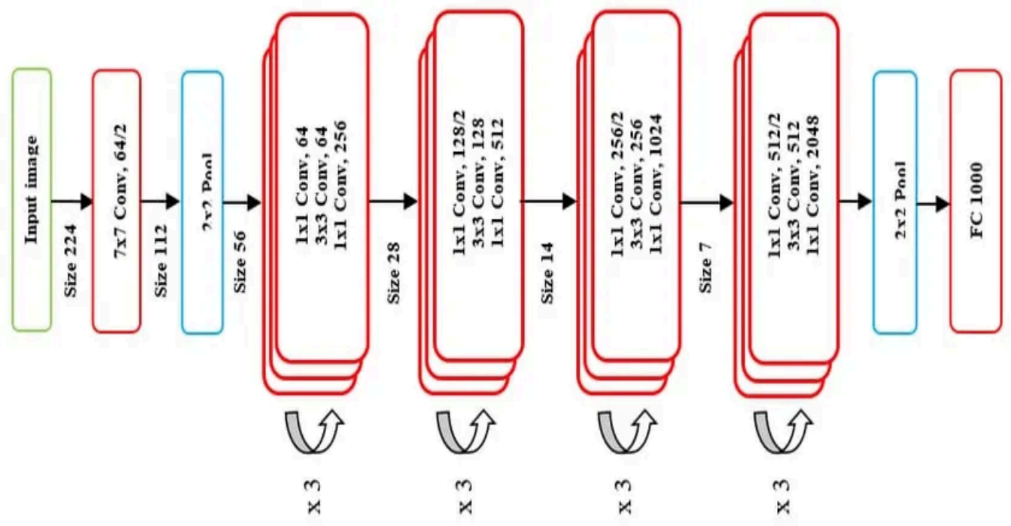


Figure 3.10: Resnet50 model architecture

The fully connected layer, average pooling layers, and convolutional layers perform the procedure in a sequential order. Finally, the categorized findings are transmitted to a SoftMax layer, which allocates categories to the obtained features in order to anticipate the final conclusion properly. By leveraging residual connections, ResNet50 solves the problem of vanishing gradients and creates more stable representations.

3.6.5 VGG19:

The VGG-19 architecture consists of nineteen layers, forming an intricate convolutional neural network. VGG-19 model of deep learning utilizes the extensive ImageNet dataset, including millions of images, to facilitate its self-learning process. The input photographs undergo preprocessing using the image component and the preprocessing input module to ensure proper scaling of pixel values for the VGG-19 model. The NumPy module is used for efficient list analysis. Subsequently, the VGG-19 model is augmented with weights obtained from the ImageNet dataset. In the VGG-19 model, a convolutional operation consists of one or more dense (or completely connected) layers.

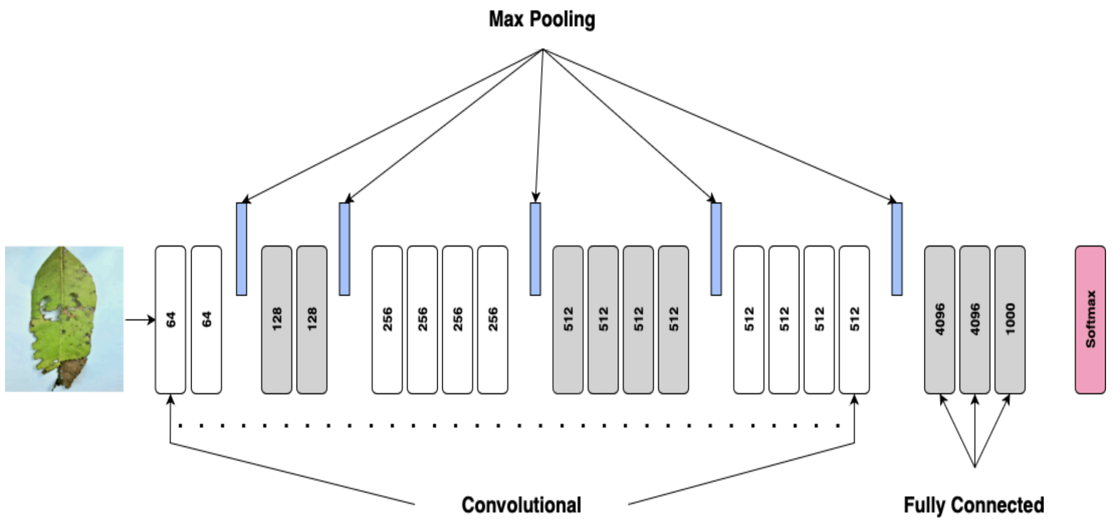


Figure 3.11: Vgg19 model architecture

Input Image: The network takes an input image with dimensions 224x224x3, which stands for width, height, and color channels (RGB).

Convolution + ReLU Layers: The image passes through a series of convolutional layers (indicated by blue blocks), where filters are applied to extract features. To introduce non-linearity, a Rectified Linear Unit (ReLU) activation function is used after each convolution.

Max Pooling: Yellow blocks represent max pooling layers that follow certain convolutional layers. These reduce the spatial dimensions of the feature maps, thus reducing the number of parameters and computation in the network.

Fully Connected + ReLU Layers: After several convolutional and max pooling layers, the network transitions to fully connected layers (purple blocks). In order to classify data, these layers combine the information learnt by the convolutional layers.

Softmax Layer: The final output layer is a softmax layer (green block), which converts the output into a probability which is being distributed over the multiple classes that the network can classify.

The VGG19 model, with its 19 layers that have learnable parameters, is known for its simplicity and depth, which contribute to its effectiveness in image classification tasks.

3.6.6 AlexNet :

Convolutional neural networks (CNNs) like AlexNet are considered pioneers in the area of deep learning, especially when it comes to image recognition tasks. This architecture, consisting of eight layers, played a vital moment for capabaling the advance role of neural networks. The first five layers of AlexNet are convolutional, whereas the last three are fully connected.

Input Layer: The network starts with an input image of size 224x224x3, where 3 refers to the RGB color channels.

Convolutional Layers: Several convolutional layers, depicted as blue bars, apply various-sized filters (noted by their dimensions, such as 11x11, 5x5, 3x3) to the input image to extract features. The "s" denotes the stride, or step size, the filter takes as it moves across the image.

Pooling Layers: These layers, which are not explicitly shown in the diagram but can be inferred from the reduction in size of feature maps, downsample the spatial dimensions to reduce the computation required for the network and to help prevent overfitting.

Fully Connected Layers (FC): When i look up in the convolutional and pooling layers, the network has fully connected layers, which are depicted as columns with multiple circles. These layers integrate the learned features and help in making the final classification decision.

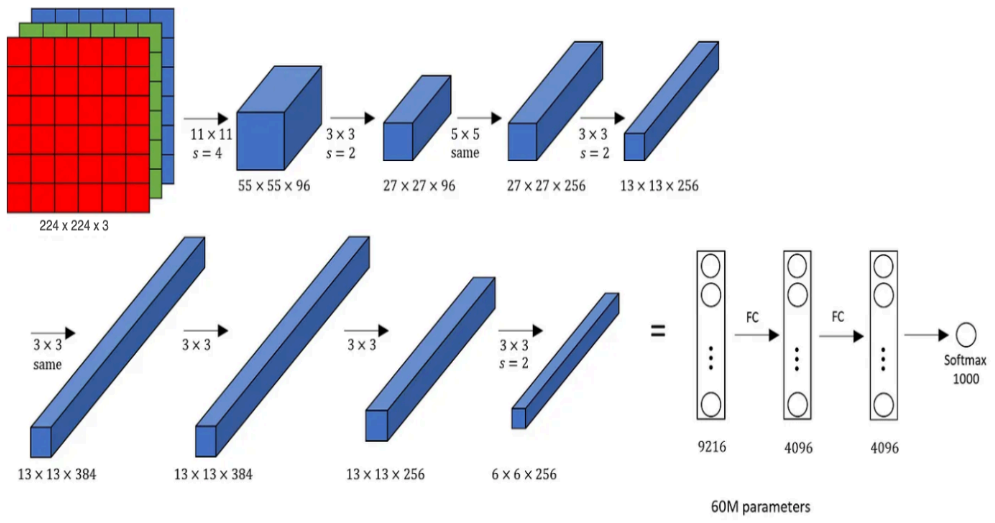


Figure 3.12: Alexnet model architecture

Softmax Layer: The last layer is a softmax layer, which converts the outputs from the fully connected layers into a probability distribution over the 1000 class labels that the network can predict.

Parameters: The diagram mentions "60M parameters," indicating the total number of trainable weights and biases in the network.

AlexNet was known for its use of the ReLU activation function, which helped it to train faster and perform better than its predecessors, and it was the first to use GPU computing to significantly reduce training times.

3.6.7 EfficientNet V2:

The architecture of EfficientNetV2, a deep learning model known for its efficiency and effectiveness in image classification tasks. The architecture figure 3.13 shows that it is structured into blocks, each with multiple convolutional layers.

It processes the input image through several blocks, each containing layers that extract features using filters of different sizes (like 3x3 or 5x5). These blocks, labeled as "MBConv," are designed to be efficient, meaning they perform well without using too much computing power. As the image moves through the blocks, it is transformed into a detailed feature map, which the network uses to identify and classify what's in the image.

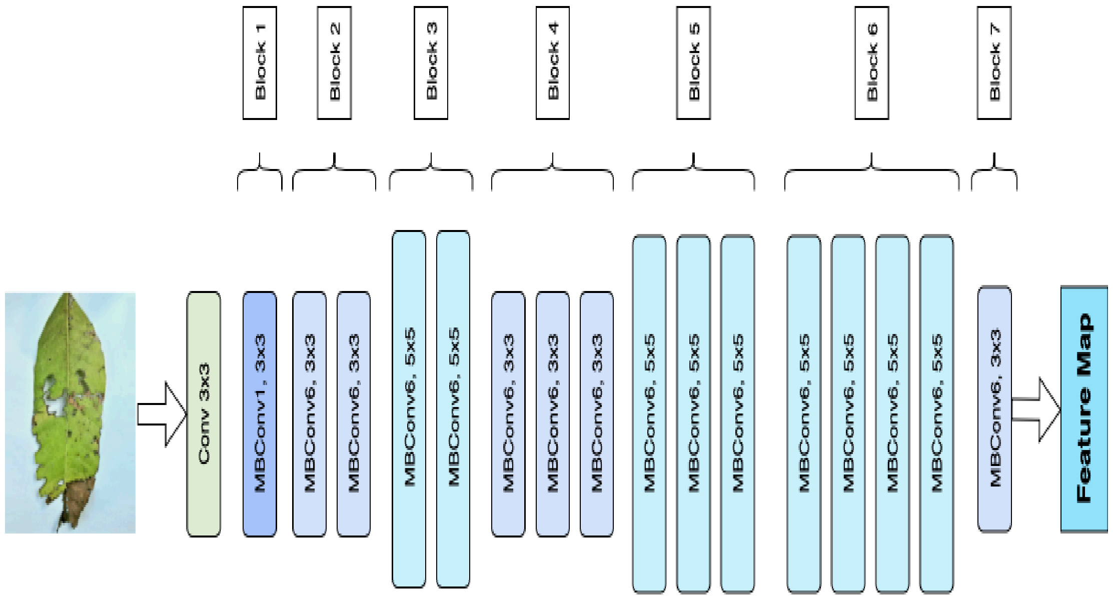


Figure 3.13: EfficientNet V2 model architecture

The workflow of figure 3.14 illustrate in a simple way as like below:

Input Image: An image is input into the model.

Processing Blocks: The image passes through several blocks (Block 1 to Block 7), where each block processes the image in stages to extract important features. The blocks vary in complexity and depth.

Feature Extraction: As the image moves through the blocks, the network extracts and refines features critical for understanding the content of the image.

Output Image: After the final block, the network outputs an image or a set of features that represents what it has learned about the image's content.

The key point of EfficientNetV2 is its ability to efficiently process images for tasks such as classification, using less computational power while maintaining high accuracy.

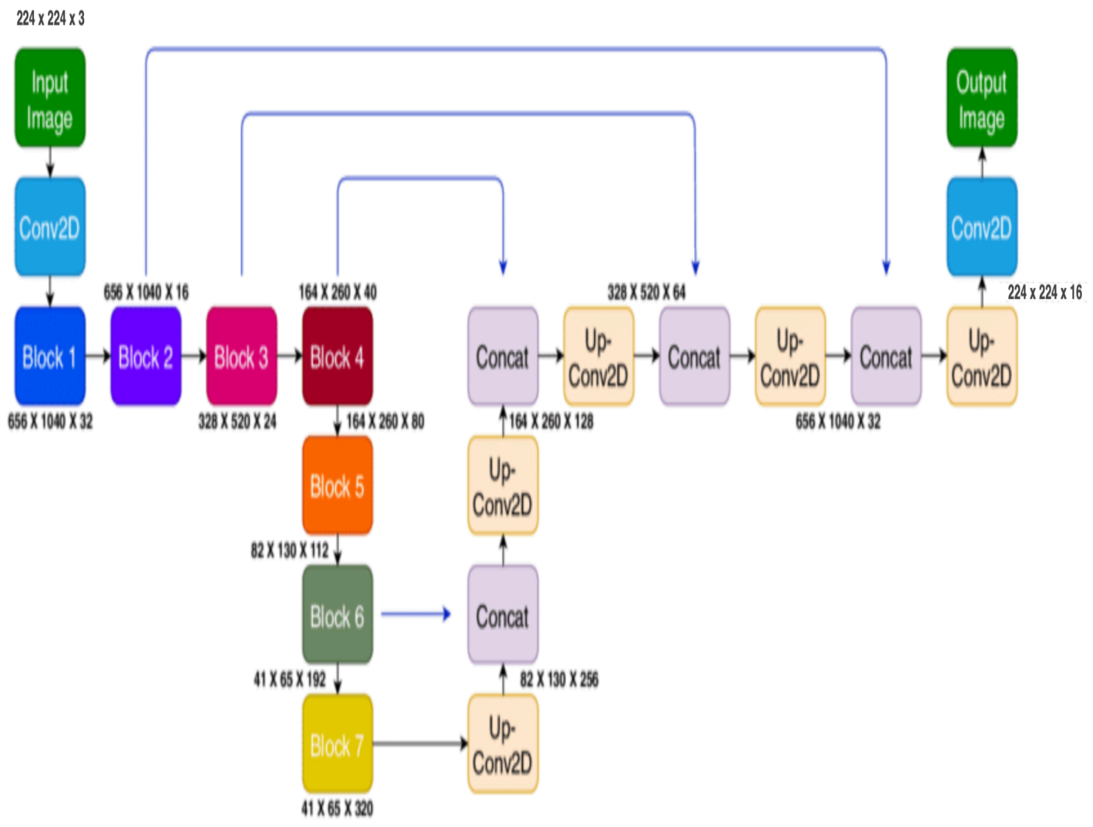


Figure 3.14: EfficientNet V2 model workflow

3.6.8 MobileNet V2:

MobileNet V2 is a combination of three convolutional layers, which are essential to computer vision tasks but can be replaced by so-called depthwise separable convolutions. The three convolution layers are called: i) Expansion layer ii) Depthwise layer iii) Projection layer. The Expansion Layer widens the network by increasing channel depth for complex feature capture, the Depthwise Layer efficiently extracts features using a filter per channel, and the Projection Layer condenses these features into a lower-dimensional space for efficiency. And the main building block look like:

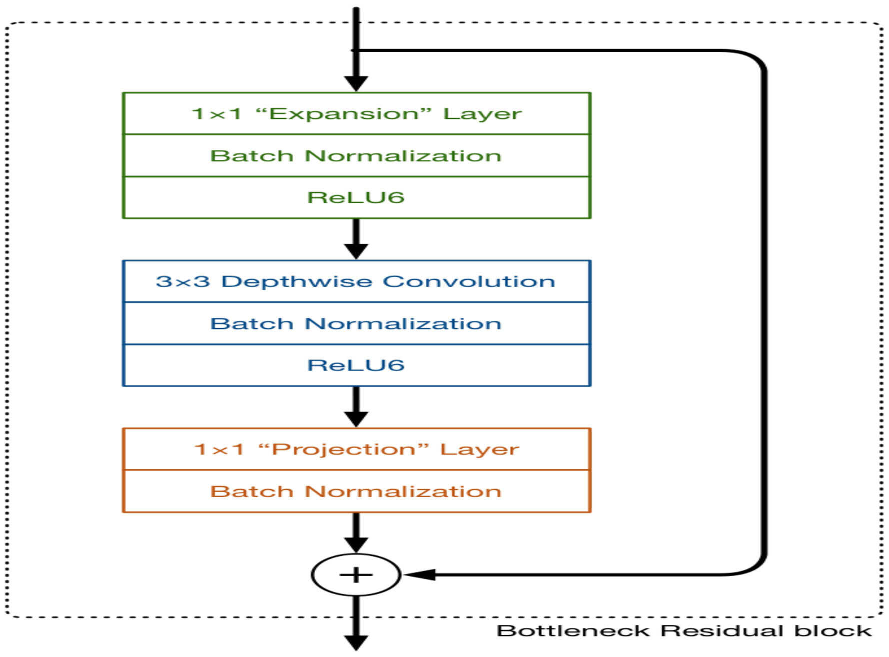


Figure 3.15: MobileNet V2 model convolution layers

MobileNet V2 replaced expensive convolutions with cheaper ones than the previous version also even if it meant using more layers than others. The projection layers and residual connections are the two primary architectural modifications in the V2 design. Upon examining the data as it traverses the network, i will see that the quantity of channels remains relatively constant in between blocks:

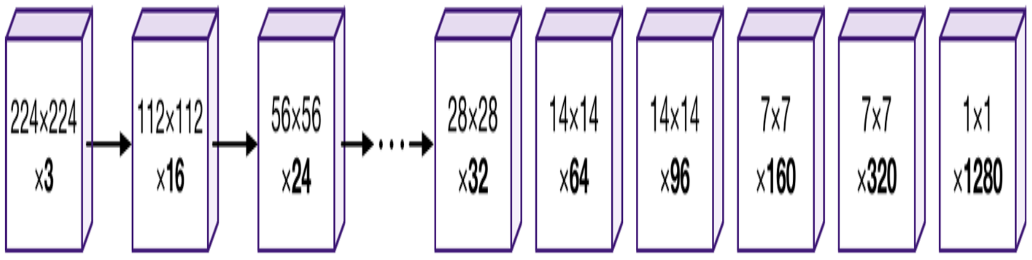


Figure 3.16: MobileNet V2 model dataflow

It's not enough to just use low-dimensional tensors. When you apply a convolutional layer to a tensor with small dimensions, you don't get very good results. Therefore, it is acceptable to work with large tensors for the goal of filtering data. MobileNet V2's block design lets us use the best parts of both systems. When data is sent between blocks, it is compressed into a form with few dimensions. It's necessary to clear the data before you can use filters on it. The following things happen in each block:

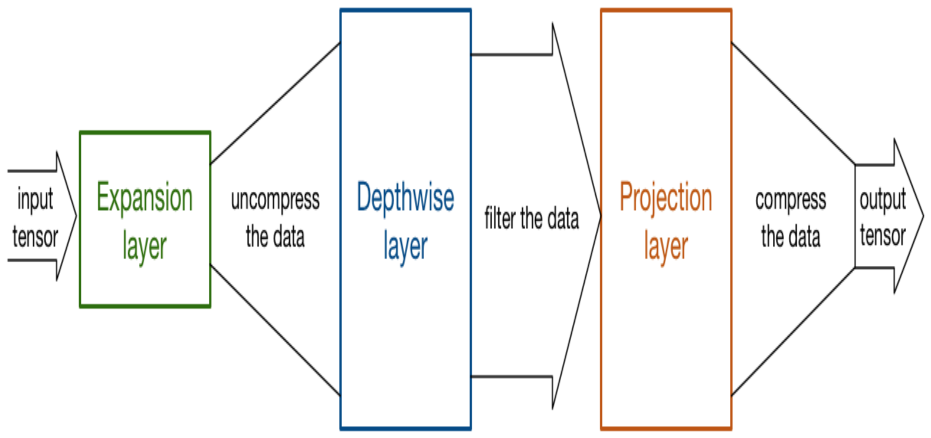


Figure 3.17: MobileNet V2 model workflow

It works a lot like unzip in that the expansion layer first decompresses the data and then puts it back in its original format. After that, the depthwise layer sorts the data based on what's important at this network level. Finally, the projection layer squeezes the data even more.

3.7 Implementation Requirements:

There are many different models and types of deep learning, and each one does a certain job. There were to be seven different kinds for the project idea. When given pictures of illnesses, these seven models are very good at figuring out what they are. Most of my files are pictures of mango leaves that I've taken in the field and in my yard. Using machine learning to accurately classify pictures and spot images of illnesses, this model gives the most accurate results.

- Macbook Pro
- Internet Connection
- Google Colab
- Photoshop
- Python

CHAPTER 4
EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction

Before I go on with this part, let me quickly go over the methods that were used in this study. This part will talk about the progress and findings that these programs have made possible. Deep learning is what I focus on most. Proposed model, InceptionV3, DenseNet121, ResNet50, AlexNet, MobileNetV2, EfficientNetV2 and VGG19 were used to classify the images. I will look closely at the results our models created in this part, which will also include a review of extra measures. I hired Adam as the only planner to help me make our models better and reduce loss as much as possible. After training the models with the training information, I expect to get very accurate results. Adam is one of the optimizers and makes a big difference in the training process.

To create models for deep learning algorithms, you need a desktop computer, laptop that has all the right tools. For training, you need a powerful graphics processing unit. I started the training process for our suggested model on a normal macbook. I chose to train my models in Google Colab because of the long processing times and the chance of getting wrong results. My use of Google Colab was very important to the success of the project's model training.

4.2 Descriptive Analysis:

The effectiveness of a deep learning classification system may be assessed by using a confusion matrix. To provide the allocation count for each class. Here, the confusion matrix will be used to determine the True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

Accuracy: How effective a model is can be measured by how many correct statements it makes.

Accuracy= ((TP+TN) / (TP+TN+FP+FN)) x 100%

Precision: It shows the percentage of positive cases successfully recognized against those mistakenly identified in the positive pulse. The ratio of correct model predictions to accurate results. Its main goal is information gathering.The formula is written as,

Precision= (TP /(TP+FP)) x 100%

Recall: The recall is found by dividing the number of true positive values by the number of false negative values. This measure is used to judge how complete the model is.

Recall = (TP/(TP+FN)) x 100%

F1-Score: The F1-score is found by taking the average of the memory and precision numbers. There are false negative and false positive results that make up the f1 score.

F1-Score = ((2 x Precision x Recall) / (Precision + Recall) x 100%

False Negative Rate (FNR): This number shows how often the model gets the negative class wrong. It is called the False Negative Rate (FNR).

FNR= (FN / (FN + TP)) x 100%

Negative Predictive Value (NPV): In the testing world, the Negative Predictive Value (NPV) is the measure of the number of real negative results to the total number of people in the community who had a fake negative result.

NPV= (TN / (TN + FN)) x 100%

False Discovery Rate(FDR): To find the false discovery rate, divide the total number of right findings by the total number of wrong ones.

FDR= (FP / (TP + FP)) x 100%

4.3 Experimental Results

The end result is an important part of any project. Everyone knows that no machine can produce results that are 100% accurate. Even though the results of my study were great, the method I used was not always correct. There are seven different types of models and two different kinds of optimizers that I use to get the best results. There was use of the Proposed model, Inception V3, DenseNet121, EfficientNet V2, AlexNet, MobileNet V2, VGG19, and ResNet50 models. The Adam optimizers were also used. In that one spot, the two optimizers came up with three different results. The best accuracy I got in a general model like DenseNet121 which gives 95.31% accuracy but my proposed model gives 97.92% accuracy. where I used an Adam planner for best performance. The EfficientNet V2 model which I also use got 82.27% accuracy which was the worst among all of them.

Table 4.1: Test result of different models and optimizer

Model Name	Optimizer	Test Precision	Test Recall	Test Fi-Score	FNR	NPV	FDR	Test Accuracy
DenseNet 121	Adam	0.92	1.00	0.96	0	1.00	0.08	95.31%
VGG 19	Adam	0.91	0.95	0.95	0	1.00	0.08	95.07%
ResNet50	Adam	0.49	0.51	0.50	0.49	0.49	0.51	93.60%
Inception V3	Adam	0.75	0.85	0.80	0.15	1.00	0.25	91.13%
AlexNet	Adam	0.91	0.91	0.91	0.08	0.91	0.08	91%
MobileNet V2	Adam	0.89	0.89	0.90	0.07	0.91	0.13	89%
Efficient Net V2	Adam	0.84	0.81	0.82	0.06	0.92	0.24	82.27%
Hybrid model	Adam	0.9674	0.88	0.92	0.11	0.89	0.03	97.92%

4.3.1 DenseNet 121 :

Here is our best model among the other six general models which gives me 95.31% accuracy. I ran 50 epochs & all time the accuracy increases & val loss becomes low. Below I show the last 10 epochs on the Table & train loss, train accuracy, value loss, value accuracy will be listed on the table. Finally, I get 91.35% accuracy, precision & recall.

Table 4.2: Train & Val accuracy of DenseNet 121 model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.0432	98.48%	0.5296	92.71
0.0409	98.23%	0.4617	94.27
0.0242	98.99%	0.5231	93.23
0.0354	98.23%	0.6851	90.62
0.0464	98.36%	0.2869	93.75

0.0349	98.61%	0.2761	94.27
0.0458	98.11%	0.3871	93.23
0.0310	98.86%	0.3928	94.79
0.0320	98.74%	0.3012	95.83
0.0145	99.24%	0.2991	95.31

It shares the difference between the model's train and validation accuracy as well as its train and validation loss after it has been trained. The accuracy of the train is always better than the accuracy of the confirmation.

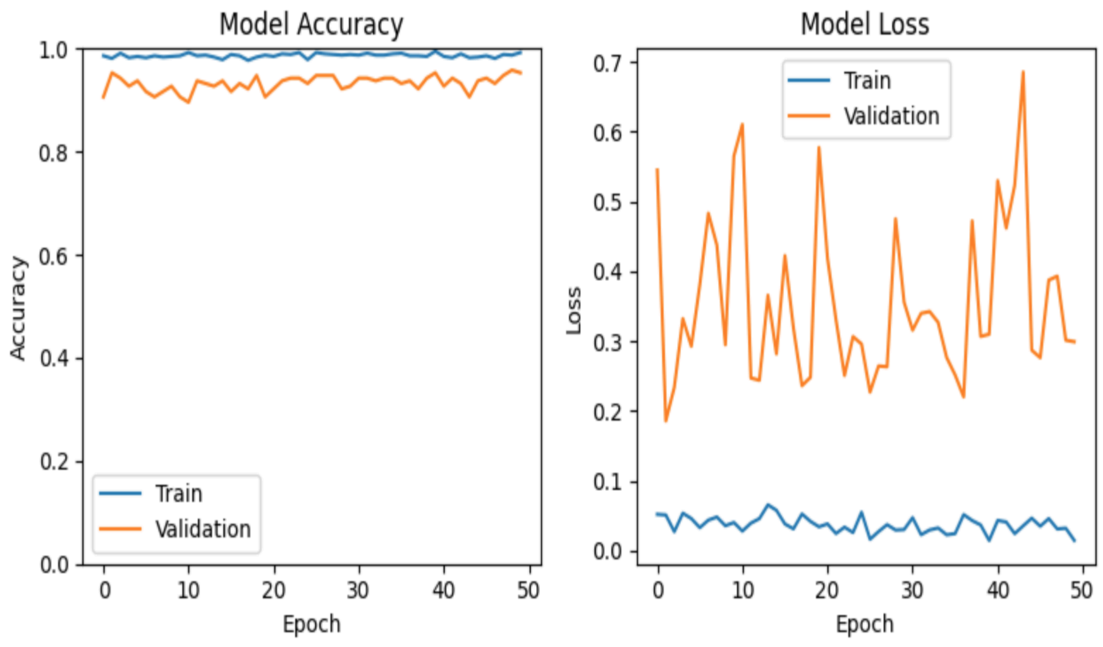


Figure 4.1: DenseNet 121 model Accuracy & Loss

There are two variables for both the real name and the expected label in the uncertainty matrix. This grid lets us figure out things like accuracy, memory, fi-score, and more.

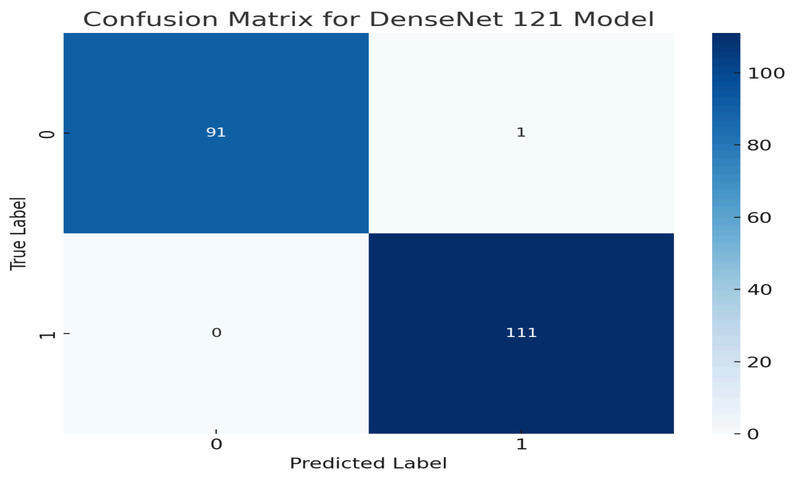


Figure 4.2: DenseNet 121 model confusion matrix

4.3.2 VGG 19:

VGG19 is the second-highest model, which gives me 95.07% accuracy. In this model, 20 epochs were run, and inside each of them, the steps per epoch were 51. In the below table, there are the last 10 epochs of the model. Here, I can see that training and val accuracy are increasing and training and val loss are decreasing. At last, after summarizing all the accuracy, it gives accuracy that is above 95%.

Table 4.3: Train & Val accuracy of VGG 19 model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.1211	95.30	0.1435	95.07
0.0857	96.16	0.2028	92.61
0.1086	95.30	0.2446	92.61
0.0922	96.53	0.1241	95.07
0.0767	97.03	0.1352	94.09
0.1093	95.92	0.1367	94.09
0.1681	94.80	0.1865	93.60
0.1235	94.06	0.2441	93.60
0.0821	96.91	0.1484	94.09
0.0923	96.04	0.1200	95.07

After training the model,Both the train and validation loss comparisons and the accuracy comparison are presented. The train accuracy is always higher than the validation accuracy.

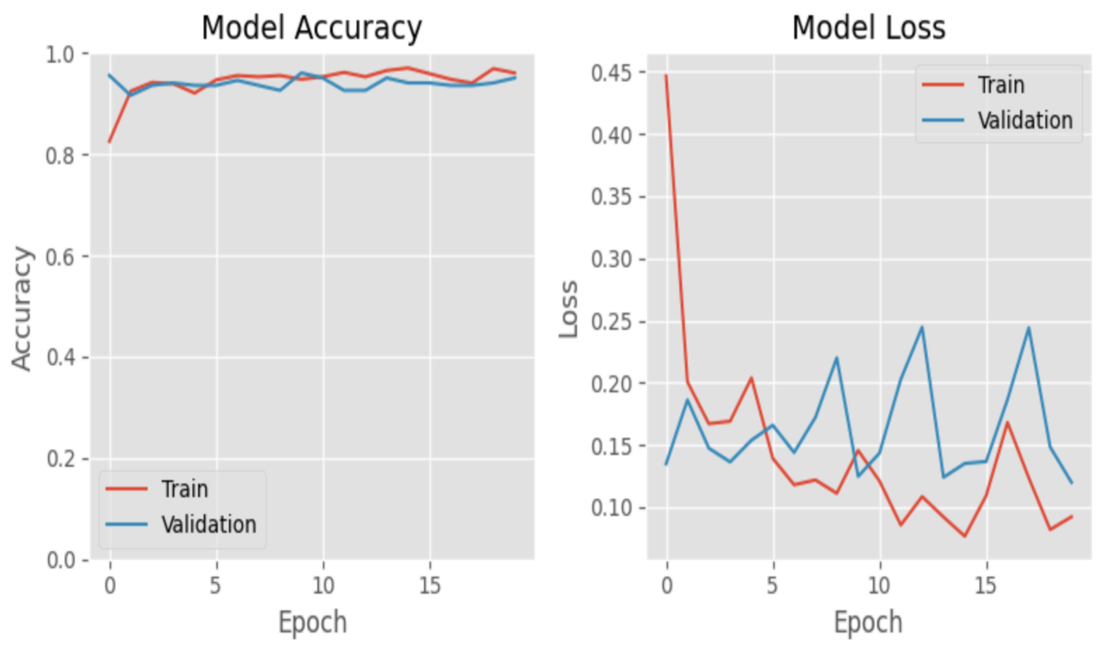


Figure 4.3: VGG 19 model Accuracy & Loss

The confusion matrix shows two dimensions for the true label and the predicted label. By calculating this matrix, i can measure precision, recall, fi-score, and so on.

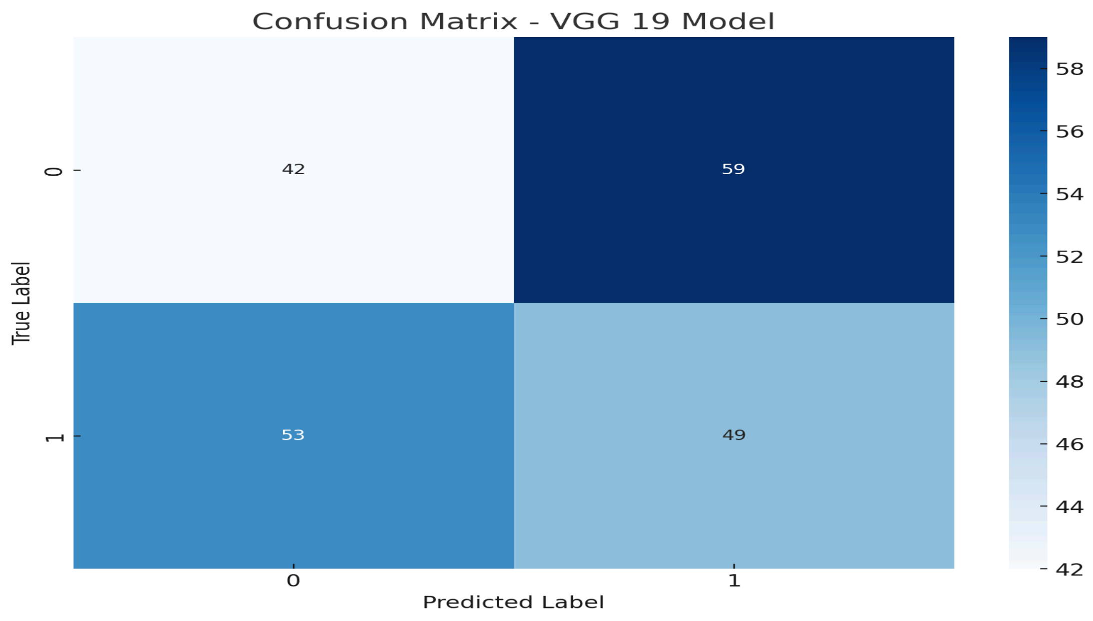


Figure 4.4: VGG 19 model confusion matrix

4.3.3 ResNet 50:

The third-best model, ResNet 50, gives me 93.60% accuracy. This model ran 50 epochs, and each one had 51 steps. Ten of the model's most recent epochs are shown below. In this case, i can see that training and val loss are going down while training and val precision are going up. When you add up all the accuracy, you get an accuracy of more than 90%.

Table 4.4: Train & Val accuracy of ResNet 50 model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.3080	99.13	4.3567	92.61
0.3099	99.01	3.6804	94.58
0.2938	98.89	3.5152	92.12
0.1103	98.89	4.0021	92.61
0.2089	99.13	3.8529	93.10
0.2306	99.13	3.6315	93.10
0.1906	99.01	3.1843	93.10
0.2824	99.26	4.0446	93.60
0.8762	98.27	5.0090	91.13
0.6300	98.89	3.8252	93.60

There is a common comparison of train and validation loss in addition to accuracy. It is always true that the train accuracy is better than the confirmation accuracy.

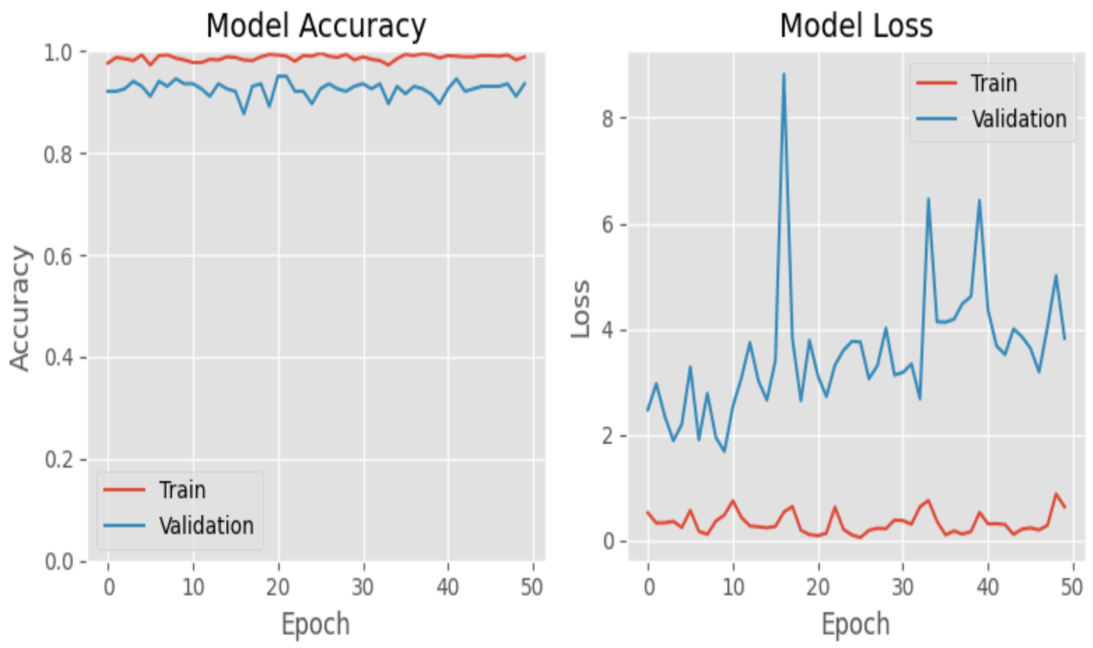


Figure 4.5: ResNet 50 model Accuracy & Loss

The confusion matrix shows that both the true label and the expected label have two dimensions. I can find out about accuracy, memory, fi-score, and other things by working out this grid.

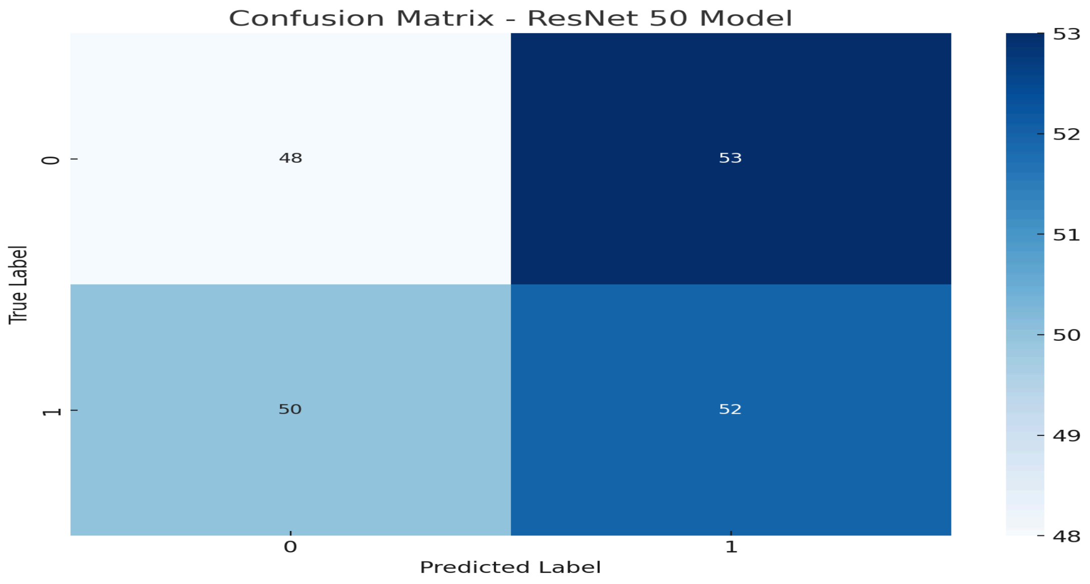


Figure 4.6: ResNet 50 model confusion matrix

4.3.4 Inception V3:

The Inception V3 model, which is the fourth-best, is 91.13% accurate for me. This model repeated itself 15 times, instead of 50 times, and each time it did 33 moves. The following images display ten of the model's most recent epochs. To be more specific, I can see that training and val loss are going down while training and val accuracy are going up. With all of these correct answers, the overall accuracy is over

91%.

Table 4.5: Train & Val accuracy of Inception V3 model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.0348	99.01	0.4465	92.61
0.0377	98.76	0.4014	91.13
0.0404	98.51	0.3524	92.61
0.0471	98.89	0.3566	93.10
0.0317	99.13	0.3435	93.10
0.0237	99.13	0.4202	92.61
0.0120	99.63	0.3952	92.12
0.0348	98.89	0.3852	94.09
0.0479	98.89	0.4851	92.61
0.0684	98.27	0.3944	91.13

After the model has been trained, the accuracy and loss difference between the train and confirmation runs are shown. Consistently, the train accuracy is higher than the proof accuracy.

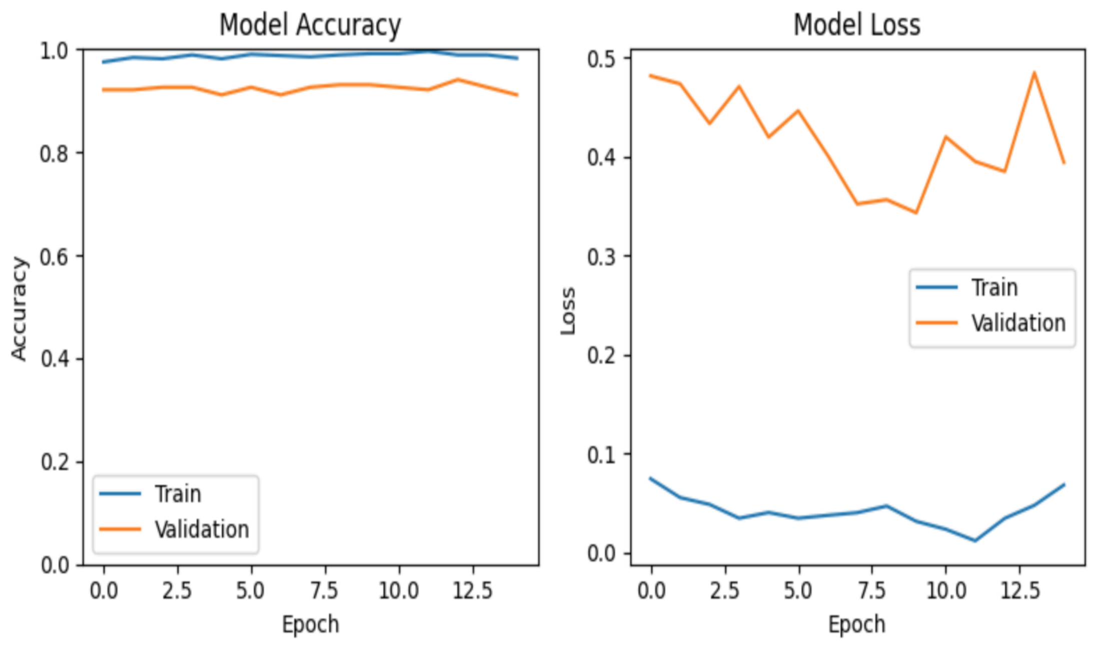


Figure 4.7: Inception V3 model Accuracy & Loss

There are two aspects to both the real name and the assumed label, as shown by the uncertainty matrix. This grid can help us learn about accuracy, memory, fi-score, and other things.

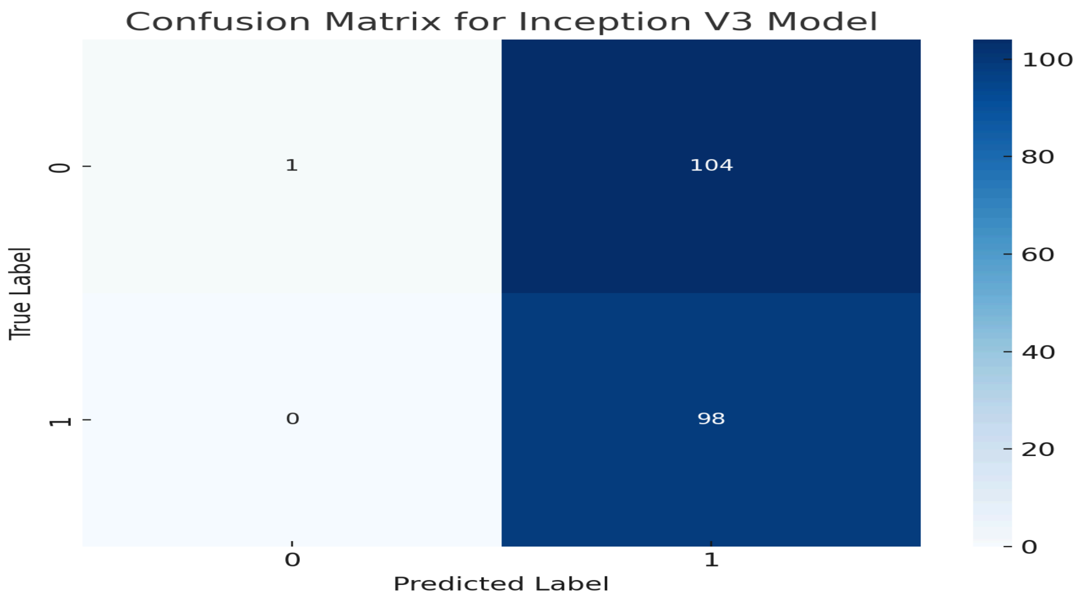


Figure 4.8: Inception V3 model confusion matrix

4.3.5 AlexNet:

That model, AlexNet, is the fifth one, and it works 91% of the time. It did 25 moves each time, so this model did itself 30 times. The next set of pictures shows the model's ten most recent epochs. I can see that training and val loss are going down, but training and val precision are going up. Like going from 86.86% to 88.02%, and in val accuracy, it was like 85.71% to 91.13%, and train loss and val loss went down like 0.3120 to 0.2742 and 0.3638 to 0.2721. So, the average score is over 90%.

Table 4.6: Train & Val accuracy of AlexNet model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.3120	86.86	0.3638	85.71
0.3057	86.86	0.2711	89.66
0.2975	87.37	0.2546	91.13
0.3037	86.21	0.2836	89.16
0.3228	87.63	0.3735	84.73
0.2703	88.66	0.3211	89.16

0.2917	88.92	0.2793	87.19
0.3162	86.73	0.3168	85.22
0.3049	86.21	0.4628	82.27
0.2742	88.02	0.2721	91.13

The precision and loss difference between the train and proof runs are shown after the model has been trained. The train accuracy is always higher than the test accuracy.

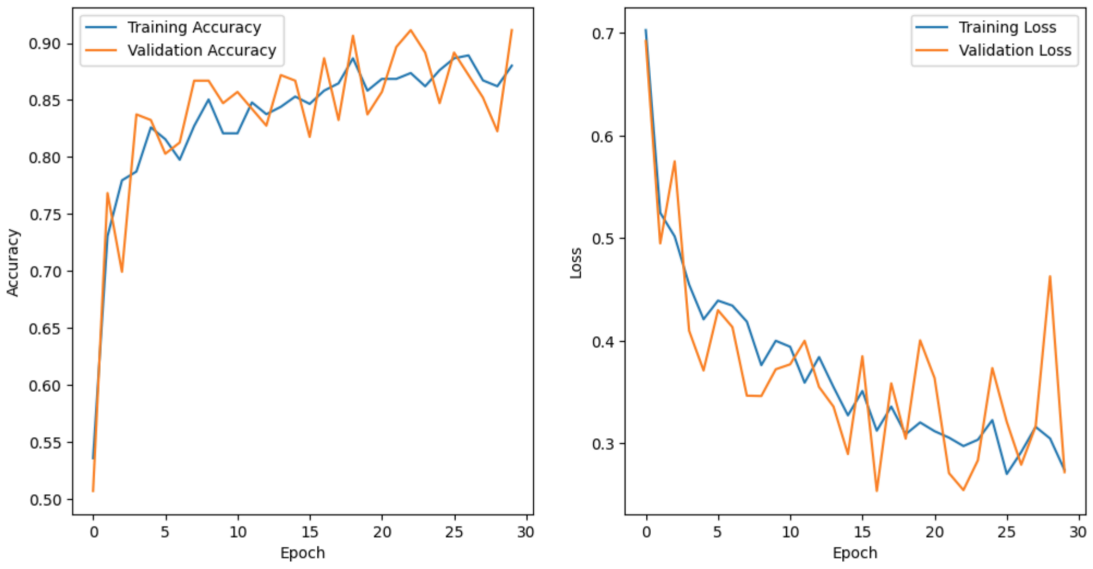


Figure 4.9: AlexNet model Accuracy & Loss

The uncertainty matrix shows that both the real name and the supposed label are made up of two parts. I can use this grid to learn about spelling, memory, fi-score, and other things.

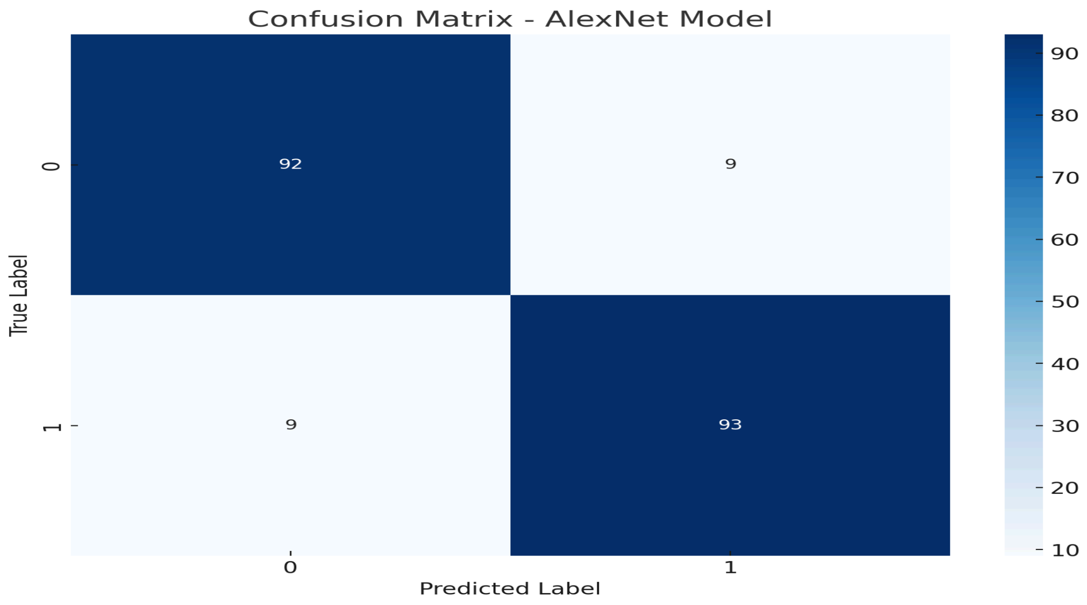


Figure 4.10: AlexNet model confusion matrix

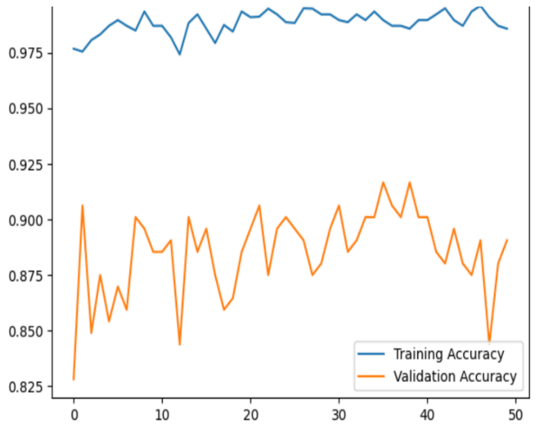
4.3.6 MobileNet V2:

The sixth one is MobileNet V2, and it works 89.13% of the time. Each time it did 50 moves, that's 25 times this model did itself. The model's last ten epochs are shown in the next set of shots. It's clear that training and val loss are going down while training and val accuracy are going up. Going from 98.97% to 99.61% in terms of accuracy, and from 90.10% to 89.06% in terms of val accuracy. Both train loss and val loss went down, by about 0.0282 to 0.0138 and 0.4338 to 0.5371. The mean score is less than 90% and more than 85%.

Table 4.7: Train & Val accuracy of MobileNet model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.0282	98.97	0.4338	90.10
0.0318	99.23	0.4926	88.54
0.0159	99.50	0.6512	88.02
0.0263	98.97	0.7495	89.58
0.0259	98.71	0.5525	88.02
0.0197	99.36	0.5256	87.50
0.0138	99.61	0.5371	89.06
0.0258	99.10	0.6300	84.38
0.0349	98.71	0.4668	88.02
0.0329	98.58	0.5148	89.06

The precision and loss difference between the train and proof runs are shown after the model has been trained. The train accuracy is always higher than the test accuracy.



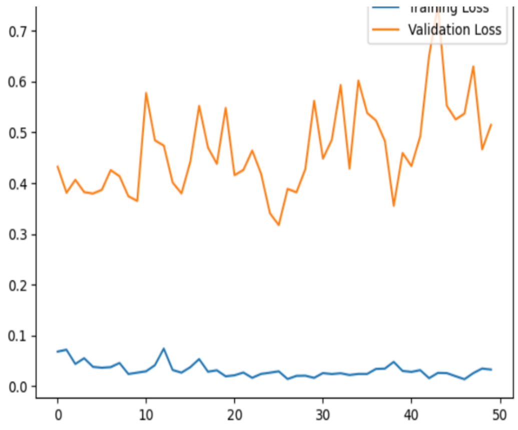


Figure 4.11: MobileNet V2 model Accuracy & Loss

The uncertainty matrix shows that both the real name and the supposed label are made up of two parts. I can use this grid to learn about spelling, memory, fi-score, and other things.

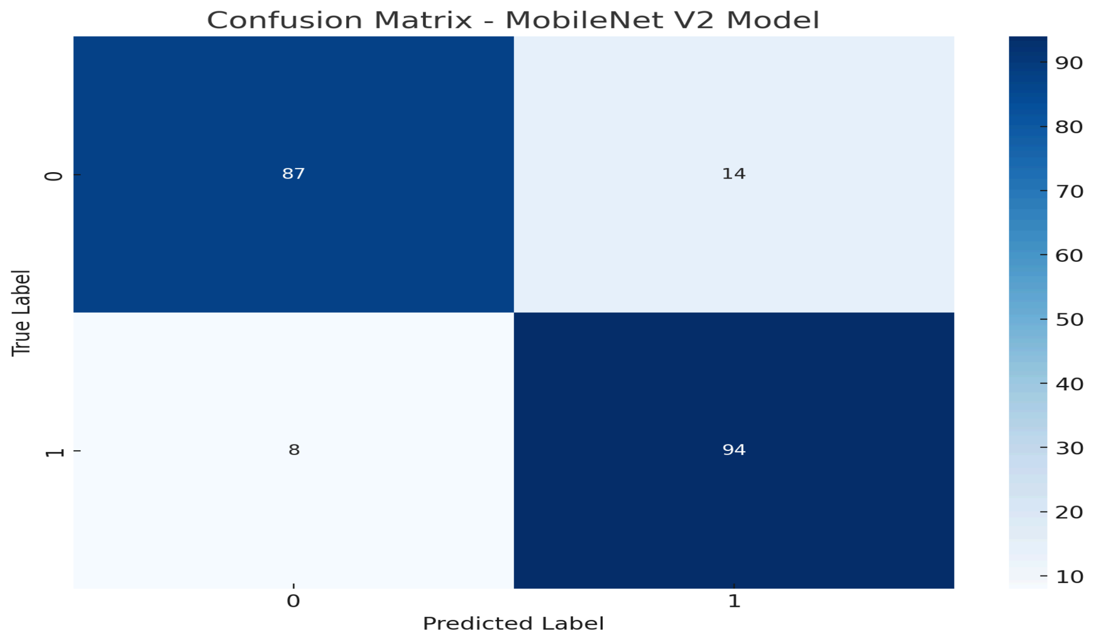


Figure 4.12: MobileNet V2 model confusion matrix

4.3.7 EfficientNet V2:

EfficientNet V2 is the seventh general model, and it operates 82.27% of the time. That means that for 50 motions it made, the model performed 50 times on its own. The following series of images displays the model's previous 10 epochs. Training and val accuracy are clearly increasing, while training and val loss are obviously decreasing. Accuracy increased from 83.71% to 83.46%, but val accuracy decreased from 82.76% to 81.77%. Train loss and val loss decreased, respectively, by about 0.3622 to 0.3557 and 0.3503 to 0.3488. Over 85% and over 80% is the mean accuracy of the train.

Table 4.8: Train & Val accuracy of EfficientNet V2 model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.3622	83.71	0.3503	82.76
0.3771	82.95	0.3897	80.30
0.3715	83.21	0.7264	71.43
0.3721	82.20	0.3964	80.79

0.3588	84.72	0.4789	78.33
0.3712	82.45	0.4130	79.80
0.3752	82.20	0.5406	72.91
0.3925	81.57	0.5084	72.41
0.3608	82.95	0.4081	78.33
0.3557	83.46	0.3488	81.77

Once the model is trained, the accuracy difference between the train and proof runs is shown. There is never a test accuracy that is lower than the train accuracy. In some cases, the val accuracy is greater than the train accuracy, and collisions happen sometimes.

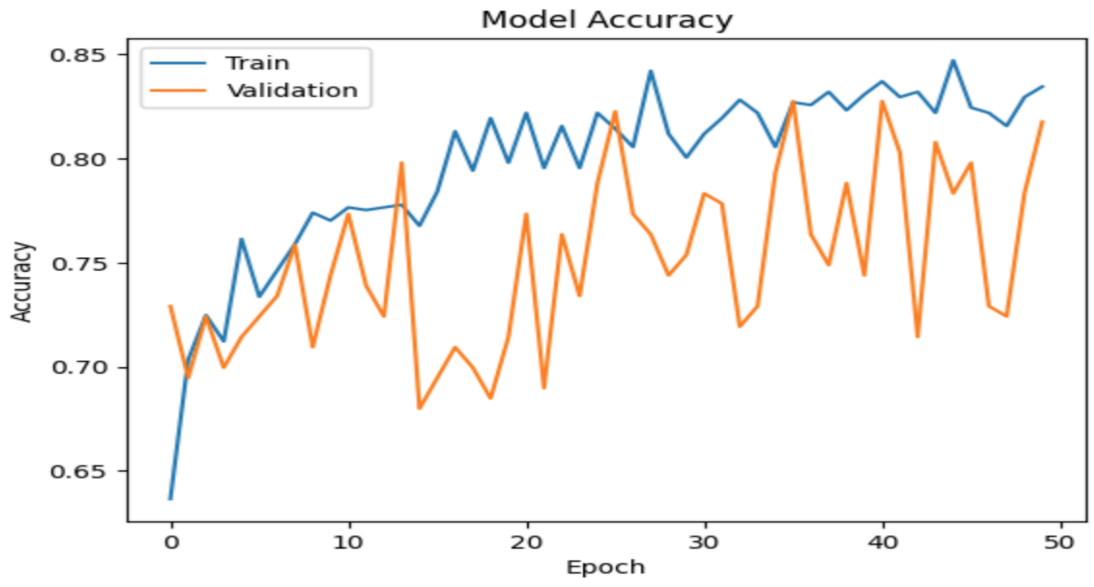


Figure 4.13: EfficientNet V2 model Accuracy & Loss

Both the True label and the Proposed label consist of two components, as the uncertainty matrix demonstrates. This grid may be used to teach us fi-score, spelling, memorization, and other subjects.

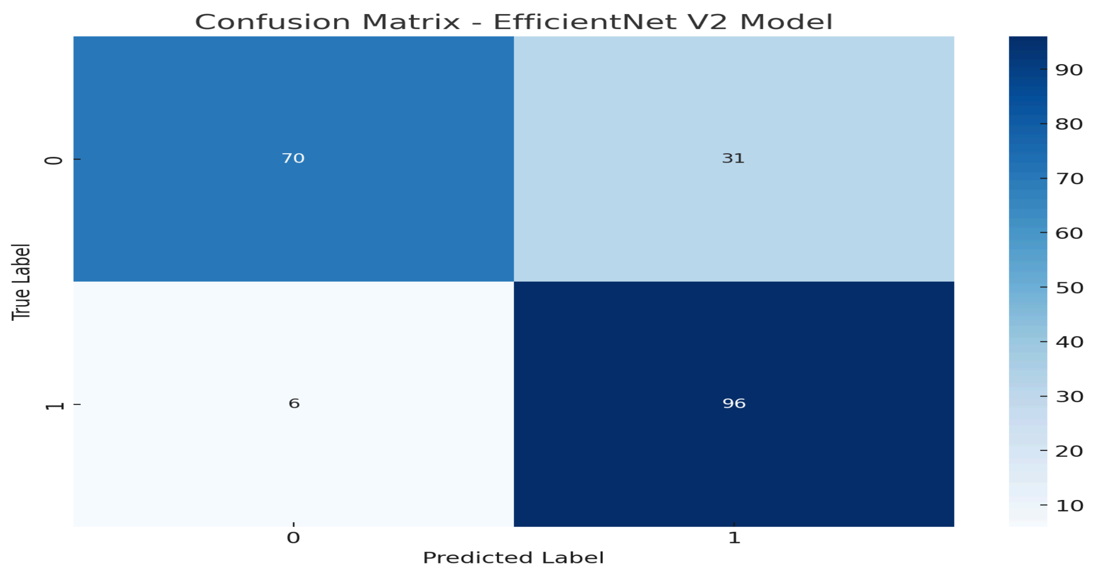


Figure 4.14: EfficientNet V2 model confusion matrix

4.3.8 Proposed model (InceptionV3 with DenseNet121):

Last one is my Proposed model, and it operates 97.27% accuracy which is better than among the other seven general CNN models. That means that for 30 motions it made, the model performed 22 times on its own. The following series of images displays the model's previous 10 epochs. Training and val accuracy are clearly increasing, while training and val loss are obviously decreasing. Accuracy was from 99.85% to 99.70%, but val accuracy was similar as 94.20% to 94.20%. Train loss and val loss decreased, respectively, by about 0.3622 to 0.3557 and 0.3503 to 0.3488. Over 85% and over 80% is the mean accuracy of the train.

Table 4.9: Train & Val accuracy of Proposed model

Train Loss	Train Accuracy	Val Loss	Val Accuracy
0.0156	0.9985	0.1328	0.9420
0.0140	0.9970	0.1628	0.9565
0.0123	0.9970	0.2184	0.9420
0.0189	0.9957	0.1960	0.9420
0.0116	0.9970	0.0124	1.0000

0.0126	0.9970	0.1962	0.9420
0.0083	1.0000	0.1995	0.9130
0.0119	0.9970	0.1162	0.9710
0.0082	1.0000	0.2697	0.9275
0.0067	1.0000	0.1463	0.9565

The difference in accuracy between the train run and the proof run is shown after the model has been taught. The accuracy of the test is never less than the precision of the train. Sometimes, the accuracy of the val is better than the accuracy of the train, and accidents do happen.

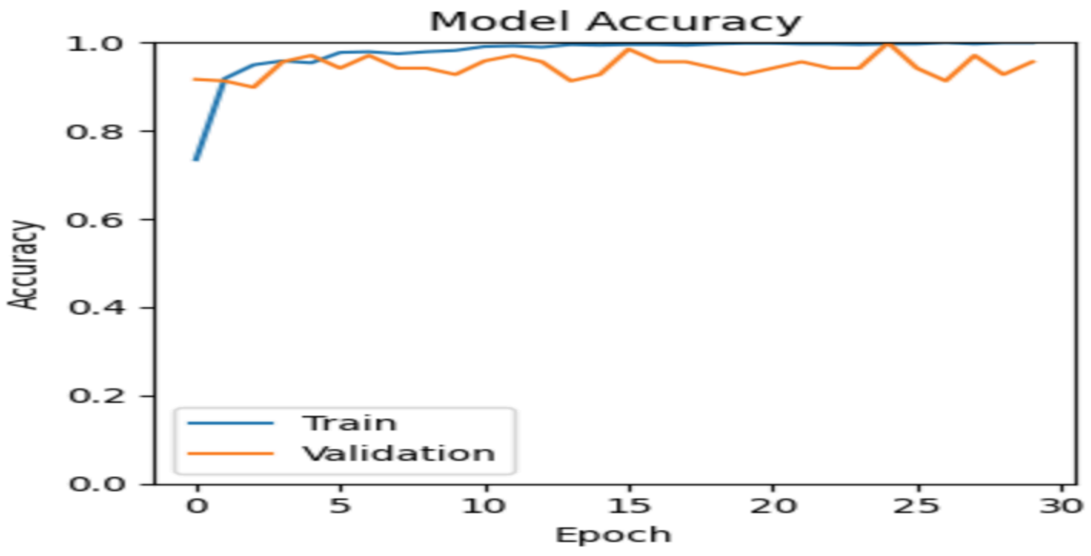


Figure 4.15: Proposed model Accuracy & Loss

The uncertainty matrix shows that both the True label and the Proposed label are made up of two parts. I can use this grid to learn fi-score, writing, how to remember things, and other things.

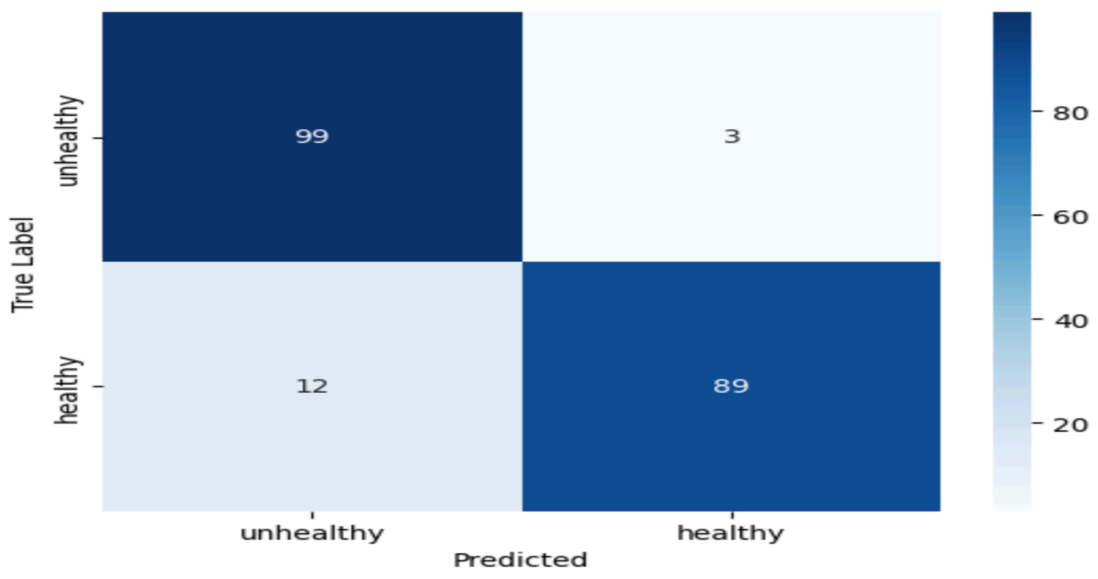


Figure 4.16: Proposed model confusion matrix

4.4 Discussion:

When compared to other classification models, experiments show that DenseNet121 has the best accuracy, F1 score, and recall between seven other models of CNN but my proposed model gets best accuracy among all other models.The most accurate and negative predictive value (NPV) model was DenseNet121. I learned that the DenseNet121 method might help us group how users feel from the studies and research I looked at above. In fact, farmers can tell from the sorting results if the leaf is free of disease or not. Farmers, on the other hand, can act quickly when they learn about diseases to protect their goods. It will make more fruits available and raise the amount that is grown.

The Google shared notepad stores all of the work from start to finish. As a first step, several pictures of sick mango leaves were scanned and identified. The pictures are picked from the dataset's collection. The prediction becomes accessible for viewing once the images have been submitted.

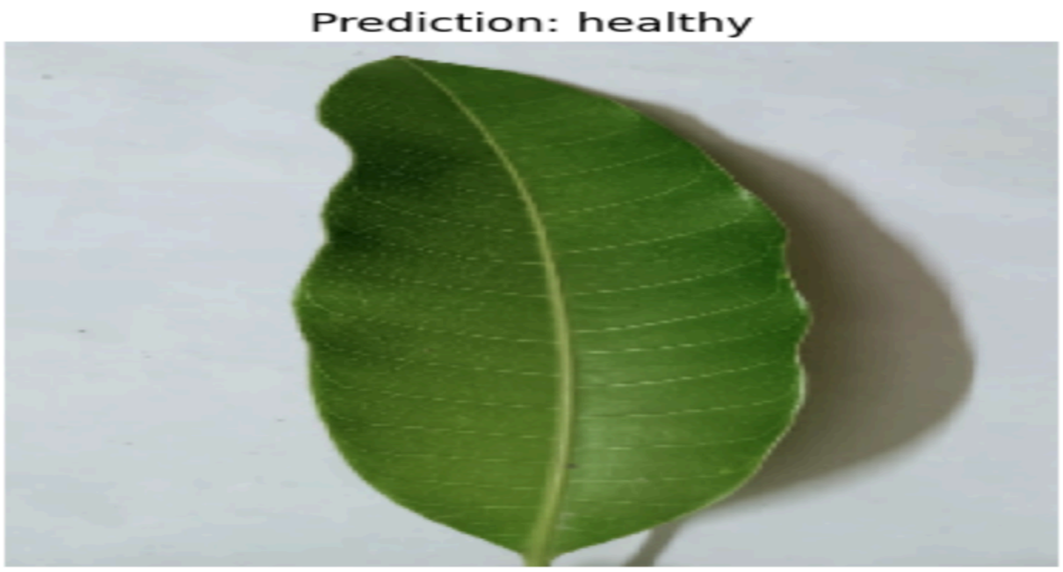


Fig. 4.17. Predicting actual leaf classes

As you can see in Figure 4.5, our model is a perfect guess for the disease. The actual sign here says "healthy," and our model said it would be "healthy.

CHAPTER 5
IMPACT ON SOCIETY, ENVIRONMENT, AND SUSTAINABILITY

5.1 Impact on Society

Mangoes are more than just a fruit in Asian countries like Bangladesh, they're a cultural and nutritional treasure. Integral to many dishes, they play a vital role in the food heritage of Bangladesh. Beyond being a popular food, mangoes are key in the food and beverage industry, used in diverse products, including alcoholic beverages. Their starch is also valuable, often used for thickening in cooking. Importantly, mangoes are crucial in agriculture, especially in disease management. Effective control of mango diseases is vital for crop health and food security. This makes disease prediction in mango cultivation a significant aspect of agriculture. Overall, mangoes are not just food; they represent a rich cultural heritage, are essential in various culinary practices, and are critical for agricultural sustainability and food security. Their importance underscores the need for ongoing research and innovation in disease management and cultivation techniques.

5.2 Impact on the Environment

The environmental impact of detecting diseases in mango cultivation is significant. By identifying diseases early, farmers can manage their crops more effectively, reducing the need for harmful pesticides and fungicides. This approach not only benefits the environment but also protects human health. The key is to detect these diseases efficiently, which can lead to a decrease in chemical usage, promoting a healthier ecosystem. Additionally, early detection means better crop management, resulting in less waste and more efficient use of resources. However, it's important to consider the environmental impact of the research methods themselves. For instance, if research involves harmful substances or practices affecting soil or water, it could harm the environment. In summary, while disease detection in mangoes can have positive environmental effects by reducing chemical use and waste, careful consideration of research methods is necessary to avoid any negative impacts.

5.3 Ethical Aspects

In researching mango leaf disease, ethical considerations are paramount. Ensuring data accuracy is essential to avoid misleading conclusions. It's also vital to respect the local communities and ecosystems affected by the research. This involves considering the wider societal implications of the study and striving to avoid any negative impacts. By focusing on these ethical aspects, the research not only gains credibility but also contributes positively to society and the environment. It's about being responsible and thoughtful in every step of the research process.

5.4 Sustainability Plan

The long-term plan for finding mango leaf diseases stresses the importance of always getting better and adapting. It means always coming up with new and better ways to deal with problems in agriculture. Adding new tools to these ways is one of the main goals for the future. Working with local farmers is also important to make sure these technologies are used in a manner that is beneficial and environmentally benign. This method is meant to keep mango trees healthy over time, making sure they are a useful and eco-friendly resource for many years to come.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Summary of the Study

The study primarily focused on using Convolutional Neural Networks (CNNs) for the classification and analysis of mango leaf diseases. Various deep learning models such as Proposed model (InceptionV3 with DenseNet121), DenseNet121, VGG19, ResNet50, AlexNet, MobileNet V2, and EfficientNet V2 were employed to classify images of mango leaves. These models were trained and tested on a dataset, and their performance metrics like accuracy, precision, recall, and F1-score were evaluated.

The Proposed model demonstrated the highest accuracy, indicating its effectiveness in the classification of mango leaf diseases. The research showed the potential of deep learning techniques in accurately identifying diseases in mango leaves, which could be a significant contribution to agriculture, particularly in disease management and prevention.

6.2 Conclusions

The research successfully demonstrated the applicability of deep learning models, particularly CNNs, in classifying and identifying diseases in mango leaves. Among the various models tested, my Proposed model shows the highest performance in terms of accuracy, F1 score, and recall. This highlights the potential of using advanced techniques in agricultural applications, offering a reliable method for early disease detection in mango leaves, which is crucial for effective disease management.

The study underscores the importance of technology in enhancing agricultural practices and supports the need for further exploration in this field. The use of CNNs in plant disease detection not only contributes to improving the quality of crop production but also has broader implications for sustainable agriculture practices.

6.3 Implication for further study

- To do this, we're going to: Use more deep learning techniques to make this model more accurate.
- I am going to use a combination method to put this idea into action soon.
- I want to grow even more data in the future so that i can classify it with leaf and fruit and make real-time recognition methods.

REFERENCES

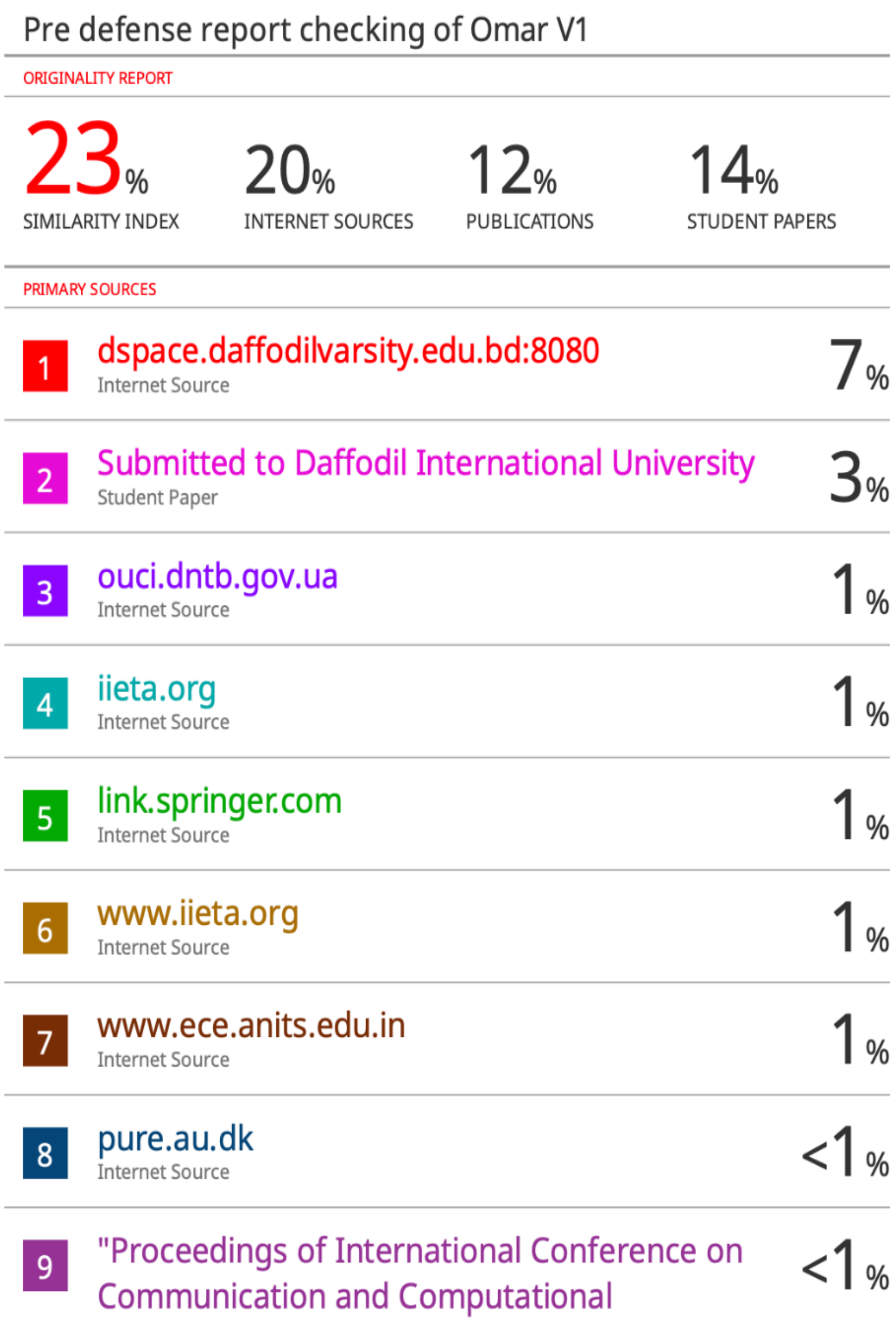
1. Tete, T. N., & Kamlu, S. (2017). Plant disease detection using different algorithms. *Computer Science and Information Systems (FedCSIS), 2019 Federated Conference On*. https://doi.org/10.15439/2017r24
2. Mohapatra, M., Parida, A. K., Mallick, P. K., Zymbler, M., & Kumar, S. (2022). Botanical leaf disease detection and classification using convolutional neural network: a hybrid metaheuristic enabled approach. *Computers*, *11*(5), 82. https://doi.org/10.3390/computers1iang, C. (2019b). Real-Time detection of Apple leaf diseases using deep learnin1050082
3. Jiang, P., Chen, Y., Liu, B., He, D., & Lg approach based on improved convolutional neuralnetworks.*IEEEAccess*,*7*,59069–59080.https://doi.org/10.1109/access.2019.2914929
4. Pham, T. T., Van Tran, L., & Dao, S. V. T. (2020). Early disease classification of mango leaves using Feed-Forward neural network and hybrid metaheuristic feature selection.*IEEEAccess*,*8*,189960–189973.https://doi.org/10.1109/access.2020.3031914
5. Chouhan, S. S., Kaul, A., Singh, U. P., & Jain, S. (2018). Bacterial Foraging Optimization Based Radial Basis Function Neural Network (BRBFNN) for identification and Classification of plant leaf diseases: An Automatic Approach towardsplantPathology.*IEEEAccess*,*6*,8852–8863.https://doi.org/10.1109/access.2018.2800685
6. Wu, C., Luo, C., Xiong, N., Zhang, W., & Kim, T. (2018). A greedy deep learning method for medical disease analysis. *IEEE Access*, *6*, 20021–20030. https://doi.org/10.1109/access.2018.2823979
7. Srivastava, P., Mishra, K., Awasthi, V., Sahu, V. K., & Pal, P. K. (2021). PLANT DISEASE DETECTION USING CONVOLUTIONAL NEURAL NETWORK. *InternationalJournalofAdvancedResearch*,*9*(01),691–698.https://doi.org/10.21474/ijar01/12346
8. Anitha, J., & Saranya, N. (2022b). Cassava Leaf Disease Identification and Detection using Deep Learning approach. *International Journal of Computers Communications & Control*, *17*(2). https://doi.org/10.15837/ijccc.2022.2.4356
9. Gining, R. a. J., Fauzi, S. S. M., Yusoff, N. M., Razak, T. R., Ismail, M. H., Zaki, N. a. M., & Abdullah, F. (2021). Harumanis mango leaf disease recognition system using image processing technique. *Indonesian Journal of Electrical EngineeringandComputerScience*,*23*(1),378.https://doi.org/10.11591/ijeecs.v23.i1.pp378-386
10. Fuentes, A., Yoon, S., Kim, S. C., & Park, D. S. (2017). A robust Deep-Learning-Based detector for Real-Time tomato plant diseases and pests recognition. *Sensors*, *17*(9), 2022. https://doi.org/10.3390/s17092022
11. Frederick, Q., Burks, T. F., Watson, A., Yadav, P. K., Qin, J., Kim, M. J., & Ritenour, M. A. (2023). Selecting Hyperspectral Bands and Extracting Features with a Custom Shallow Convolutional Neural Network to Classify Citrus Peel Defects.*SmartAgriculturalTechnology*,*6*,100365.ttps://doi.org/10.1016/j.atech.2023.100365

12. Yohannes Agegnehu Bezabih, Biniyam Mulugeta Abuhayi, Aleka Melese Ayalew, Habtamu Ayenew Asegie, Classification of Pumpkin Disease by Using a Hybrid Approach,*SmartAgriculturalTechnology*(2024).https://doi.org/10.1016/j.atech.2024.100398

13. Kusrini, K., Suputa, S., Setyanto, A., Agastya, I. M. A., Priantoro, H., Chandramouli, K., & Izquierdo, E. (2020). Data augmentation for automated pest classification in Mango farms. *Computers and Electronics in Agriculture*, *179*, 105842. https://doi.org/10.1016/j.compag.2020.105842
14. Argüeso, D., Picón, A., Irusta, U., Medela, A., San-Emeterio, M. G., Bereciartua, A., & Alvarez-Gila, A. (2020). Few-Shot Learning approach for plant disease classification using images taken in the field. *Computers and Electronics in Agriculture*, *175*, 105542. https://doi.org/10.1016/j.compag.2020.105542
15. Sujatha, R., Chatterjee, J. M., Jhanjhi, N. Z., & Brohi, S. N. (2021). Performance of deep learning vs machine learning in plant leaf disease detection. *MicroprocessorsandMicrosystems*,*80*,103615.https://doi.org/10.1016/j.micpro.2020.103615
16. Fiona, R., Thomas, S., Maria, I. J., & Hannah, B. (2019). Identification of ripe and unripe citrus fruits using artificial neural network. *Journal of Physics: ConferenceSeries*,*1362*(1),012033.https://doi.org/10.1088/1742-6596/1362/1/012033
17. Tian, X., Liu, X., He, X., Zhang, C., Li, J., & Huang, W. (2023). Detection of early bruises on apples using hyperspectral reflectance imaging coupled with optimal wavelengths selection and improved watershed segmentation algorithm. *Journal of the Science of Food and Agriculture*, *103*(13), 6689–6705. https://doi.org/10.1002/jsfa.12764
18. Özgüven, M. M., & Adem, K. (2019). Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms. *Physica A: StatisticalMechanicsandItsApplications*,*535*,122537.https://doi.org/10.1016/j.physa.2019.122537
19. Barbedo, J. G. A. (2019c). Plant disease identification from individual lesions and spots using deep learning. *Biosystems Engineering*, *180*, 96–107. https://doi.org/10.1016/j.biosystemseng.2019.02.002
20. Hu, G., Yang, X., Zhang, Y., & Wan, M. (2019). Identification of tea leaf diseases by using an improved deep convolutional neural network. *Sustainable Computing:InformaticsandSystems*,*24*,100353.https://doi.org/10.1016/j.suscom.2019.100353
21. Zhang, S., Huang, W., & Zhang, C. (2019). Three-channel convolutional neural networks for vegetable leaf disease recognition. *Cognitive Systems Research*, *53*, 31–41. https://doi.org/10.1016/j.cogsys.2018.04.006
22. Sladojević, S., Arsenović, M., Anderla, A., Ćulibrk, D., & Stefanović, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *ComputationalIntelligenceandNeuroscience*,*2016*,1–11.https://doi.org/10.1155/2016/3289801

23. Ramesh, S., & Vydeki, D. (2020). Recognition and classification of paddy leaf diseases using Optimized Deep Neural network with Jaya algorithm. *Information ProcessinginAgriculture*,*7*(2),249–260. https://doi.org/10.1016/j.inpa.2019.09.002
24. Pantazi, X. E., Moshou, D., & Tamouridou, A. A. (2019). Automated leaf disease detection in different crop species through image features analysis and One Class Classifiers. *Computers and Electronics in Agriculture*, *156*, 96–104. https://doi.org/10.1016/j.compag.2018.11.005
25. Rangarajan, A. K., Raja, P., & Ramesh, A. (2018). Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia Computer Science*, *133*, 1040–1047. https://doi.org/10.1016/j.procs.2018.07.070

PLAGIARISM REPORT



Pre defense report checking of Omar V1

ORIGINALITY REPORT

23%	20%	12%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	7%
2	Submitted to Daffodil International University Student Paper	3%
3	ouci.dntb.gov.ua Internet Source	1%
4	iieta.org Internet Source	1%
5	link.springer.com Internet Source	1%
6	www.iieta.org Internet Source	1%
7	www.ece.anits.edu.in Internet Source	1%
8	pure.au.dk Internet Source	<1%
9	"Proceedings of International Conference on Communication and Computational	<1%