BACHELOR OF SCIENCE IN COMPUTER AND COMMUNICATION ENGINEERING

Enhancing Plant Disease Detection: A Deep Learning Approach and Comparative Analysis of Pretrained Models with Explainable AI Visualization.

Supervised by

Engr. Mohammad Nadib Hasan
Lecturer
Department of Computer and Communication Engineering
International Islamic University Chittagong

Co-Supervised By

Engr. Md. Jiabul Hoque
Assistant Professor
Department of Computer and Communication Engineering
International Islamic University Chittagong

Submitted By

Md. Zia Ul Hassan Chowdhury
Matric ID: E193009

Department of Computer and Communication Engineering
International Islamic University Chittagong
Kumira, Sitakunda, Chittagong – 4318

BOARD OF EXAMINERS

The thesis/project titled "**Enhancing Plant Disease Detection: A Deep Learning Approach and Comparative Analysis of Pretrained Models with Explainable AI Visualization.**" submitted by **MD ZIA UL HASSAN CHY(E193009)**, **Session: Spring 2023** has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Bachelor of Science in **Computer and Communication Engineering (CCE)** held on **23rd September 2024**.

Prof. Md. Razu Ahmed Professor Dept. of Computer and Communication Engineering International Islamic University Chittagong	Chairman
Dr. Mohammed Saifuddin Assistant Professor Dept. of Computer and Communication Engineering International Islamic University Chittagong	Convener & Internal Member
Engr. Md Jiabul Hoque Assistant Professor Dept. of Computer and Communication Engineering International Islamic University Chittagong	Internal Member
Engr. Mohammad Nadib Hasan Lecturer Dept. of Computer and Communication Engineering International Islamic University Chittagong	Supervisor
Prof. Dr. Mohammad Osiur Rahman Professor Dept. of Computer Science & Engineering University of Chittagong	External Member

Certificate of Approval

The thesis entitled "**Enhancing Plant Disease Detection: A Deep Learning Approach and Comparative Analysis of Pretrained Models with Explainable AI Visualization**" was submitted by **Md. Zia Ul Hassan Chowdhury** bearing ID No. **E193009**, to the **Department of Computer and Communication Engineering** of **International Islamic University Chittagong (IIUC),** has been accepted as satisfactory for the partial fulfillment of the requirements for the Degree of Bachelor in Computer and Communication Engineering and approved as to its style and contents for the examination held on **23rd September 2024**.

Approved By-

Engr. Mohammad Nadib Hasan

Supervisor

Lecturer

Department of Computer and Communication Engineering

International Islamic University Chittagong

Engr. Md. Jiabul Hoque

Co-Supervisor

Assistance Professor

Department of Computer and Communication Engineering

International Islamic University Chittagong

Declaration of Candidate

It is hereby declared that the work presented here in is genuine work done by me and has not concurrently submitted in candidature for any degree. The result of the thesis that have found totally depends on my own investigation/work. This work was done under the guidance of **Engr. Mohammad Nadib Hasan**, Lecturer of Computer and Communication Engineering (CCE), the International Islamic University Chittagong.

MD. Zia Ul Hassan Chowdhury

(Signature of Candidate)

Md. Zia Ul Hassan Chowdhury

Matric ID: E193009

Program: B. Sc. In CCE

Acknowledgments

The journey leading to the completion of this thesis has been one marked by immense dedication, invaluable support, and profound encouragement from numerous individuals. Their unwavering belief in my work, coupled with their selfless contributions of time, knowledge, and creativity, has made this accomplishment possible. I would like to express my deepest gratitude to all those who have played a significant role in bringing this study to fruition.

First and foremost, I wish to extend my heartfelt thanks to my esteemed thesis supervisor, **Engr. Mohammad Nadib Hasan**. His insightful guidance, continuous support, and steadfast commitment to excellence have been the foundation upon which this work was built. His expertise, coupled with his unyielding encouragement, served as a beacon, constantly motivating me to push the boundaries of my potential and aim for the highest academic standards. I also wish to express my sincere appreciation to the Department of Computer and Communication Engineering (CCE) at the International Islamic University Chittagong. The department's support has been a cornerstone in enabling me to pursue this study with confidence and clarity. To my dear friends, your companionship along this challenging path has been a source of solace and strength. The intellectually stimulating conversations, moments of shared laughter, and unwavering support you provided have been vital in maintaining my mental well-being throughout this journey. Your presence has been invaluable, and I am deeply grateful for the joy and encouragement you brought into my life. Lastly, I would like to extend my profound gratitude to all those who have contributed to my academic and personal growth in various ways, yet may not be named here. Your influence, though behind the scenes, has been integral to the development of this thesis and to my journey as a scholar.

Thank you all, from the depths of my heart, for your continued belief in me and for helping me reach this milestone.

With warm regards,

Md. Zia Ul Hassan Chowdhury

ID: E193009

Author

Abstract

The accurate detection of plant diseases is essential for safeguarding food security and promoting agricultural sustainability. This study presents an advanced deep learning approach using Convolutional Neural Networks (CNNs) for the classification of plant leaf diseases. Utilizing the Plant Village dataset, a custom CNN model was developed, which achieved superior accuracy of 96.11%, surpassing popular transfer learning models such as ResNet152-v2 (94%), VGG16 (91%), and Inception-v3 (88%). Key aspects of this work include the implementation of robust data preprocessing techniques like image augmentation and the design of an optimized CNN architecture to enhance performance while maintaining computational efficiency. Furthermore, explainability was explored using Layer CAM, which provided more detailed visual explanations compared to Grad-CAM, thereby increasing the model's transparency and trustworthiness. These results confirm the effectiveness of the proposed method for accurate plant disease identification, positioning it as a promising tool for agricultural disease diagnosis. Future efforts will focus on expanding the dataset, improving model generalization, and deploying the model for real-time field applications.

Keywords: Plant disease detection, CNN, Deep Learning, Transfer Learning, Layer CAM, Agriculture, Plant Village dataset, Grad-CAM, Explainable AI

Table Of Contents

List of Figure

List of Table

Chapter 1

Introduction

1.1 Background

In recent years, the application of Artificial Intelligence (AI) has revolutionized various fields by offering advanced solutions to complex problems. AI encompasses a range of techniques and technologies designed to enable machines to perform tasks that typically require human intelligence. Machine Learning (ML) and Deep Learning (DL) are two of the many subfields of artificial intelligence (AI) that have drawn a lot of attention because of their capacity to analyze massive datasets, find patterns, and make predictions. Applications for these technologies can be found in a variety of fields, such as agriculture, healthcare, and finance. Plant disease control and identification are essential to agriculture in order to maintain crop health and productivity. Plant diseases pose a threat to food security and can result in significant financial losses. Manual inspections are a common component of traditional plant disease detection techniques, although they are time- and labor-intensive. With the introduction of AI, disease detection procedures might be improved and automated, giving farmers and agricultural specialists access to timely and reliable information. [1].

1.2 Artificial Intelligence

Artificial Intelligence (AI) is the field of technology focused on creating systems capable of performing tasks that typically require human intelligence. This includes a wide range of skills, such as problem-solving, experience-based learning, and data-driven decision-making. AI technologies are categorized into three main types based on their scope and complexity. Narrow AI refers to systems designed for specific tasks, such as facial recognition or language translation, operating within a limited domain of functionality [10]. General AI, a more advanced and largely theoretical concept, envisions systems that possess the ability to understand, learn, and apply intelligence across a diverse range of tasks, akin to human cognitive abilities. Finally, super intelligent AI represents a hypothetical future state where artificial intelligence not only matches but exceeds human intelligence in all aspects, including creativity, problem-solving, and decision-making. These categories outline the progression from specialized, task-specific applications to broader, more ambitious goals in the realm of AI development.

1.3 Machine Learning

Machine Learning (ML) is a critical subject within Artificial Intelligence focusing on building systems and techniques that enable computers to learn from data and make judgments or predictions without explicit programming. Unlike traditional computer programs that execute commands written by a programmer, ML systems continuously improve their performance by analyzing data and adjusting their algorithms based on what they learn. This capability allows machines to perform complex tasks and solve problems by recognizing patterns and making informed decisions.ML encompasses three primary categories, each tailored to different types of data and problem-solving approaches. Supervised Learning is the most prevalent form of machine learning, where models are trained on datasets that include both input data and corresponding correct outputs. During training, the model learns to map inputs to outputs by comparing its predictions with the actual results and adjusting its parameters to enhance accuracy. Supervised learning includes two main tasks: classification and regression. In classification, the model categorizes data into predefined classes or labels, such as distinguishing between "spam" and "not spam" emails. In regression, the model predicts continuous values, like estimating house prices based on features like size and location. Unsupervised Learning deals with data that lacks predefined labels or outcomes, aiming to identify hidden patterns or structures within the data. This category includes techniques such as clustering, which groups data points based on their similarities to uncover natural divisions within the data, like identifying customer segments with similar purchasing behaviors. Another method, called dimensionality reduction, makes complicated datasets easier to view or handle by lowering the number of features while maintaining critical information. An example of this would be compressing image data for more effective storage. The process of teaching an agent to make decisions based on interactions with its surroundings is known as reinforcement learning. In this paradigm, the agent receives feedback in the form of rewards or penalties for its behaviors, with the goal of increasing cumulative rewards over time. Reinforcement Learning is employed in diverse applications such as game playing, robotics, and autonomous vehicles, where agents learn to perform tasks like navigating a maze or controlling a robot arm [11].

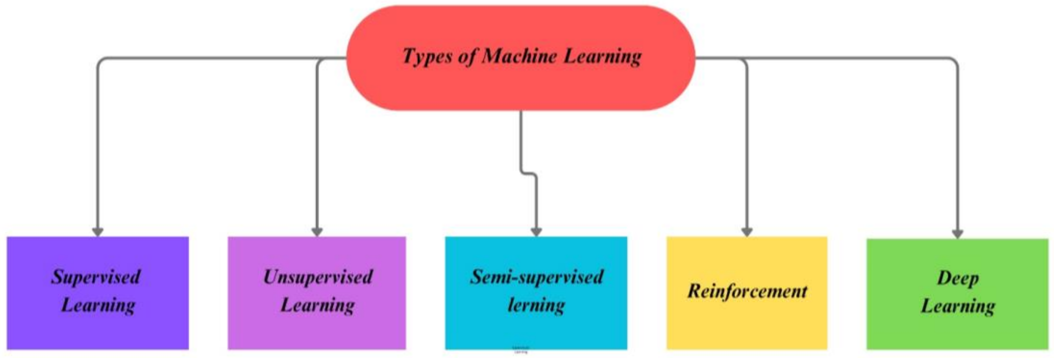


Figure 1.1: Types of machine learning

1.4 Deep Learning

Deep Learning represents a sophisticated subset of Machine Learning focused on leveraging multi-layered neural networks to tackle intricate data analysis tasks. Fundamentally, deep learning is the construction and training of neural networks. Networks made up of multiple layers of connected nodes, or neurons that mimic the way the human brain processes information. An input layer, many hidden layers, and an output layer make up these deep neural networks. Numerous nodes in each layer modify data using activation functions and weights that have been learned. Because Deep Learning can automatically learn and extract hierarchical characteristics from raw data, it is a powerful tool for a variety of challenging tasks, including speech and image recognition. [4].

In image classification, Convolutional Neural Networks (CNNs) are pivotal. CNNs apply convolutional layers that use filters to detect various features in images, such as edges, textures, and shapes, progressively combining these features to identify objects or patterns. This hierarchical approach allows CNNs to recognize complex structures and perform tasks like identifying objects in images or diagnosing diseases from medical scans. Recurrent neural networks, or RNNs, are an important part of deep learning since they are made to handle sequential input. With connections that create directed cycles, RNNs, in contrast to feedforward neural networks, are able to preserve a kind of memory and identify temporal relationships in input. For applications like speech recognition, language translation, and time series prediction, RNNs are especially well-suited. Through techniques like backpropagation and optimization algorithms, Deep Learning models are trained to minimize errors and improve predictions [12]. The ability of Deep Learning models to learn from vast amounts of data and their scalability make them invaluable for solving complex real-world problems across various domains, from healthcare to autonomous driving.

1.5 Crop Disease

A significant obstacle to agriculture is crop disease, which has an international influence on plant health and agricultural productivity. The Food and Agriculture Organization (FAO) estimates that 20–40% of global crop losses occur each year as a result of crop diseases. This significant loss translates into billions of dollars in economic damage each year. For instance, a 2020 survey by the International Food Policy Research Institute (IFPRI) estimated that plant diseases cause economic losses of around $220 billion annually, with over 50% of these losses affecting staple crops such as wheat, rice, and maize. The pathogens responsible for these diseases include fungi, bacteria, and viruses, each capable of inflicting severe damage to crops. The American Phytopathological Society (APS) reported that fungal diseases alone are responsible for about 60% of the global crop disease burden. Plant diseases must be identified and treated early to slow their spread and minimize financial damages. In order to greatly reduce the impact of various disorders, effective disease management strategies frequently include prompt intervention and the use of cutting-edge technologies for early detection. Thus, there is a pressing need for innovative solutions to enhance the efficiency and accuracy of plant disease detection [13].

1.6 Problem Statement

Crop diseases present a significant challenge to global food security and agricultural productivity, causing substantial economic losses each year by diminishing both the quality and quantity of crop yields. Traditional methods for diagnosing these diseases, such as visual inspections and expert evaluations, are often slow, labor-intensive, and require specialized knowledge, making them impractical for large-scale agricultural operations. These conventional approaches are not only inefficient but also suffer from delays in disease detection and management. To overcome these limitations, there is a pressing need for automated and efficient solutions that can provide timely and accurate disease diagnosis [3]. Deep learning methods, particularly Convolutional Neural

Networks (CNNs), have shown great promise in automating image-based classification tasks, offering the potential for faster and more reliable disease detection [5]. A customized CNN model is created in this study to overcome the shortcomings of conventional plant disease classification techniques. The proposed model aims to deliver a scalable, automated solution for classifying plant diseases, thereby enhancing classification accuracy, reducing reliance on expert knowledge, and providing a more efficient method for managing crop health.

1.7 Goal & Motivation

1.7.1 Goal

The primary objective of this research is to develop a deep learning model capable of accurately classifying plant leaf diseases using the Plant Village dataset. This study compares the performance of pre-trained models like VGG16 [16], Inception V3 [17], and ResNet152-v2 [15] with the custom Convolutional Neural Network (CNN) model. Several optimization and hyperparameter tuning strategies will be used in the analysis to determine the best configurations for maximizing model performance. The ultimate objective is to improve the precision and effectiveness of plant disease detection by offering a scalable, dependable solution for actual agricultural settings.

1.7.2 Motivation

Plant diseases pose a significant threat to global agriculture, leading to substantial economic losses and threatening food security. Traditional methods of disease detection, which rely on manual inspections and expert knowledge, are often slow, labor-intensive, and not scalable. There has never been a more critical time to revolutionize this field with advanced technological solutions. The convergence of agricultural expertise and artificial intelligence offers a beacon of hope amid these challenges [1]. Imagine a world where farmers and agricultural professionals can quickly and accurately diagnose plant diseases, leading to timely interventions and healthier crops. This vision drives the motivation behind this research. Convolutional Neural Networks (CNNs) have the potential to transform how we detect and classify plant diseases, offering significant advancements in accuracy and efficiency. This promise extends beyond mere precision; it symbolizes liberation from the uncertainties and inefficiencies that plague traditional methods. We are exploring new ground here by comparing the performance of several CNN architectures and fine-tuning their parameters. The aspiration is to achieve a harmonious blend of accuracy and speed, where advanced AI models replace manual analysis with superior diagnostic capabilities. This research is not just about technological innovation; it's about enhancing the resilience of agricultural systems and empowering those who depend on them. Our research is a testament to the relentless pursuit of improving agricultural disease management. It calls for a shift from traditional methods to innovative solutions, emphasizing that the fight against crop diseases is not merely about addressing challenges but also about discovering solutions. From image analysis to coding, every stage of our research contributes to a new story: one in which technology opens the door for a more productive and sustainable agricultural future.

1.8 Objective

The primary objective of our research is to develop an advanced deep-learning model to enhance the classification of plant leaf diseases using images from the Plant Village dataset. Plant diseases pose a significant threat to global agriculture, requiring timely and accurate diagnosis to mitigate their impact. This research aims to create a reliable diagnostic tool that surpasses traditional methods in accuracy and efficiency. To achieve this objective, our research is structured around several key steps. Firstly, we will preprocess and augment the Plant Village dataset to optimize it for machine learning applications. This preparation is crucial to ensure The quality and diversity of the dataset improved the performance of our models. Next, we will design and implement a custom Convolutional Neural Network (CNN) tailored for plant disease classification. This custom model will be built To efficiently capture the intricate patterns and features contained in plant leaf photos, thereby improving classification accuracy. In parallel, we will train and evaluate three pre-trained deep learning models: VGG16 [16], Inception V3 [17], and ResNet152-v2 [15]. These established architectures are known for their robustness and high performance in image classification tasks. By employing these models, we aim to set a benchmark for our custom CNN. A comprehensive comparison between the custom CNN model and the pre-trained models will be conducted to assess their performance. Important metrics including accuracy, precision, recall, and F1 score will be the main emphasis of this comparison, giving a thorough grasp of the advantages and disadvantages of each model. Our goal is to identify the most effective approach for plant disease classification, considering both accuracy and practical applicability.

Furthermore, we will fine-tune the hyperparameters of all models, exploring various configurations to achieve optimal performance. This tuning process will involve adjusting learning rates, batch sizes, and other critical parameters, ensuring that each model is operating at its best.

Through this research, we aim to make a significant contribution to agricultural disease management. An advanced deep learning model, refined through rigorous analysis and optimization, promises to improve diagnostic accuracy and support farmers in maintaining crop health. This research not only advances the subject of AI in agriculture, but it also lays the path for more efficient and scalable solutions to one of the most important issues in global food production.

Thesis Outlines

This thesis is organized into the following chapters:

- **Chapter 1:** Introduction - Provides the background, problem statement, goals, and structure of the thesis.
- **Chapter 2:** Literature Review - Reviews related work and current advancements in plant disease detection and deep learning.
- **Chapter 3:** Methodology - Details the dataset, preprocessing techniques, model development, and evaluation methods.
- **Chapter 4:** Model Implementation and Experimental Results - Presents and analyzes the results of the custom CNN model and pre-trained models.
- **Chapter 5:** Result Analysis, Evaluation, and Discussion - Discusses the findings, challenges, and future directions for the research.
- **Chapter 6:** Conclusion & Future Work - Summarizes the research outcomes and contributions.

Chapter 2

Literature Review

2.1 Crop Disease

Agriculture plays a crucial role in the modern globalized world, with its deep historical roots connecting human civilization to the land. In countries like Bangladesh, where agriculture is not only an economic activity but also a primary means of livelihood for many people, the importance of crop production is profound. Whether on small subsistence farms or expansive commercial agricultural operations, the production of crops remains a vital aspect of human existence and food security. However, crop diseases brought on by a variety of pathogens, such as bacteria, fungus, viruses, and other microbes, pose serious problems to the agricultural industry. [2]. These diseases can lead to severe reductions in crop yields, with losses ranging from 10% to 95%, which adversely affects both the quantity and quality of agricultural output. Crop productivity must be kept high in order to ensure food security, and these diseases must be effectively managed and identified early. Traditionally, the detection of crop diseases has relied on manual inspection methods. Farmers and agricultural experts would visually assess symptoms on plant leaves, stems, and roots to identify potential diseases. This approach, while foundational, is often limited by the subjective nature of visual inspections, which can be inaccurate and time-consuming. Furthermore, many farmers lack access to advanced agricultural technologies for disease detection and management, which can delay the identification of infections and exacerbate crop losses. To address these challenges, there is a growing need for modern technological solutions that can enhance the efficiency and accuracy of crop disease detection [3]. Convolutional Neural Networks (CNNs), one of the prominent deep learning algorithms, offer a promising advancement in this field. CNNs have demonstrated remarkable efficacy in deciphering intricate patterns seen in extensive datasets, such as high-definition images of plant diseases. These models allow for automated, precise, and fast disease diagnosis, making them a strong substitute for conventional techniques. The introduction of deep learning technologies has transformed the landscape of plant disease detection. Techniques such as CNNs utilize advanced algorithms to analyze digital images of plants, detecting subtle symptoms of diseases that may be missed by manual inspection. When compared to conventional methods, the use of CNNs in agriculture has shown notable improvements in the accuracy of disease diagnosis. [6]. Deep learning models have shown remarkable performance in identifying and classifying plant diseases through the analysis of detailed visual features. Despite these achievements, there remain difficulties to face. Existing deep learning models frequently specialize in specific diseases or plant species, which limits their general usefulness. To address these constraints, recent research has focused on increasing the efficiency and universality of illness detection technology using techniques such as transfer learning. Transfer learning leverages pre-trained models and adapts them to new tasks, thereby improving the performance of disease detection systems across diverse agricultural contexts [7]. Recent research shows how deep learning has significantly advanced the field of plant disease identification. For example, Chen et al. (2020) demonstrated the effectiveness of combining Inception modules with VGGNet for rice plant disease classification, achieving an impressive average accuracy of 92.00% despite complex backgrounds [21]. Sunil et al. (2022) proposed a low-cost method for early disease identification using a combination of AlexNet, ResNet50, and VGG16, which yielded high accuracy rates of up to 100% for binary classification tasks and 99.53% for multi-class datasets [22]. The work of Gayathri et al. (2020), who created the INAR-SSD model for the real-time diagnosis of apple leaf diseases, shows further developments. On a recently constructed apple leaf disease dataset, they achieved a mean average precision of 78.80% and a detection speed of 23.13 frames per second by merging Rainbow concatenation with the GoogLeNet Inception framework [23]. Similar to this, Jiang et al. (2019) achieved a detection accuracy of 78.80% and a speed of 23.13 FPS for apple leaf diseases by combining Rainbow concatenation with the GoogleNet Inception architecture. Other studies, such as the one presented by a team utilizing the Caffe Deep Learning framework, achieved precision levels ranging from 91% to 98% for diagnosing 13 different plant diseases [24]. This research demonstrated the effectiveness of deep convolutional networks for disease classification tasks. Additionally, a comparative study of CNN architectures like VGG-16 and VGG-19 revealed that data augmentation techniques significantly improved model accuracy from around 88% to 95% for diagnosing olive leaf diseases [25]. These advancements underscore the potential of deep learning technologies to revolutionize crop disease detection. Plant disease control is entering a new era of accuracy and efficiency by fusing cutting-edge computational techniques with time-tested agricultural procedures. The continual advancement of sophisticated models and methods presents encouraging prospects for improving crop health and guaranteeing the sustainability of agriculture.

2.2 Convolutional Neural Networks (CNNs)

2.2.1 Convolutional Layer

Convolutional Neural Nets (CNNs) are a significant class of deep learning algorithms intended to handle grid-like input, including images. Because of their distinctive architecture, CNNs perform incredibly well in tasks involving visual pattern detection and feature extraction. Convolutional layers, which are essential to CNNs, carry out convolution operations by swiping tiny filters over the input data to identify characteristics like edges, textures, and patterns. Since the Conv-2D layer conducts two-dimensional convolutions over the height and breadth of the input image, it is essential for tasks like object detection and image categorization. By adding a depth dimension, the Conv-3D layer expands on this idea and makes it possible to analyze volumetric or video data. Another essential element, pooling layers, downsamples feature maps to lower spatial dimensions while keeping important information, improving computational effectiveness and generalization. In addition, CNNs have extra layers including flattening layers that help with the final classification by converting 2D feature maps into 1D vectors. While batch normalization layers aid in training stabilization and speed, dropout layers guard against overfitting. While Softmax functions are utilized for classification tasks, activation functions like ReLU (Rectified Linear Unit) introduce nonlinearity, enabling the network to learn complex patterns. CNNs use backpropagation to iteratively modify filter weights in order to reduce error and enhance model performance. Adam and other optimizers are essential to this process because they dynamically modify the learning rate. CNNs have changed computer vision and image processing, leading to improvements in facial recognition, image segmentation, and object detection. Their significant importance in contemporary AI and machine learning applications is highlighted by their capacity to learn from large datasets and adapt to a variety of visual tasks. [7-9].

2.2.2 Pooling Layer

Pooling layers in Convolutional Neural Networks (CNNs) are critical for lowering the dimensionality of feature maps while keeping significant data. Pooling layers improve CNN efficiency and performance by reducing computational complexity and lowering the danger of overfitting. There are several types of pooling algorithms. The most prevalent is max pooling, which selects the maximum value from a set of values inside a feature map [8]. This method effectively retains the most prominent features of the input image, ensuring that essential details are preserved while reducing the spatial dimensions. Min pooling, although less common, selects the minimum value from a set of values in the feature map. This technique can be advantageous in specific applications where minimizing feature values is beneficial, providing a different perspective on the feature extraction process. Average pooling computes the average value from a collection of data in the feature map. As a result, it produces a more generalized representation of the input data, which can be beneficial in a variety of situations where a smoothed or averaged feature map is desired. These pooling strategies improve CNNs' ability to handle complicated visual data by quickly summarizing and decreasing feature maps, hence contributing to the network's overall performance and accuracy in tasks like picture classification and object detection.

2.2.3 Flatten Layer

Convolutional neural networks (CNNs) require the flattened layer in order to turn two-dimensional feature maps into one-dimensional vectors. This transformation is critical for closing the gap between convolutional and fully connected layers, allowing the network to successfully understand and process extracted data. By flattening the feature maps, the model can utilize these features in dense layers to perform classification or regression tasks. This step ensures that the spatial hierarchies learned during the convolutional process are preserved and utilized for making accurate predictions, enhancing the overall performance of the neural network in various tasks [9].

2.2.4 Dense Layer

Dense layers, also known as fully connected layers, serve an important function in neural networks since they combine the data extracted from previous layers to make final predictions. In these layers, each neuron is connected to every neuron in the previous layer, making it easier to learn complex data correlations. This wide connectedness enables the network to aggregate and interpret features at a higher level of abstraction, improving its ability to discern complex patterns and dependencies. Dense layers are essential for transforming the spatially structured information from convolutional layers into meaningful predictions for classification or regression tasks [9].

2.2.5 Dropout Layer

The dropout layer is a regularization technique that prevents overfitting in neural networks. During training, it randomly turns a portion of the input units to zero, essentially "dropping out" these units from the network. This method drives the network to develop more robust and redundant representations of the data because it cannot rely on a single neuron too much. By promoting redundancy and preventing co-adaptation of neurons, dropout improves the model's generalization capabilities, leading to better performance on unseen data. This technique is particularly effective in deep learning models, where overfitting is a common challenge [9].

2.2.6 Batch Normalization

In neural networks, batch normalization is a technique used to normalize the inputs of each layer such that the variance is one and the mean is zero. This normalization technique speeds up the training process, improves the network's stability, and increases its capacity to generalize to new data. By normalizing the inputs, batch normalization lowers internal covariate shift, allowing the model to learn more efficiently. This causes faster convergence throughout training, which can result in improved overall performance. It can also assist reduce difficulties like vanishing and bursting gradients [9].

2.2.7 Activation Function

Activation functions incorporate nonlinearity into neural networks, allowing them to describe complicated relationships in data. The Rectified Linear Unit (ReLU) is a widely used activation function in Convolutional Neural Networks (CNNs).

2.2.7.1 Rectified Linear Unit

The Rectified Linear Unit (ReLU) is a fundamental component in modern neural networks. This activation function transforms input values by applying a simple operation: it outputs the input value itself if it is positive, and zero if it is negative. This straightforward mechanism introduces non-linearity into the model, which is crucial for learning intricate patterns and features from data. As a result, ReLU allows the network to learn more effectively from the data and accelerates the training process's convergence. Its ease of use and efficiency make it a popular choice for deep learning initiatives. However, ReLU has drawbacks, including the well-known "dying ReLU" phenomenon, in which neurons become inactive and cease to learn. To address this, alternative activation functions such as Leaky ReLU and ELU have been proposed. Despite these challenges, ReLU remains a key element in enhancing the learning capabilities of CNNs and driving advancements in image recognition and other deep learning applications [26].

2.2.7.2 SoftMax

The SoftMax activation function is a crucial component in classification tasks within machine learning. It takes a vector of raw scores, or logits, and converts them into a probability distribution over multiple classes. By applying the exponential function to each score and then normalizing the results so that they sum to 1, SoftMax transforms these scores into interpretable probabilities. This function ensures that the output values are in the range of 0 to 1 and reflect the likelihood of each class. SoftMax is predominantly used in the final layer of neural networks for multi-class classification problems, enabling the model to make well-calibrated predictions by emphasizing the most probable class and diminishing the influence of less likely ones. This approach helps in better convergence during training and supports effective decision-making in classification tasks [26].

2.2.8 Loss Function

Loss functions are important in machine learning models because they quantify the difference between expected and actual output values. They serve as an indicator of how well or poorly the model performs, directing the optimization process during training. By computing the loss, these functions provide input to the model, allowing it to alter the weights and biases to reduce error and increase performance.

2.2.8.1 Catagorical_crossentrophy

A popular loss function for multi-class classification issues when each input is assigned to one of multiple classes is categorical cross-entropy. The difference between the true class labels and the anticipated probability distribution is computed by this loss function. It measures how well the model's predicted probabilities align with the actual class labels, where a lower cross-entropy value indicates better model performance. During training, categorical cross-entropy penalizes inaccurate predictions more severely, pushing the model to assign higher probability to the correct classes and therefore improving overall classification accuracy. [9].

2.2.9 Backpropagation

A crucial method for training neural networks in a way that minimizes the loss function is backpropagation. The forward pass and the reverse pass are the two steps involved. In order to produce predictions, input data is routed through the network's layers during the forward pass. The loss function calculates the difference between the predicted and actual results. In the backward pass, this loss is propagated back through the network to compute the gradients of the loss function with respect to each weight using the chain rule of calculus. These gradients represent the direction and size of the modifications required to mitigate the loss. The optimizer then updates the network's weights based on these gradients to improve performance. This iterative process continues until the model achieves optimal accuracy and generalization on the training data [27].

2.2.10 Optimizer

Optimizers adjust a model's weights during training by using gradients calculated through backpropagation. This procedure entails adjusting weights to reduce the loss function and improve the model's performance using the estimated gradients.

2.2.11 Adam Optimizer

Adam, or Adaptive Moment Estimation, is an advanced optimization methodology that expands upon the features of two earlier approaches, Adagrad and RMSProp. First- and second-order gradient estimations are used to adjust the learning rate for each parameter. Adam maintains a moving average of both gradients (first moment) and squared gradients (second moment). These averages help to dynamically alter the learning rate, improving convergence and efficiency. Adam provides a robust and effective technique for training deep learning models by combining the advantages of Adagrad's adaptive learning rates and RMSProp's learning rate normalization. It is extensively used because it can handle sparse gradients, make adaptive adjustments during training, and perform well over a wide range of neural network architectures [9].

2.2.12 RMSProp Optimizer

RMSProp (Root Mean Square Propagation) is an advanced optimization algorithm designed to enhance the training of deep learning models by addressing the limitations of earlier methods like Adagrad. The main objective of RMSProp is to modify the learning rate for every parameter according to the gradients' most recent magnitudes. It achieves this by maintaining a moving average of the squared gradients, which helps to normalize the learning rate for each parameter. This approach prevents the learning rate from becoming too small, a common issue with Adagrad, particularly when dealing with non-stationary or highly varied gradients. RMSProp guarantees that the model may continue learning efficiently even in situations where the gradients vary greatly across distinct parameters by modifying the learning rate in accordance with the gradient's magnitude. RMSProp is particularly effective in scenarios with noisy or sparse gradients, making it well-suited for training deep neural networks. Recurrent neural networks (RNNs) and other complicated models, in particular, have found widespread use in neural network topologies due to its ability to stabilize the learning process and encourage convergence. The algorithm's balance between simplicity and performance has established RMSProp as a reliable and efficient optimizer in the deep learning community [9].

2.3 Network Architecture

Network architectures define the structure of neural networks, including how layers and connections are organized to achieve specific tasks. The design of these architectures determines the network's effectiveness for various applications and problem-solving. Different architectures are tailored to handle distinct types of data and learning objectives. For example, Convolutional Neural Networks (CNNs) are very good at tasks involving images because they can identify features and spatial hierarchies. [28]. Recurrent Neural Networks (RNNs), on the other hand, excel at processing sequential data such as time series or text due to their capacity to maintain information over time [29]. Modern network architectures incorporate advanced techniques like residual connections, attention mechanisms, and memory cells to enhance learning efficiency and address challenges. Selecting the appropriate architecture is crucial for optimizing performance across different machine learning and deep learning applications. Below is a quick summary of how various data kinds correspond with particular network architectures:

- **Vector Data**: Best suited for densely connected networks or Dense layers.
- **Image Data**: Typically handled by 2D Convolutional Networks (CNNs).
- **Sound Data**: 1D Convolutional Networks (preferred) or RNNs can be used.
- **Text Data**: Either 1D Convolutional Networks (preferred) or RNNs.
- **Time-series Data**: RNNs (preferred) or 1D Convolutional Networks.
- **Sequence Data**: RNNs or 1D Convolutional Networks, with a preference for RNNs when data ordering is crucial.
- **Video Data**: 3D Convolutional Networks for capturing motion or a combination of 2D CNNs for feature extraction followed by RNNs or 1D Convolutional Networks for sequence processing.

2.4 Transfer Learning

In machine learning, transfer learning is the process of applying knowledge from one solved problem to another that is similar but not identical. This method allows models pre-trained on large, diverse datasets to be fine-tuned for specific applications, and is especially helpful when there is a lack of labeled data available for the new task. By leveraging pre-trained models, transfer learning can dramatically reduce training time and improve performance on tasks where data might be limited or complex. Because they have previously acquired a wide range of features from a large dataset, the pre-trained models offer a solid foundation that can be modified for the new task. [14]. This technique is a fundamental strategy in deep learning, especially for applications such as image classification, where vast datasets and computational resources are often required.

2.4.1 ResNet152-v2

ResNet152-v2, or Residual Network with 151 layers, is an advanced deep convolutional neural network architecture built to tackle the complexities of training very deep networks. Like its predecessors, it employs residual learning through the use of skip connections, also known as residual connections, which allow the input of a block to be added to its output. This technique ensures that gradients can flow more easily through the network during backpropagation, mitigating the vanishing gradient problem that can occur as network depth increases [15]. ResNet152-v2 enhances this approach by incorporating modifications such as pre-activation within its residual blocks, where batch normalization and ReLU activations precede the convolutional layers. This adjustment further improves gradient flow and model convergence. The architecture consists of batch normalization, ReLU activations, and a complex configuration of convolutional layers integrated into residual blocks. ResNet152-v2 excels in training very deep networks while preserving high performance and efficiency, making it a powerful tool for complex image classification tasks and other applications requiring deep learning models with exceptional accuracy and robustness.

2.4.2 VGG16

VGG16 is a deep convolutional neural network model developed by the Visual Geometry Group (VGG) at the University of Oxford. It is characterized by its simple and uniform architecture, consisting of 16 layers: 13 convolutional layers and 3 fully connected layers. The VGG16 model uses small 3x3 convolutional filters with a consistent architecture throughout the network. This design choice helps to capture fine details in images while maintaining a manageable number of parameters. VGG16 has been known for its high performance in image classification tasks and has been a benchmark model in computer vision tasks. Its straightforward architecture, with a sequence of convolutional layers followed by max-pooling layers and fully connected layers, allows for effective feature extraction and classification. The model's deep structure enables it to learn complex hierarchical features from images, making it a popular choice for transfer learning applications where high accuracy is required for tasks such as object recognition and scene understanding [16].

2.4.3 Inception V3

Inception V3 is a deep convolutional neural network architecture that is optimized for large-scale image categorization tasks. It belongs to the Inception family, which is renowned for its cutting-edge strategy for boosting computer effectiveness without sacrificing superior precision. Inception V3 uses a novel modular design known as Inception modules, which are made up of numerous convolutional filters of varied sizes applied in tandem. This enables the network to record elements at many scales at the same time, improving its capacity to spot patterns in images with varying levels of detail. A key aspect of Inception V3 is its use of factorized convolutions, where larger convolutional filters are broken down into smaller ones. This reduces the computational cost without sacrificing performance. Additionally, Inception V3 incorporates techniques such as batch normalization, label smoothing, and auxiliary classifiers to improve model generalization and stability during training. The architecture of Inception V3 is structured with a series of these Inception modules, along with convolutional layers, pooling layers, and fully connected layers. This design enables the network to extract rich feature representations from input images while keeping the number of parameters manageable. Inception V3 is widely used in various applications, from image classification to object detection, due to its balance of accuracy, efficiency, and scalability [17].

Model	Architecture	Key focus	Significant variants
ResNet152-v2	Deep, layered CNN	Utilizes identity skip connections to address vanishing gradients	ResNet-151, ResNet-101, ResNet-50
VGG16	Multi-layer CNN with small filters	Deep stack of convolutional layers with 3x3 filters	VGG19
Inception-v3	Multi-layer CNN with inception modules	Various convolutional operations with different filter sizes	Inception-v1, Inception-v2

Table 2.1: Summary of transfer learning models

2.5 Explainable Artificial Intelligence (XAI)

The term "explainable artificial intelligence" (XAI) describes a collection of approaches and strategies meant to increase the transparency and human understandability of AI systems. Understanding the decision-making processes of AI models, especially deep learning algorithms, gets harder as they get more complicated. XAI seeks to bridge this gap by providing insights into how models arrive at their predictions, which is crucial for trust, accountability, and refinement [18].

XAI techniques are designed to elucidate the inner workings of AI models, making them more accessible to both experts and non-experts. This transparency helps in diagnosing model failures, ensuring fairness, and complying with regulatory requirements. XAI promotes a deeper comprehension of model behavior and helps with better decision-making processes in a variety of applications by making AI systems more interpretable.

2.5.1 Grad-CAM

Grad-CAM, or Gradient-weighted Class Activation Mapping, is an effective technique for visualizing picture regions that influence a CNN's predictions. Grad-CAM generates heatmaps that emphasize the crucial portions of a picture that influence the model's judgment. Gradients across the final convolutional layer are utilized to determine which areas of the image are more important for the projected class. Grad-CAM creates a weighted mixture of these maps by computing the gradient of the target class in relation to the feature maps. This combination is then upsampled to the dimensions of the input image to create the heatmap. This visualization technique allows you to examine the features the model concentrates on, providing insights into the model's decision-making process. Grad-CAM is especially beneficial for assessing model behavior and verifying that the model's focus is consistent with expert knowledge of the problem domain [18].

2.5.2 Layer-CAM

Layer-CAM, or Layer-wise Class Activation Mapping, builds upon the concepts of Grad-CAM to provide a more detailed visualization of the CNN's decision-making process. Unlike Grad-CAM, which focuses on the gradients of the last convolutional layer, Layer-CAM allows for the exploration of intermediate layers within the network. This technique generates activation maps from various layers to show how different layers contribute to the final classification decision. By examining these intermediate feature maps, Layer-CAM helps to understand which layers and features are important at different stages of the model's processing pipeline. This approach not only visualizes the regions of the image influencing the prediction but also reveals how features are hierarchically combined to form the final decision. Layer-CAM is beneficial for in-depth model analysis and debugging, offering a comprehensive view of the feature extraction and decision-making processes [19].

2.6 Data Processing

Data processing techniques are crucial for enhancing the quality and effectiveness of datasets used in training machine learning models. These techniques help ensure that the data is well-prepared, which can lead to improved model performance and generalization capabilities.

2.6.1 Batch Normalization

Batch normalization is a technique for stabilizing and accelerating deep neural network training. By deducting the batch mean and dividing by the batch standard deviation, it normalizes the output of an earlier activation layer. This procedure modifies the distribution of activations and gradients across the network, assisting in the mitigation of difficulties such as vanishing and bursting gradients. Batch Normalization calculates the mean and variance of each feature throughout the batch during training. These statistics are used to normalize the features, and learnable parameters are added to scale and shift the normalized values, allowing the model to retain its representational capability. This method results in faster convergence during training since the model can employ larger learning rates without risking instability. Furthermore, Batch Normalization serves as a regularizer, lowering the need for dropout and possibly increasing the model's generalization performance. Overall, batch normalization is a fundamental deep learning technique that improves the robustness and efficiency of neural networks.

2.6.2 Augmentation

Data Augmentation is a strategy used to artificially increase the size and diversity of training datasets by applying a range of transformations to existing data samples. This technique helps improve the model’s ability to generalize to new, unseen data by introducing variations that mimic real-world scenarios. Common augmentation techniques include rotation, scaling, flipping, and cropping, which modify the original images to create new training examples. For instance, rotating an image by a few degrees or flipping it horizontally can produce different perspectives of the same object. Other techniques might include adjustments in brightness, contrast, or adding noise to simulate different environmental conditions. By increasing the diversity of the training data, augmentation helps the model become more resilient to variations in input data and reduces the risk of overfitting. It allows the model to learn more robust features and patterns, thereby enhancing its performance on test datasets. Overall, data augmentation is a powerful tool for extending datasets and improving the robustness of machine learning models, particularly in image classification tasks.

2.7 Model Evaluation

A crucial phase in the machine learning process is model assessment, which assesses how effectively a trained model functions on data that it did not encounter during training. This step evaluates the model's capacity to generalize to new, unknown data while also testing its accuracy on the training set. The major purpose is to assess whether the model can make accurate predictions in real-world circumstances. By assessing a model's performance, we may determine whether it is just memorizing training data or learning patterns that can be applied to fresh samples. Effective model evaluation entails computing measures like accuracy, precision, recall, and F1 score, which provide information about many elements of the model's performance. Accuracy is the proportion of correct predictions; precision is the proportion of true positives among all positive predictions; recall is the proportion of actual positives correctly identified; and the F1 score is a combination of precision and recall that provides a single performance metric. These measures let us understand not only how frequently the model is correct, but also how well it identifies relevant events and avoids making incorrect predictions. In essence, model evaluation is more than just obtaining high performance on test data; It also involves making sure that the model's predictions hold up in real-world scenarios and are applicable. This step is critical for ensuring that the model can handle fresh data and perform real-world activities.

2.8 Previous Studies

Agriculture plays a fundamental role in modern society, contributing significantly to both the economy and daily life. In regions such as Bangladesh, where agriculture is a primary livelihood for a substantial portion of the population, the significance of crop production is profound. Yet, this vital industry is continually threatened by diseases caused by bacteria, fungi, viruses, and other pathogens, which can reduce crop yields by as much as 95% and severely impact both the quantity and quality of agricultural produce. Early and accurate detection of crop diseases is crucial to mitigating these effects and improving productivity. Traditional procedures, on the other hand, are usually insufficient because of their time-consuming nature and possibility for inaccuracy. These methods typically include manual inspections and professional expertise. In recent years, there has been a substantial trend toward using modern technologies to identify plant diseases. This movement is motivated by the desire for more efficient and dependable solutions to supplement or replace old procedures. Convolutional Neural Networks (CNNs), a sort of deep learning model, have emerged as an effective tool for this task. CNNs excel at interpreting complicated visual patterns and have been effectively used in a variety of image-based classification tasks, including plant disease diagnosis. These models have shown considerable promise in enhancing diagnostic accuracy and offering automated solutions for disease detection.

The use of deep learning models to assess high-resolution plant leaf photos and detect minute disease indicators is one noteworthy development in this field. For instance, recent research has demonstrated the effectiveness of CNN-based approaches in improving disease classification accuracy. Chen et al. (2020) explored the use of deep learning for plant disease diagnosis, combining the Inception module with VGGNet to achieve an impressive The average accuracy for rice plant picture classification is 92.00%. Their research demonstrated the potential of transfer learning, which involves adapting pre-trained models for new tasks, to attain excellent performance even in complicated circumstances.

Similarly, Sunil et al. (2022) addressed the challenges of feeding a growing global population and the impact of plant diseases on crop yields. Their research involved integrating For plant leaf image analysis, deep learning models such as AlexNet, ResNet50, and VGG16 achieved excellent accuracy rates of 100% for binary classification tasks and 99.53% for multi-class datasets. This study demonstrated the effectiveness of combining different CNN architectures to enhance disease detection capabilities.

In another significant contribution, Gayathri et al. (2020) developed the INAR-SSD model, which utilized Rainbow concatenation with the GoogLeNet Inception framework to identify five major apple leaf diseases. Their approach, which included considerable data augmentation and annotation, resulted in a detection rate of 23.13 FPS and an accuracy of 78.80% mAP. This study demonstrated the potential for real-time disease detection and established new performance standards for apple leaf disease diagnosis.

Jiang et al. (2019) introduced a similar deep learning technique that combined Rainbow concatenation with the GoogleNet Inception architecture to detect five primary types of apple leaf diseases. Their model achieved a detection speed of 23.13 FPS and an accuracy of 78.80% mAP, demonstrating that advanced CNN techniques can deliver both speed and precision in disease detection.

Additionally, recent studies have explored various CNN architectures for plant disease classification. For example, researchers employed VGG-16 and VGG-19 to compare transfer learning scenarios, showing that data augmentation techniques could significantly enhance model accuracy from approximately 88% to around 95% for olive plant disease classification. This work underscored the effectiveness of combining deep convolutional networks with robust data processing methods for plant disease diagnosis.

Overall, these studies reflect a growing body of research focused on leveraging deep learning technologies to advance plant disease detection. The shift from traditional methods to sophisticated AI-driven solutions has not only improved diagnostic accuracy but also opened new avenues for developing adaptive, scalable, and efficient tools for managing plant health. The continuous advancements in CNN architectures and transfer learning techniques promise to further enhance these capabilities and drive future innovations in the field of plant disease detection.

2.9 Comment about Previous Studies

The examination of prior research highlights a transformative shift in plant disease detection from traditional methodologies to advanced deep learning techniques. Historically, plant disease management relied on expert knowledge and manual inspection, which were subjective and time-consuming. Recent studies illustrate a significant evolution through the adoption of Convolutional Neural Networks (CNNs), utilizing sophisticated algorithms capable of analyzing large and complex datasets with high accuracy. The integration of transfer learning techniques has further enhanced these models, allowing effective adaptation of pre-trained networks to new tasks, improving performance and extending their utility. Additionally, Explainable Artificial Intelligence (XAI) methods, such as Grad-CAM and Layer-CAM, have introduced interpretability to these models, providing transparency in AI systems. Despite these advancements, challenges remain, such as data quality, generalizability across different plant species, and practical implementation in resource-limited settings. While models like ResNet50 and VGG16 have achieved high accuracy, their performance can be sensitive to dataset variations. The reviewed studies reflect a broader trend towards leveraging state-of-the-art technologies to overcome traditional limitations, promising future innovations in plant disease management. Continued advancements in deep learning architectures and efforts to make these technologies more accessible are essential for future improvements in plant disease detection systems.

Chapter 3

Proposed Methodology

3.1 Proposed Method of CNN

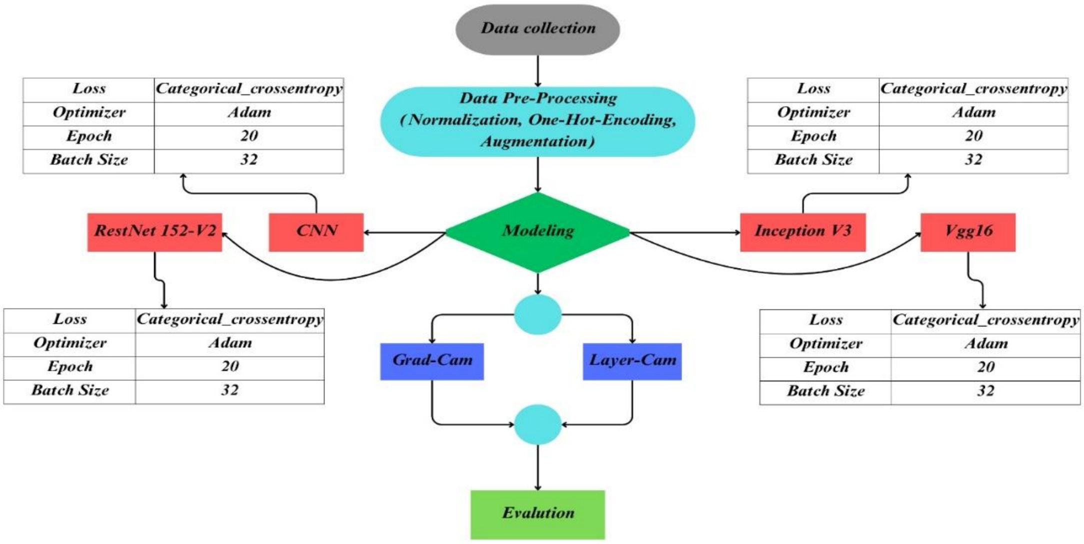


Figure 3.1: Proposed Methodology

3.2 Dataset Description

For this study, the Plant Village dataset was selected as the primary source, acquired from the Kaggle platform. The original dataset comprises 38 classes of plant diseases, including both healthy and diseased leaf images, totaling 54,707 images. Each class represents a distinct category of plant disease or a healthy plant condition, with the number of images per class varying significantly. Given the diverse nature of the dataset, a subset of 16 classes was chosen to streamline the training process for the proposed model.

This new dataset includes a balanced number of images for each class to ensure effective training and evaluation. Each of the selected classes contains approximately 2000 images, with a few slight variations. The final dataset for this study, therefore, comprises 32,127 images.

This curated dataset reflects a range of common plant diseases and healthy conditions across several crops, including apples, corn, grapes, and tomatoes. By focusing on these specific classes, the study aims to develop a robust model capable of accurately diagnosing a diverse set of plant conditions.

partition	number of images
train	**22,488**
validation	**6,425**
test	**3,214**

Table 3.1: dataset

3.3 Data Preprocessing

In the pre-processing phase of the plant disease classification project, the dataset was meticulously prepared to ensure optimal performance of the Convolutional Neural Network (CNN) model. The images were initially loaded from their respective folders, with each folder representing a distinct class of plant disease. Each image was converted to RGB format and resized to a standard resolution of 256x256 pixels. This resizing was essential to maintain uniformity across the dataset and to meet the input requirements of the CNN.

Following image resizing, the pixel values of all images were normalized to a range between 0 and 1. This normalization process was achieved by converting the pixel values to the float32 data type and then dividing by 255, the maximum pixel value in an 8-bit image. Normalization helps to standardize the input data and improves the convergence rate during model training.

The target labels for each image were then mapped from their string representations to numerical indices. This conversion was necessary for the model to process categorical data effectively. A dictionary was created to associate each unique label with a corresponding index, which facilitated the transformation of the string labels into numerical values.

The numeric labels were then translated to a categorical representation by one-hot encoding. This approach encodes each label as a binary vector of the same length as the number of classes, with just the index corresponding to the class label set to one. Due to its ability to match the output format required by the softmax activation function in the model's final layer, this encoding is essential for multi-class classification problems.

To complete the preprocessing, the data was divided into training, validation, and test sets. Each subset was handled independently, ensuring that the model training and evaluation were done on different parts of the dataset. The meticulous preparation of the dataset was critical in establishing accurate and dependable performance measures for the CNN model.

3.4 Data Augmentation

To enhance the dataset and improve the model's robustness, data augmentation techniques were employed. Utilizing TensorFlow's Keras library, the ImageDataGenerator class was instrumental in applying a variety of transformations to the images, thereby increasing their diversity and the overall size of the dataset.

The augmentation process included several key transformations: images were rotated randomly within a 40-degree range to simulate different viewing angles, shifted both horizontally and vertically by up to 20% to account for positional variations, and subjected to shearing and zooming effects up to 20%. These modifications help to simulate real-world variations that the model might encounter.

Additionally, images were flipped horizontally and vertically to introduce further variability, and their brightness was adjusted within a range of 0.5 to 1.5 to emulate different lighting conditions. The fill_mode parameter was set to 'nearest', ensuring that any newly created pixels during augmentation were filled with the nearest existing pixel values, thus maintaining visual consistency.

The primary objective of this augmentation process was to generate an additional 2000 images to balance the dataset and provide the model with a more comprehensive range of examples. This was achieved by randomly selecting images from the original dataset, applying the aforementioned transformations, and saving the resulting augmented images.

The goal of the data augmentation method was to improve plant disease classification by improving the model's capacity to generalize and function accurately in a variety of real-world circumstances by expanding the dataset size and adding variability.

3.5 CNN Architecture

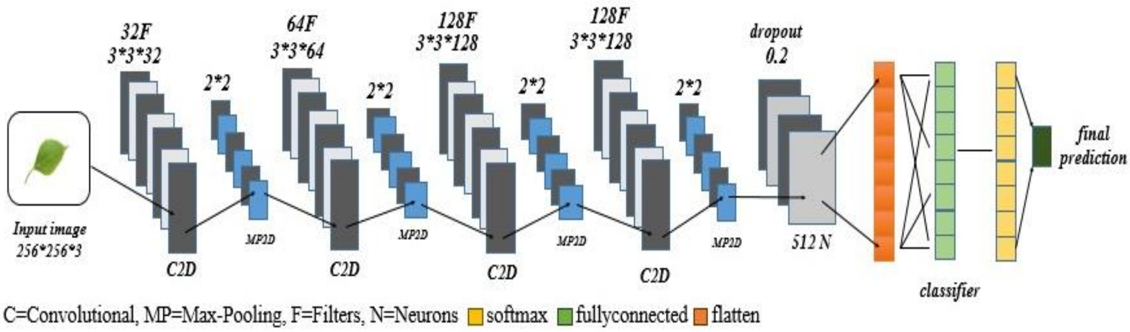


Figure 3.2: Proposed CNN architecture

The convolutional neural network (CNN) architecture used for plant disease classification is intended to efficiently capture and understand complicated patterns from photos. The design begins with a sequence of convolutional layers, starting with 32 filters of size 3×33 \times 33×3, applied to the input picture of dimensions 256×256×3256 \times 256 \times 3256×256×3. The initial layer is followed by a max pooling layer with a pool size of 2×22 \times 22×2. This reduces the spatial dimensions of the feature maps while keeping key characteristics.

The network continues with an additional convolutional layer consisting of 64 filters, again with a 3×33 \times 33×3 kernel, followed by another max pooling layer. This procedure is repeated, with 128 filters being used in the third and fourth convolutional layers, respectively, with max pooling operations coming after each. These convolutional and pooling operations progressively extract and downsample features, enabling the model to learn complex patterns while reducing the dimensionality of the data.

Following the convolutional and pooling layers, the design includes a flattening layer that converts 3D feature maps to 1D vectors. This flattened output is then fed through a dense layer of 512 units, which creates nonlinearity and records complex interactions using a ReLU activation function. A dropout layer with a dropout rate of 0.2 is added to stop overfitting. During training, it randomly sets a portion of the input units to zero. Finally, a dense layer with 16 units and a softmax activation function are used to generate the final class probabilities, which correspond to the sixteen disease classes. The total number of parameters in the model is approximately 13,094,608, all of which are trainable. This comprehensive architecture is designed to effectively classify images of plant diseases, utilizing deep learning methods to attain superior performance in differentiating between different disease categories.

```
Model: "sequential"
_________________________________________________________________
Layer (type)            Output Shape           Param #
=================================================================
conv2d (Conv2D)           (None, 254, 254, 32)     896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 127, 127, 32)     0
_________________________________________________________________
conv2d_1 (Conv2D)          (None, 125, 125, 64)     18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 62, 62, 64)       0
_________________________________________________________________
conv2d_2 (Conv2D)          (None, 60, 60, 128)      73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 30, 30, 128)      0
_________________________________________________________________
conv2d_3 (Conv2D)          (None, 28, 28, 128)      147584
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 128)      0
_________________________________________________________________
flatten (Flatten)          (None, 25088)           0
_________________________________________________________________
dense (Dense)             (None, 512)           12845568
_________________________________________________________________
dropout (Dropout)          (None, 512)            0
_________________________________________________________________
dense_1 (Dense)           (None, 16)            8208
=================================================================
Total params: 13,094,608
Trainable params: 13,094,608
Non-trainable params: 0
_________________________________________________________________
```

3.6 Data Implementation

An Intel Core i5 processor, 12 GB of RAM, and a 250 GB SSD comprised the machine that the model was trained on. TensorFlow, Keras, NumPy, OpenCV, and tqdm were among the important packages and libraries used for model training, picture preprocessing, and augmentation. Python 3.8.15 was the programming language used for the model implementation.

The dataset was input into the CNN model in batches during training, with a batch size of 32. Using the estimated first and second moments of the gradients as a guide, the ADAM optimizer—a well-liked variation of stochastic gradient descent—was used to optimize the model. For multi-class classification problems, the categorical cross-entropy loss function was utilized. After the completely connected layers, a dropout rate of 0.2 was imposed to avoid overfitting.

The 32,127 photos in the prepared dataset are split into 16 classes in order to facilitate the training, validation, and testing of the Convolutional Neural Network (CNN) model. Models for the classification of plant diseases are trained and evaluated using this dataset, which is a subset of the original Plant Village dataset.

Approximately 70% of the dataset was set aside for training, 15% for validation, and 15% for testing after it was divided into training, validation, and test sets. In order to comply with the CNN model's needed input shape, each image was scaled to 256 by 256 pixels. Normalization was also used to scale pixel values to a range of [0, 1], which increased the stability and training efficiency of the model. To improve the model's capacity for generalization, data augmentation techniques were used, including arbitrary transformations like rotation, shifting, zooming, and flipping.

In order to minimize the loss function, the model's parameters were iteratively changed during training using the validation set for hyperparameter tuning and the training set for learning. The model was tested on an independent test set following training to determine how well it performed and how well it could generalize to new data. The CNN model was robustly trained and precisely validated thanks to this thorough data implementation approach, which produced dependable performance in the classification of plant diseases.

3.7 Transfer Learning Models for Comparison

Transfer learning is a potent deep learning technique that makes use of pre-trained models to tackle new problems, especially in situations when data availability is constrained. Transfer learning makes it possible to employ a model that has previously been trained on a huge dataset—like ImageNet—instead of starting from zero and refining it on a smaller, task-specific dataset. This method works well because the pre-trained model can more easily adapt to the new task with less data because it has already learnt general features that can be used to a variety of image identification tasks. Transfer learning greatly reduces the time and computing power required for training while maintaining acceptable performance in situations when huge datasets are hard to get.

In this thesis, transfer learning was employed to compare the performance of pre-trained models with the custom CNN model developed for plant disease classification. By utilizing pre-trained models such as VGG16, ResNet152-v2, and Inception V3, their effectiveness in extracting features and classifying plant diseases was assessed. Each of these models was fine-tuned using the specific dataset created for this study, and their results were compared to determine the optimal model for this task. Transfer learning not only provided a baseline for performance comparison but also highlighted the potential benefits of using well-established architectures in real-world applications of plant disease detection.

3.7.1 VGG16

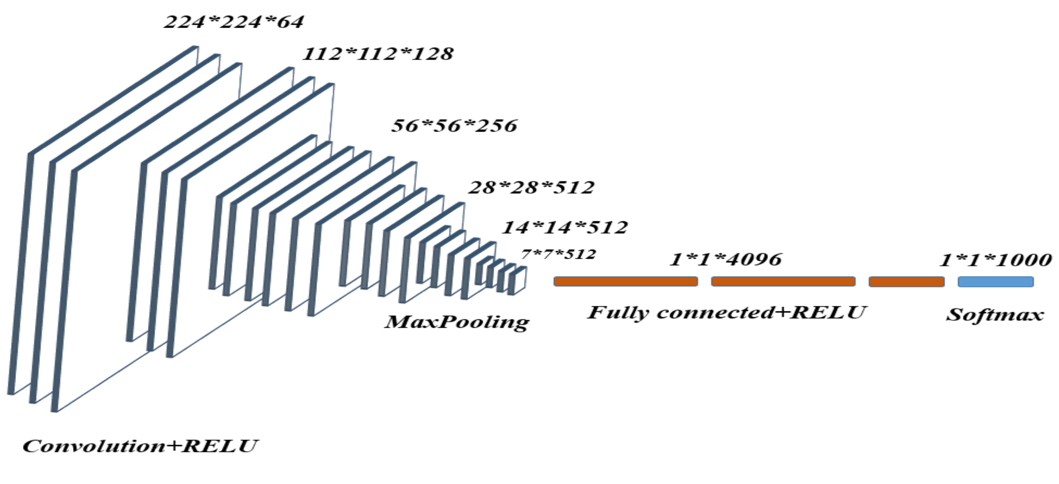


Figure 3.3: VGG 16 architecture

One well-known deep learning model is VGG16, which is well-known for its simple yet efficient architecture. VGG16, created by the University of Oxford's Visual Geometry Group, has a sequence of convolutional layers that are followed by fully linked layers. The architecture of the model is uniform, with 2x2 max-pooling layers and tiny 3x3 convolutional filters used all over the network. Because of its deep stack of convolutional layers, VGG16's design allows it to gradually extract higher-level features from images, allowing it to catch complicated patterns.VGG16 is particularly notable for its performance in image classification tasks due to its depth and the simplicity of its architecture. The model can learn fine-grained information thanks to the use of small convolutional kernels, which enhances its ability to discriminate between various image classes.

VGG16 requires a lot of memory and processing power because to its depth, which also results in high computational demands. This aspect can be a limitation when working with constrained hardware or requiring rapid model deployment [16].

In this study, VGG16 was employed as a pre-trained model for the task of plant disease classification. The model was used without the top classification layer (i.e., include_top=False), and a custom classifier was added on top to modify the architecture for the particular purpose. The input size for VGG16 was adjusted to 256x256 pixels to match the dimensions of the dataset used in this study.

To preserve the characteristics that were learned from the ImageNet dataset, the underlying model was made non-trainable. Meanwhile, a new classifier was built, which was composed of a flattening layer, a dense layer with 16 units, and a softmax activation function.

Accuracy was used as the assessment metric when the model was constructed using the Adam optimizer with categorical cross-entropy loss. This method made use of VGG16's strong feature extraction capabilities while allowing it to be tuned for the particular classification goal.

3.7.2 ResNet152-v2

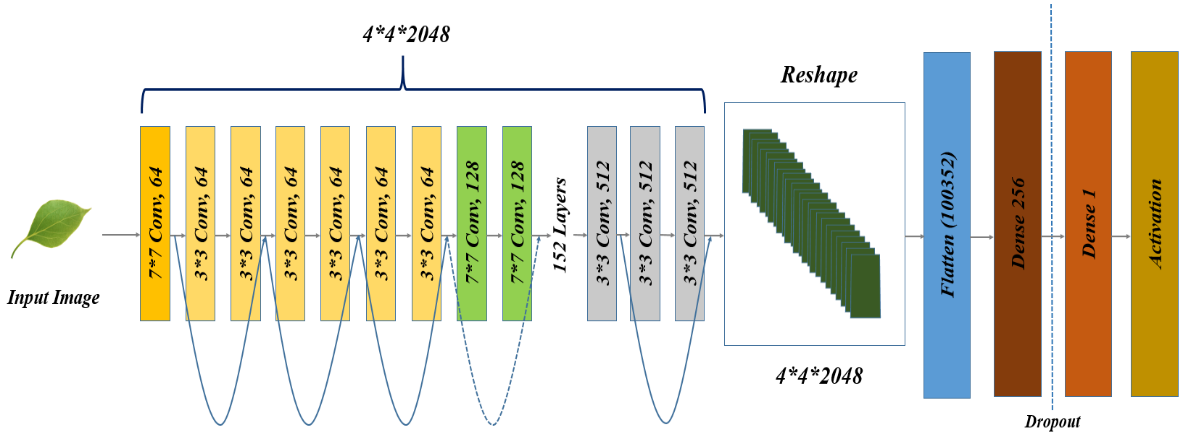


Figure 3.4: ResNet152-v2 architecture

The unique deep learning architecture known as ResNet, or Residual Networks, is well-known for its usage of residual connections, which solve the vanishing gradient issue that frequently causes difficulties while training deep networks. By addressing gradient propagation-related problems, residual connections enable the model to learn an identity function in addition to the primary function, hence enabling the training of much deeper networks.

Among the various ResNet variants, ResNet152-v2 was selected for this study due to its considerable depth and demonstrated efficacy in handling complex tasks. The ResNet152-v2 model features 151 layers, making it suitable for capturing intricate patterns and representations in image data. Its architecture includes advanced features such as batch normalization and improved residual blocks, which enhance its performance and robustness.

In this research, ResNet152-v2 was integrated into the experiments by utilizing its pre-trained weights as a base model. The model was adapted for plant disease classification by freezing the base layers to retain the learned features and adding a new classifier layer on top. With this strategy, the model was able to concentrate on the particular classification task while utilizing ResNet152-v2's comprehensive feature extraction capabilities. In order to assess the ResNet152-v2 model's efficacy in comparison to other models for plant disease identification, its accuracy and performance on the validation set were taken into consideration [15].

3.7.3 Inception V3

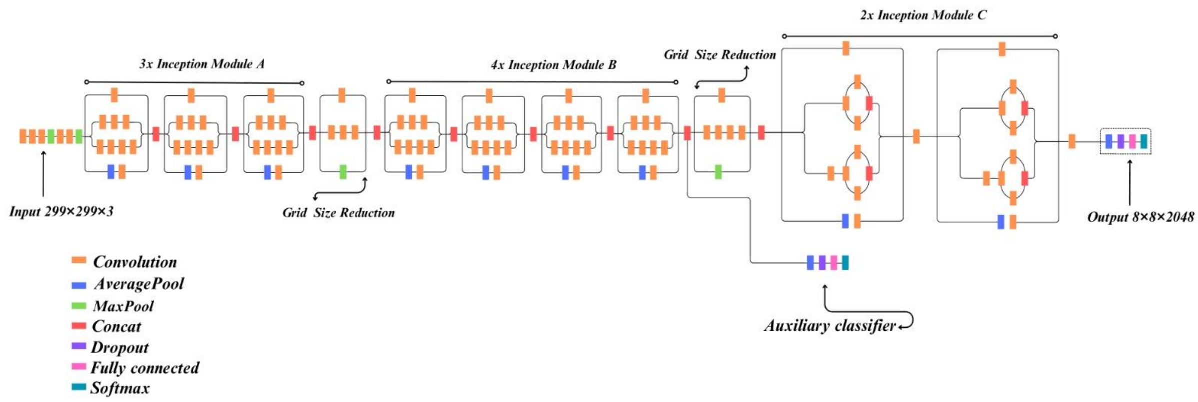


Figure 3.5: Inception-v3 Architecture

Using inception modules, Inception V3 is a sophisticated deep learning model that increases computing efficiency and expands the capabilities of deeper networks. The core innovation of Inception V3 lies in its inception modules, which consist of multiple parallel convolutional layers with different filter sizes, pooling operations, and other transformations. This architecture enables the model to extract features at various scales and levels of abstraction, which is crucial for handling diverse patterns and structures in image data.

The inception modules are structured to process input images through multiple branches simultaneously. Each branch performs convolution operations with different kernel sizes and pooling operations, capturing different aspects of the image. These features are then concatenated and passed to subsequent layers, allowing the model to aggregate multi-scale features effectively. This design not only improves the model's ability to recognize complex patterns but also optimizes computational efficiency by reducing the need for excessive network depth. For this thesis, Inception V3 was applied to the plant disease classification dataset by leveraging a pre-trained version of the model. The pre-trained Inception V3 model, which had been trained on a large dataset such as ImageNet, was used as a feature extractor by excluding its top classification layers. The model's weights were frozen during training to prevent them from being updated, ensuring that the pre-trained features remained intact. A dense layer with 16 output units, which corresponded to the number of classes in the dataset, a flattening layer, and a dropout layer with a rate of 0.2 were all included in the custom classifier that was constructed on top of the Inception V3 base. The model was trained for 20 epochs using a training dataset and a validation dataset. It was constructed using the Adam optimizer with categorical cross-entropy loss function. The accuracy with which Inception V3 was able to classify plant illnesses was used to assess its performance, and the outcomes were compared to those of other models [16].

3.8 Explainable Artificial Intelligence

In the realm of artificial intelligence (AI), particularly in deep learning, explainability has become an increasingly critical aspect. Explainable AI (XAI) refers to techniques and methods designed to make the decisions and processes of AI models transparent and understandable to humans. This is especially important in applications where decisions significantly impact stakeholders, such as in plant disease detection, where accurate and reliable results are essential for effective disease management and prevention. Despite their great potential, deep learning models are frequently seen as "black boxes" because of their detailed decision-making processes and complicated topologies. Understanding how these models arrive at specific predictions can be challenging, which may hinder trust and confidence in their outputs. Understanding the reasoning behind a model's predictions can give stakeholders, such farmers or agronomists, important insights on plant health and disease patterns, enabling them to make better decisions. [18]. To address these challenges, XAI techniques offer methods to demystify the inner workings of deep learning models.. Two well-known XAI approaches, Layer-CAM and Grad-CAM, were used in this thesis to shed light on the trained models' decision-making processes.

3.8.1 Grad-CAM

A popular explainable AI method called gradient-weighted class activation mapping, or Grad-CAM, offers visual representations of the decisions made by deep learning models. Grad-CAM produces heatmaps that show which areas of an input image most significantly affect the predictions made by the model. This image is produced by computing the gradients of a target class with respect to the feature maps of the final convolutional layer. This helps to highlight the regions that the model deems significant for a certain prediction [18].

In my work, Grad-CAM was implemented to analyze the internal workings of the convolutional neural network (CNN) model used for plant disease detection. By generating Grad-CAM visualizations, I was able to identify which parts of the plant leaf images were contributing most to the model's classification decisions. This provided valuable insights into how the model processed various plant diseases and whether it focused on relevant areas of the leaf, such as spots, discoloration, or other disease symptoms. Upon applying Grad-CAM to the predictions made by the model on the dataset, several key findings emerged. For instance, in cases where the model accurately predicted the disease, the Grad-CAM heatmaps clearly highlighted the affected areas of the leaves. However, in instances where the model's predictions were incorrect, the heatmaps sometimes showed attention to irrelevant regions, indicating areas where the model's focus needed improvement. These insights from Grad-CAM helped refine the model by guiding adjustments to the training process and hyperparameters.

3.8.2 Layer-CAM

Layer-wise Class Activation Mapping (Layer-CAM) extends the Grad-CAM approach by providing more detailed and accurate visualizations. While Grad-CAM focuses on the final convolutional layer, Layer-CAM generates heatmaps using information from multiple layers of the model. This captures hierarchical feature representations that are essential for precise prediction and enables a more thorough understanding of how the model processes information at various stages of the network [19]. In this research, Layer-CAM was applied alongside Grad-CAM to compare the effectiveness of these techniques in interpreting the model's decision-making. By utilizing Layer-CAM, I was able to obtain a deeper analysis of the model's attention throughout the network, not just at the final layer. The visualizations from Layer-CAM provided a more granular view of the model's focus areas, offering insights into how different layers contributed to the overall prediction. The specific findings from my thesis indicate that Layer-CAM produced more accurate and detailed visualizations compared to Grad-CAM. The ability of Layer-CAM to capture information from multiple layers resulted in clearer and more relevant heatmaps, which better represented the features influencing the model's decisions. This enhanced interpretability was crucial for validating the model's performance and ensuring that it focused on the correct areas of the input images for plant disease detection.

Chapter 4

Model Implementation and Experimental Result

This chapter presents the implementation of various models used in this thesis and their corresponding experimental results. The primary focus is on comparing the performance of different deep learning models for plant disease classification, with special emphasis on a custom CNN model and popular pre-trained models. The data source is covered in detail at the beginning of the chapter, which is then followed by the installation of the suggested CNN model and transfer learning models, such as ResNet152-v2, VGG16, and Inception-V3. The results of these experiments are presented and discussed, highlighting the effectiveness of each model in terms of accuracy and other evaluation metrics.

4.1 Data Source

The Plant Village dataset, a publicly accessible dataset from Kaggle, was employed in this study [20]. This dataset originally contains 54,707 images categorized into 38 distinct classes of plant diseases. For the purpose of this thesis, a subset of 16 classes was selected, consisting of 32,127 images in total. The collection contains tagged photos of both healthy and damaged plant leaves, and these classes represent a range of plant diseases.

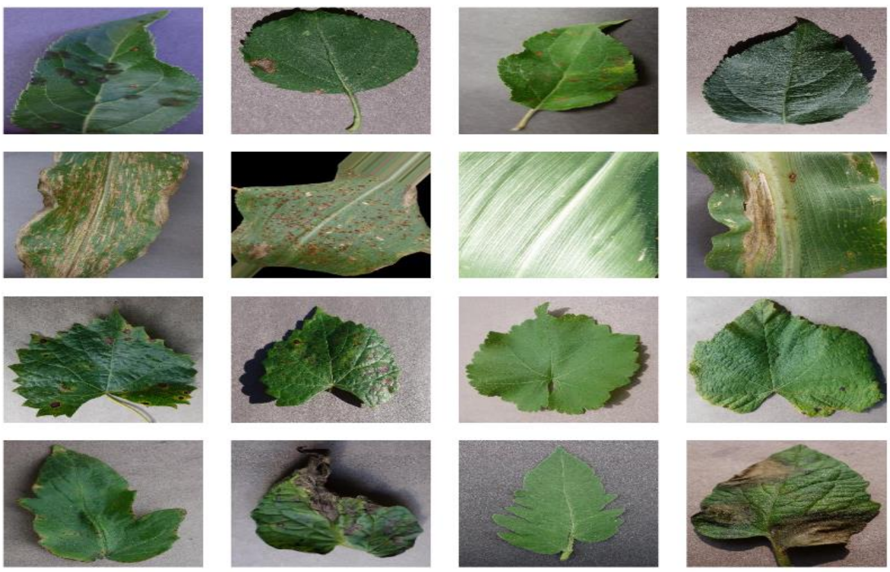


Figure 4.1: Image of Plant Leaf

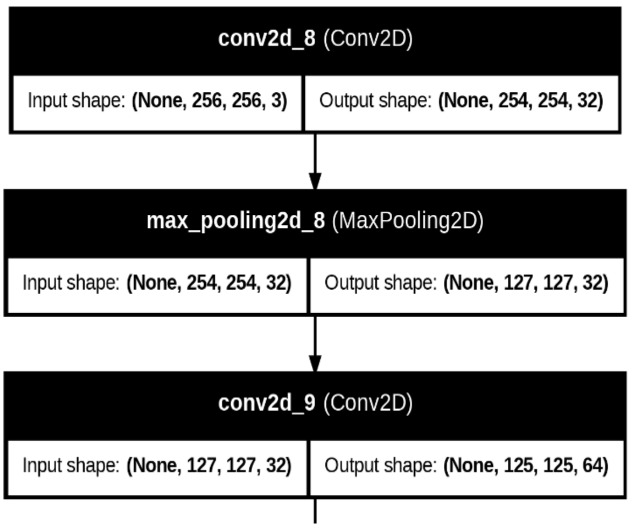
The dataset is highly imbalanced in terms of class distribution, making it essential to employ techniques like data augmentation to address this issue. The dataset was pre-processed, including resizing images to 256x256 pixels and normalization, in preparation for model training.

4.2 Implementation

The implementation of the models was carried out using Python (version 3.8.15) and relevant deep-learning libraries, such as TensorFlow and Keras. The models were trained on a machine with an Intel Core i5 processor, 12 GB RAM, and a 250 GB SSD. Four models were implemented and compared: a proposed CNN model and three transfer learning models—ResNet152-v2, VGG16, and Inception-V3. Each model was fine-tuned to optimize performance on the given dataset.

4.2.1 Proposed CNN model

The unique CNN model was created from the ground up to categorize photos of plant diseases. It is made up of four convolutional layers, followed by a max-pooling layer and a fully linked dense layer. Due to the design of the model, spatial characteristics may be extracted from photos efficiently. The model was trained with 32 batches, the Adam optimizer, and categorical cross-entropy as the loss function. To avoid overfitting, a dropout rate of 0.2 was applied. This model was implemented and trained on the Plant Village dataset, and the experimental results are compared with the pre-trained models discussed below.



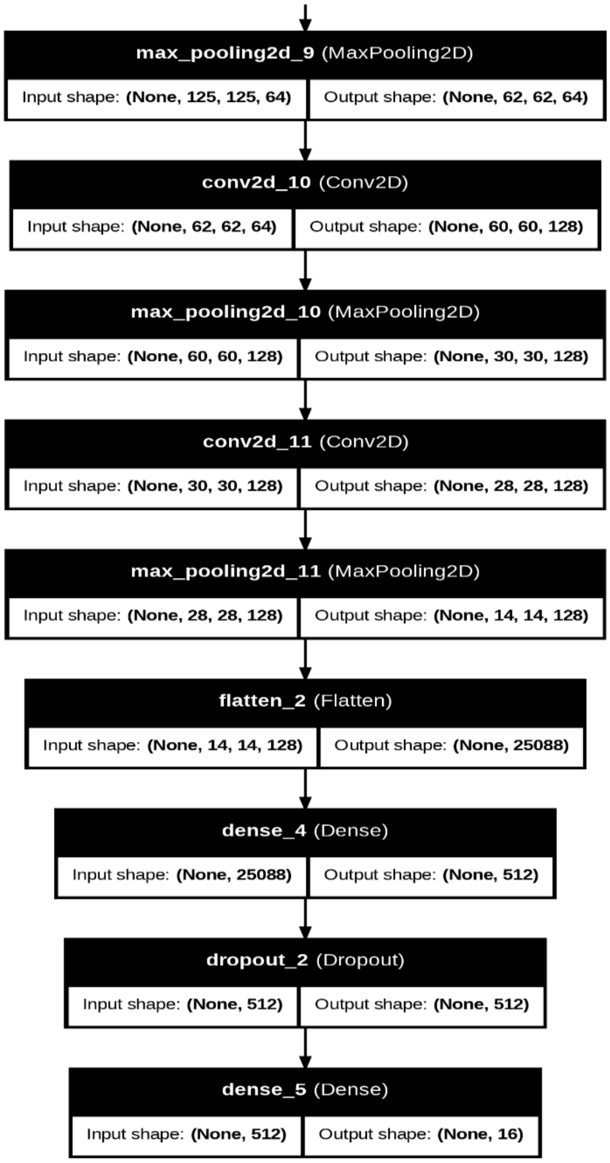


Figure 4.2: Plant leaf classification of CNN model

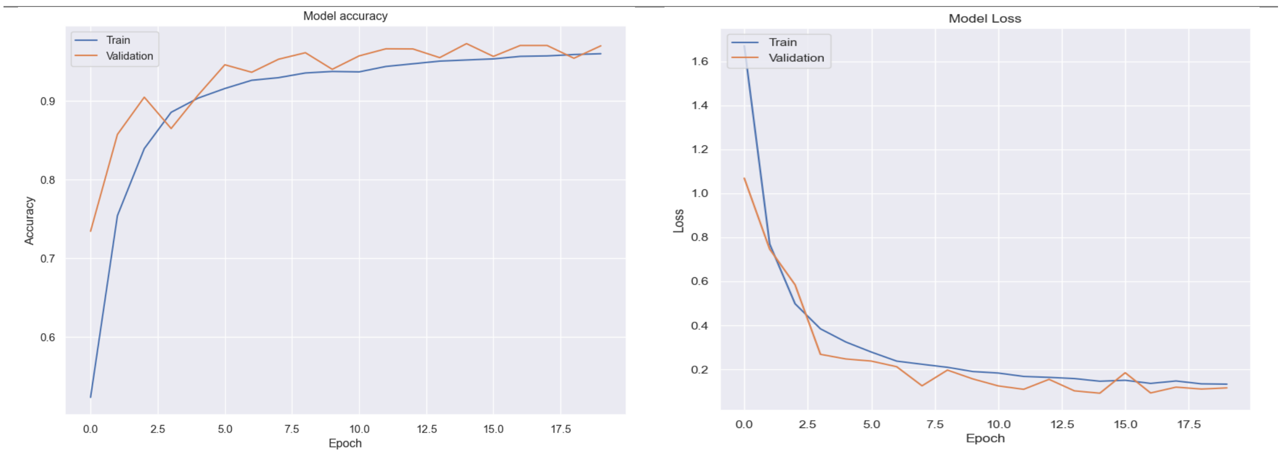


Figure 4.3: Accuracy and Loss Curve of Proposed CNN

4.2.2 ResNet152-v2 Model

ResNet152-v2 is a deep residual network that employs skip connections to alleviate the vanishing gradient problem in very deep networks. This model was chosen due to its proven effectiveness in image classification tasks. The ResNet152-v2 model was fine-tuned by replacing the top layers with a custom classifier to match the number of classes in the dataset. The model was initialized using the pre-trained weights from the ImageNet dataset, and it was then trained using similar training conditions as the proposed CNN model on the plant disease dataset. This model's output is contrasted with that of the custom CNN and other pre-trained models.

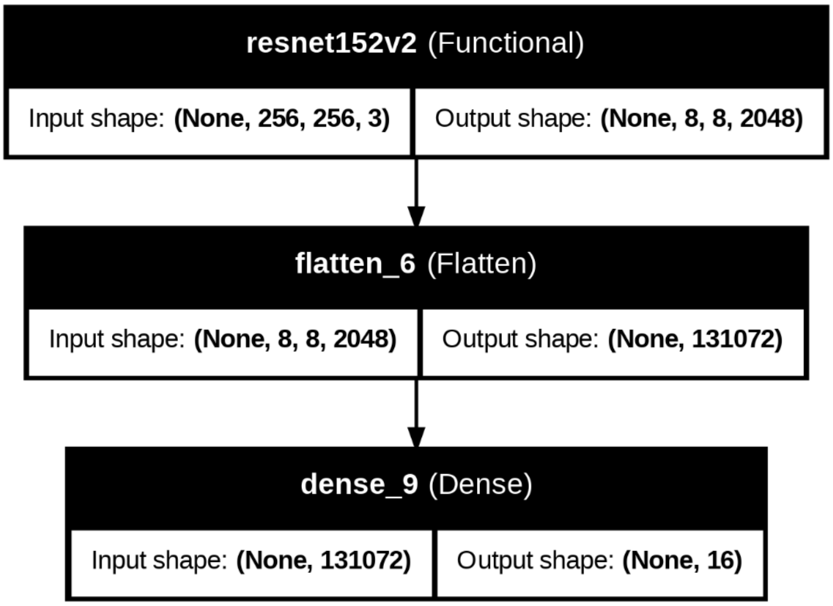


Figure 4.4: ResNet152-v2 model for plant leaf classification

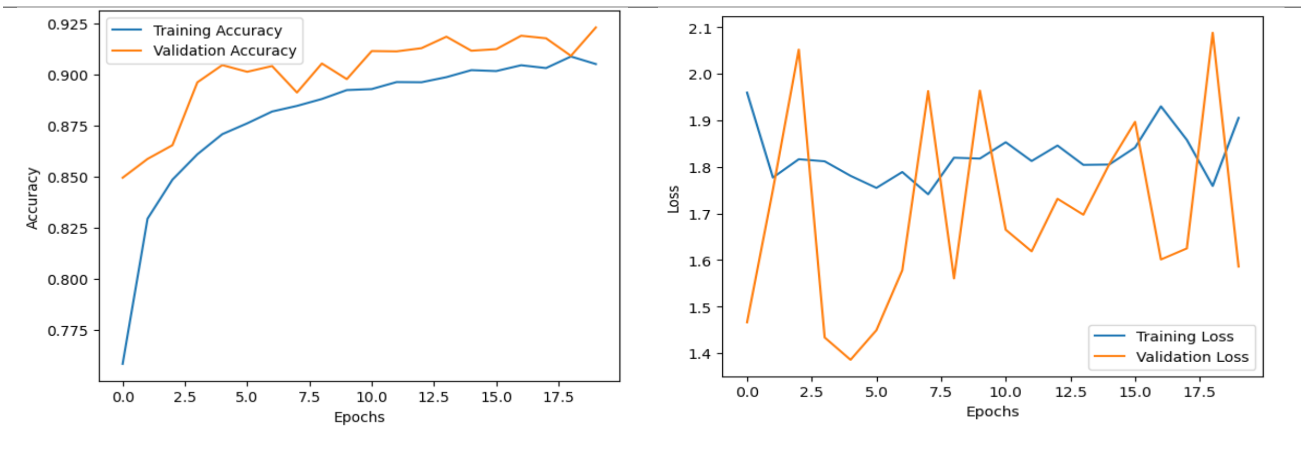


Figure 4.5: Accuracy and Loss Curve of ResNet152-v2

4.2.3 VGG16 Model

VGG16 is a popular deep learning model known for its simple yet effective architecture, consisting of 16 layers. Because of its reliable performance on a variety of datasets, the model is frequently employed for image classification tasks. Using pre-trained weights from the ImageNet dataset, VGG16 was utilized as a transfer learning model in this thesis. The top layers were changed to accommodate the plant disease dataset. The model was trained and fine-tuned, and its performance was evaluated against the proposed CNN and other pre-trained models.

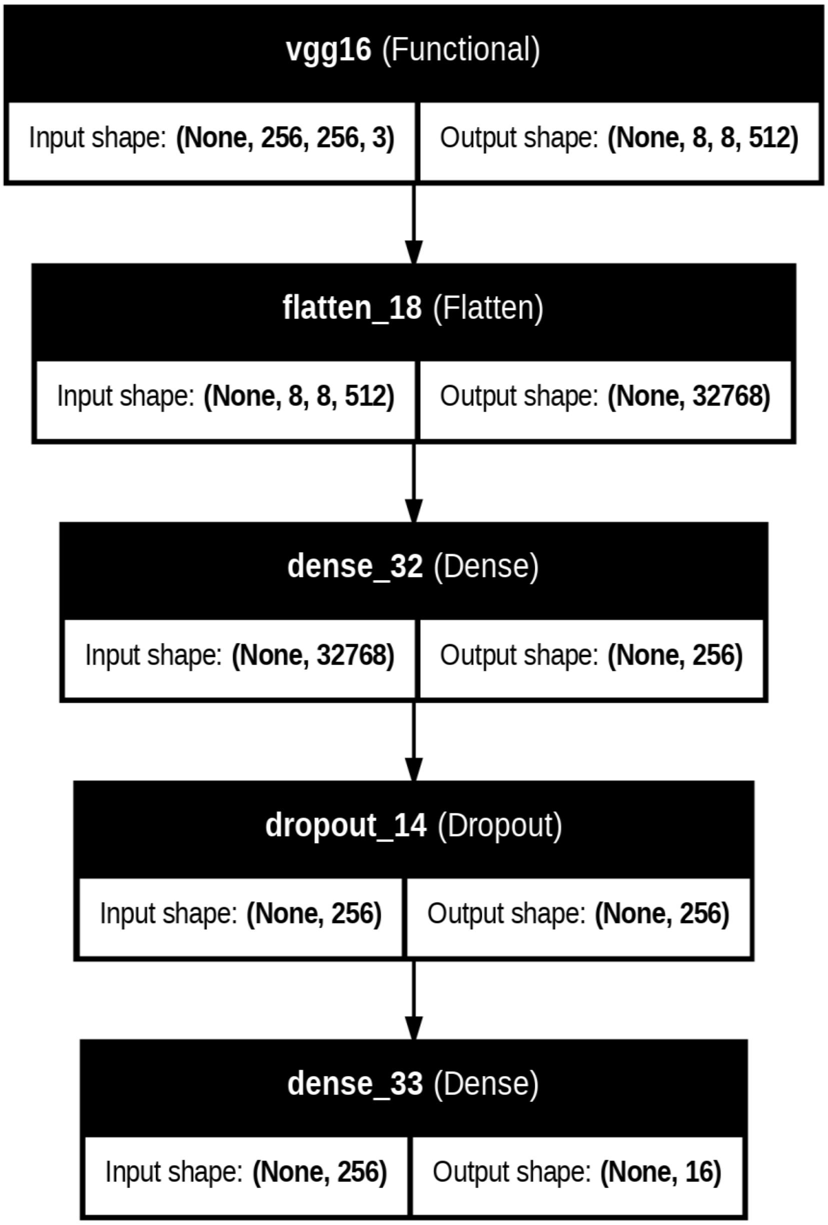


Figure 4.6: VGG16 model for plant leaf classification

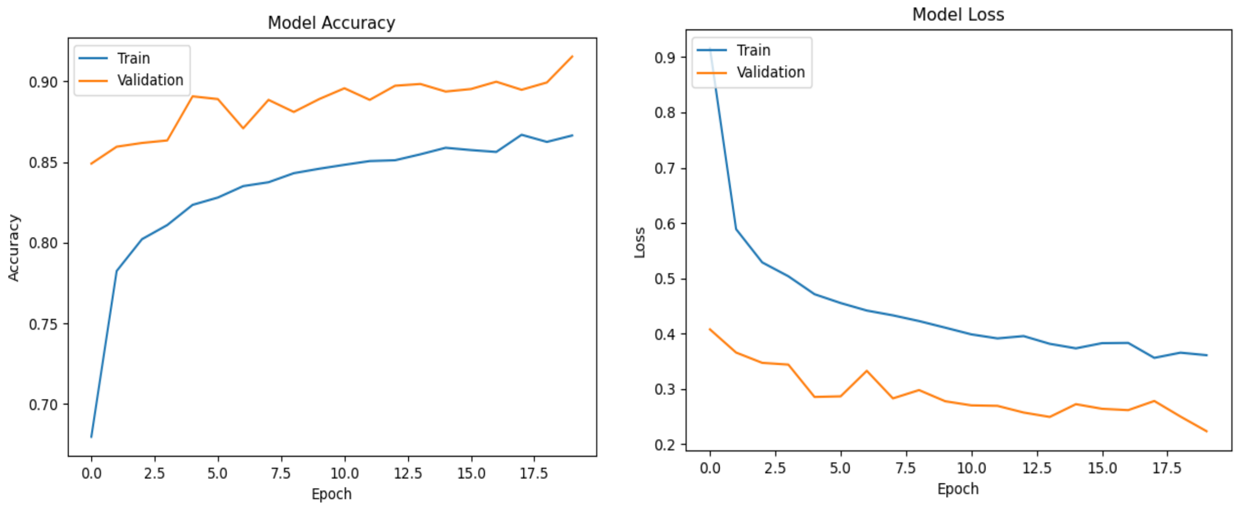


Figure 4.7: Accuracy and Loss Curve of VGG16

4.2.4 Inception-v3 Model

Inception-v3 is a deep learning model that uses inception modules to capture multi-scale information efficiently. This model is particularly known for its ability to perform well on large and complex datasets. Using pre-trained weights from the ImageNet dataset, Inception-v3 was used as a transfer learning model for this study. To match the plant disease dataset, the top layers were changed, and the model was adjusted by fine-tuning. Inception-v3's performance was compared with the other models, and its effectiveness in handling plant disease classification was analyzed.

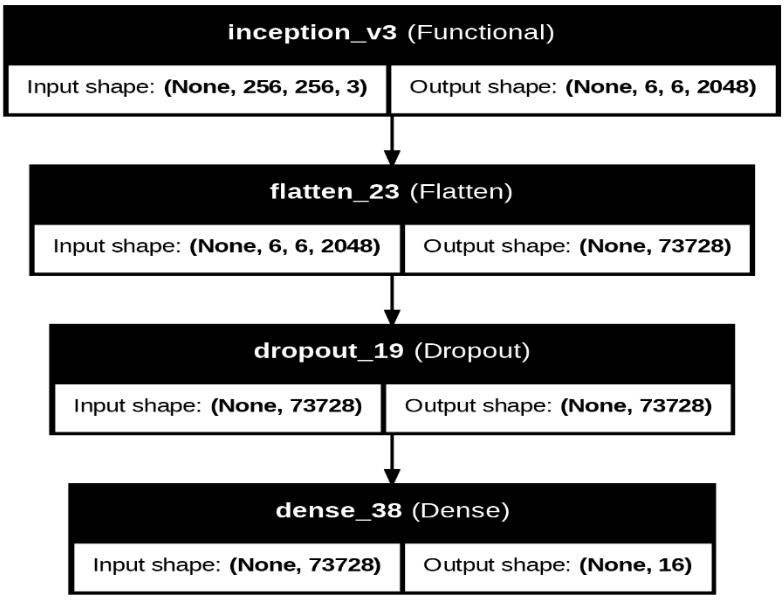


Figure 4.8: Model description of Inception-v3

Figure 4.9: Accuracy and Loss Curve of Inception-v3

Model	Train-MSE	Test-MSE	Bias	Variance
Proposed CNN	0.000138	0.000019	0.000137	0.249725
ResNet152-v2	0.041599	0.044467	0.041899	0.191598
VGG16	0.251162	0.250074	0.251162	0.000528
Inception-v3	0.128295	0.134735	0.128295	0.125323

Table 4.1 Performance characteristics of proposed CNN, ResNet151-v2, VGG16, and Inception-v3

Chapter 5

Result Analysis, Evaluation, and Discussion

This chapter presents the results obtained from the implementation of the models discussed in the previous chapter. Key performance indicators, such as recall, accuracy, precision, F1 score, and confusion matrix analysis, are used to assess each model's performance. Additionally, this chapter provides a comprehensive analysis of the Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC) for each model to assess their ability to distinguish between classes. The results are thoroughly discussed to understand the effectiveness of each model and their applicability in plant disease classification.

5.1 Result Analysis

The training and evaluation results of the proposed Convolutional Neural Network (CNN) model are shown in this part together with the transfer learning models, Inception V3, VGG16, and ResNet152-v2. Each model's performance is evaluated using a variety of metrics, including accuracy, precision, recall, and F1 score. Additionally, confusion matrices are provided to provide insight into how different classes fare in terms of classification.

5.1.1 Result Table

The following table summarizes the evaluation metrics for each model. The metrics provide a comparative overview of the effectiveness of each model in plant disease classification.

Model Names	Accuracy	Precision	Recall	F1-Score
Proposed CNN	96.11%	97%	97%	97%
Inception V3	87.92%	88%	88%	88%
Vgg16	88.64%	91%	91%	91%
ResNet152-v2	93.71%	94%	94%	94%

Table 5.1: result comparison between proposed cnn, inception-v3, vgg16, and resnet152-v2

Out of all the models, the proposed CNN model performed the best, with 96.11% accuracy, 97% F1 score, precision, and recall. These results demonstrate that the proposed CNN model effectively classifies plant diseases, showing minimal misclassifications. In contrast, the Inception V3 model, despite its robustness, performed comparatively lower with an accuracy of 87.92% and a consistent precision, recall, and F1 score of 88%. This performance could be attributed to the complexity of the Inception V3 model and its potential overfitting on certain classes. The VGG16 model achieved an accuracy of 88.64%, with precision, recall, and F1 score all at 91%. Known for its deep yet simple architecture, VGG16 performed consistently across the evaluation metrics. With an accuracy of 93.71% and steady precision, recall, and F1 score of 94%, ResNet152-v2 demonstrated impressive performance. The deep residual connections of ResNet152-v2 contributed to effective feature extraction, making it a strong candidate for this classification task.

The proposed CNN model achieved the highest overall performance, with an accuracy of 97%. Class-wise, the model excelled across most categories, such as *Apple___Black_rot*, *Tomato___Bacterial_spot*, and *Grape___healthy*, achieving near-perfect precision and recall. Specific classes, like *Tomato___Late_blight* and *Corn_(maize)___Northern_Leaf_Blight,* saw slightly lower performance, with F1 scores around 93%. Despite these minor variations, the model's overall precision, recall, and F1 score reflect its strong ability to classify plant diseases with minimal misclassifications.

The VGG16 model, while solid, achieved a slightly lower overall accuracy of 92%. It demonstrated strong performance in classes like *Corn_(maize)__Common_rust* and *Tomato___Bacterial_spot,* but had challenges with *Grape___Black_rot* and *Apple___Black_rot,* where F1 scores dropped to around 84%. The simplicity of VGG16's architecture allowed it to perform well across various classes but struggled with more complex variations.

Inception V3, while robust, showed an accuracy of 88%. Its class-wise performance was mixed, excelling in classes such as *Corn_(maize)__Common_rust* and *Tomato___healthy,* but facing difficulties in classes like *Grape___Esca_(Black_Measles)* and *Tomato___Early_blight,* where F1 scores were lower, around 77%. The complexity of the Inception V3 architecture may have led to overfitting in certain classes, contributing to the variance in performance.

ResNet152-v2 displayed strong performance with an accuracy of 94%, showcasing high precision and recall in most classes, such as *Tomato___healthy* and *Grape___Leaf_blight_(Isariopsis_Leaf_Spot)*, where it achieved F1 scores close to 98%. However, there were minor dips in performance in classes like *Corn_(maize)___Northern_Leaf_Blight* and *Apple___Black_rot*, with F1 scores in the mid-80s. The deep residual connections in ResNet152-v2 contributed to effective feature extraction, making it a top performer for this classification task.

5.1.2 Confusion Matrix and Evaluation Equations

To better understand the classification performance of each model, confusion matrices were analyzed. The confusion matrix provides detailed insight into how well the models are classifying each class and where they might be making mistakes.

The confusion matrix's structure identifies True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The examples that were accurately categorized as being in the positive class are known as True Positives. True Negatives are the examples that were accurately categorized as not being in the positive class. False Negatives are cases that were mistakenly classified as not belonging to the positive class, whereas False Positives are instances that were mistakenly classed as positive. These components of the confusion matrix are crucial for calculating evaluation metrics such as accuracy, precision, recall, and F1 score.

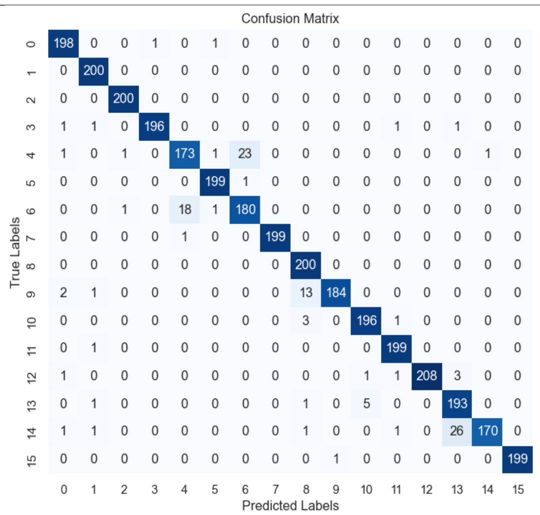


Figure 5.1: Confusion Matrix of Proposed CNN

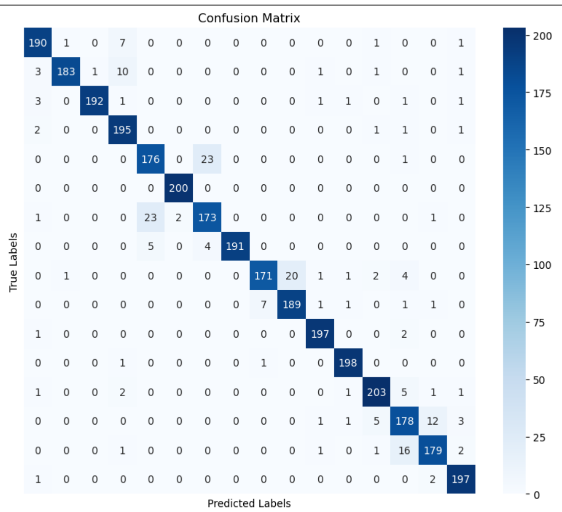


Figure 5.2: Confusion Matrix of ResNet152-v2

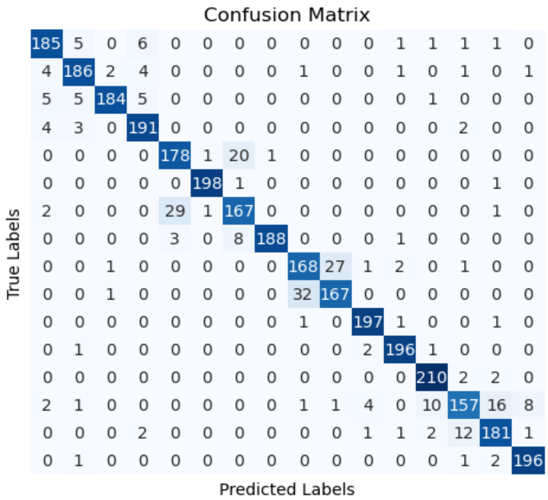


Figure 5.3: Confusion Matrix of VGG16

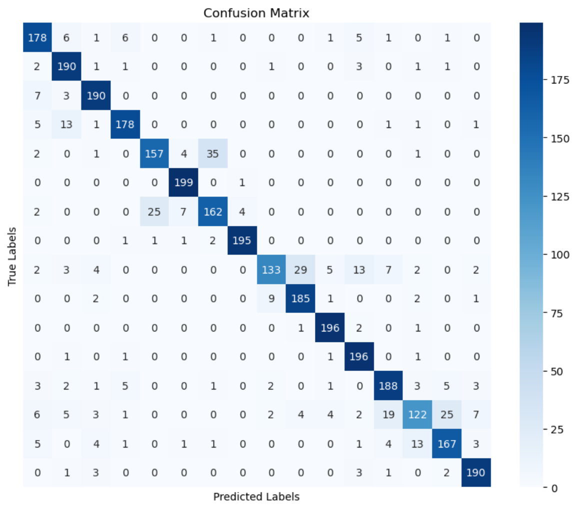


Figure 5.4: Confusion Matrix of Inception-v3

The accuracy of the model measures its overall correctness and is calculated as the ratio of correctly predicted instances (both true positives and true negatives) to the total instances. The formula for accuracy is given by:

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad \text{.......} (1)$$

The precision of a model is the ratio of true positive forecasts to all positive predictions. It is particularly important in scenarios where the cost of false positives is high. The formula for precision is:

$$Precision = \frac{TP}{\text{TP+FP}} \quad \text{.......}(2)$$

Recall, also known as sensitivity, represents the ability of the model to correctly identify positive instances. High recall is essential in cases where missing positive instances (false negatives) would have significant consequences. The formula for recall is:

$$Recall = \frac{TP}{TP+FN} \quad \text{.......}(3)$$

The F1 score is a balanced statistic that takes into account both false positives and false negatives. It is calculated as the harmonic mean of precision and recall. When precision and recall are equally critical, or when there is an unequal class distribution, it is especially helpful. The formula for the F1 score is:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

By evaluating each model using these metrics, we can assess the balance between precision and recall, the overall accuracy, and the robustness of the models in different scenarios. These assessment criteria offer a thorough grasp of each model's performance in the classification of plant diseases.

5.2 ROC and AUC Curves

The Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) are crucial metrics for evaluating the performance of classification models, particularly in multiclass problems. The trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR) across different categorization thresholds is graphically represented by the ROC curve. A model with a well-performing ROC curve will approach the top-left corner of the plot, indicating high sensitivity and low false positives. In this section, we analyze the ROC curves and AUC scores for the proposed CNN model, as well as the pre-trained models Inception-v3, VGG16, and ResNet152-v2.

For the proposed CNN model, the ROC curves for all 16 classes exhibit outstanding performance, with each class achieving an AUC of 1.00, indicating perfect discrimination. Similarly, the Inception-v3 model shows strong performance, with AUC values ranging from 0.97 to 1.00, highlighting its ability to differentiate effectively between the classes. The VGG16 model follows suit, with high AUC values between 0.99 and 1.00, demonstrating its robustness. ResNet152-v2 also performs exceptionally well, with most classes achieving an AUC of 1.00, except for a few classes where the AUC score is 0.99.

These AUC scores reflect the high discriminatory power of all the models, with the proposed CNN model standing out by achieving perfect AUC scores across all classes. The Inception-v3 model, while slightly lower, still performs at an impressive level with AUC scores nearing 1.00. The VGG16 and ResNet152-v2 models also demonstrate excellent class-wise performance with minimal variation in AUC scores. The ROC-AUC curves for these models will be presented in the following figure.

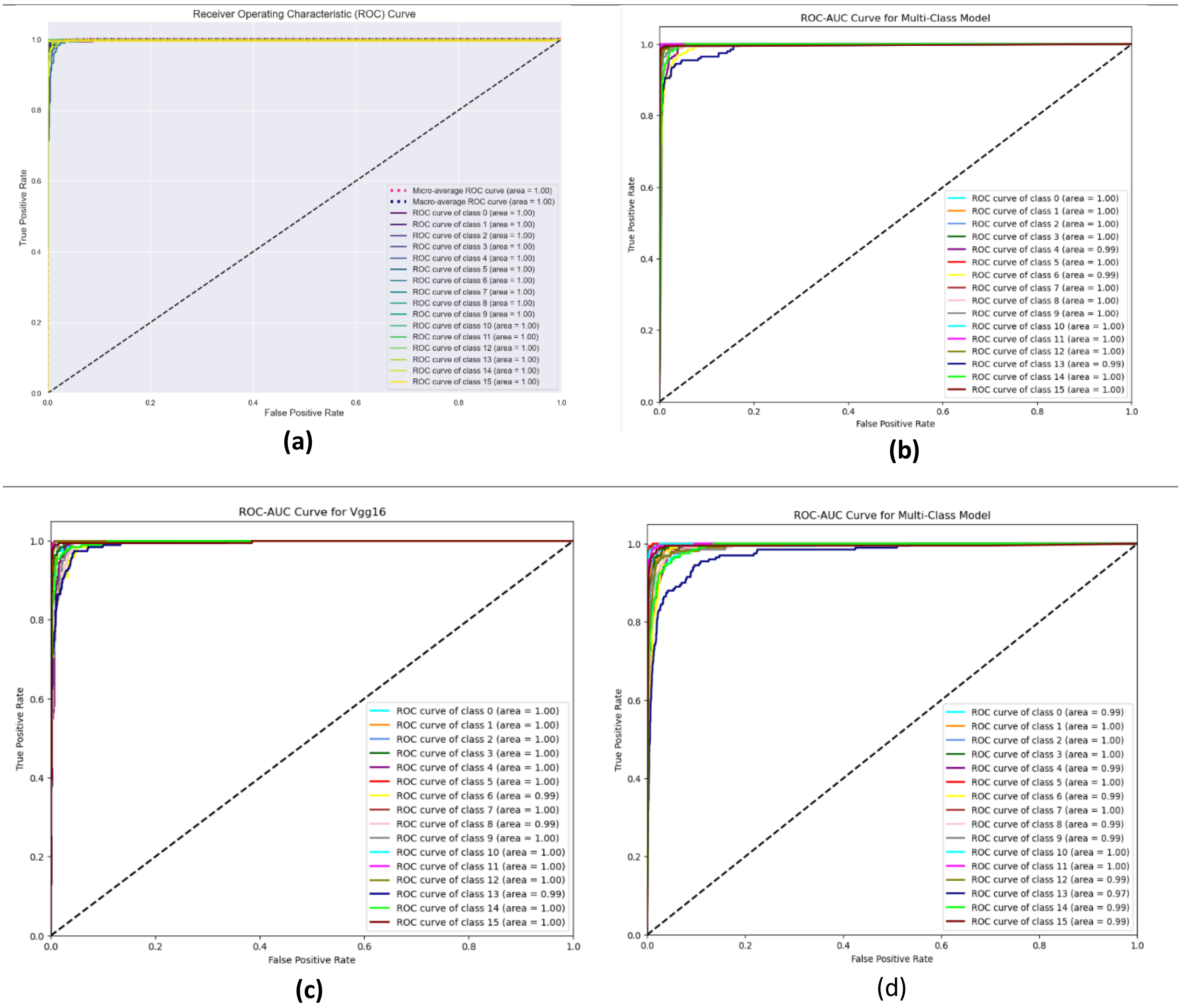


Figure 5.5: ROC-AUC Curve (a) proposed CNN, (b) ResNet152-v2, (c) VGG16, (d) Inception-v3

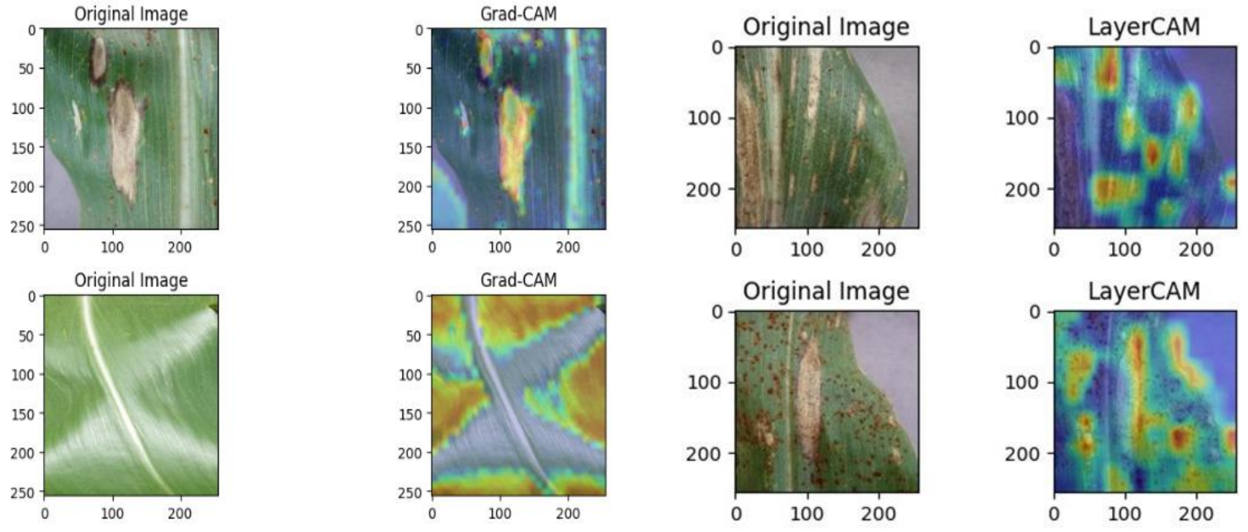
5.3 Explainable AI Results

In this section, we explore the application of Explainable AI techniques to visualize and interpret the model's predictions. Through the use of visualization techniques like Grad-CAM and Layer-CAM, We are able to determine which portions of the input images have the greatest impact on the model's conclusions. By using these techniques, we may better comprehend the deep learning models' interpretability and identify the model's emphasis areas during classification.

The Grad-CAM method, which stands for Gradient-weighted Class Activation Mapping, produces heat maps that emphasize the significant areas inside the input image that impacted the model's prediction. It provides a coarse localization map that shows where the model is looking when it makes a decision. While Grad-CAM proved useful in interpreting the model's decisions, the generated heatmaps were relatively broader and less focused in certain instances.

Layer-CAM, on the other hand, utilizes the outputs of individual convolutional layers to generate more refined and detailed heat maps. This approach enables better visualization of the specific areas the model is focusing on at various stages of the decision-making process. Through our findings, Layer-CAM consistently provided more accurate and focused visualizations compared to Grad-CAM, revealing more precise regions of interest that contributed to the model's classification decisions.

Upon comparing the results of both techniques, Layer-CAM demonstrated superior performance in terms of generating clearer and more informative visual explanations. This finding suggests that Layer-CAM is better suited for interpreting the model's predictions in the context of plant disease classification.



(a) **(b)**

Figure 5.6: explainable AI visualization (a) Grad-CAM, (b) Layer-CAM

5.4 Discussion

This section provides a detailed examination of the experimental findings, paying close attention to how well the proposed CNN model performed. and the transfer learning models, Inception-v3, VGG16, and ResNet152-v2. The comparison between these models highlights the strengths and limitations of each approach and their applicability in real-world plant disease classification. The results indicate that transfer learning models, especially Inception-v3, significantly outperform the custom CNN model across various evaluation metrics, including accuracy and ROC-AUC scores.

Inception-v3's superior performance is largely attributable to its pre-trained knowledge from large-scale datasets like ImageNet, which enables it to generalize more effectively to new and unseen data. The custom CNN model, Although it performs competitively, it is not as well as the transfer learning models, which use pre-trained weights to gain more accuracy and improved class discriminating. Moreover, explainable AI techniques, such as Grad-CAM and Layer-CAM, were applied to provide visual interpretations of the models' predictions. The findings reveal that Layer-CAM consistently provided more precise and informative visualizations compared to Grad-CAM. The Layer-CAM visualizations offered clearer insights into the specific regions of the images that influenced the model's decisions, making it a more effective tool for model interpretability in this context. These visualization techniques enhance the transparency of the deep learning models, enabling a better understanding of how they arrive at their predictions.

In terms of practical applications, the ability to accurately classify plant diseases has far-reaching implications for agriculture. Early and accurate diagnosis of plant diseases can significantly improve crop management, reduce losses due to disease outbreaks, and ultimately boost agricultural productivity. The deployment of these models in mobile or cloud-based systems could provide farmers with real-time diagnostics and actionable recommendations, empowering them to make informed decisions in managing crop health. This emphasizes the potential for integrating AI-driven solutions into agricultural practices, contributing to more sustainable and efficient farming methods. Overall, while the proposed CNN model demonstrated promising results, Plant disease classification applications in the actual world might benefit more from the increased performance and robustness of the transfer learning models, especially Inception-v3. The application of these models is further strengthened by the incorporation of explainable AI techniques, which not only give high accuracy but also visible and interpretable results—a crucial component in building trust in AI-driven agricultural systems.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This research has made significant strides in enhancing plant disease detection through deep-learning approaches. The research centered on creating a customized Convolutional Neural Network (CNN) model and assessing its efficacy in comparison to many well-known transfer learning models, including Inception V3, VGG16, and ResNet152-v2. The accuracy of the suggested CNN model was a noteworthy 96.11%, and its precision, recall, and F1-score all reached 97%. This excellent result highlights how well the customized CNN classified plant diseases from the Plant Village dataset. In comparison, the transfer learning models demonstrated varying levels of performance. Inception V3 achieved an accuracy of 87.92%, with precision, recall, and F1-score all at 88%. VGG16 performed slightly better with an accuracy of 88.64% and precision, recall, and F1-score of 91%. ResNet152-v2 exhibited strong results with an accuracy of 93.71%, and precision, recall, and F1-score all at 94%. These findings illustrate that the proposed CNN model not only outperforms the transfer learning models in accuracy but also in precision, recall, and F1-score. This implies that the customized CNN model is a useful tool for plant disease classification since it is quite good at differentiating between various plant disease classes. The advantages of deep learning for plant disease identification were also emphasized by the study. By leveraging advanced neural network architectures, the study has demonstrated how deep learning can significantly improve diagnostic accuracy and efficiency. The ability to accurately identify plant diseases can lead to better crop management, reduced losses, and improved food security. The use of Explainable AI techniques, such as Layer-CAM and Grad-CAM, provided further insights into model predictions, with Layer-CAM offering more precise visualizations. This added interpretability is crucial for practical applications, as it helps users understand and trust the model’s decisions. Overall, the research confirms that deep learning techniques, particularly custom CNN models, offer substantial advantages in plant disease classification. The ability to achieve high accuracy and provide actionable insights paves the way for more effective and real-time disease management solutions, benefiting farmers and contributing to global agricultural practices.

6.2 Future Work

While this research has achieved promising results in plant disease classification using deep learning, several avenues for future work remain to be explored to further enhance model performance and applicability. Expanding the collection to encompass a wider range of plant species and diseases is a crucial subject for future research. The current study utilized a subset of the Plant Village dataset, which, while comprehensive, may not cover all possible plant diseases. By incorporating more diverse and extensive datasets, models can be trained to generalize better across different plant types and conditions. Another potential improvement could be the incorporation of additional advanced neural network architectures.

Even while the suggested CNN model and the transfer learning models employed in this investigation performed well, investigating more modern architectures like Vision Transformers (ViTs) or sophisticated hybrid models may produce even better outcomes. Data augmentation techniques could also be enhanced. Although the study employed various augmentation methods to increase dataset diversity, experimenting with more sophisticated techniques, such as generative adversarial networks (GANs) for image synthesis, could improve model robustness and performance. Additionally, integrating multi-modal data, such as combining image data with environmental factors (e.g., soil moisture, weather conditions) and other sensor data, could lead to a more comprehensive disease detection system.

This integration would enable models to leverage a wider range of inputs for more accurate and context-aware predictions. Further exploration into model interpretability is also crucial. While Layer-CAM provided valuable insights, developing and applying other explainable AI techniques could enhance understanding of model decisions and improve trust in automated systems. This could include refining visualization methods or incorporating user feedback mechanisms to continuously improve the model's interpretability.

Finally, deploying these models in real-world scenarios and evaluating their performance in practical settings is essential. Implementing the models in mobile or cloud-based applications for real-time disease detection could provide valuable feedback on their usability and effectiveness in agricultural practices. Pilot studies or field trials could offer practical insights and guide further refinements. In summary, future work should focus on expanding datasets, exploring advanced architectures, enhancing data augmentation, integrating multi-modal data, improving model interpretability, and conducting real-world evaluations. Plant disease classification will benefit from these efforts as more reliable, accurate, and useful solutions are developed.

Reference

[1] M. Albahar, "A survey on deep learning and its impact on agriculture: Challenges and opportunities," Agriculture, vol. 13, no. 3, p. 540, 2023.

[2] T. Islam, T. A. Chisty, and A. Chakrabarty, "A deep neural network approach for crop selection and yield prediction in bangladesh," in 2018 IEEE region 10 humanitarian technology conference (R10-HTC), pp. 1– 6, IEEE, 2018.

[3] H. Orchi, M. Sadik, and M. Khaldoun, "On using artificial intelligence and the internet of things for crop disease detection: A contemporary survey," Agriculture, vol. 12, no. 1, p. 9, 2021.

[4] T. Islam, T. A. Chisty, and A. Chakrabarty, "A deep neural network approach for crop selection and yield prediction in bangladesh," in 2018 IEEE region 10 humanitarian technology conference (R10-HTC), pp. 1– 6, IEEE, 2018.

[5] L. Li, S. Zhang, and B. Wang, "Plant disease detection and classification by deep learning—a review," IEEE Access, vol. 9, pp. 56683–56698, 2021.

[6] Tugrul, B.; Elfatimi, E.; Eryigit, R. Convolutional Neural Networks in Detection of Plant Leaf Diseases: A Review. Agriculture 2022, 12, 1192. https://doi.org/10.3390/agriculture12081192

[7] Shafik, W., Tufail, A., De Silva Liyanage, C. et al. Using transfer learning-based plant disease classification and detection for sustainable agriculture. BMC Plant Biol 24, 136 (2024). https://doi.org/10.1186/s12870-024-04825-y

[8] A. Jafar, N. Bibi, R. A. Naqvi, A. Sadeghi-Niaraki, and D. Jeong, "Revolutionizing agriculture with artificial intelligence: plant disease detection methods, applications, and their limitations," Frontiers in Plant Science, vol. 15, p. 1356260, 2024.

[9] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition.

[10] A. Jafar, N. Bibi, R. A. Naqvi, A. Sadeghi-Niaraki, and D. Jeong, "Revolutionizing agriculture with artificial intelligence: plant disease detection methods, applications, and their limitations," Frontiers in Plant Science, vol. 15, p. 1356260, 2024.

[11] Karnan, A., Ragupathy, R. (2024). A Comprehensive Study on Plant Classification Using Machine Learning Models. In: Kaiser, M.S., Xie, J., Rathore, V.S. (eds) ICT: Smart Systems and Technologies. ICTCS 2023. Lecture Notes in Networks and Systems, vol 878. Springer, Singapore. https://doi.org/10.1007/978-981-99-9489-2_17

[12] M. Shoaib, B. Shah, S. Ei-Sappagh, A. Ali, A. Ullah, F. Alenezi, T. Gechev, T. Hussain, and F. Ali, "An advanced deep learning modelsbased plant disease detection: A review of recent research," Frontiers in Plant Science, vol. 14, p. 1158933, 2023.

[13] CABI. *Global Burden of Crop Loss*. 2020. [Global Burden of Crop Loss - CABI.org]

[14] S. M. Hassan, A. K. Maji, M. Jasinski, Z. Leonowicz, and E. Jasi ´ nska, ´ "Identification of plant-leaf diseases using cnn and transfer-learning approach," Electronics, vol. 10, no. 12, p. 1388, 2021.

[15] He, Kaiming, et al. "Deep Residual Learning for Image Recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi:10.48550/arXiv.1512.03385.

[16] Sherly, K.K., Sonia, A. (2024). Hybrid CNN Models for Plant Species Recognition and Disease Detection. In: Arai, K. (eds) Intelligent Computing. SAI 2024. Lecture Notes in Networks and Systems, vol 1016. Springer, Cham. https://doi.org/10.1007/978-3-031-62281-6_3

[17] Qiang, Z., He, L., Dai, F. (2019). Identification of Plant Leaf Diseases Based on Inception V3 Transfer Learning and Fine-Tuning. In: Wang, G., El Saddik, A., Lai, X., Martinez Perez, G., Choo, KK. (eds) Smart City and Informatization. iSCI 2019. Communications in Computer and Information Science, vol 1122. Springer, Singapore. https://doi.org/10.1007/978-981-15-1301-5_10

[18] Sagar, S., Javed, M., & Doermann, D. S. (2023). Leaf-Based Plant Disease Detection and Explainable AI. *arXiv preprint arXiv:2404.16833*. https://doi.org/10.48550/arXiv.2404.16833

[19] P.-T. Jiang, C.-B. Zhang, Q. Hou, M.-M. Cheng, and Y. Wei, "Layercam: Exploring hierarchical class activation maps for localization," IEEE Transactions on Image Processing, vol. 30, pp. 5875–5888, 2021.

[20] "PlantVillage Dataset", Dataset of diseased plant leaf images and corresponding labels. https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset

[21] Chen, Ruoling, Ming Tao, Hang Xing, Xiuli Yang, Chuang Liu, and Long Qi. "Automatic Diagnosis of Rice Diseases Using Deep Learning." *Frontiers in Plant Science* 12 (2020): 701038. https://doi.org/10.3389/fpls.2021.701038

[22] C. K. Sunil, C. D. Jaidhar, and N. Patil. Binary class and multi-class plant disease detection using ensemble deep learning-based approach. International Journal of Sustainable Agricultural Management and Informatics, 8(4):385–407, 2022.

[23] S. Gayathri, D. J. W. Wise, P. B. Shamini, and N. Muthukumaran. Image analysis and detection of tea leaf disease using deep learning. In 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), pages 398–403. IEEE, July 2020.

[24] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang. Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks. IEEE Access, 7:59069–59080, 2019.

[25] Sinan Uğuz and Nese Uysal. Classification of olive leaf diseases using deep convolutional neural networks. Neural computing and applications, 33(9):4133–4149, 2021.

[26] Datta, S. (2020). A Review on Convolutional Neural Networks. In: Bera, R., Pradhan, P.C., Liu, CM., Dhar, S., Sur, S.N. (eds) Advances in Communication, Devices and Networking. ICCDN 2019. Lecture Notes in Electrical Engineering, vol 662. Springer, Singapore. https://doi.org/10.1007/978-981-15-4932-8_50

[27] Renner, A., Sheldon, F., Zlotnik, A., Tao, L., & Sornborger, A. (2020). The Backpropagation Algorithm Implemented on Spiking Neuromorphic Hardware. *Frontiers in Neuroscience*, 14, 138. doi:10.3389/fnins.2020.00138

[28] C. Sunil, C. Jaidhar, and N. Patil, “Binary class and multi-class plant disease detection using ensemble deep learning-based approach,” International Journal of Sustainable Agricultural Management and Informatics, vol. 8, no. 4, pp. 385–407, 2022

[29] Schmidt, R. M. (2019). Recurrent Neural Networks (RNNs): A Gentle Introduction and Overview. *arXiv preprint arXiv:1904.01383*.

Appendix

Proposed CNN

```
from tensorflow.keras import layers
from tensorflow.keras import models
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(256, 256, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(layers.Dense(16, activation='softmax'))
# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
# Train the model with data augmentation
history = model.fit(
    train_generator,
    steps_per_epoch=len(input_train) // 32,
    epochs=20,
    validation_data=valid_generator,
    validation_steps=len(input_valid) // 32
)
```

ResNet152-v2

```
from tensorflow.keras.applications import ResNet152V2
# Load the ResNet152V2 model, excluding the top layers
resnet152v2_base = ResNet152V2(include_top=False, input_shape=(128, 128, 3))
# Freeze the base model layers so they won't be updated during training
resnet152v2_base.trainable = False
# Build the classifier on top of the base model
resnet152v2_classifier = Sequential()
resnet152v2_classifier.add(resnet152v2_base)
resnet152v2_classifier.add(Flatten())
resnet152v2_classifier.add(Dense(16, activation='softmax'))
# Compile the model
resnet152v2_classifier.compile(optimizer='adam',
                               loss='categorical_crossentropy',
                               metrics=['accuracy'])
# Train the ResNet152V2 model
history_resnet152v2 = resnet152v2_classifier.fit(
    train_generator,
    steps_per_epoch=len(input_train) // 32,
    epochs=20,
    validation_data=valid_generator,
    validation_steps=len(input_valid) // 32
)
```

VGG16

```
import tensorflow as tf
from tensorflow.keras.applications import VGG16
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers, models, optimizers
from tensorflow.keras.layers import Flatten, Dense, Dropout
classifier=Sequential()
classifier.add(base_model)
classifier.add(Flatten())
classifier.add(layers.Dense(256, activation='relu'))
classifier.add(Dropout(0.2))
classifier.add(Dense(16,activation='softmax'))
classifier.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
# Train the model with data augmentation
history = classifier.fit(
    train_generator,


    steps_per_epoch=len(input_train) // 32,
    epochs=20,
    validation_data=valid_generator,
    validation_steps=len(input_valid) // 32
)
```

Inception-v3

```
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.layers import Dropout, Flatten, Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# Load the InceptionV3 model, excluding the top layers
inception_base = InceptionV3(include_top=False, input_shape=(128, 128, 3))
# Freeze the base model layers so they won't be updated during training
inception_base.trainable = False
# Build the classifier on top of the base model
inception_classifier = Sequential()
inception_classifier.add(inception_base)
inception_classifier.add(Flatten())
inception_classifier.add(Dropout(0.2))  # Add dropout layer with 0.2 dropout rate
inception_classifier.add(Dense(16, activation='softmax'))
# Compile the model
inception_classifier.compile(optimizer='adam',
                             loss='categorical_crossentropy',
                             metrics=['accuracy'])
# Train the InceptionV3 model
history_inception = inception_classifier.fit(
    train_generator,
    steps_per_epoch=len(input_train) // 32,
    epochs=20,
    validation_data=valid_generator,
    validation_steps=len(input_valid) // 32
)
```