Using Convolutional Neural Network for Fruit Disease Detection in Machine Learning Approach

Submitted By:

Md.Imran Hossain

191-35-2722

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY

Supervised By:

Ms. Fatama Binta Rafiq

Lecturer (Senior Scale)

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY

This Thesis report has been submitted in fulfillment of the requirements for the Degree of Bachelor of Science in Software Engineering

Spring-2023

APPROVAL

This thesis titled on **"Using Convolutional Neural Network for Fruit Disease Detection in Machine Learning Approach"**, submitted by **Md.Imran Hossain (ID: 191-35-2722)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

--- **Chairman**

Afsana Begum
Assistant Professor
Department of Software Engineering
Faculty of Science and Information
Technology Daffodil International University

--- **Internal Examiner 1**

Md Shohel Arman
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

--- **Internal Examiner 2**

Md. Rajib Mia
Lecturer
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

--- **External Examiner**

Dr. Md. Sazzadur Rahman
Associate Professor
Institute of Information Technology, Jahangirnagar University

DECLARATION

It hereby declares that this thesis has been completed by me under the supervision of **Ms. Fatama Binta Rafiq**, Lecturer (Senior Scale), Department of Software Engineering (SWE), Daffodil International University. It is also declared that neither this work nor any part of this has been submitted elsewhere for award of any degree by me.

Md. Imran Hossain

Md.Imran Hossain

Student ID: 191-35-2722

Department of Software Engineering

Daffodil International University.

Certified By:

Fatama Binta Rafiq

Ms. Fatama Binta Rafiq

Lecturer (Senior Scale)

Department of Software Engineering

Faculty of Science & Information Technology

Daffodil International University

ACKNOWLEDGMENT

I would like to express my deepest gratitude and also profoundly grateful to all the individuals who have provided their unwavering support and contributed to the successful completion of this thesis.

First and foremost, I would like to thank my supervisor, **Ms. Fatama Binta Rafiq, Lecturer (Senior Scale)** in the "Department of Software Engineering" at "Daffodil International University", for her invaluable guidance, patience, and unwavering support throughout the entire research process. Her expertise, insightful feedback, and constant encouragement have been instrumental in shaping the orchestration and quality of this thesis. I am truly grateful for her counsel and the opportunity to learn from their wealth of knowledge.

I extend my heartfelt appreciation to the members of my thesis committee for their time, valuable insights, and constructive criticism, and also extend my sincere appreciation to **Dr. Imran Mahmud,** Head of the "Software Engineering" Department, Faculty of Science and Information Technology. I would also like to express my gratitude to my family for their unwavering love, understanding, and encouragement. Their support and belief in my abilities have been the driving force behind my academic pursuits.

Finally, I want to acknowledge the countless particulars whose work and publications have laid this research's foundation. Their contributions to the field have laid the groundwork for further exploration and understanding.

ABSTRACT

Crop cultivation occupies a key place in the agricultural industry and is essential to supplying the world's food needs. The negative effects of diseased crops, which result in decreased production rates and resultant food loss, provide a serious concern.

Machine learning techniques' quick development has created new opportunities for solving practical problems. In this study, we investigate the use of Convolutional Neural Networks (CNN) for fruit disease detection, utilizing the strength of machine learning techniques to automate fruit illness detection and diagnosis.

This study uses machine learning to create a precise and effective system that can identify different fruit diseases for guava. We extract complex information from images, and CNN, a deep learning algorithm, promises to improve the precision of disease identification and classification.

To do this, a sizable dataset containing pictures of both healthy and fruits with various ailments has been gathered. These pictures serve as instruction. We testing data for the CNN pre-trained InceptionV3, VGG16, and Resnet50 model. We aim to attain high overall accuracy in disease identification through intensive experimentation and model optimization. The findings of this study will help fruit disease detection systems develop by enabling early disease detection and facilitating quick action, thereby lowering crop losses. We also find the F1 score, confusion matrix, and accuracy. By offering effective and automated methods for fruit disease diagnosis, the suggested approach also illustrates the potential of machine learning techniques to revolutionize the agricultural economy.

Table of Content

List of Figure

CHAPTER 1

1 INTRODUCTION

To fulfill the rising demands of a growing population and ensure global food security, the agricultural sector is essential. However, the prevalence of illnesses in crops, particularly in fruits, is one of the main issues that farmers worldwide must contend with. Fruit infections can negatively affect crop quality and productivity, resulting in major financial losses and food shortages. Effective disease management and mitigation techniques depend on prompt and accurate disease detection.

Traditional illness detection methods, which rely on visual inspection by human experts, are labor-intensive, frequently subjective, and prone to inaccuracy. Consequently, there is a critical need for automated and effective methods to identify and categorize fruit illnesses. The concept of **"inception modules,"** created by Google's InceptionV3 architecture, allows for the proficient acquisition of data at various scales by using various kernel sizes concurrently. This architecture has displayed astounding performance in several computer vision tasks and is anticipated to produce encouraging outcomes in the identification of fruit diseases.

The Visual Geometry Group at the University of Oxford developed VGG16, which is recognized for its efficiency and clarity. It is made up of many convolutional layers with small filter sizes, and fully linked layers come next. VGG16 is a well-liked option for transfer learning in CNN-based applications because of its impressive accuracy on image classification tasks.

A deep convolutional neural network architecture called ResNet50 has attracted a lot of attention in the field of image recognition and computer vision. It is recognized for its capacity to solve the vanishing gradient issue and make very deep neural networks trainable. Skip connections, often referred to as residual connections, are a component of ResNet50 that enable the network to efficiently transmit gradients and learn more intricate features. A deep convolutional neural network architecture called ResNet50 has attracted a lot of attention in the field of image recognition and computer vision.

Furthermore, we investigate the impact of varying hyperparameters and optimization techniques to maximize the CNN models' performance. Aspects such as learning rates, batch sizes, and optimizers are fine-tuned to strike the right balance between model convergence and overfitting avoidance. By carefully tuning these parameters, we aim to achieve robust and reliable fruit disease detection models capable of handling real-world scenarios with diverse environmental conditions and fruit varieties.

It is recognized for its capacity to solve the vanishing gradient issue and make very deep neural networks trainable. Skip connections, often referred to as residual connections, are a component of ResNet50 that enable the network to efficiently transmit gradients and learn more intricate features.

1.1 BACKGROUND

In the realm of agriculture, ensuring the health and productivity of crops is of utmost importance to meet the growing global demand for food. However, the occurrence of diseases in plants, including fruit-bearing plants, poses a significant challenge to farmers and the agricultural industry. Early detection and timely intervention are crucial to prevent the spread of diseases, minimize yield losses, and optimize resource utilization.

The issues of disease identification in agriculture have drawn more attention since the development of machine learning and deep learning techniques. Convolutional Neural Networks (CNNs) have become an effective tool for computer vision tasks, especially in picture analysis and recognition. CNNs were created with the particular purpose of automatically learning and extracting features from images, simulating how the human visual system works. Because it might be crucial to correctly identify small visual cues, this makes them very ideal for fruit disease diagnosis.

The integration of CNNs into agricultural practices offers several advantages over traditional methods of disease detection. Visual inspection by human experts is time-consuming, subjective, and often limited by human biases and expertise. In contrast, CNNs can analyze large volumes of image data rapidly, allowing for efficient and objective disease detection. This technology has the potential to revolutionize the agricultural sector by providing farmers with automated and reliable tools for early disease diagnosis and management.

In the field of CNN architectures, several models have demonstrated exceptional performance in image recognition tasks. InceptionV3, VGG16, and ResNet50 are among the most renowned and widely used architectures. These models have been extensively evaluated on various benchmark datasets and have shown impressive capabilities in capturing intricate patterns and distinguishing between different classes of objects.

InceptionV3, introduced the concept of inception modules, which optimize feature extraction by utilizing filters of different sizes in parallel. This architecture enhances the model's ability to capture both fine and coarse details, making it effective in detecting subtle signs of diseases in fruit images.

The model VGG16, which is distinguished by its straightforward and consistent construction. It offers a rich representation of visual characteristics by utilizing numerous convolutional layers with modest filter sizes. The impressive performance that VGG16 has shown on several picture recognition benchmarks demonstrates the promise of this algorithm for detecting fruit diseases.

The Reset family member ResNet50 tackles the problem of training very deep neural networks. It alleviates the vanishing gradient problem by incorporating skip connections or residual connections, which allow gradients to flow smoothly throughout training. This architecture is a good contender for fruit disease detection tasks due to its depth, which enhances feature extraction and classification.

This thesis compares and contrasts InceptionV3, VGG16, and ResNet50 performance in the context of fruit disease detection. We want to find the best efficient model for precisely recognizing and classifying fruit illnesses by comparing their performance on a large dataset. The results of this study will offer insightful advice for the creation and application of effective machine-learning techniques in agricultural practices, enhancing the control of diseases, boosting crop output, and ensuring the sustainability of food production.

1.2 MOTIVATION OF THE RESEARCH

In Bangladesh's agriculture industry, there is a dearth of use of contemporary technologies. This project intends to introduce the application of contemporary technology in its unique dimension in Bangladesh's agriculture. Lack of early diagnosis of agricultural diseases results in excessively large yield losses. Because current technologies are not as accurate in identifying diseases, this research makes use of Deep Learning, whose accuracy is higher.

Fruits like guava, apple, mangoes, and others are regarded as cash crops and provide significant earnings for farmers in our nation. To help the farmer, disease detection in fruits will be extremely helpful in identifying the disease in the fruit early on. In Figure *1* we saw that the GDP for agriculture is decreasing every year and the main part is also the crops and fruits diseases.

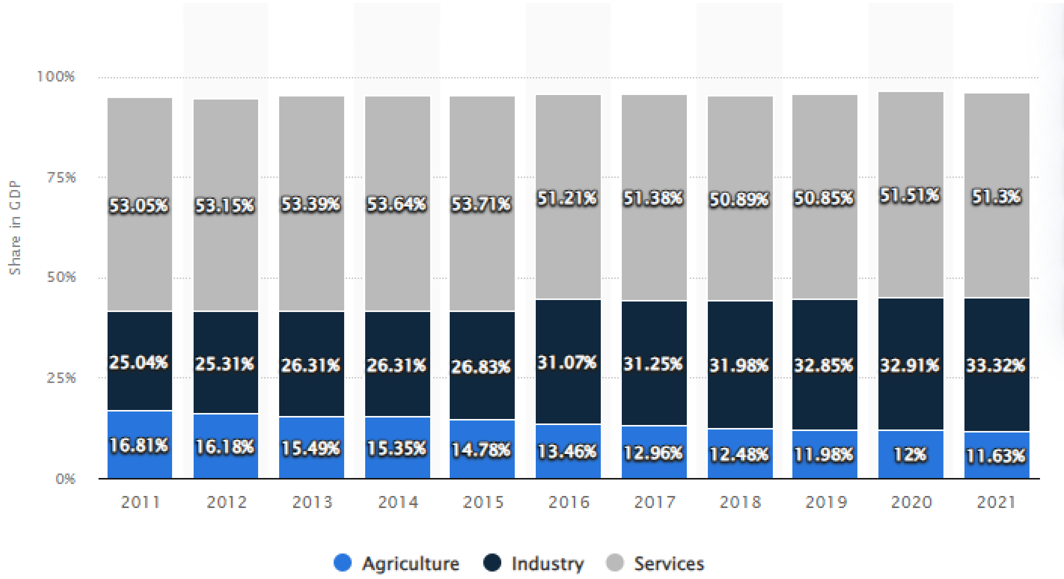


Figure 1 Share of economic sectors in the GDP in Bangladesh

The urgent need to treat fruit diseases in agriculture and the potential for Convolutional Neural Networks (CNNs) to completely transform disease detection and management are the driving forces behind this research. It is impossible to overstate the negative effects of fruit diseases on crop quality and output, which can result in considerable financial losses for producers and possibly exacerbate consumer food shortages. Inefficient, subjective, and sensitive to human error, manual inspection is a major component of traditional techniques of illness detection. To improve the productivity and efficacy of agricultural activities, there is a critical need for automated and precise disease detection systems.

1.3 PROBLEM STATEMENT

The detection and timely management of fruit diseases are critical for maintaining crop health, maximizing yield, and ensuring food security. However, traditional methods of disease identification in agriculture heavily rely on manual inspection by human experts, which is laborious, subjective, and prone to errors. These limitations hinder the efficiency and effectiveness of disease detection, leading to delayed interventions and significant economic losses for farmers.

The research seeks to investigate the usage of Convolutional Neural Networks (CNNs) in a machine-learning strategy for fruit disease detection to address this issue. CNNs are a viable technique for automated illness identification because they have demonstrated impressive success in image recognition tasks. To assure their accuracy, efficiency, and applicability for real-world agricultural scenarios, CNNs must first undergo a thorough examination and evaluation before being applied to the detection of fruit disease.

The problem can be further categorized into the following sub-problems:

Accurate Disease Detection: The primary challenge is to train a CNN model that can accurately distinguish between healthy fruits and those affected by various diseases. This requires the model to effectively learn and extract disease-specific features and patterns from input images.

Generalization Across Varieties: Fruits exhibit significant variations in appearance, shape, and texture across different varieties and stages of ripeness. The CNN model should be capable of generalizing its learning across various fruit types and detecting diseases accurately, regardless of the fruit variety.

Real-Time Detection: Efficient real-time disease detection is essential for timely interventions. The CNN model needs to process images quickly and provide prompt and accurate disease predictions to support proactive disease management practices.

1.4 RESEARCH QUESTIONS

"What is the effectiveness of using Convolutional Neural Networks (CNNs), specifically InceptionV3, VGG16, and ResNet50, in the machine learning approach for fruit disease detection?"

Sub-questions:

- How accurate are the CNN models (InceptionV3, VGG16, and ResNet50) in detecting and classifying fruit diseases compared to traditional manual inspection methods?
- How well do the CNN models generalize across different fruit varieties and stages of ripeness, ensuring accurate disease detection regardless of fruit type?
- How practical and accessible are the CNN models (InceptionV3, VGG16, and ResNet50) for deployment on resource-constrained devices or edge computing platforms in agricultural environments?

1.5 RESEARCH OBJECTIVE AND SCOPE

Fruit disease diagnosis is crucial since it aids the farmer in attaining a high crop. We already saw this in Figure 1. The goal of this initiative is to improve the farmer's options for taking preventative measures to safeguard his harvest by focusing on the early diagnosis of fruit illnesses. The major goals of this project are to use technology in agriculture, and utilizing image processing with deep learning makes it easier to get precise findings or detection.

The research aims to:

- Develop a comprehensive dataset of fruit images, including healthy fruits and various diseased fruits.
- Preprocess and augment the dataset to improve its quality and diversity.
- Train and optimize Convolutional Neural Network (CNN) models, specifically InceptionV3, VGG16, and ResNet50, for fruit disease detection.
- Evaluate and compare the performance of the trained CNN models in terms of accuracy
- Analyze the computational efficiency of the CNN models for real-time disease detection.

- Investigate the practical deployment of the trained models on resource-constrained devices or edge computing platforms.
- Provide recommendations and guidelines for selecting the most suitable CNN model and implementing CNN-based fruit disease detection systems in agriculture. These objectives aim to advance the field of fruit disease detection, enabling farmers to adopt efficient and accurate automated solutions for timely disease management and improved crop productivity.

CHAPTER 2

2 LITERATURE REVIEW

There has been extensive research on the diagnosis of leaf and fruit diseases for many years. Researchers have looked into several techniques that use pattern recognition and machine learning to improve the accuracy of disease diagnosis. These innovative techniques are applied to many other types of crops, such as wheat [2], rice [3], maize [4], and corn [5]. Golhani et al. [6] have reported a variety of experiments on neural network techniques used for identifying and categorizing diseases from images of plant leaves and fruit. To improve the precision of spot and rot detection in apples, pears, oranges, and pears, this research discusses the application of a particular sort of CNN called Mask R CNN. Based on Faster R CNN, Mask R-CNN is a better target detection method. Although the detection model suggested in the article has excellent detection accuracy, the detection speed is a little slower, according to Hongjun, Qisong, Youjun, and Hui.[11]. Citrus canker and Huanglongbing (HLB) were detected using an SVM and a fluorescent imaging system by Wetterich et al. [15]. Citrus canker and scab were correctly classified by the approach with an accuracy of 97.8%, while HLB and zinc deficiency were correctly classified with an accuracy of 95%. An SVM with ANNs is also used to increase the high classification speeds. The approach proposed will substantially engineer precise fruit disease identification and automatic classification [17]. The paper explains four features that are shape, size and color, and texture. It is also noted that the outcome of the SVM classification changes as there is a change in the training/testing ratio [18]. Liu et al. [24] trained MobileNetV2 to classify six common citrus diseases and to diagnose them. Comparing the model correctness, model size, and model validation speed with other network models, we can see that MobileNetV2 is good at classifying and recognizing citrus diseases

CHAPTER 3

3 SOFTWARE DETAILS

3.1 Language: Python

Python is a high-level, general-purpose programming language that is interpreted. Python's design philosophy places a strong emphasis on code readability and makes considerable use of whitespace. Its linguistic features and object-oriented methodology are intended For both modest and large-scale projects, programmers create logical, understandable code.

Python turns out to be a good option for these image-processing approaches. This is a result of its rising reputation as a programming language for scientific applications and the abundance of cutting-edge image-processing tools in its ecosystem.

Why do we use Python in Machine learning?

Wide Range of Libraries: Python provides a rich ecosystem of libraries and frameworks specifically designed for deep learning, such as TensorFlow, PyTorch, Keras, and Theano. These libraries offer pre-built functions and tools for building, training, and evaluating deep learning models, making it easier and more efficient to implement complex neural networks.

Simplicity and Readability: Python is known for its simplicity and readability, making it an ideal language for beginners and experienced developers. Its clean syntax allows for intuitive code organization and easier debugging, fostering faster development and easier collaboration among researchers and practitioners.

Community Support: Python has a vibrant and active community of developers, researchers, and enthusiasts. This community contributes to the development of deep learning frameworks, shares code repositories and provides support through online forums and communities. This active community ensures abundant resources and documentation are available, making it easier to find solutions and troubleshoot issues.

Versatility and Integration: Python is a flexible language that is simple to use in conjunction with other devices and software. It enables efficient data manipulation and analysis through seamless integration with libraries for data processing and visualization including NumPy, pandas, and matplotlib. Python can also easily interface with other fields, such as web development or data science, making it easier to create and implement end-to-end pipelines.

Performance Optimization: The efficient execution of neural networks is ensured by deep learning frameworks like TensorFlow and PyTorch, which implement low-level operations in optimized C++ or CUDA. Python also gives developers the freedom to use high-performance computing tools like GPUs to speed up deep learning computations.

Educational Resources: There are many instructional materials, tutorials, and online courses available thanks to Python's popularity in data science and machine learning. This encourages innovation and information sharing by making it simpler for researchers and students to understand and use deep learning techniques using Python.

3.1.1 OS: Windows 11(64bits)

The most recent operating system from Microsoft is called Windows 11. Its goal is to deliver a cutting-edge, user-friendly computer environment. Windows 11's 64-bit edition provides better performance, more memory, and compatibility with a variety of software programs and hardware gadgets. It enables users to fully utilize 64-bit processors' capabilities, allowing for quicker multitasking, improved security measures, and support for bigger quantities of system memory. For both home and business users, Windows 11 64-bit offers a reliable and effective platform that creates an immersive computing experience. Microsoft made the official announcement of Windows 11 on June 24, 2021. The initial announcement offered updated operating system components, new design elements, and new functions. However, the actual launch date of Windows 11 for general availability was October 5, 2021.

3.1.2 IDE: Anaconda Navigator

An essential component of the Anaconda distribution, a well-liked open-source platform for Python data research and scientific computing, is Anaconda Navigator, a graphical user interface (GUI). Users can access and use Anaconda's several components more easily since it offers a straightforward approach to managing packages, environments, and development tools.

With Anaconda Navigator, users can efficiently manage Python packages and dependencies. It offers a user-friendly interface to search, install, update, and remove packages with just a few clicks. This simplifies the process of setting up a Python environment and ensures that the necessary libraries and tools are readily available. Additionally, Anaconda Navigator allows users to create and manage virtual environments, which are isolated environments with their own Python installations and package dependencies. These virtual environments enable users to work on different projects with distinct requirements without worrying about conflicts or compatibility issues.

Additionally, Anaconda Navigator gives users access to well-liked development tools including Jupyter Notebook, Spyder, and JupyterLab, which are frequently used for scientific computing, machine learning, and data analysis. By allowing users to run these tools directly from the Navigator user interface, workflow is streamlined and code development, experimentation, and visualization are made easier. The Navigator interface is user-friendly and attractive to the eye, with a straightforward structure and segmented sections for packages, environments, and tools. Users of all skill levels may quickly navigate through the various areas and access the needed capabilities thanks to this. In summary, Anaconda Navigator is a powerful tool that enhances the user experience of Anaconda by offering a graphical interface for package management, environment creation, and access to popular development tools. It simplifies the process of setting up a Python data science environment and allows users to focus on their analysis, coding, and research tasks.

3.1.3 CNN: Convolutional Neural Network

Convolutional Neural Networks (CNNs) are a particular class of deep learning models that are intended for the analysis of visual data, such as photographs and movies. In a variety of applications, such as image classification, object recognition, and picture segmentation, they have achieved exceptional accuracy and performance, revolutionizing computer vision jobs. The design of the human brain's visual cortex served as an inspiration for CNNs. Convolutional, pooling, and fully connected layers are among the many layers that make them up. A collection of trainable filters are applied by the convolutional layers to the input images to identify regional patterns and characteristics. The pooling layers downsample the output, reducing spatial dimensions and extracting the most relevant information. Finally, the fully connected layers process the extracted features and make predictions. One of the key strengths of CNNs lies in their ability to automatically learn hierarchical representations of visual data. The lower layers learn simple features like edges and textures, while deeper layers learn more complex and abstract features, such as shapes and objects. This hierarchical feature extraction enables CNNs to handle variations in scale, rotation, and translation, making them robust to changes in input data.

Popular pre-trained CNN models include VGG16, VGG19, InceptionV3, ResNet50, and MobileNet, among others. These models have been trained on massive image datasets like ImageNet, which contains millions of labeled images spanning thousands of classes. Training CNNs typically involves feeding labeled data into the network and adjusting the model's parameters through backpropagation and gradient descent optimization. With a large amount of annotated data and computational resources, CNNs can learn to recognize and classify objects with exceptional accuracy.

Overall, CNNs have made important contributions to the field of computer vision and are now the standard architecture for the processing of visual input. Numerous applications have been transformed by their capacity to automatically learn and extract features from photos, which has made strides in fields like autonomous driving, medical imaging, and object recognition possible.

3.1.4 Max Pooling Layer

Convolutional neural networks (CNNs) use the Max Pooling layer as a key building block to extract robust features from input data. The input is divided into non-overlapping sections, and the maximum value within each zone is chosen to operate. Max pooling helps to reduce the spatial dimensions of the input, resulting in enhanced computational efficiency and spatial invariance. The Max Pooling layer improves the network's ability to generalize and handle variations in scale and translation by assisting in the retention of the most important features while rejecting less important information. This layer is essential for capturing hierarchical representations and expanding the network's receptive field, which improves how well CNNs perform overall in tasks like object detection and image categorization.

3.1.5 TensorFlow

Google's open-source TensorFlow machine learning framework has become quite well-known in the deep learning community. With a focus on neural networks, it offers a complete ecosystem for creating, honing, and deploying machine learning models.

The ability of TensorFlow to describe calculations as data flow graphs is one of its fundamental characteristics. Models in TensorFlow are described as a collection of linked nodes, where each node is a single operation and edges denote the data flow between operations. Automatic differentiation and effective parallel processing are made possible by this graph-based method, which is essential for deep neural network training.

Numerous different layer types, activation functions, loss functions, and optimization techniques are just a few of the many functionalities supported by TensorFlow. Additionally, it provides strong capabilities for model evaluation, visualization, and data manipulation and preparation. TensorFlow offers GPU and CPU acceleration, enabling users to take advantage of the GPUs' computing ability to train models more quickly. TensorFlow offers high-level APIs like Keras in addition to its fundamental features, making it easier to create and train deep learning models. Deploying TensorFlow on several platforms is also supported, including computers, servers, mobile phones, and even edge devices.

In the domains of deep learning and machine learning, TensorFlow has mostly established itself as the standard framework. Its adaptability, scalability, and wide range of capabilities make it an effective tool for solving a variety of machine learning tasks, from straightforward regression and classification issues to challenging computer vision and natural language processing issues.

3.2 Keras

Developed in Python, Keras is a high-level neural network API that may be used with Theano, TensorFlow, or other popular deep learning frameworks. It offers a simple interface for creating and refining deep learning models. Keras makes it easier to define layers, establish connections, and configure model parameters, enabling developers to quickly prototype and test out various designs. Keras is appropriate for a variety of applications since it provides a huge selection of pre-built layers, activation functions, loss functions, and optimization techniques. In the deep learning field, Keras is a well-liked option for both newcomers and seasoned practitioners due to its simplicity and usability.

3.3 Softmax Function

In neural networks, the softmax function is a popular activation function, especially in multiclass classification issues. It normalizes an input vector of real values into a probability distribution across many classes.

The mathematical definition of the softmax function is as follows:

softmax(x_i) = exp(x_i) / sum(exp(x_j)) for all j

where exp stands for the exponential function and x_i is the input element at index i. Simply expressed, the softmax function multiplies each input value by the sum of the multiplied values for all classes before dividing it. This transformation makes sure that the probabilities are represented by values that are between 0 and 1 and add up to 1.

The softmax function is useful in multiclass classification tasks because it assigns higher probabilities to the class with larger input values while suppressing the probabilities of other classes. This allows the network to make confident predictions by selecting the class with the highest probability. During the training process, the softmax function is often combined with a suitable loss function, such as categorical cross-entropy, to measure the discrepancy between predicted probabilities and true labels. This helps guide the model's learning by optimizing the network's weights and biases to minimize the loss.

In summary, the softmax function is a critical component of neural networks for multiclass classification problems. It transforms input values into a probability distribution, enabling the model to assign probabilities to multiple classes and make confident predictions based on the highest probability.

CHAPTER 4

4 RESEARCH METHODOLOGY

Research methodology refers to the systematic approach and techniques used in conducting research. It involves the selection and application of appropriate methods, tools, and procedures to gather and analyze data in order to answer research questions or test hypotheses. Research methodology encompasses various aspects such as research design, data collection methods, sampling techniques, data analysis, and interpretation of results. We use VGG16, Resnet50 and InceptionV3 for detect diseases. The choice of research methodology depends on the nature of the research problem, the research objectives, and the available resources.

4.1 DATA COLLECTION

We did not find any suitable dataset from online. For this, we collect data from the different gardens and fruits shop. We collect three types of diseases of guava and also collect normal data. The data class are given below:

1. Normal Dataset.
2. Phytopthora Dataset.
3. Scab Dataset.
4. Stylar end Rot.

Figure 2 Normal Guava Dataset

Figure 3 Phytophthora Diseases Dataset

Figure 4 Scab Diseases Dataset

Figure 5 Stylar end Rot Diseases Dataset

4.2 DATA PREPROCESSING

After collecting data from fields, then we preprocess the data and also visualize it. We visualize all four folders of the dataset, With three types of diseases and one normal dataset.

Figure 6 Visualize Dataset

4.3 VGG16

The VGG16 model is a deep convolutional neural network (CNN) architecture that has been widely used in computer vision tasks, including image classification and object detection. We use this mode because, In the context of fruit disease detection, the VGG16 model can be used as a powerful tool for accurate and reliable identification of diseased fruits. Here are a few reasons why VGG16 is commonly employed in fruit disease detection:

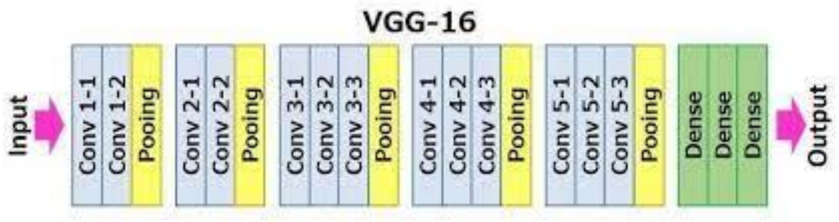


Figure 7 VGG16 CNN Model

Deep feature extraction: The VGG16 model has a deep architecture with multiple convolutional layers. These layers are capable of automatically extracting meaningful features from images, including textures, shapes, and patterns. This is crucial in distinguishing between healthy and diseased fruit images, as diseases often manifest as distinct visual abnormalities.

Pre-trained Model: VGG16 is frequently used as a pre-trained model, meaning it has already undergone extensive training on a large dataset (such as ImageNet) to recognize a variety of objects and patterns. The VGG16 can collect general features that are helpful for a variety of picture classification tasks, including fruit disease diagnosis, by utilizing pre-trained models. When there is a limited amount of training dataset available, this is especially advantageous.

Transfer Learning: Building on the previous point, the pre-trained VGG16 model can be fine-tuned on a specific fruit disease dataset. By fine-tuning, we adjust the parameters of the pre-trained model to better adapt it to the target task. This approach allows us to take advantage of the knowledge and representations learned by the VGG16 on a large dataset and apply it to the specific problem of fruit disease detection.

High performance: VGG16 has demonstrated impressive performance in various image classification benchmarks, achieving high accuracy rates. Its deep architecture and ability to capture intricate details contribute to its effectiveness in detecting subtle disease symptoms in fruit images.

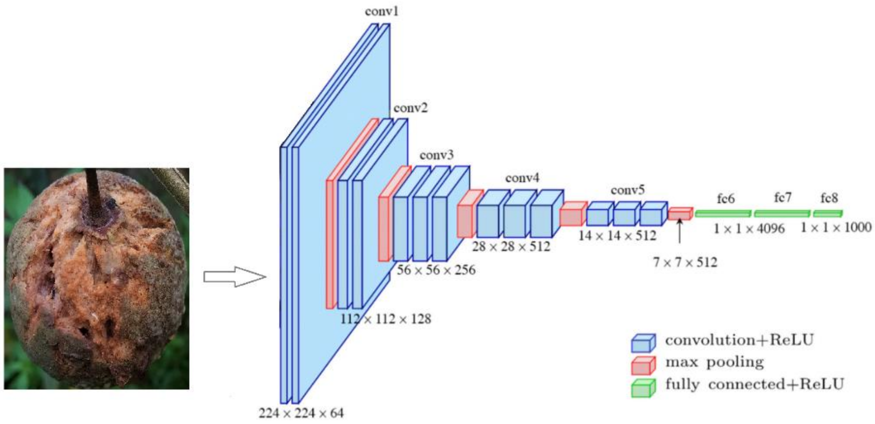


Figure 8 VGG16 Architecture

The 13 convolutional layers that make up the heart of VGG16 are each followed by a rectified linear unit (ReLU) activation function. These layers take minute details and patterns, such edges, textures, and forms, from the input photos. After each pair of convolutional layers, max pooling layers are used to downsample the feature maps, which lowers computational complexity while preserving crucial spatial data. Then, fully connected layers, often referred to as dense layers, are transferred over the flattened feature maps. Based on the features that were collected, these layers learn high-level representations and provide predictions. Softmax is the last layer to be fully connected, and it creates a probability distribution across several classes.

4.4 RESNET50

We also use Resnet50 in our research, it is another deep convolutional neural network (CNN) architecture commonly used in computer vision tasks, including fruit disease detection. Here are some reasons why ResNet50 is beneficial in this context:

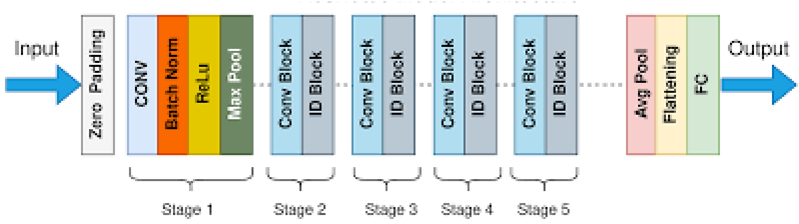


Figure 9 Resnet50 Model

Addressing the vanishing gradient problem: Deep neural networks often encounter the issue of vanishing gradients, where the gradients become extremely small as they propagate backward through the layers during training. This can hinder the learning process. ResNet50 introduced the concept of residual connections, which allow the network to learn residual mappings. By adding skip connections that bypass certain layers, ResNet50 can effectively address the vanishing gradient problem and facilitate the training of deeper networks.

Deeper architecture: ResNet50 is a deeper network compared to VGG16. It consists of 50 layers, including residual blocks with skip connections. This increased depth allows ResNet50 to capture more complex and abstract features from images, which can be crucial for accurately identifying subtle disease symptoms in fruit images.

Transfer learning capabilities: Similar to VGG16, ResNet50 is often used as a pre-trained model that has been trained on large-scale datasets like ImageNet. By leveraging transfer learning, the pre-trained ResNet50 model can be fine-tuned on a smaller dataset of fruit images, enabling it to learn specific patterns and features related to fruit diseases. This approach is particularly useful when the available dataset for training is limited.

Performance and accuracy: ResNet50 has shown impressive performance in various computer vision benchmarks, including image classification tasks. Its deeper architecture and residual connections allow for improved learning and more accurate predictions. This makes ResNet50 a suitable choice for fruit disease detection, where precise identification and classification of diseases are crucial for effective disease management and crop yield optimization.

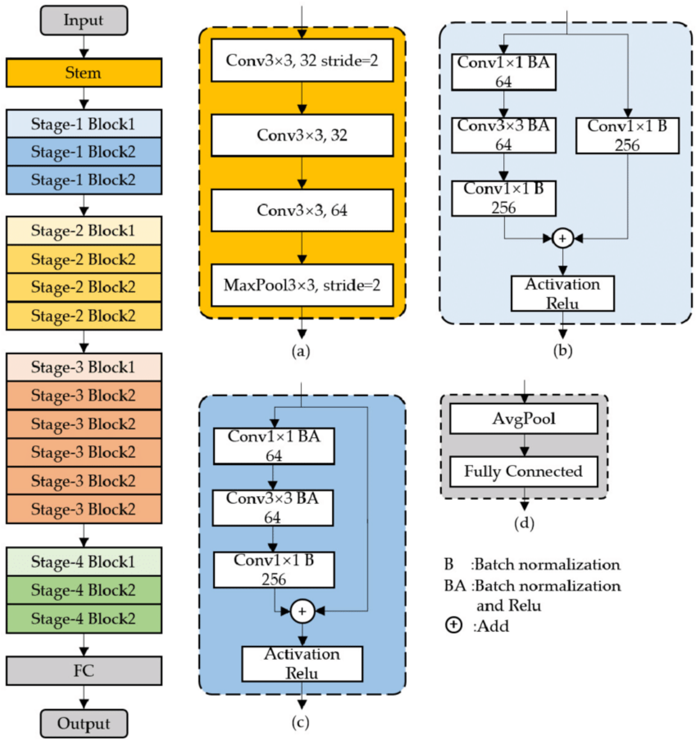


Figure 10 Resnet50 Architecture

In summary, ResNet50's ability to address the vanishing gradient problem, its deeper architecture, transfer learning capabilities, and high-performance metrics make it a valuable tool in fruit disease detection tasks. Its use enables the accurate identification of diseases and aids in developing effective solutions for crop protection and management.

4.5 INCEPTION V3

The last model we use that is InceptionV3, is also a deep convolutional neural network (CNN) architecture that has been widely used in fruit disease detection. The main reason we use this mode is given below:

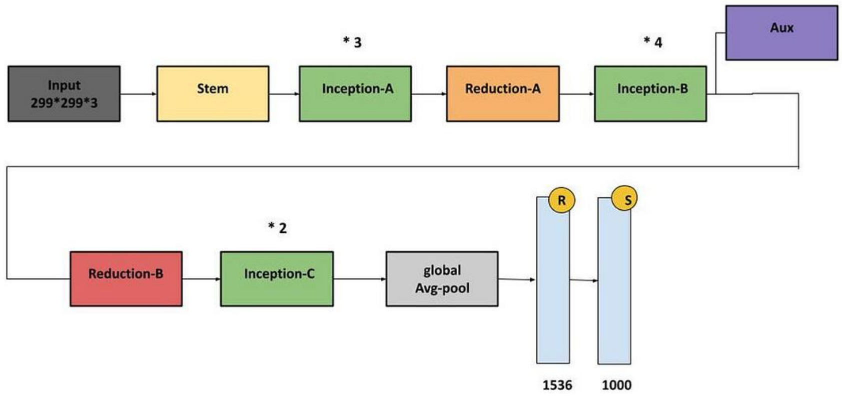


Figure 11 Inception V3 Model

Computational efficiency: InceptionV3 is designed to be computationally efficient while maintaining high accuracy. It achieves this by using multiple parallel convolutional layers with different filter sizes (1x1, 3x3, 5x5) to capture features at multiple scales. This architecture allows for effective feature extraction while minimizing the number of parameters, resulting in faster computations.

Multi-level feature extraction: InceptionV3 uses a technique called "Inception module" that performs multi-level feature extraction. This module consists of convolutional layers with different filter sizes concatenated together. By capturing features at various scales simultaneously, InceptionV3 can detect both fine-grained details and high-level semantic information, which is crucial for accurately identifying diseases in fruit images.

Performance and accuracy: InceptionV3 has demonstrated excellent performance in image classification tasks, achieving high accuracy rates. Its ability to capture multi-scale features and leverage transfer learning makes it suitable for fruit disease detection, where precise identification and classification of diseases are essential for effective crop management.

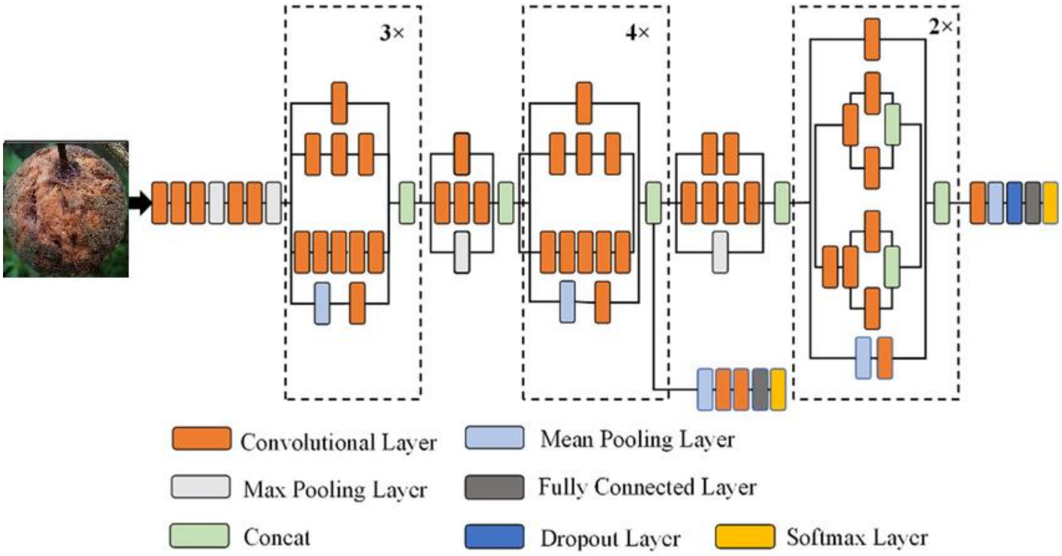


Figure 12 Inception V3 Architecture

Overall, InceptionV3's computational efficiency, multi-level feature extraction, transfer learning capabilities, and strong performance metrics make it a valuable architecture for fruit disease detection. Its use can facilitate accurate disease identification, help in implementing timely interventions, and support sustainable fruit cultivation practices.

4.6 EVALUATION METHODS

4.6.1 Accuracy

The accuracy of a machine's outcome prediction depends on how good the model is. When each class is equally important, something important has occurred. Every class is crucial to our area of work. As a result, precision is essential in determining the model's accuracy.

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{False Positives} + \text{False Negatives} + \text{True Negatives}}$$

Figure 13 Accuracy Formula

4.6.2 Precision

Precision provides information about the model's ability to avoid false positives, which means correctly identifying positive instances and minimizing the rate of falsely predicted positive cases.

The formula for precision is:

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Positive(FP)}$$

Figure 14 Precision Formula

Where:

TP (True Positives) is the number of correctly predicted positive instances. FP (False Positives) is the number of negative instances that were incorrectly predicted as positive.

4.6.3 Recall

In the context of binary classification, recall is a performance metric that measures the ability of a model to correctly identify positive instances (true positives) out of all actual positive instances (true positives + false negatives). It is also known as sensitivity or true positive rate (TPR).

The formula for the recall is:

$$Recall = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Negative(FN)}$$

Figure 15 Recall Formula

Where:

TP (True Positives) is the number of correctly predicted positive instances. FN (False Negatives) is the number of positive instances that were incorrectly predicted as negative. Recall provides information about the model's ability to avoid false negatives, which means correctly identifying positive instances and minimizing the rate of missed positive cases.

4.6.4 F1 Score

The F1 score is a single metric that combines both precision and recall to provide an overall evaluation of a model's performance in binary classification tasks. It is particularly useful when the dataset is imbalanced, meaning the number of positive and negative instances differs significantly.

The F1 score is calculated using the following formula:

$$\text{F1 Score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Figure 16 F1 Score Formula

The harmonic mean of both precision and recall is taken into account by the F1 score. It offers a fair evaluation of a model's capacity for accurate positive instance detection (precision) and avoiding false negatives (recall).

The F1 score has a range of 0 to 1, with 0 denoting subpar performance and 1 denoting flawless precision and recall. A higher F1 score indicates that the model has successfully balanced recall and precision.

4.6.5 Mean Average Precision(mAP)

Mean Average Precision (mAP) is a widely used evaluation metric in object detection and information retrieval tasks. It is used to assess the accuracy and performance of models in scenarios where multiple classes or categories are present. In the context of object detection, mAP measures the precision of the model at different recall levels across multiple classes. It considers both the accuracy of object localization (bounding box prediction) and the correct classification of the detected objects.

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

$$AP_k = \text{ the AP of class } k$$

$$n = \text{ the number of classes}$$

Figure 17 Mean Average Precision(mAP) Formula

4.6.6 Confusion Matrix

A confusion matrix is a table that summarizes the performance of a classification model by showing the counts of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. It provides insights into the model's ability to correctly classify instances and helps in evaluating its performance across different classes.

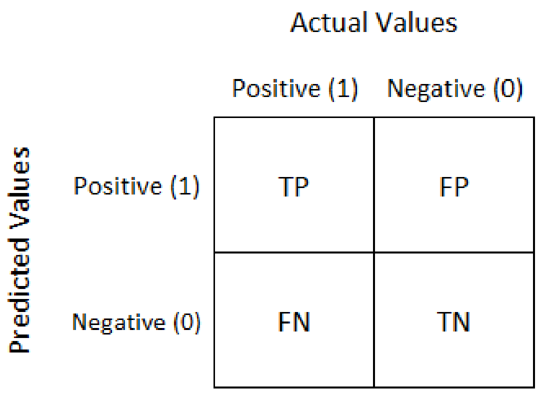


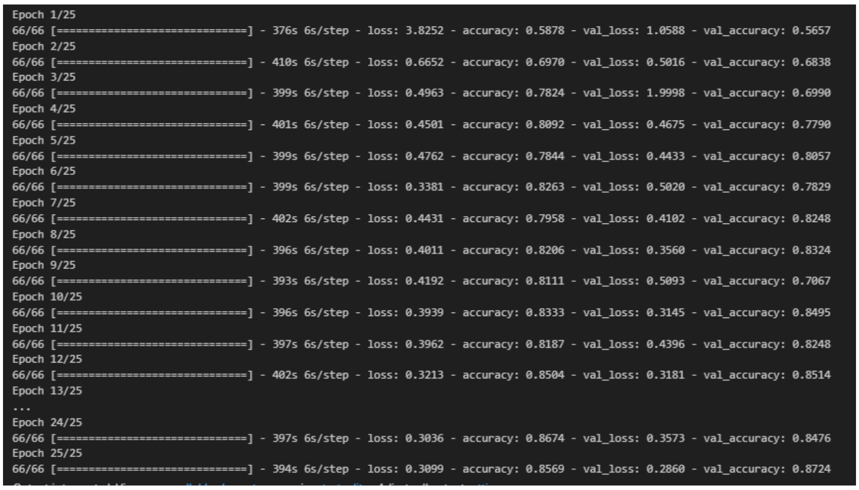
Figure 18 Confusion Matrix Diagram

CHAPTER 5

5 RESULTS AND DISCUSSION

After 25 epochs in **VGG16** we get 87% accuracy. We use the training and test data for the model

The result is given below:



```
Epoch 1/25
66/66 [==============================] - 376s 6s/step - loss: 3.8252 - accuracy: 0.5878 - val_loss: 1.0588 - val_accuracy: 0.5657
Epoch 2/25
66/66 [==============================] - 410s 6s/step - loss: 0.6652 - accuracy: 0.6970 - val_loss: 0.5016 - val_accuracy: 0.6838
Epoch 3/25
66/66 [==============================] - 399s 6s/step - loss: 0.4963 - accuracy: 0.7824 - val_loss: 1.9998 - val_accuracy: 0.6990
Epoch 4/25
66/66 [==============================] - 401s 6s/step - loss: 0.4501 - accuracy: 0.8092 - val_loss: 0.4675 - val_accuracy: 0.7790
Epoch 5/25
66/66 [==============================] - 399s 6s/step - loss: 0.4762 - accuracy: 0.7844 - val_loss: 0.4433 - val_accuracy: 0.8057
Epoch 6/25
66/66 [==============================] - 399s 6s/step - loss: 0.3381 - accuracy: 0.8263 - val_loss: 0.5020 - val_accuracy: 0.7829
Epoch 7/25
66/66 [==============================] - 402s 6s/step - loss: 0.4431 - accuracy: 0.7958 - val_loss: 0.4102 - val_accuracy: 0.8248
Epoch 8/25
66/66 [==============================] - 396s 6s/step - loss: 0.4011 - accuracy: 0.8206 - val_loss: 0.3560 - val_accuracy: 0.8324
Epoch 9/25
66/66 [==============================] - 393s 6s/step - loss: 0.4192 - accuracy: 0.8111 - val_loss: 0.5093 - val_accuracy: 0.7067
Epoch 10/25
66/66 [==============================] - 396s 6s/step - loss: 0.3939 - accuracy: 0.8333 - val_loss: 0.3145 - val_accuracy: 0.8495
Epoch 11/25
66/66 [==============================] - 397s 6s/step - loss: 0.3962 - accuracy: 0.8187 - val_loss: 0.4396 - val_accuracy: 0.8248
Epoch 12/25
66/66 [==============================] - 402s 6s/step - loss: 0.3213 - accuracy: 0.8504 - val_loss: 0.3181 - val_accuracy: 0.8514
Epoch 13/25
...
Epoch 24/25
66/66 [==============================] - 397s 6s/step - loss: 0.3036 - accuracy: 0.8674 - val_loss: 0.3573 - val_accuracy: 0.8476
Epoch 25/25
66/66 [==============================] - 394s 6s/step - loss: 0.3099 - accuracy: 0.8569 - val_loss: 0.2860 - val_accuracy: 0.8724
```

Figure 19 VGG-16 Epoch Result

In **Resnet50** we continue with only 20 epochs and we get almost 85% accuracy. Here we use the same dataset with the same folder testing and training data.

The result is given below:

```
Epoch 1/20
26/26 [==============================] - 83s 3s/step - loss: 0.9993 - accuracy: 0.5746 - val_loss: 0.4551 - val_accuracy: 0.6571
Epoch 2/20
26/26 [==============================] - 76s 3s/step - loss: 0.3574 - accuracy: 0.6928 - val_loss: 0.2906 - val_accuracy: 0.7010
Epoch 3/20
26/26 [==============================] - 77s 3s/step - loss: 0.2933 - accuracy: 0.7189 - val_loss: 0.2313 - val_accuracy: 0.7752
Epoch 4/20
26/26 [==============================] - 77s 3s/step - loss: 0.2749 - accuracy: 0.7699 - val_loss: 0.3699 - val_accuracy: 0.7810
Epoch 5/20
26/26 [==============================] - 80s 3s/step - loss: 0.2992 - accuracy: 0.7823 - val_loss: 0.2714 - val_accuracy: 0.7295
Epoch 6/20
26/26 [==============================] - 83s 3s/step - loss: 0.2715 - accuracy: 0.7649 - val_loss: 0.2000 - val_accuracy: 0.7943
Epoch 7/20
26/26 [==============================] - 83s 3s/step - loss: 0.2417 - accuracy: 0.7861 - val_loss: 0.1989 - val_accuracy: 0.7962
Epoch 8/20
26/26 [==============================] - 83s 3s/step - loss: 0.2564 - accuracy: 0.7836 - val_loss: 0.2244 - val_accuracy: 0.7714
Epoch 9/20
26/26 [==============================] - 82s 3s/step - loss: 0.2188 - accuracy: 0.8147 - val_loss: 0.2061 - val_accuracy: 0.8076
Epoch 10/20
26/26 [==============================] - 83s 3s/step - loss: 0.3359 - accuracy: 0.8010 - val_loss: 0.2313 - val_accuracy: 0.8248
Epoch 11/20
26/26 [==============================] - 82s 3s/step - loss: 0.3047 - accuracy: 0.7724 - val_loss: 0.1956 - val_accuracy: 0.8286
Epoch 12/20
26/26 [==============================] - 83s 3s/step - loss: 0.2248 - accuracy: 0.8047 - val_loss: 0.2208 - val_accuracy: 0.7752
Epoch 13/20
...
Epoch 19/20
26/26 [==============================] - 83s 3s/step - loss: 0.2177 - accuracy: 0.8408 - val_loss: 0.2915 - val_accuracy: 0.7733
Epoch 20/20
26/26 [==============================] - 83s 3s/step - loss: 0.2380 - accuracy: 0.8408 - val_loss: 0.1647 - val_accuracy: 0.8571
```

Figure 20 Resnet50 Epoch Result

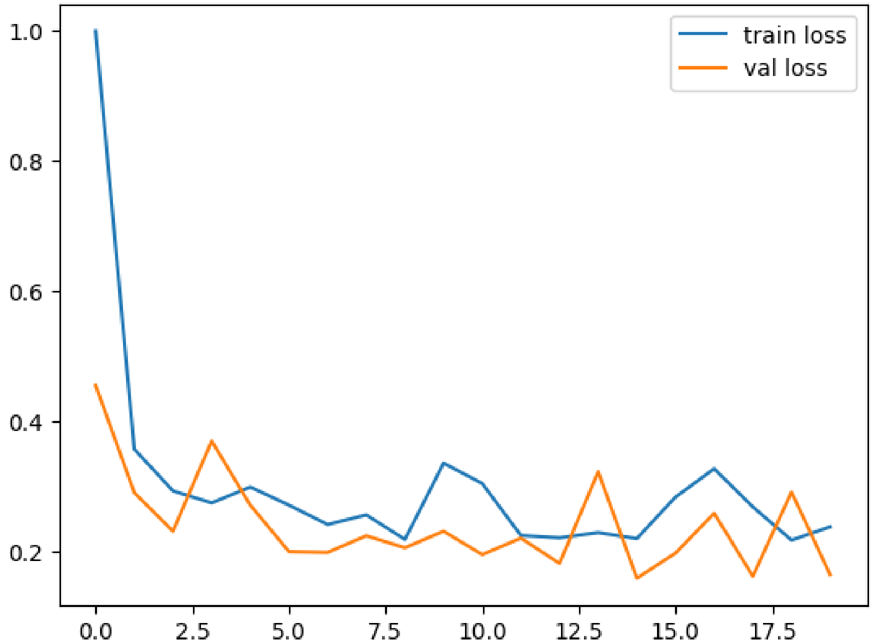


Figure 21 Resnet50 Train and Val Loss

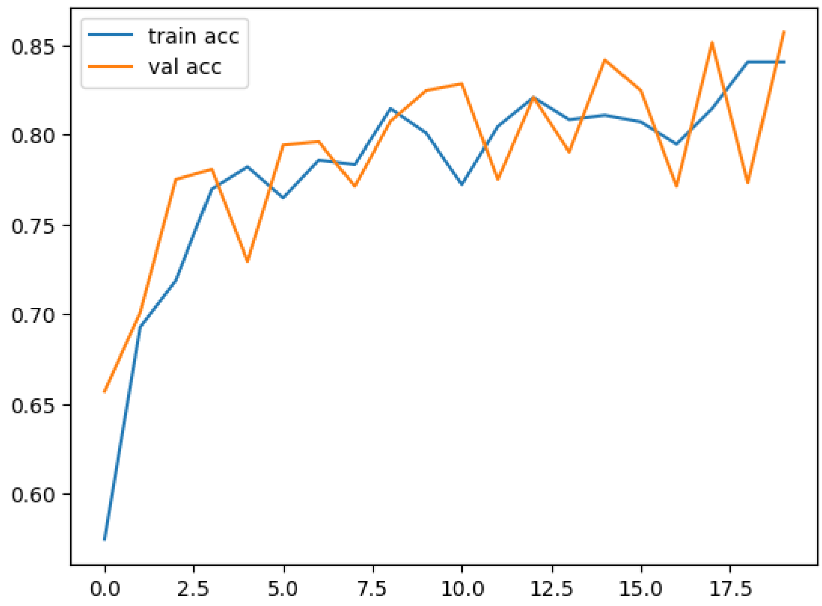


Figure 22 Resnet50 Train and Val acc

Then our last model is **InceptionV3**, we start our model with 10 epochs and we get the highest result so far which is 94% accuracy. We use the same data and folder for this model.

The result is given below:

```
Epoch 1/10
26/26 [==============================] - 46s 2s/step - loss: 0.7129 - accuracy: 0.9154 - val_loss: 0.4652 - val_accuracy: 0.9371
Epoch 2/10
26/26 [==============================] - 46s 2s/step - loss: 0.5806 - accuracy: 0.9341 - val_loss: 0.5718 - val_accuracy: 0.9314
Epoch 3/10
26/26 [==============================] - 46s 2s/step - loss: 0.6749 - accuracy: 0.9266 - val_loss: 0.4339 - val_accuracy: 0.9486
Epoch 4/10
26/26 [==============================] - 46s 2s/step - loss: 0.5720 - accuracy: 0.9440 - val_loss: 0.2331 - val_accuracy: 0.9562
Epoch 5/10
26/26 [==============================] - 46s 2s/step - loss: 0.4665 - accuracy: 0.9403 - val_loss: 0.2653 - val_accuracy: 0.9600
Epoch 6/10
26/26 [==============================] - 46s 2s/step - loss: 0.1930 - accuracy: 0.9739 - val_loss: 0.0754 - val_accuracy: 0.9810
Epoch 7/10
26/26 [==============================] - 46s 2s/step - loss: 0.3295 - accuracy: 0.9664 - val_loss: 0.1004 - val_accuracy: 0.9790
Epoch 8/10
26/26 [==============================] - 51s 2s/step - loss: 0.3286 - accuracy: 0.9565 - val_loss: 0.5253 - val_accuracy: 0.9352
Epoch 9/10
26/26 [==============================] - 51s 2s/step - loss: 0.6592 - accuracy: 0.9415 - val_loss: 0.1460 - val_accuracy: 0.9733
Epoch 10/10
26/26 [==============================] - 51s 2s/step - loss: 0.7410 - accuracy: 0.9366 - val_loss: 0.5864 - val_accuracy: 0.9486
```

Figure 23 InceptionV3 Epoch Result

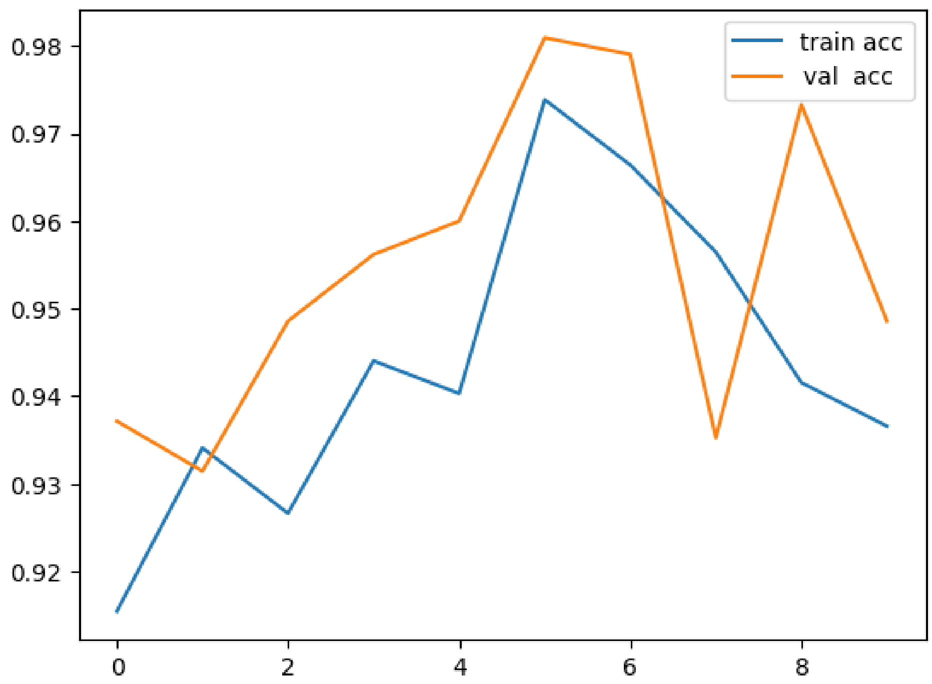


Figure 24 InceptionV3 Train and Val acc

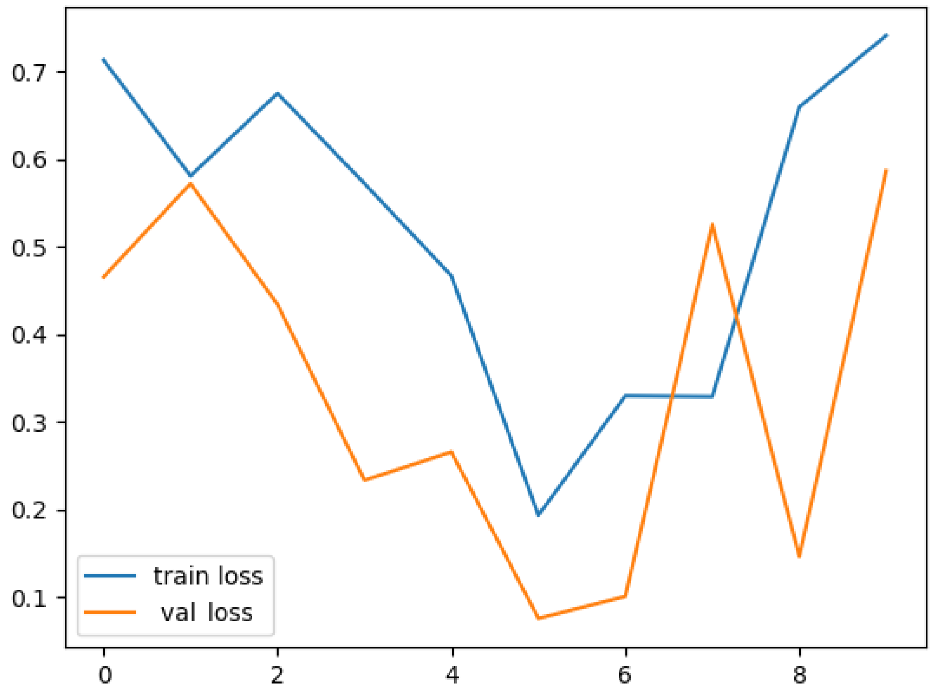


Figure 25 InceptionV3 Train and Val Loss

Detect Diseases by Rsnet50:

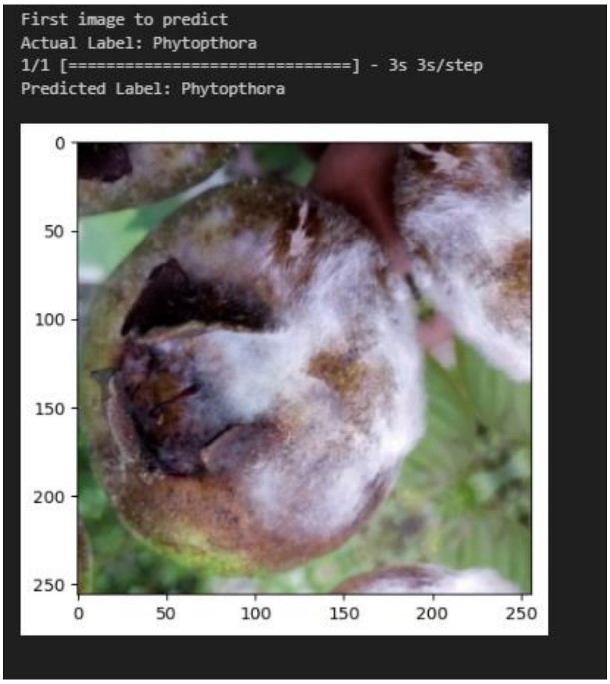


Figure 26 Resnet50 Diseases detection-01

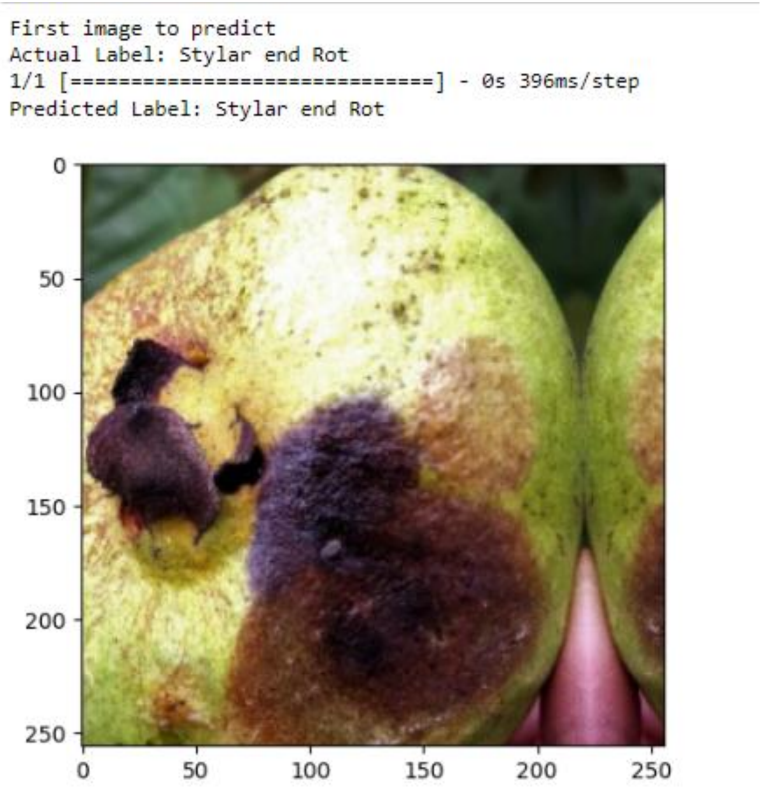


Figure 27 Resnet50 Diseases detection-02

Overall Result for Test Data:

Classification Report:

	precision	recall	f1-score	support
Phytopthora	0.73	0.85	0.79	13
Scab	0.60	0.82	0.69	11
Stylar end Rot	0.77	0.53	0.62	19
normal	1.00	1.00	1.00	75
accuracy			0.89	118
macro avg	0.78	0.80	0.78	118
weighted avg	0.90	0.89	0.89	118

Figure 28 Overall Result-test data

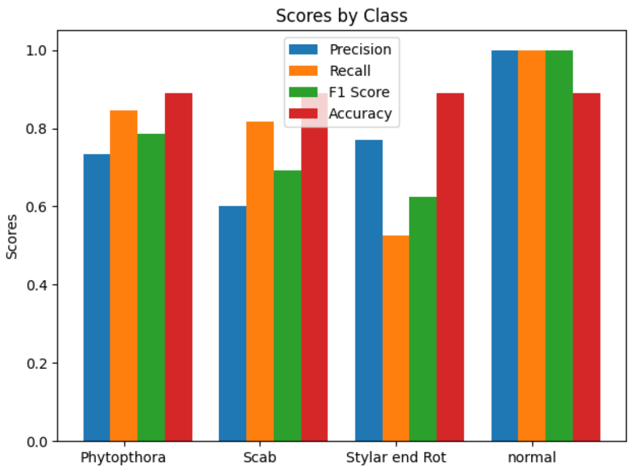


Figure 29 Overall Result Plot-test data

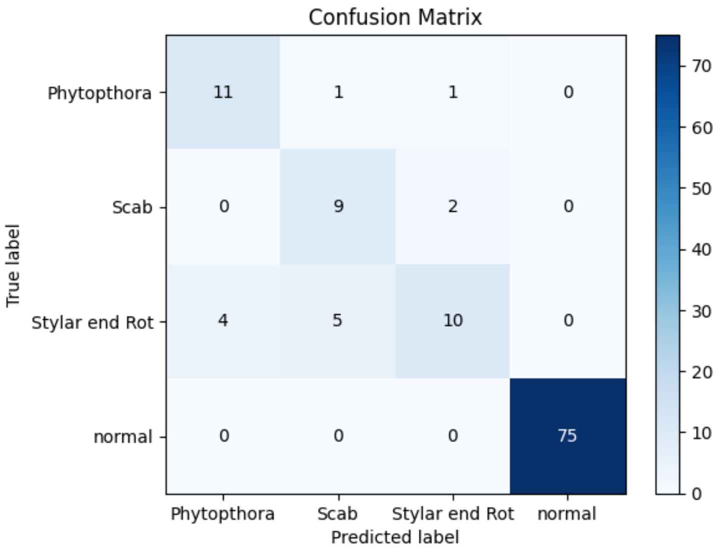


Figure 30 Confusion Matrix

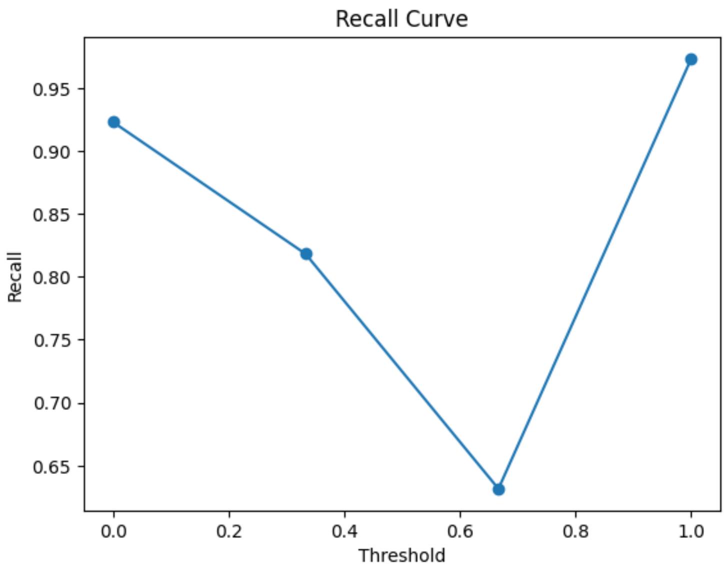


Figure 31 Recall Curve-test data

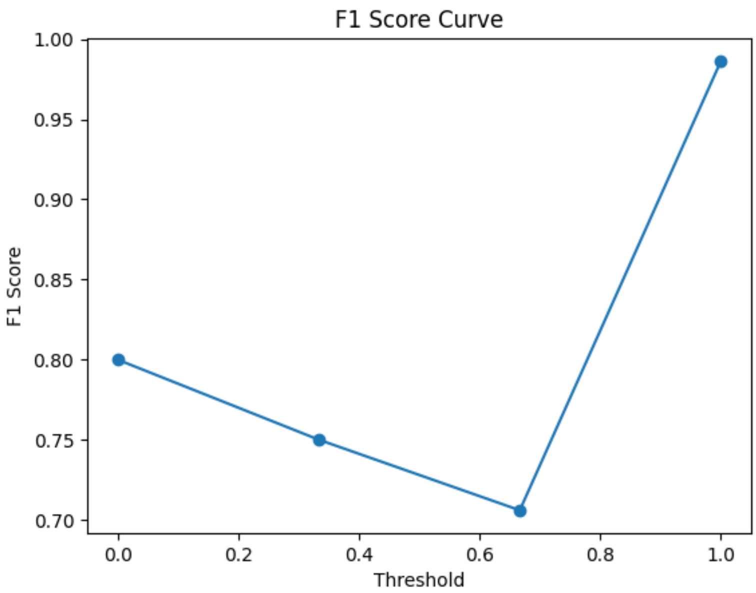


Figure 32 F1 Score Curve-test data

Best Model:

I use three model and the best accuracy I get that is 94% in InceptionV3.

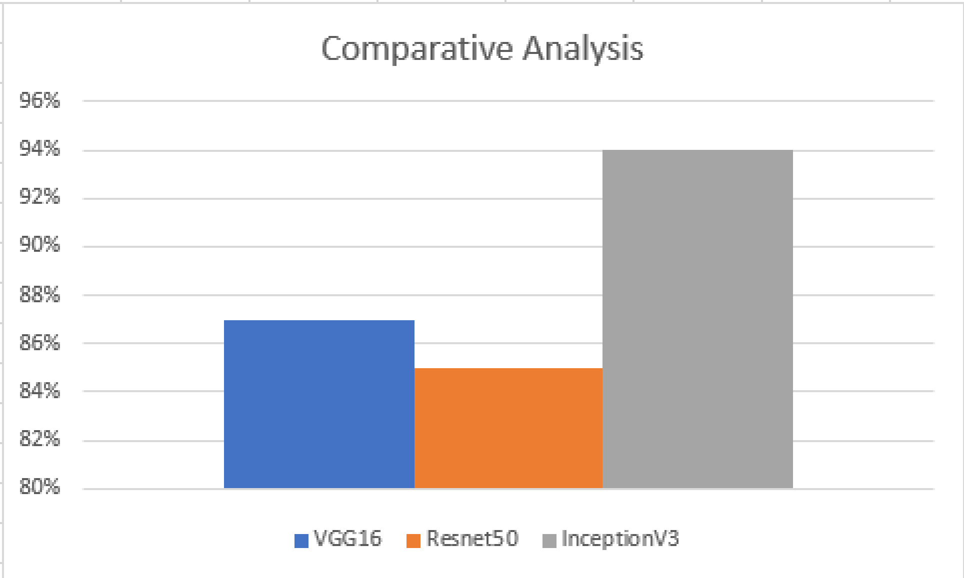


Figure 33 Best Model

CHAPTER 6

6 REFERENCES

[1] M. Sharif, M. A. Khan, Z. Iqbal, M. F. Azam, M. I. U. Lali, and M. Y. Javed, ‘‘Detection and classification of citrus diseases in agriculture based on optimized weighted segmentation and feature selection,’’ Comput. Electron. Agricult., vol. 150, pp. 220–234, Jul. 2018.

[2] Z. Lin, S. Mu, F. Huang, K. A. Mateen, M. Wang, W. Gao, and J. Jia, ‘‘A unified matrix-based convolutional neural network for finegrained image classification of wheat leaf diseases,’’ IEEE Access, vol. 7, pp. 11570–11590, 2019.

[3] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, ‘‘Identification of rice diseases using deep convolutional neural networks,’’ Neurocomputing, vol. 267, pp. 378–384, Dec. 2017.

[4] B. Richey, S. Majumder, M. V. Shirvaikar, and N. Kehtarnavaz, ‘‘Realtime detection of maize crop disease via a deep learning-based smartphone app,’’ Proc. SPIE, vol. 11401, Apr. 2020, Art. no. 114010A.

[5] S. Mishra, R. Sachan, and D. Rajpal, ‘‘Deep convolutional neural network based detection system for real-time corn plant disease recognition,’’ Procedia Comput. Sci., vol. 167, pp. 2003–2010, 2020.

[6] K. Golhani, S. K. Balasundram, G. Vadamalai, and B. Pradhan, ‘‘A review of neural networks in plant disease detection using hyperspectral data,’’ Inf. Process. Agricult., vol. 5, no. 3, pp. 354–371, Sep. 2018.

[7] A. R. Luaibi, T. M. Salman, and A. H. Miry, ‘‘Detection of citrus leaf diseases using a deep learning technique,’’ Int. J. Electr. Comput. Eng. (IJECE), vol. 11, no. 2, p. 1719, Apr. 2021.

[8] M. Z. Asghar, F. Subhan, M. Imran, F. M. Kundi, S. Shamshirband, A. Mosavi, P. Csiba, and A. R. Varkonyi-Koczy, ‘‘Performance evaluation of supervised machine learning techniques for efficient detection of emotions from online content,’’ 2019, arXiv:1908.01587. [Online]. Available: https://arxiv.org/abs/1908.01587

[9] Y. Wang, F. Subhan, S. Shamshirband, M. Zubair Asghar, I. Ullah, and A. Habib, ‘‘Fuzzy-based sentiment analysis system for analyzing Student feedback and satisfaction,’’ Comput., Mater. Continua, vol. 62, no. 2, pp. 631–655, 2020.

[10] M. Ali, M. Z. Asghar, and A. Baloch, ‘‘An efficient approach for subimage separation from large-scale multi-panel images using dynamic programming,’’ Multimedia Tools Appl., vol. 80, no. 4, pp. 5449–5471, Feb. 2021.

[11] Research on "Detection Technology of various Fruit Disease Spots Based on Mask R-CNN”, Tianjin University of Technology, Tianjin China, 2016 [12] L. Yang, Q. Song, and Y. Wu, ‘‘Attacks on state-of-the-art face recognition using attentional adversarial attack generative network,’’ Multimedia Tools Appl., vol. 80, no. 1, pp. 855–875, Jan. 2021.

[13] J. Fang, B. Qu, and Y. Yuan, ‘‘Distribution equalization learning mechanism for road crack detection,’’ Neurocomputing, vol. 424, pp. 193–204, Feb. 2021. [14] H. B. Yedder, B. Cardoen, and G. Hamarneh, ‘‘Deep learning for biomedical image reconstruction: A survey,’’ 2020, arXiv:2002.12351. [Online]. Available: http://arxiv.org/abs/2002.12351

[14] M. Ji, L. Zhang, and Q. Wu, ‘‘Automatic grape leaf diseases identification via unitedmodel based on multiple convolutional neural networks,’’ Inf. Process. Agricult., vol. 7, no. 3, pp. 418–426, Sep. 2020.

[15] C. B. Wetterich, R. Felipe de Oliveira Neves, J. Belasque, and L. G. Marcassa, ‘‘Detection of citrus canker and Huanglongbing using fluorescence imaging spectroscopy and support vector machine technique,’’ Appl. Opt., vol. 55, no. 2, pp. 400–407, 2016.

[16] Z. Iqbal, M. A. Khan, M. Sharif, J. H. Shah, M. H. ur Rehman, and K. Javed, ‘‘An automated detection and classification of citrus plant diseases using image processing techniques: A review,’’ Comput. Electron. Agricult., vol. 153, pp. 12–32, Oct. 2018.

[17] B. Doh, D. Zhang, Y. Shen, F. Hussain, R. F. Doh, and K. Ayepah, “Automatic citrus fruit disease detection by phenotyping using machine learning,” in 25th IEEE International Conference on Automation and Computing, 2019, pp. 1–5, doi: 10.23919/IConAC.2019.8895102.

[18] H. Patel, R. Prajapati, and M. Patel, “Detection of Quality in Orange Fruit Image using SVM Classifier,” in 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 74–78.

[19] S. Xing, M. Lee, and K.-K. Lee, ‘‘Citrus pests and diseases recognition model using weakly dense connected convolution network,’’ Sensors, vol. 19, no. 14, p. 3195, Jul. 2019.

[20] U. Barman, R. D. Choudhury, D. Sahu, and G. G. Barman, ‘‘Comparison of convolution neural networks for smartphone image based real time classification of citrus leaf disease,’’ Comput. Electron. Agricult., vol. 177, Oct. 2020, Art. no. 105661.

[21] M. Khanramaki, E. A. Asli-Ardeh, and E. Kozegar, ‘‘Citrus pests classification using an ensemble of deep learning models,’’ Comput. Electron. Agricult., vol. 186, Jul. 2021, Art. no. 106192, doi:10.1016/j.compag.2021.106192.

[22] V. Kukreja and P. Dhiman, ‘‘A deep neural network based disease detection scheme for citrus fruits,’’ in Proc. Int. Conf. Smart Electron. Commun (ICOSEC), Sep. 2020, pp. 97–101.

[23] V. Partel, L. Nunes, P. Stansly, and Y. Ampatzidis, ‘‘Automated visionbased system for monitoring Asian citrus psyllid in orchards utilizing artificial intelligence,’’ Comput. Electron. Agricult., vol. 162, pp. 328–336,Jul. 2019.

[24 H. T. Rauf, B. A. Saleem, M. I. U. Lali, M. A. Khan, M. Sharif, and S. A. C. Bukhari, ‘‘A citrus fruits and leaves dataset for detection and classification of citrus diseases through machine learning,’’ Data Brief,vol. 26, Oct. 2019, Art. no. 104340.

[25] D. P. Hughes and M. Salathe, ‘‘An open access repository of images on plant health to enable the development of mobile disease diagnostics,’’ 2015, arXiv:1511.08060. [Online]. Available:http://arxiv.org/abs/1511.08060