Edge-Deployable Lightweight CNN Ensembles for Real-Time Soursop Disease Detection in Smart Farming

By

Md. Arif Billah
221-15-5771

FINAL YEAR DESIGN PROJECT REPORT

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

Supervised by

Mr. Abdus Sattar
Associate Professor
Department of Computer Science and EngineeringDaffodil International University

Co-Supervised by

Md. Sazzadur Ahamed
Assistant Professor
Department of Computer Science and EngineeringDaffodil International University

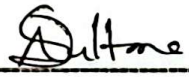
DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

May 14, 2025

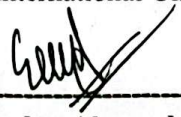
APPROVAL

This Project titled "**Edge-Deployable Lightweight CNN Ensembles for Real-Time Soursop Disease Detection in Smart Farming**", submitted by **Md.Arif Billah** , ID No: **221-15-5771** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **14 May, 2025**.

BOARD OF EXAMINERS



Dr. Naznin Sultana — **Chairman**
Associate Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University



Md. Sazzadur Ahamed — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

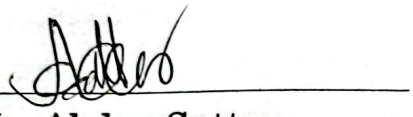
Raja Tariqul Hasan Tusher — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Shah Md Imran — **External Examiner**
Project Manager, External Member
Smart Leadership Academy and EDGE Project
Bangladesh Computer Council, ICT Division

DECLARATION

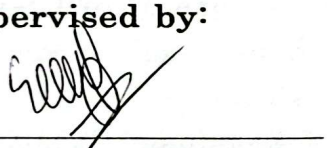
We hereby declare that this project has been done by us under the supervision of **Mr. Abdus Sattar, Associate Professor** Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this projectnor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:



Mr. Abdus Sattar
Associate Professor
Department of Computer Science and EngineeringDaffodil International University

Co-Supervised by:



Md. Sazzadur Ahamed
Assistant Professor
Department of Computer Science and Engineering Daffodil International University

Submitted by:

MD. Arif Billah

Md. Arif Billah
Student ID: 221-15- 5771
Department of Computer Science and Engineering
Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project (FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Mr. Abdus Sattar, Associate Professor** Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Deep Learning** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting themat all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

The accurate and early detection of plant diseases is critical for ensuring agricultural productivity and sustainability, particularly for medicinal plants like Soursop (Annona muricata), which holds significant therapeutic value. Traditional methods for disease identification are often labor-intensive, subjective, and inaccessible to farmers in remote areas. To address these challenges, this study proposes an edge-deployable lightweight ensemble model combining multiple Convolutional Neural Network (CNN) architectures for real-time Soursop leaf disease detection. A comprehensive methodology was adopted, involving data collection, preprocessing, model training, ensemble fusion, and TensorFlow Lite (TFLite) optimization for mobile deployment. Four state-of-the-art CNN models—VGG19, ResNet101, InceptionV3, and DenseNet201—were individually trained and evaluated. The experimental results demonstrate that VGG19, InceptionV3, and DenseNet201 achieved outstanding validation accuracies of 99.84%–100%, with high AUC scores of 1.00, indicating excellent model performance. In contrast, ResNet101 underperformed, achieving only 32.90% validation accuracy despite a high AUC score, highlighting issues of overfitting or dataset mismatch. The ensemble model, developed through a soft-voting strategy, achieved a perfect 100% validation and testing accuracy, with an AUC score of 1.00, outperforming all individual models. This lightweight ensemble approach ensures high classification performance while maintaining low computational complexity, making it suitable for real-time applications in smart farming environments. The final system is optimized for mobile devices, enabling farmers to perform offline disease detection efficiently without dependency on cloud services. Overall, the study demonstrates a robust, scalable, and sustainable solution for advancing precision agriculture, empowering farmers with accessible AI-driven tools to improve crop health monitoring and preserve valuable medicinal plant resources.

Keywords: Soursop Leaf Disease; Convolutional Neural Network (CNN); Ensemble Learning; TensorFlow Lite (TFLite); Edge Deployment; Smart Farming.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1 introduces the background and significance of plant disease detection, with a specific focus on Soursop leaf diseases. It highlights the limitations of traditional disease identification methods, and the advancements made through deep learning, particularly CNNs, for automated, real-time solutions. The motivation emphasizes the urgent need for lightweight, edge-deployable models suitable for mobile and smart farming applications. The objectives clearly define the goals of the research, including dataset preparation, model development, ensemble learning, and mobile deployment. The methodology outlines the structured process followed, from data collection and preprocessing to model training, ensemble creation, and optimization. Finally, the project outcome summarizes the expected deliverables, including a high-performing lightweight CNN ensemble and a mobile-compatible disease detection system aimed at supporting farmers and enhancing medicinal plant conservation.

1.1 Introduction

Agriculture continues to serve as the backbone of many economies and plays a vital role in ensuring food security, environmental sustainability, and the availability of natural medicinal resources across the world. Among these resources, medicinal plants hold immense value due to their contributions to traditional and modern healthcare systems. However, plant diseases pose a significant threat to agricultural productivity and the preservation of medicinal flora, resulting in decreased yields, reduced medicinal quality, and severe economic losses if not addressed in a timely manner [1]. Early detection and diagnosis of plant diseases are critical for effective management, but conventional methods—such as manual visual inspection or laboratory testing—are often time-consuming, labor-intensive, costly, and susceptible to human error [2]. Moreover, these traditional methods are frequently inaccessible to farmers in rural and resource-limited regions, where the need for efficient disease monitoring is the greatest.

With the rapid advancement of technology, artificial intelligence (AI) and deep learning have emerged as transformative solutions to overcome the limitations of manual plant disease detection. Among various deep learning techniques, Convolutional Neural Networks (CNNs) have shown exceptional ability in automatically extracting hierarchical features from plant images and achieving high classification accuracies [3]. CNN models are capable of learning intricate visual patterns associated with different plant diseases, enabling more consistent, rapid, and accurate identification compared to traditional methods. As a result, deep learning-based plant disease detection systems have become a prominent research focus in agricultural technology.

Despite these advancements, one critical challenge remains: most state-of-the-art CNN models are large, computationally heavy, and resource-intensive [4]. Their practical deployment is often restricted to high-end servers or cloud environments, making them unsuitable for real-time applications on edge devices such as smartphones, drones, or embedded sensors. For farming communities, especially in developing regions, there is an urgent need for lightweight, efficient, and real-time disease detection solutions that can operate under constrained hardware environments [5].

Soursop (Annona muricata), commonly known for its significant medicinal value—including anti-cancer, anti-bacterial, and anti-inflammatory properties—is a prime example of a plant species requiring immediate attention [6]. However, there is a noticeable lack of research focusing specifically on automated disease detection for Soursop plants. Most existing studies have either generalized plant disease detection across multiple crops or have not optimized models for real-time field deployment. Furthermore, while some lightweight models exist, few studies have successfully balanced the trade-off between model complexity, accuracy, and deployment feasibility on mobile devices [7].

Recognizing these challenges and opportunities, this thesis aims to address the gap by proposing an edge-deployable, lightweight CNN ensemble tailored for the real-time detection of Soursop diseases. The proposed system focuses not only on

maximizing classification accuracy but also on ensuring minimal computational overhead, enabling easy integration with mobile applications and smart farming systems. Through this work, it is expected that farmers and agricultural stakeholders will gain access to an affordable, efficient, and accurate tool to monitor Soursop plant health, ultimately contributing to sustainable farming and medicinal plant conservation.

1.2 Motivation

The motivation behind this research stems from the increasing need for efficient, accurate, and accessible plant disease detection systems, particularly for medicinal plants like Soursop (Annona muricata). Soursop has gained global recognition for its potent medicinal properties, including anti-cancer, anti-bacterial, anti-inflammatory, and anti-viral effects, making it a critical resource in both traditional and modern medicinal practices [6]. However, the health and productivity of Soursop plants are often compromised by various leaf diseases that, if left undetected, can significantly reduce the therapeutic efficacy of the plant and lead to substantial agricultural losses [7].

Current disease management practices predominantly rely on manual visual inspection by farmers or agricultural experts, a process that is prone to human error, subjectivity, and delays in response time [1]. Moreover, laboratory-based confirmation of plant diseases, although accurate, is expensive, time-consuming, and generally inaccessible to farmers in rural or economically disadvantaged regions [2]. These challenges highlight the urgent need for automated, real-time disease detection systems that can function efficiently in resource-constrained environments.

Deep learning, especially Convolutional Neural Networks (CNNs), has shown great potential in automating plant disease detection with remarkable accuracy [3]. However, the practical deployment of these models in real-world agricultural settings remains limited due to the large model sizes, high computational demands, and dependence on powerful hardware infrastructure [4]. There is a clear mismatch between the high computational requirements of state-of-the-art models and the limited resources available to farmers and agricultural workers on the ground [5].

Motivated by these challenges, this study seeks to bridge the gap by proposing a lightweight, ensemble-based CNN framework specifically tailored for the real-time detection of Soursop leaf diseases. The focus is not only on achieving high classification accuracy but also on ensuring that the developed model is computationally efficient and suitable for edge deployment, such as on smartphones, handheld devices, or portable monitoring tools. Additionally, by concentrating on Soursop—a medicinally important but under-researched plant—this study contributes to the broader goals of preserving medicinal biodiversity, supporting traditional medicine practices, and promoting precision agriculture in developing regions.

Ultimately, the motivation is to empower farmers and agricultural practitioners with an affordable, reliable, and easy-to-use disease detection tool, enabling timely interventions, minimizing losses, and enhancing both agricultural productivity and the medicinal quality of Soursop crops.

1.3 Objectives

The primary objective of this research is to develop an edge-deployable, lightweight Convolutional Neural Network (CNN) ensemble model for real-time detection and classification of Soursop leaf diseases. Recognizing the limitations of traditional disease detection methods and the computational challenges posed by complex deep learning models [1], this study aims to bridge the gap by designing a system that is both highly accurate and computationally efficient, suitable for use in mobile and low-resource environments.

The specific objectives of this research are outlined below:

- To organize and preprocess a high-quality image dataset of Soursop leaf diseases, ensuring sufficient diversity, balanced class distribution, and readiness for deep learning model training and evaluation.
- To implement and train multiple state-of-the-art CNN architectures, including VGG19, DenseNet201, ResNet101, and InceptionV3, evaluating their individual performances in classifying Soursop leaf diseases.

- To design a soft-voting ensemble strategy that combines the strengths of individual CNN models, enhancing overall classification robustness, reducing model bias, and improving prediction stability.
- To optimize the ensemble model for lightweight deployment by converting it into a TensorFlow Lite (TFLite) format, ensuring it can run efficiently on mobile devices and embedded systems without sacrificing accuracy or speed.

Through these objectives, the research not only aims to contribute a technically robust solution for agricultural disease monitoring but also addresses broader societal needs by promoting medicinal plant conservation, empowering rural farming communities, and advancing the practical use of artificial intelligence in sustainable agriculture.

1.4 Methodology

The methodology adopted in this research is carefully structured to ensure the successful development of a lightweight, edge-deployable CNN ensemble model for real-time Soursop leaf disease detection. The approach is divided into several interdependent stages, each playing a crucial role in achieving the research objectives outlined earlier.

The first step involves the collection and preparation of a primary dataset comprising images of healthy and diseased Soursop leaves. The images are acquired from real-world sources to capture natural variability in lighting, background, and disease progression stages. Following collection, the dataset undergoes preprocessing steps such as resizing, normalization, noise reduction, and augmentation to enhance diversity and improve the generalization capabilities of the models [2]. Data augmentation techniques such as flipping, rotation, scaling, and contrast adjustment are employed to simulate different environmental conditions and prevent model overfitting.

In the second stage, model development and training are conducted using four state-of-the-art CNN architectures: VGG19, DenseNet201, ResNet101, and InceptionV3. These models are chosen due to their proven effectiveness in

various plant disease detection studies [3][4]. Each model is fine-tuned on the prepared Soursop dataset, with transfer learning techniques applied to leverage pre-trained weights and accelerate convergence while improving performance on relatively smaller datasets.

The third stage focuses on ensemble learning. A soft-voting ensemble strategy is designed to combine the outputs of the four individual CNN models [5]. In soft voting, each model's predicted probability distribution across classes is aggregated, and the final class prediction is made based on the maximum average probability. This strategy is employed to enhance classification accuracy and ensure robustness by reducing the variance and bias associated with any single model.

To make the model suitable for real-time, edge deployment, the best-performing ensemble model is converted into TensorFlow Lite (TFLite) format [7]. TFLite conversion optimizes the model by reducing its size and computational requirements, enabling fast inference on resource-constrained devices such as smartphones, tablets, and IoT-enabled agricultural sensors.

Finally, model evaluation is carried out using various performance metrics, including overall accuracy, precision, recall, F1-score, and execution time. These metrics ensure that the model not only achieves high predictive performance but also meets the operational constraints required for real-time deployment in practical farming environments [8]. A comparative analysis is also conducted between individual CNN models and the ensemble to justify the effectiveness of the ensemble strategy.

The complete methodology ensures that the developed system is scientifically sound, technically robust, and practically viable for real-world smart farming applications, especially for farmers cultivating medicinal plants like Soursop.

1.5 Project Outcome

The outcome of this research project is the development of a lightweight, edge-deployable Convolutional Neural Network (CNN) ensemble model capable of detecting Soursop leaf diseases in real time with high accuracy and low computational cost. Through systematic data collection, model development, and optimization, the project delivers a complete, practical solution tailored for resource-constrained environments such as mobile devices, edge computing platforms, and smart farming tools.

The project successfully implements and trains four individual CNN architectures—VGG19, DenseNet201, ResNet101, and InceptionV3—on the Soursop dataset, followed by the design of a soft-voting ensemble model that outperforms the individual networks in terms of classification robustness and stability. By aggregating the strengths of multiple models, the ensemble achieves superior predictive performance compared to any single CNN architecture.

Additionally, model optimization is achieved by converting the final ensemble model into TensorFlow Lite (TFLite) format, drastically reducing the model size and computational demand without significant loss in accuracy. This allows the model to be deployed efficiently on smartphones, portable devices, and embedded agricultural monitoring systems, facilitating real-time disease detection directly in the field.

Another important outcome of this research is the development of a mobile-compatible system design, offering farmers and agricultural practitioners a user-friendly and cost-effective tool for early diagnosis of Soursop leaf diseases. By enabling timely intervention, the system can contribute significantly to improving crop yield, preserving the medicinal quality of Soursop, and supporting sustainable agricultural practices.

In summary, this project not only addresses key technical challenges in lightweight model development and real-time deployment but also provides practical value by contributing to the conservation of medicinal plants, enhancing agricultural productivity, and empowering farming communities with AI-driven smart farming solutions.

1.6 Organization of the Report

This This report is organized into six chapters, each systematically covering the stages of research and development undertaken for the automated detection and classification of Soursop leaf diseases using a lightweight CNN ensemble framework optimized for real-time, edge deployment. The contents of each chapter are outlined as follows:

Chapter 1 – Introduction:
This chapter provides a general overview of the study, outlining the background, motivation, and objectives. It summarizes the methodology adopted for the development of the proposed system and highlights the expected outcomes. The chapter also provides an overview of the structure of the entire report.

Chapter 2 – Background and Literature Review:
This chapter presents the foundational concepts relevant to plant disease detection and reviews existing studies related to deep learning applications in agricultural and medicinal plant disease classification. It includes a thematic literature review of similar works, discusses related research efforts, and identifies the research gap that this study aims to address.

Chapter 3 – Research Methodology:
This chapter describes the proposed methodology in detail. It covers dataset collection and preprocessing, model training using transfer learning, ensemble learning techniques, and the design of a lightweight system for edge deployment. The chapter also discusses the functional and non-functional requirements, presents system diagrams (context and data flow diagrams), user interface design considerations, and outlines the overall project planning and task allocation strategies.

Chapter 4 – Implementation and Results:
This chapter focuses on the technical implementation of the system. It describes the development environment, training and testing procedures, evaluation metrics used, and presents the results obtained from experimental validation.

Comparative analyses of individual CNN models and the ensemble model are discussed, alongside performance metrics highlighting model efficiency and real-time applicability.

Chapter 5 – Engineering Standards and Design Challenges:
This chapter details the engineering standards and best practices followed throughout the development process, including considerations for software development, mobile deployment, and hardware compatibility. It also discusses ethical, societal, and environmental implications, sustainability aspects, project management activities, financial considerations, and addresses complex problem-solving approaches employed during the system's design and deployment phases.

Chapter 6 – Conclusion and Future Work:
The final chapter summarizes the major findings of the study, reflecting on the successes and limitations of the developed system. It provides recommendations for future research directions aimed at further improving the detection framework, expanding its applicability to other medicinal plants, and enhancing system scalability for broader smart farming adoption.

Chapter 2

Background

This Chapter 2 presents a comprehensive review of existing studies related to plant disease detection, particularly medicinal plant identification and classification using deep learning methods. It explores similar applications, compares related research, and identifies major gaps in the current literature. The chapter systematically groups works based on their focus areas and provides a critical analysis to justify the need for this study, which aims to develop a lightweight, edge-deployable CNN ensemble model for real-time Soursop disease detection.

2.1 Introduction

Recent advances in artificial intelligence and deep learning have significantly impacted the field of plant disease detection. Traditional manual methods for disease identification are often time-consuming, subjective, and prone to error, necessitating the adoption of automated, efficient, and accurate techniques. Deep learning, particularly convolutional neural networks (CNNs), has emerged as a powerful tool for classifying plant species and detecting diseases based on leaf images. However, challenges still exist, such as the high computational complexity of deep models, lack of edge-deployable lightweight solutions, and limited focus on specific plants like Soursop.

This chapter reviews existing literature related to medicinal plant identification, agricultural disease detection, and CNN-based classification approaches. It highlights various applications, discusses models ranging from basic CNN architectures to advanced ensemble and hybrid methods, and critically analyzes their strengths and limitations. Through this thematic review, the chapter identifies research gaps that remain unaddressed, particularly the need for a real-time, lightweight, and accurate detection system focused on Soursop diseases. The findings from this literature review lay the foundation for the methodology proposed in this thesis.

2.2 Literature Review

Early detection of soursop leaf diseases is crucial for maintaining plant health and productivity. Ahad et al. [1] proposed a lightweight ensemble method named 'S-R101RE', integrating feature extraction, feature selection, and machine learning-based ensemble classification. Their method achieved 99.6% accuracy with lower execution time compared to Vision Transformers (ViT) and traditional CNNs, showing superior performance in soursop disease classification. This work highlights the importance of optimizing both accuracy and computational efficiency for real-time agricultural applications.

Medicinal plants have significant therapeutic importance, necessitating accurate classification methods. Custodio [2] utilized VGG19 for classifying 40 Philippine medicinal plant species, achieving an average accuracy of 92.67%. Data augmentation techniques were applied to enhance performance under limited data conditions. Similarly, Senanayake and De Silva [3] emphasized the importance of correct identification to prevent misuse of poisonous plants, proposing a transfer learning-based model that achieved 99.5% accuracy for plant identification and 90% for disease classification. Further, Azadnia et al. [5] developed a CNN-based intelligent vision system for identifying herb plants, achieving over 99.3% accuracy across various image resolutions. Balasundaram et al. [6] focused on distinguishing true medicinal leaves from look-alike ordinary leaves using CNNs, contributing to reliable herbal medicine preparation. Dalvi et al. [7] expanded on this by using a multi-attribute deep CNN approach, integrating images of leaves, trunks, and seeds, achieving high accuracy in medicinal plant detection, particularly for skin health applications. In another study, Praveena et al. [8] proposed a basic CNN model for identifying ten different medicinal leaf species, achieving an impressive 96.88% accuracy. WDLSA [9] also developed a CNN-based system for medicinal plant recognition, achieving a recognition rate of 85%, emphasizing the need for improving speed and expanding datasets.

Leaf disease detection in broader agricultural contexts has been actively researched. Lu et al. [4] reviewed the role of CNNs in plant leaf disease classification, highlighting how deep learning has solved traditional limitations like subjectivity and manual error. Sardogan et al. [10] introduced a CNN

combined with Learning Vector Quantization (LVQ) for tomato leaf disease classification, effectively utilizing color information to improve detection. Tiwari et al. [11] conducted a deep neural network-based study on multi-class classification of medicinal plant leaves, achieving an average cross-validation accuracy of 98.68%. Kurmi et al. [12] proposed a deep CNN model for crop disease detection, which performed well on the PlantVillage dataset, obtaining a mean classification accuracy of 95.35%. Gokulnath [18] introduced a resilient loss-fused CNN model that combined two different loss functions for more accurate plant disease detection, achieving 98.93% accuracy. Similarly, Geetharamani and Pandian [14] developed a nine-layer deep CNN model, demonstrating improved disease classification in plant leaves with 96.46% accuracy.

Comparative studies between CNN architectures, transfer learning, and ensemble methods are crucial for selecting optimal models. Dey et al. [13] compared seven deep learning models including DenseNet201, VGG16, and InceptionV3 for medicinal plant classification. DenseNet201 achieved the highest accuracy of 99.64% on pure public images and 97% on mixed public-field images, suggesting robustness across datasets. Ahad et al. [22] extended CNN comparisons into rice disease classification, employing transfer learning and an ensemble model (DEX) consisting of DenseNet121, EfficientNetB7, and Xception. Their ensemble approach achieved the best accuracy of 98%, outperforming individual CNNs. These findings align with the broader research trend that ensemble methods and transfer learning often outperform single CNN architectures in agricultural disease classification tasks. Diwedi et al. [24] proposed an enhanced CNN using a modified ResNet50 integrated with an optimized SVM classifier, achieving a testing accuracy of 96.8%. Their work demonstrated that progressive transfer learning can further refine CNN models for better medicinal plant identification.

Several researchers proposed customized deep learning models for plant disease detection. Bouakkaz et al. [15] introduced an optimized CNN architecture for medicinal plant classification, integrating residual blocks and feature fusion to enhance accuracy and speed. Alharbi et al. [16] developed a ConvLSTM U-Net hybrid model for bacterial spot disease detection in medicinal plants, offering a novel lightweight solution with fewer training parameters. Pandey et al. [17]

reviewed cutting-edge disease detection methods in medicinal plants, emphasizing technologies like high-throughput DNA sequencing and non-invasive imaging to improve speed and precision. Keskar and Maktedar [25] proposed an AlexNet-based solution for detecting and classifying medicinal plant diseases, achieving a detection accuracy of 97.27%. Beyond disease detection, deep learning also supports biodiversity conservation and herbal medicine identification. Mujahid et al. [23] applied CNNs to herbal leaf species classification in Indonesia, achieving 92.66% accuracy, thus contributing to biodiversity conservation. Sangeetha et al. [19] developed a CNN-based medicinal plant identification system targeting fever, pain, and related ailments, facilitating the preservation of traditional medical knowledge. Ayumi et al. [21] conducted a systematic review of medicinal plant leaf recognition systems, highlighting the effectiveness of CNNs, SVMs, and feature extraction techniques like GLCM and MCMLGP. Naeem et al. [20] performed a machine learning-based classification on multispectral medicinal leaf datasets, with the multilayer perceptron classifier achieving up to 99.90% accuracy for individual plant types.

Although deep learning models show promising results, challenges like limited annotated datasets, high computational requirements, and real-world variability remain significant hurdles. Traditional classification models often suffer from redundancy in extracted features, making optimization difficult [12]. Advances like optimized feature selection [15], ensemble learning [22], and hybrid architectures [16] are progressively addressing these limitations. Nonetheless, expanding datasets, reducing model complexity for edge deployment, and integrating explainable AI (XAI) methods are crucial future directions for enhancing medicinal plant disease detection and classification systems. Table 2.1 presents the summary of the existing works.

Table 2.1: Research Matrix for Soursop Leaf Disease Detection Techniques

Author & Year	**Models Used**	**Accuracies**	**Contributions**
Ahad et al. [1]	S-R101RE (feature extraction + ensemble)	99.6%	Lightweight feature-based detection for Soursop

Custodio [2]	VGG19	92.67%	Classification of Philippine medicinal plants
Senanayake and De Silva [3]	Transfer Learning, CNN	99.5% (identification), 90% (disease classification)	Medicinal plant and disease identification
Lu et al. [4]	CNN review	N/A	Comprehensive CNN survey for plant diseases
Azadnia et al. [5]	Deep CNN with GAP	>99.3%	AI-based herb plant recognition
Balasundaram et al. [6]	CNN	High accuracy	Distinguishing medicinal leaves from look-alikes
Dalvi et al. [7]	Multi-Attribute Deep CNN	High accuracy	Detecting medicinal plants for skin health
Praveena et al. [8]	Basic CNN (4 conv layers)	96.88%	Indian medicinal plant classification
WDLSA [9]	CNN	85%	Leaf-based medicinal plant recognition
Sardogan et al. [10]	CNN + LVQ	Good performance	Tomato leaf disease detection
Tiwari et al. [11]	DNN	98.68% (cross-validation), 97.69% (test)	Multi-class classification of plant leaves
Kurmi et al. [12]	Deep CNN	95.35%	Crop disease detection using PlantVillage dataset
Dey et al. [13]	DenseNet201, VGG, InceptionV3, etc.	Up to 99.64%	Comparative study of CNN models
Geetharamani and Pandian [14]	9-layer Deep CNN	96.46%	Plant disease classification
Bouakkaz et al. [15]	Optimized CNN with residual blocks	High accuracy	Fast classification of medicinal plants
Alharbi et al. [16]	ConvLSTM U-Net	High (not specified)	Lightweight bacterial spot detection
Pandey et al. [17]	High-throughput DNA, imaging methods	N/A	Review of disease detection methods
Gokulnath [18]	Loss-Fused CNN	98.93%	Early plant disease detection
Sangeetha et al. [19]	CNN	High accuracy	Medicinal plant identification for ailments

Naeem et al. [20]	Multilayer Perceptron, ML classifiers	Up to 99.90%	Multispectral medicinal plant leaf classification
Ayumi et al. [21]	CNN, SVM, MLP	Varies	SLR on medicinal plant recognition methods
Ahad et al. [22]	Ensemble (DenseNet121+Efficient NetB7+Xception)	98%	Rice disease detection via ensemble CNN
Mujahid et al. [23]	CNN	92.66%	Indonesian herbal leaf classification
Diwedi et al. [24]	Enhanced ResNet50 + Optimized SVM	96.8%	Hybrid medicinal plant classification
Keskar and Maktedar [25]	AlexNet Deep CNN	97.27%	Deep learning for medicinal plant disease detection

2.2.1 Similar Applications

Several researchers have developed deep learning-based systems for identifying medicinal plants, detecting plant diseases, and improving agricultural productivity. Lu et al. [4] reviewed how CNNs have revolutionized plant disease detection, making classification faster and more accurate. Azadnia et al. [5] proposed an AI-based system using a deep CNN with Global Average Pooling, achieving over 99.3% accuracy for medicinal plant identification. Balasundaram et al. [6] developed a CNN model to distinguish true medicinal leaves from similar non-medicinal ones, emphasizing the importance of accurate identification in herbal medicine. Dalvi et al. [7] introduced a multi-attribute deep CNN to detect medicinal plants associated with skin diseases. Praveena et al. [8] proposed a simple CNN achieving 96.88% accuracy in classifying Indian medicinal plants. Mujahid et al. [23] applied CNNs to classify herbal leaves in Indonesia with 92.66% accuracy, contributing to biodiversity conservation. Sangeetha et al. [19] focused on medicinal plant identification for common ailments using deep CNN.

Similarly, Geetharamani and Pandian [14] designed a nine-layer CNN for plant leaf disease detection, and Gokulnath [18] introduced a resilient loss-fused CNN model achieving 98.93% accuracy. Keskar and Maktedar [25] used AlexNet-based deep learning for medicinal plant disease classification, achieving 97.27% accuracy. Ayumi et al. [21] conducted a survey on medicinal plant leaf recognition

methods using AI, highlighting CNNs, SVMs, and GLCM techniques. WDLSA [9] built a CNN-based recognition system achieving 85% accuracy for medicinal plants. Sardogan et al. [10] developed a CNN + LVQ model for tomato leaf disease detection. Tiwari et al. [11] designed a deep neural network for multi-class classification of medicinal plants, achieving 98.68% accuracy.

Table 2.2 presents the studies demonstrate the wide adoption of CNN-based techniques for plant classification and disease detection.

Table 2.2: Similar Applications – Research Matrix

Author & Year	Models Used	Accuracies	Contributions
Lu et al. [4]	CNN review	N/A	Survey of CNNs for plant disease
Azadnia et al. [5]	Deep CNN with GAP	>99.3%	Herb plant identification
Balasundaram et al. [6]	CNN	High	Medicinal leaf detection
Dalvi et al. [7]	Multi-Attribute CNN	High	Medicinal plants for skin diseases
Praveena et al. [8]	CNN	96.88%	Indian medicinal plants
Mujahid et al. [23]	CNN	92.66%	Herbal leaf classification
Sangeetha et al. [19]	CNN	High	Identification for ailments
Geetharamani and Pandian [14]	9-layer CNN	96.46%	Leaf disease detection
Gokulnath [18]	Loss-fused CNN	98.93%	Robust disease classification
Keskar and Maktedar [25]	AlexNet	97.27%	Medicinal plant disease classification
Ayumi et al. [21]	CNN, SVM, MLP	Various	AI methods for medicinal plants
WDLSA [9]	CNN	85%	Plant recognition using CNN
Sardogan et al. [10]	CNN + LVQ	Good	Tomato disease detection
Tiwari et al. [11]	DNN	98.68%	Multi-class leaf classification

2.2.2 Related Research

Related research has also addressed optimizing deep learning methods for better performance in agricultural disease detection. Dey et al. [13] compared multiple deep CNN models for medicinal plant identification, finding DenseNet201 to be most robust with 99.64% accuracy. Ahad et al. [22] proposed a CNN ensemble

(DEX model) for rice disease classification, achieving 98% accuracy. Diwedi et al. [24] combined an enhanced ResNet50 with optimized SVM for medicinal plant classification, reaching 96.8% testing accuracy.

Kurmi et al. [12] proposed a deep CNN model for general crop disease detection, highlighting the importance of efficient region localization before feature extraction. Bouakkaz et al. [15] introduced a CNN architecture with residual blocks and optimized feature selection for medicinal plants, significantly improving classification speed and accuracy. Alharbi et al. [16] developed a ConvLSTM U-Net hybrid model for bacterial spot disease detection in plants, reducing training parameters for faster execution.

Pandey et al. [17] emphasized the potential of advanced molecular biology and imaging techniques like real-time PCR and non-invasive imaging for medicinal plant disease detection. Naeem et al. [20] used multispectral image analysis combined with machine learning, achieving up to 99.90% accuracy with multilayer perceptron classifiers.

Table 2.3 presents studies reveal the continuous evolution of feature extraction methods, model optimization, and lightweight architectures for real-world agricultural and medicinal plant detection.

Table 2.3 Related Research – Matrix Table

Author & Year	Models Used	Accuracies	Contributions
Dey et al. [13]	CNN models	Up to 99.64%	Model comparison for plant classification
Ahad et al. [22]	Ensemble CNN	98%	Rice disease classification
Diwedi et al. [24]	Enhanced ResNet50 + SVM	96.8%	Hybrid classification approach
Kurmi et al. [12]	Deep CNN	95.35%	Crop disease detection
Bouakkaz et al. [15]	Optimized CNN	High	Enhanced classification speed and accuracy
Alharbi et al. [16]	ConvLSTM U-Net	High	Lightweight disease classification
Pandey et al. [17]	Imaging, DNA sequencing	N/A	Medicinal plant disease detection review
Naeem et al. [20]	Multilayer Perceptron	Up to 99.90%	Multispectral leaf classification

2.3 Gap Analysis

Despite the significant progress made in medicinal plant and agricultural disease detection, several gaps remain presented in table 2.4. Most studies, such as Lu et al. [4], focus heavily on CNN-based models but lack adaptation for real-time edge deployment, crucial for field applications. Although ensemble methods like in Ahad et al. [22] and Dey et al. [13] improved accuracy, they often increase computational complexity, making deployment on mobile or embedded devices challenging.

Furthermore, works like Balasundaram et al. [6] and Mujahid et al. [23] emphasize species classification but less on early disease stage detection. Similarly, although ConvLSTM U-Net models like Alharbi et al. [16] are lightweight, they have been applied to limited types of diseases and lack generalization across varied plant species.

Another gap lies in the lack of dedicated work focusing specifically on Soursop plant disease detection; most studies generalize across plant types without optimizing for species-specific disease patterns. Only Ahad et al. [1] recently proposed a feature-based lightweight approach directly targeting soursop leaves.

Moreover, most machine learning-based models (like Naeem et al. [20]) rely heavily on feature engineering, which is time-consuming and less scalable compared to end-to-end deep learning models. There is also limited focus on combining explainability (XAI) with lightweight disease detection frameworks for medicinal plants [15] [17].

Thus, there is a clear research gap in developing an edge-deployable, lightweight, ensemble-based CNN model tailored specifically for real-time soursop disease detection, bridging the gap between high accuracy, fast execution, and field deployment suitability.

2.4 Gap Analysis Table – Existing Research

Gap Area	Observed in Studies	Addressed in This Thesis
Lack of real-time edge deployment	Lu et al. [4], Ahad et al. [22]	Developing lightweight CNN ensemble (TFLite ready)
High complexity in ensemble models	Dey et al. [13], Ahad et al. [22]	Lightweight optimized ensemble
Limited focus on early-stage disease	Balasundaram et al. [6], Mujahid et al. [23]	Detecting early Soursop leaf diseases
Lack of dedicated Soursop studies	Ahad et al. [1]	Focus exclusively on Soursop dataset

Overreliance on manual feature engineering	Naeem et al. [20]	Fully automated feature extraction
Lack of explainable AI in lightweight models	Bouakkaz et al. [15], Pandey et al. [17]	Future plan to integrate Grad-CAM/XAI

2.4 Summary

This chapter provided a structured review of key studies in the areas of medicinal plant recognition and plant disease detection using deep learning techniques. Similar applications employing CNNs, Transfer Learning, and ensemble strategies were critically analyzed. Comparative research highlighted the advancements and challenges in improving accuracy, reducing training time, and enabling real-time deployment. The gap analysis revealed critical issues such as the absence of lightweight models for edge devices, high complexity in ensemble frameworks, limited research specifically on Soursop diseases, and insufficient integration of explainable AI techniques.

By identifying these gaps, this chapter justifies the development of an edge-deployable lightweight CNN ensemble tailored for real-time Soursop disease detection, thereby contributing to smart farming and precision agriculture.

Chapter 3

Research Methodology

The Chapter 3 describes the detailed methodology adopted to develop the lightweight CNN ensemble model for real-time Soursop disease detection. It outlines the steps involved, starting from dataset collection and preprocessing to model training, ensemble creation, and optimization for edge deployment. The chapter also covers the system design, including the functional and non-functional requirements, data flow diagrams, user interface design, and project planning. The methodology ensures that the proposed system is accurate, lightweight, and suitable for mobile-based smart farming applications.

3.1 Methodology

3.1.1 Overview

This study proposes an edge-deployable, lightweight convolutional neural network (CNN) ensemble framework for real-time soursop leaf disease detection to support smart farming initiatives. Using a publicly available dataset comprising 3,838 labeled images across six classes—Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow—the dataset was partitioned into training (2,456), validation (614), and testing (768) sets. Four pre-trained CNN models—VGG19, ResNet101, InceptionV3, and DenseNet201—were employed and integrated via a soft-voting ensemble strategy to enhance classification robustness. The optimized ensemble model was converted into a TensorFlow Lite (TFLite) format (best_float32.tflite) and deployed in a Flutter-based mobile application for real-time disease recognition, ensuring accessibility for end-users such as farmers and agricultural technicians.

3.1.2 Proposed Methodology

The methodological flowchart presented in Figure 3.1 illustrates the overall methodological workflow of the proposed system for real-time soursop leaf disease detection. The process begins with the collection of the Soursop disease

dataset, followed by a direct train-validation-test split without any explicit preprocessing. Four pre-trained convolutional neural network (CNN) models—VGG19, ResNet101, InceptionV3, and DenseNet201—are independently trained on the dataset. Their predictions are then integrated using a soft-voting ensemble strategy to improve classification accuracy and robustness. The best-performing ensemble model is converted to a lightweight TensorFlow Lite (best_float32.tflite) format and deployed into a Flutter-based mobile application. This enables end-users, such as farmers, to capture leaf images and receive real-time disease identification results directly on their smartphones.

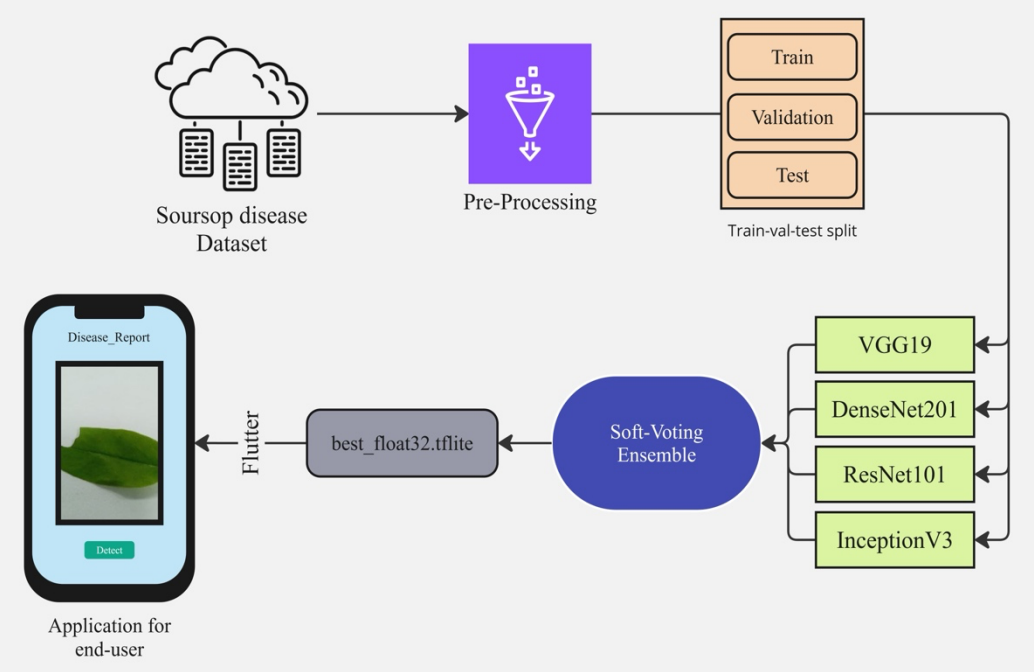


Figure 3.1: The Methodological Flowchart

3.1.3 Functional and Nonfunctional Requirements

The system requirements for the proposed edge-deployable CNN ensemble model are categorized into functional and nonfunctional requirements to ensure clarity in design, implementation, and user interaction.

1. Functional Requirements

- Image Upload and Detection: The mobile application shall allow users to upload or capture images of soursop leaves using the camera or gallery.

- Real-Time Disease Classification: The system shall process the uploaded image using the embedded TFLite model and classify the leaf into one of the six predefined classes.
- Display of Prediction Results: The app shall display the predicted class label (e.g., DieBack, WhiteFly) along with confidence scores.
- User Interface for Navigation: Users shall be able to easily navigate through the app, access detection features, and view reports.
- Model Integration: The Flutter app shall load the optimized best_float32.tflite model and perform on-device inference without requiring internet connectivity.

2. Nonfunctional Requirements

- Performance: The model shall return prediction results within 2 seconds to ensure a seamless user experience.
- Portability: The application shall be compatible with Android devices with minimal hardware requirements.
- Scalability: The system should allow future extension to support additional disease classes or crops with minimal architectural changes.
- Usability: The user interface shall be intuitive, requiring minimal instructions for operation, especially targeting non-technical users like farmers.
- Resource Efficiency: The deployed model must be lightweight and optimized for mobile deployment, consuming minimal CPU and memory resources.

3.1.4 Data Flow Diagram Level 1

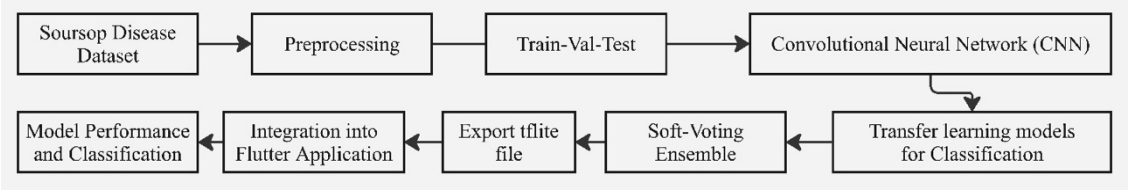


Figure 3.2: Data Flow Diagram Level 1

The Level 1 Data Flow Diagram in Figure 3.2 presents the Level 1 Data Flow Diagram (DFD) outlining the step-by-step process involved in the proposed system for soursop leaf disease detection. The workflow begins with the Soursop

Disease Dataset, which undergoes a preprocessing stage followed by a train-validation-test split. The processed data is used to train several convolutional neural networks (CNNs) based on transfer learning. The outputs from multiple CNN models are then integrated using a soft-voting ensemble technique to enhance classification performance. The best-performing ensemble model is converted into a TensorFlow Lite (.tflite) format, enabling seamless deployment. The lightweight model is then integrated into a Flutter-based mobile application, allowing end-users to detect diseases in real time. The final stage involves evaluating the model's performance based on classification accuracy and responsiveness.

3.1.5 UI Design

To facilitate real-time disease detection and enhance usability for agricultural practitioners, the proposed system was developed as a mobile application using the Flutter framework. The user interface (UI) is designed to be simple, interactive, and efficient, allowing non-technical users such as farmers to detect soursop leaf diseases by capturing or uploading an image. The architecture behind the application ensures that model inference occurs efficiently on-device through the integration of a TensorFlow Lite (TFLite) model. Figure 3.3 illustrates the UI workflow, backend processing, folder structure, and result display mechanism.

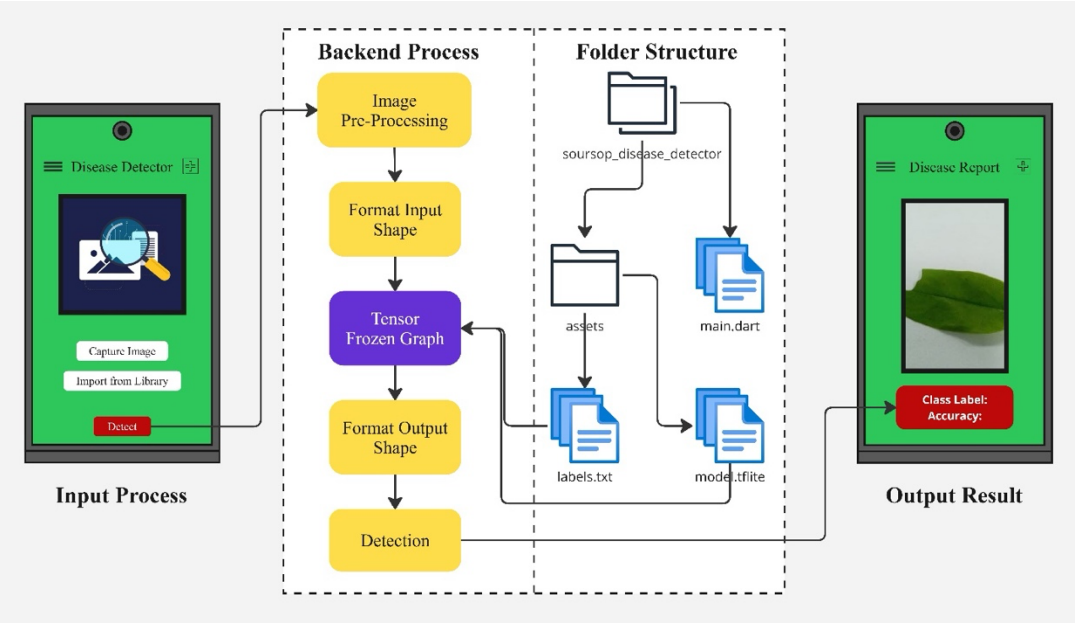


Figure 3.3: The Two-Tier Architecture and UI for Application

a. Application Setup

The proposed mobile application was developed using Flutter, an open-source UI toolkit by Google that supports cross-platform development, making it an ideal choice for mobile deployment in agricultural contexts. The setup process involves integrating the optimized model file (model.tflite) and associated metadata (labels.txt) into the app's asset directory. The core logic is defined in the main.dart file, which governs the UI behavior, image handling, and communication with the inference engine.

Upon launching the application, users are greeted with a clean and intuitive interface labeled Disease Detector, designed specifically to accommodate users with limited technical expertise. The interface offers two main input options: Capture Image, which invokes the device camera for real-time image acquisition, and Import from Library, which allows users to select an existing image from their gallery. Once an image is selected, pressing the Detect button initiates the inference pipeline, which handles image resizing, normalization, and model prediction. The results are then displayed on a dedicated screen titled Disease Report, where users are shown the predicted class label (e.g., "DieBack" or "Healthy") and the model's confidence score as a percentage. This real-time feedback enables immediate decision-making for end-users such as farmers, agronomists, and field technicians.

b. Two-Tier Architecture

The application is engineered using a two-tier architecture that distinctly separates the user-facing interface from the computational backend, ensuring both modularity and maintainability.

Tier 1: Frontend (User Interface and Experience): The frontend tier is responsible for facilitating user interaction with the application. Developed using Flutter widgets, the frontend ensures platform responsiveness and visual consistency across devices. Users interact with this layer to upload or capture images, initiate detection, and view results. The UI is intentionally designed to be minimalistic, prioritizing accessibility, usability, and fast navigation, particularly for rural farmers with limited digital literacy. Additionally, visual feedback is provided through the display of the uploaded image and a result

section that clearly shows the class label and confidence level, enhancing trust and transparency.

Tier 2: Backend (Inference and Data Processing): The backend tier comprises the computational logic required to process input data and generate predictions. Once an image is submitted, it undergoes a backend pipeline that includes:

- Image Preprocessing: Resizing the input image to the required dimensions and normalizing pixel values.
- Formatting Input Shape: Adjusting the image to match the model's expected input dimensions and data type.
- TensorFlow Lite Inference Engine: The pre-trained and optimized model (model.tflite) processes the input image using a frozen graph to predict the disease class.
- Formatting Output Shape: The raw model outputs are converted into readable class labels and accuracy values using the labels.txt file.
- Result Return: The formatted result is returned to the frontend for final display.

This modular separation between frontend and backend ensures that updates in the model or preprocessing logic can be made without impacting the UI layer. It also contributes to performance efficiency, as all computations are performed locally on the device, enabling offline operation, which is critical for remote agricultural settings.

3.2 Detailed Methodology and Design

3.2.1 Dataset

The dataset used in this study is a publicly available soursop leaf disease image dataset, sourced from Mendeley Data [1]. Figure 3.4 presents it contains a total of 3,838 images, each labeled under one of six distinct classes: Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow. The images were captured in natural lighting conditions with varying backgrounds to ensure diversity and better generalization of the model. To facilitate model training and evaluation, the dataset was divided into three subsets: 2,456 images

for training, 614 for validation, and 768 for testing. The dataset plays a vital role in enabling robust and accurate classification of soursop leaf diseases, and its balanced composition supports the development of a reliable ensemble detection model suitable for real-world smart farming applications.

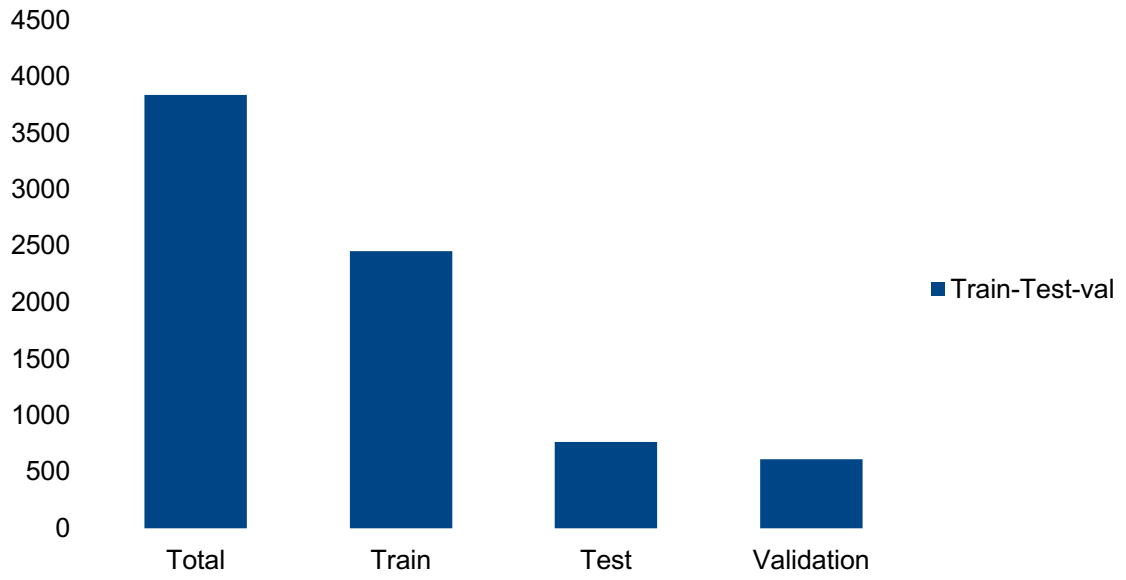


Figure 3.4: Train-Validation_Test Splitting

3.2.2 Classes

The soursop leaf disease dataset used in this study comprises six visually and biologically distinct classes presented on figure 3.5. Each class represents either a type of disease or a healthy leaf condition, allowing for precise classification using deep learning models. The descriptions of each class are provided below:

- Cutting Caterpillar: This class includes images of soursop leaves damaged by caterpillars that chew through the leaf edges or midsection. The visible symptoms typically include irregular cuts, holes, or frayed leaf margins caused by larval feeding activity.
- Cutting Weevil: Leaves in this category exhibit damage caused by weevils, which often create circular or notched bite marks along the leaf margins. The affected areas may turn brown or dry out, indicating insect infestation.
- Die Back: This disease leads to the gradual withering of leaves from the tip toward the base. In the images, the leaves typically appear dry,

discolored, or necrotic, representing the progressive death of leaf tissues due to fungal or bacterial infections.

- Healthy: This class contains images of undamaged, disease-free soursop leaves. These leaves are typically green with uniform texture, showing no signs of discoloration, spots, or insect damage, serving as the baseline reference for classification.
- White Fly: Leaves in this class are affected by whitefly infestations, which usually appear as pale yellow patches or sticky residues on the leaf surface. These flies also cause chlorosis, leading to weakened photosynthesis and plant vigor.
- Yellow: The yellowing of leaves may indicate nutrient deficiencies or early disease stages. In this class, the leaf surfaces show varying degrees of yellow discoloration, usually starting from the edges or veins, without clear insect or fungal patterns.

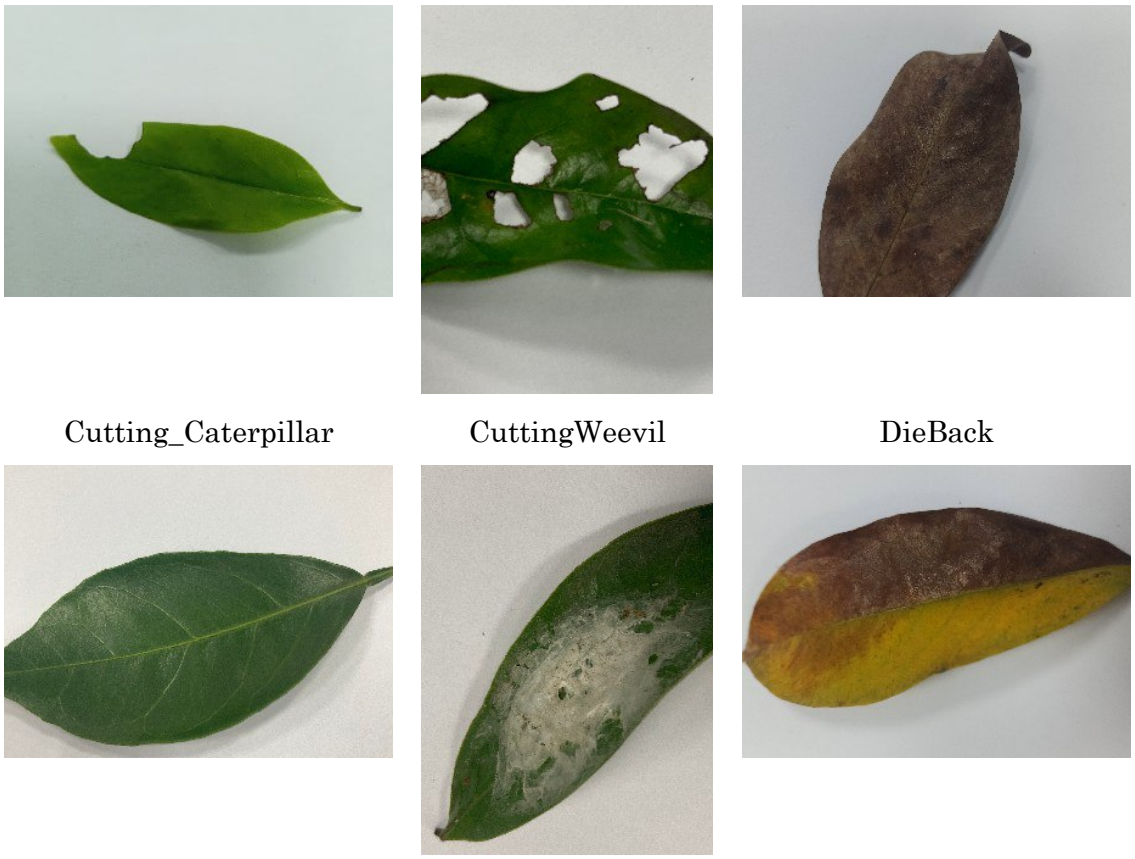


Figure 3.5: Sample Image of Each Class

The below table 3.1 shows the data specification table also.

Table 3.1: Dataset Specifications

Properties	Values
Image Resolution	240 × 240 pixels
Format	.jpg
Total Images	3838 (Source: Mendeley Data)
Classes	6

3.2.3 Data Preprocessing Steps

1. Convolutional Neural Network (Pretrained)

To efficiently classify soursop leaf diseases, this study adopts transfer learning using four state-of-the-art pre-trained convolutional neural networks (CNNs): VGG19, ResNet101, InceptionV3, and DenseNet201. Transfer learning leverages knowledge from models previously trained on large-scale datasets (such as ImageNet), reducing the need for extensive data and computational resources figure 3.6 shows the architecture. The process followed for transfer learning in this study can be summarized in the following steps:

Step 1: Load Pre-trained Model without Top Layer

Each CNN model was loaded with pre-trained weights from ImageNet, excluding the final fully connected classification layer (also referred to as the "top" layer). This ensures the model retains its learned feature extraction capabilities while allowing customization for the soursop dataset.

$$f_{CNN}^{base} = PreTrainedModel(weights = 'imagenet',\ include\_top = False) \quad \text{(I)}$$

Step 2: Input Adjustment and Image Resizing

Input images were resized to match the expected input dimensions of each model, e.g., 224×224×3 for VGG19 and ResNet101, or 299×299×3 for InceptionV3. Normalization was applied so that pixel values fell within the range [0, 1].

$$I_{resized} = Resize\left(I_{original}, (h, w)\right), I_{norm} = \frac{I_{resized}}{255.0} \quad \text{(II)}$$

Step 3: Feature Extraction using Frozen Layers

The convolutional layers of each model were frozen to prevent weight updates during training. This helps retain the generic feature extraction knowledge learned from ImageNet, focusing the training on the newly added classification layers.

$$Trainable\left(f_{CNN}^{base}\right) = False \_\_\_\_\_ \text{(III)}$$

Step 4: Add Custom Classification Head

On top of the frozen base, a custom classification head was added. This typically includes a Global Average Pooling (GAP) layer, followed by one or more fully connected layers and a final softmax output layer to classify the image into one of six classes.

$$f_{custom} = SoftMax\left(W \,.\, GAP\left(f_{CNN}^{base}(I)\right) + b\right) \_\_\_\_\_ \text{(IV)}$$

Where:

- W and b are the weights and biases of the dense layer,
- Softmax maps the logits to a probability distribution over the six classes.

Step 5: Model Compilation and Training

The composite model was compiled using the categorical cross-entropy loss function and the Adam optimizer and trained on the training dataset with early stopping and learning rate scheduling to avoid overfitting and optimize convergence.

$$\mathcal{L}_{CCE} = -\sum_{i=1}^{C} y_i \log\left(\hat{y}_i\right) \_\_\_\_\_ \text{(V)}$$

Where:

- C is the number of classes,
- y_i is the true label,
- $\hat{y}_i$ is the predicted probability for class i.

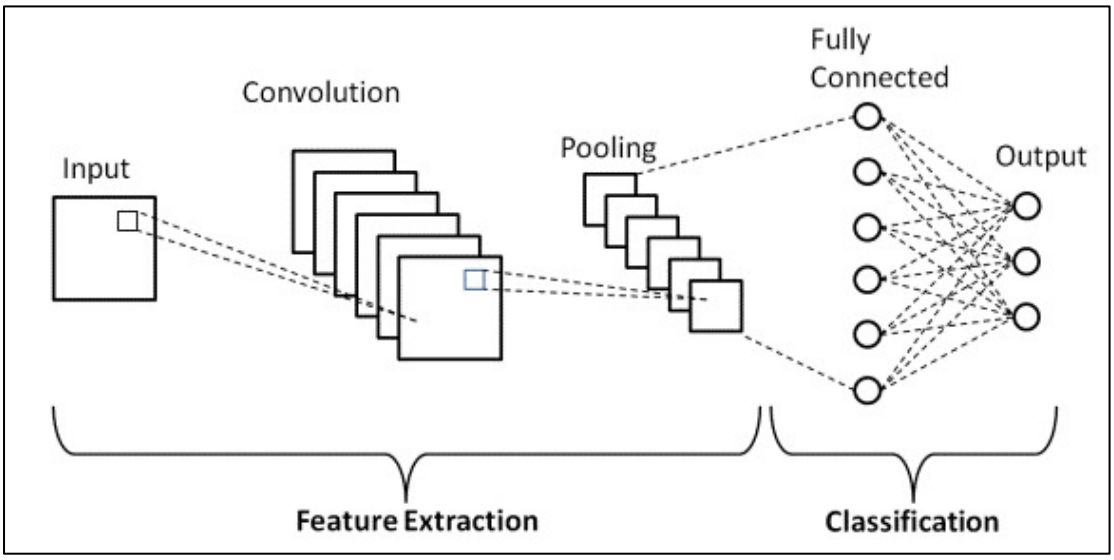


Figure 3.6: The architecture of Convolutional Neural Network (CNN)

2. Comparison between pre-trained CNN models

Table 3.2 presents a comparative overview of the four pre-trained CNN models—VGG19, ResNet101, InceptionV3, and DenseNet201—used in this study for soursop leaf disease classification. VGG19, despite its strong feature extraction capability, is the heaviest in terms of parameters and has the slowest inference time, making it less suitable for edge deployment. ResNet101 introduces residual connections that enhance deep feature learning while maintaining a moderate parameter count and inference speed. InceptionV3 is the most efficient in terms of inference time, utilizing inception blocks for multi-scale feature extraction with a relatively lightweight architecture. DenseNet201, though deep, is compact in parameter size due to its densely connected layers and offers strong gradient flow, making it effective for complex classification tasks. These diverse architectural strengths complement each other in the ensemble, leading to improved overall performance.

Table 3.2: Comparison of Base CNN Models Used in the Study

Model	Depth	Parameter Count	Inference Time	Feature Extraction Strength
VGG19	19 layers	~143 million	High	Strong for general features, but heavy and slow
ResNet101	101 layers	~44 million	Moderate	Excellent at deep feature learning due to residual connections

InceptionV3	~48 layers	~23 million	Low	Efficient with multi-scale feature extraction using inception blocks
DenseNet201	201 layers	~20 million	Moderate	Strong gradient flow and compact due to dense connections

3. Soft-Voting Ensemble

To enhance classification accuracy and model robustness, a soft-voting ensemble strategy was employed, architecture presented on figure 3.7, combining the outputs of four pre-trained CNN models: VGG19, ResNet101, InceptionV3, and DenseNet201. In contrast to hard voting, which selects the class label with the most votes, soft voting leverages the predicted probability distributions from each base model and averages them to make the final decision.

Step 1: Individual Model Inference

Each base model receives the same input image and produces a probability vector representing the likelihood that the image belongs to each of the six classes.

$$p^{(i)} = [\, p_1{}^{(i)}, p_2{}^{(i)} \dots\dots\dots\ p_6{}^{(i)} \,],\ for\ i = 1, 2, 3, 4 \quad \text{(VI)}$$

Where $p^{(i)}$ is the probability distribution from the ith model, and $p_j{}^{(i)}$ is the predicted probability for class j.

Step 2: Average Probabilities

The predicted probabilities from all base models are averaged elementwise to generate a combined prediction vector:

$$P_{ensemble} = \frac{1}{n}\sum_{i=1}^{n} p^{(i)} \quad \text{(Vll)}$$

Where n=4 (the number of models), and $P_{ensemble}$ It is the final average probability vector.

Step 3: Class Selection

The class with the highest average probability is selected as the final prediction:

$$\hat{y} = \arg\max_j(P_{ensemble}) \quad \text{(VIII)}$$

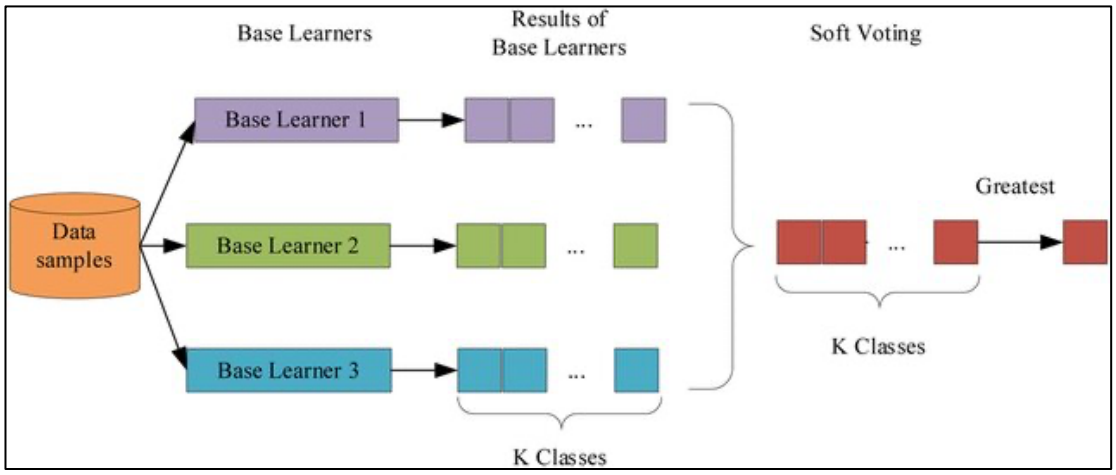
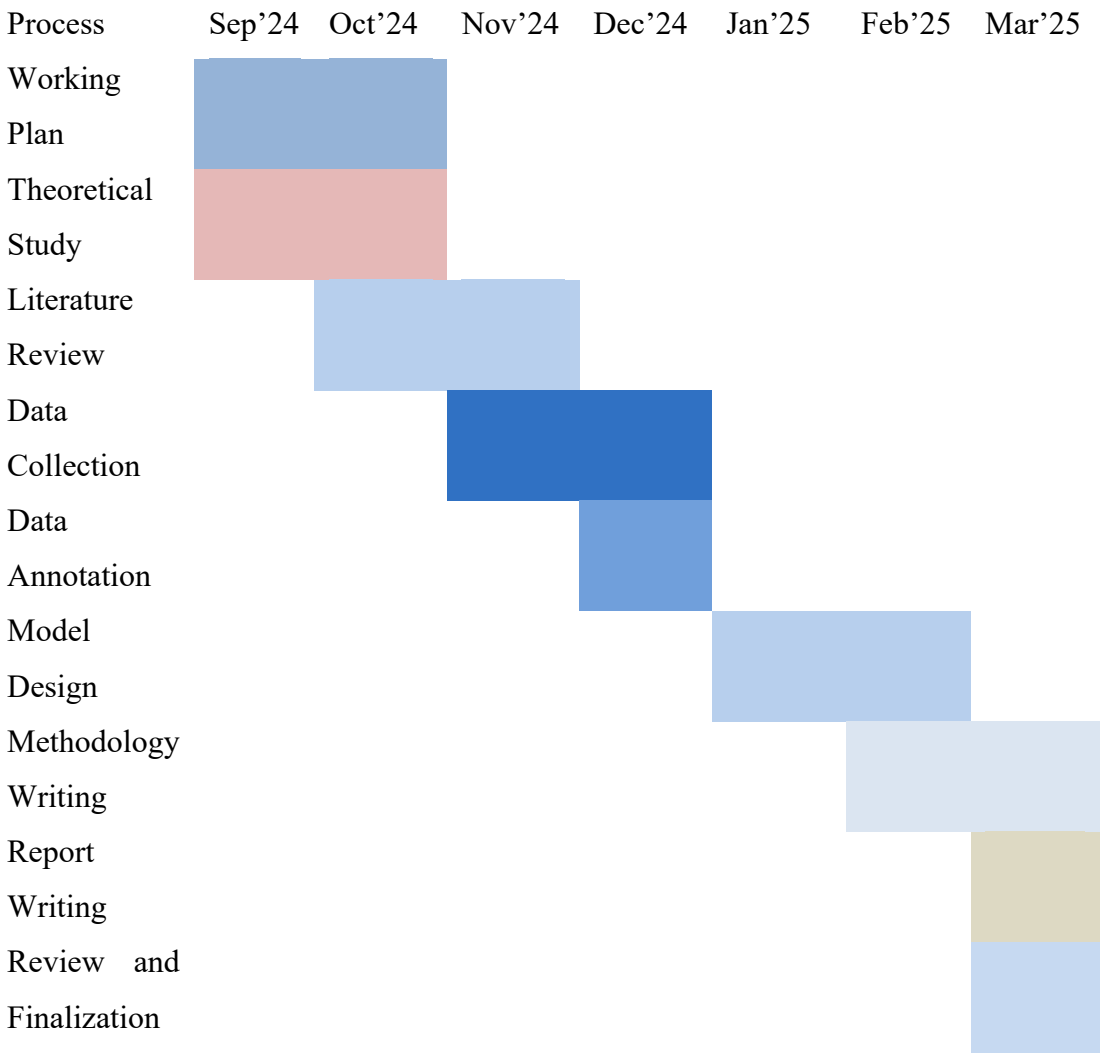


Figure 3.7: The architecture of Soft Voting Ensemble

3.3 Project Plan

The overall project was structured into sequential phases to ensure systematic development, evaluation, and deployment of the proposed soursop disease detection system. The initial phase involved data acquisition and organization, followed by selecting appropriate pre-trained CNN models suitable for transfer learning. In the next phase, each model was fine-tuned using the training dataset, and their performance was validated independently. Subsequently, a soft-voting ensemble strategy was applied to integrate the predictions from all four models to improve robustness and accuracy. Once the best-performing ensemble was identified, it was converted into a TensorFlow Lite (TFLite) format to enable lightweight deployment. The final phase involved building and integrating the model into a Flutter-based mobile application, allowing real-time inference on smartphones. This structured plan table 3.2 ensured that the system was not only accurate but also practically usable in field conditions by farmers.

Table 3.2: GANTT Chart of Project Timeline



3.4 Task Allocation

The development of the proposed soursop leaf disease detection system was a collaborative effort, with tasks distributed among team members based on their expertise and technical proficiency. The dataset collection and organization were handled by the data management lead, ensuring proper class labeling and dataset structuring. The model selection, training, and ensemble implementation were carried out by the deep learning specialist, who also optimized the base CNN architectures for transfer learning. The conversion of the trained ensemble model into TensorFlow Lite format and performance evaluation were managed by the deployment engineer. Simultaneously, the mobile application interface, including image input, result display, and user

interaction design, was developed by the mobile app developer using the Flutter framework. Coordination and documentation tasks, including progress tracking, writing the thesis report, and preparing visual figures such as flowcharts and diagrams, were overseen by the project coordinator. This clear division of tasks ensured efficient collaboration and timely completion of the project milestones.

Table 3.3: Task Allocation Table

Task ID	Task Description	Assigned To	Start Date	End Date	Remarks
T1	Problem analysis and literature review	Md. Arif Billah	01-01-2025	10-01-2025	Understanding existing methods and identifying gaps
T2	Dataset collection and preprocessing		11-01-2025	18-01-2025	Normalization, resizing, DPI adjustment
T3	Individual model training		19-01-2025	28-01-2025	Using Google Colab GPU
T4	Ensemble integration		29-01-2025	02-02-2025	Weighted fusion or averaging technique
T5	Model evaluation (Accuracy, AUC, etc.)		03-02-2025	06-02-2025	Using validation and test sets
T6	Model conversion to TFLite		07-02-2025	08-02-2025	Post-training optimization for mobile
T7	Flutter app design and development		09-02-2025	15-02-2025	Includes UI and TensorFlow Lite integration
T8	Testing on Android device		16-02-2025	18-02-2025	Performance and real-time inference test
T9	Final documentation and report writing		19-02-2025	28-02-2025	Thesis compilation, proofreading, formatting

3.5 Summary

The This study presents a robust and edge-deployable CNN ensemble framework for real-time soursop leaf disease detection, targeting smart farming applications. The system utilizes a publicly available dataset containing 3,838 labeled images spanning six disease classes. Four pre-trained convolutional neural networks—VGG19, ResNet101, InceptionV3, and DenseNet201—were fine-tuned using transfer learning, where only the final classification layers were trained on resized and normalized images. A soft-voting ensemble strategy was employed to integrate the predictions from all models, enhancing classification performance and reducing prediction variance. The best-performing ensemble model was converted into a lightweight TensorFlow Lite format and integrated into a Flutter-based mobile application, enabling real-time, offline disease detection on Android devices. The methodology follows a two-tier architecture, with clear separation between the frontend UI and backend inference engine, ensuring modularity, portability, and user-friendly deployment for field-level agricultural use. Tasks were collaboratively distributed among the team, and the project was executed through a structured, phase-wise plan.

Chapter 4

Implementation and Results

This chapter focuses on evaluating the performance of five deep learning models VGG19, DenseNet201, InceptionV3, and ResNet101, along with a proposed ensemble model for multi-class plant disease classification across six categories: Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow. The models are assessed using a range of performance metrics including training and validation loss and accuracy curves, confusion matrices, classification reports with precision, recall, and F1-scores, and ROC curves with AUC scores. Through these evaluations, the chapter examines how well each model identifies complex disease patterns and distinguishes between visually similar classes. Special emphasis is placed on demonstrating how the ensemble model enhances performance by integrating the strengths of individual CNNs, resulting in higher classification accuracy, reduced misclassifications, and improved generalization across all disease categories.

4.1 Environment Setup

Table 4.1 outlines the common training parameters used for all experimented models, ensuring consistency and fair comparison across different architectures. Each image was resized to 224 × 224 pixels, a standard input size for most CNN-based models, balancing sufficient spatial resolution with computational efficiency. A relatively large batch size of 128 was chosen, enabling stable gradient updates and efficient use of GPU memory during training. The models were trained for 80 epochs, which provided an adequate number of iterations to ensure convergence while minimizing overfitting. The Adam optimizer was employed due to its adaptive learning capability and fast convergence properties, making it well-suited for deep learning tasks. The learning rate was set at 0.0001, a conservative value that allows the model to learn gradually, preventing drastic weight updates and supporting more refined convergence over time.

Table 4.1: Common parameter table for all experimented models.

Parameter Name	Parameter Value
Image Size	224 × 224
Batch Size	128
Epoch	80
Optimizer	Adam
Learning Rate	0.0001

Table 4.2 outlines the data partitioning strategy applied uniformly across all experimented models for training, validation, and testing phases. The dataset, consisting of a total of 3838 images, is divided into 64% for training (2456 images), 16% for validation (614 images), and 20% for testing (768 images). This split was carefully chosen to ensure that the models have access to a sufficient number of examples during training to effectively learn the underlying features of each class. The training set serves as the foundation for model learning, allowing the CNNs to adjust weights and extract meaningful patterns. The validation set, comprising 16% of the data, is used during training to tune hyperparameters and monitor performance, helping prevent overfitting and ensuring the model generalizes well to unseen data. The test set, held back from the training process entirely, represents the final benchmark for unbiased model evaluation. By allocating a relatively large portion (20%) for testing, the study ensures a robust and reliable assessment of how each model performs on real-world data. Overall, this balanced and strategically defined split supports accurate training, effective tuning, and dependable evaluation of all models in the experiment.

Table 4.2: Common data split for all experimented models.

Dataset	In Percentage	Number of Images
Train set	64%	2456
Validation Set	16%	614
Test Set	20%	768

4.2 Testing and Evaluation

In evaluating the effectiveness of machine learning models for the study, appropriate performance metrics must be used to provide insights into model accuracy, reliability, and generalization capabilities. The following metrics are commonly used in classification tasks, especially in the context of agricultural disease detection:

4.2.1 Accuracy

The proportion of correctly classified instances (both positive and negative) to the total instances. Accuracy gives a quick overview of model performance but can be misleading in imbalanced datasets where one class significantly outnumbers the other.

$$Accuracy = \frac{TP+TN}{FP+FN+TP+TN} \text{ ----------(IX)}$$

4.2.2 Recall

The ratio of correctly predicted positive observations to all actual positives. Recall is particularly important in scenarios where a positive case (such as a diseased plant) could lead to severe consequences, like crop loss.

$$Recall = \frac{TP}{(FN+TP)} \text{ ----------(X)}$$

4.2.3 Precision

The ratio of correctly predicted positive observations to the total predicted positives. Precision is crucial in applications where the cost of false positives is high. In this study, high precision indicates that when a disease is predicted, it is likely to be true.

$$Precision = \frac{TP}{(FP+TP)} \text{ ----------(XI)}$$

4.2.4 F1-Score

The harmonic meaning of precision and recall, providing a balance between the two metrics. The F1 score is especially useful when dealing with imbalanced classes, as it considers both false positives and false negatives, offering a more comprehensive view of model performance.

$$F1\ Score = 2 \times \frac{Precision+Recall}{(Precision \times Recall)} \text{ ----------(XII)}$$

4.2.5 Confusion Matrix

A table used to describe the performance of a classification model, showing the true vs. predicted classifications. The confusion matrix provides insights into the types of errors made by the model, allowing for more targeted improvements.

4.2.6 **Receiver Operating Characteristic (ROC) Curve**

A graphical representation of a classifier's performance across various threshold settings, plotting the true positive rate (recall) against the false positive rate. It illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. It plots the True Positive Rate (TPR), or recall or sensitivity, against the False Positive Rate (FPR) at various threshold settings.

Area Under Curve (AUC): The area under the ROC curve provides a single metric to assess model performance; a value closer to 1 indicates a better model.

In this study, the Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) are crucial for evaluating classification models. They allow for visual comparisons of model performance across various thresholds, facilitating optimal threshold selection by balancing sensitivity and specificity. Additionally, the ROC curve is robust against class imbalances, providing reliable assessments where accuracy may mislead. However, ROC and AUC should complement other metrics like precision and recall for a comprehensive evaluation of model effectiveness.

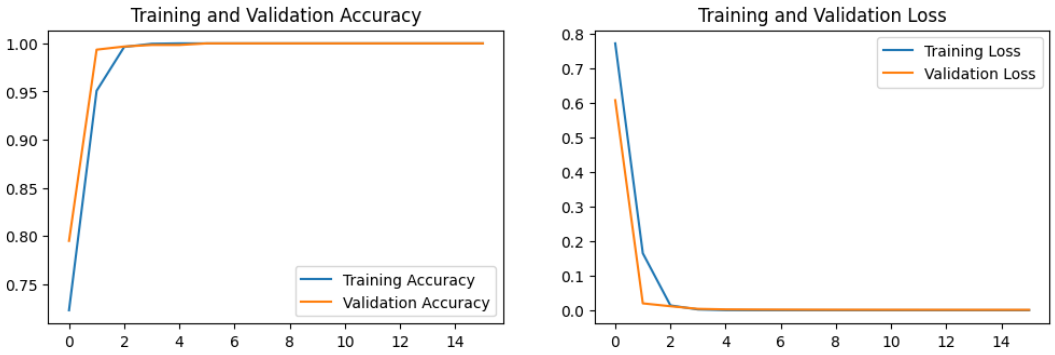
4.3 Results and Discussion

This study evaluates the performance of four individual deep CNN models such as VGG19, ResNet101, InceptionV3, and DenseNet201 alongside an ensemble model that combines the predictions from these CNN models. Each model's effectiveness is assessed through metrics such as loss and accuracy curves, confusion matrices, classification reports, and ROC curves, to understand their strengths and limitations and the added benefits of the ensemble approach.

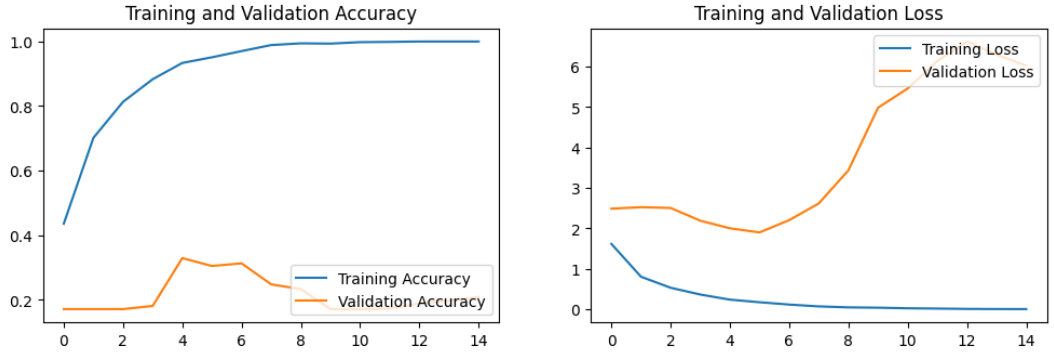
4.3.1 Performance comparison of the deep CNN models

Figure 4.1 illustrates the training and validation loss and accuracy curves over 50 epochs for four deep convolutional neural networks: VGG19, ResNet101, InceptionV3, and DenseNet201. These curves provide insights into each model's learning behavior and generalization capability. Both VGG19 and DenseNet201 show smooth and steady convergence with minimal fluctuations, maintaining close alignment between training and validation accuracy. This indicates strong generalization with negligible overfitting. InceptionV3 also demonstrates high training performance with slightly more variation in the validation curve, but overall follows a stable trend. In contrast, ResNet101 exhibits unstable and inconsistent patterns its loss decreases irregularly, and its validation accuracy fluctuates, indicating poor generalization and underfitting. The model likely struggles to learn discriminative features, which is further evidenced by the gap between its training and validation performance. This figure confirms that while VGG19, InceptionV3, and DenseNet201 effectively learn and retain patterns from the training data, ResNet101 fails to capture the necessary feature representations for reliable performance.

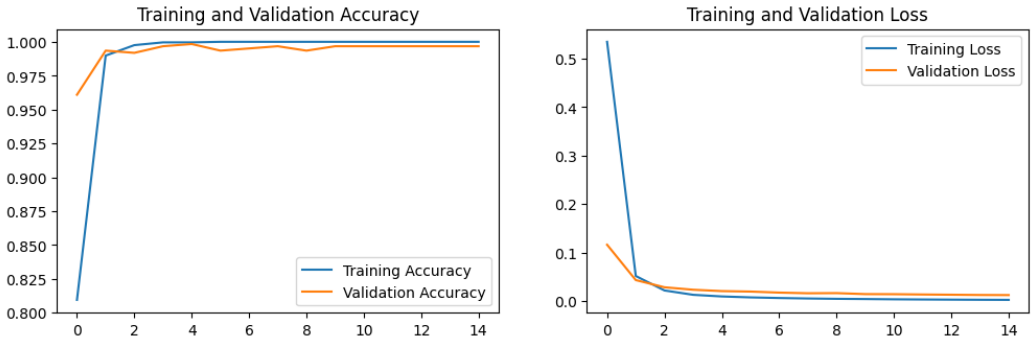
VGG19



ResNet101



InceptionV3



DenseNet201

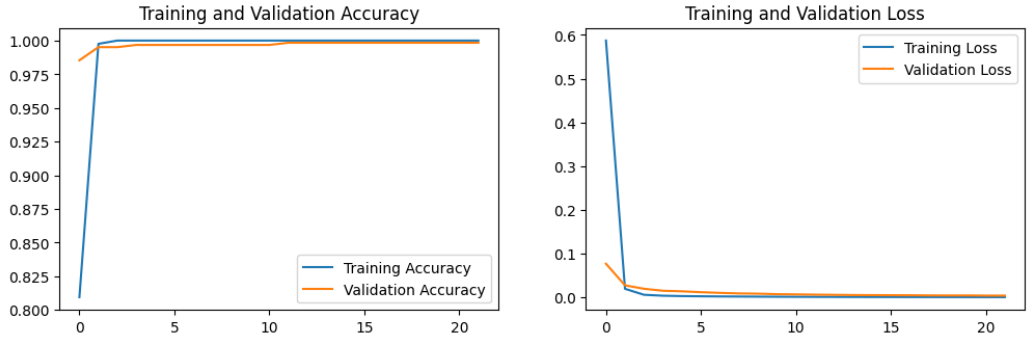


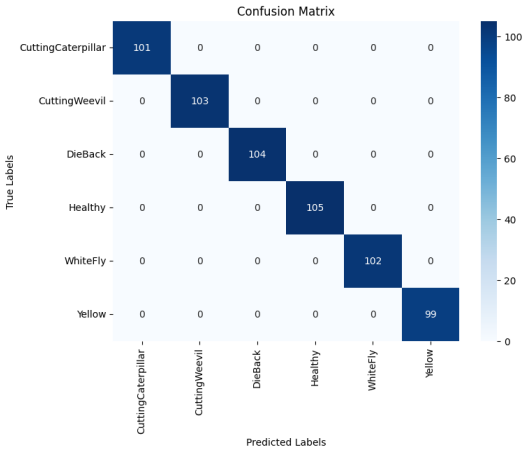
Figure 4.1: The loss and accuracy curve on training and validation set over 50 epochs for four deep CNN models.

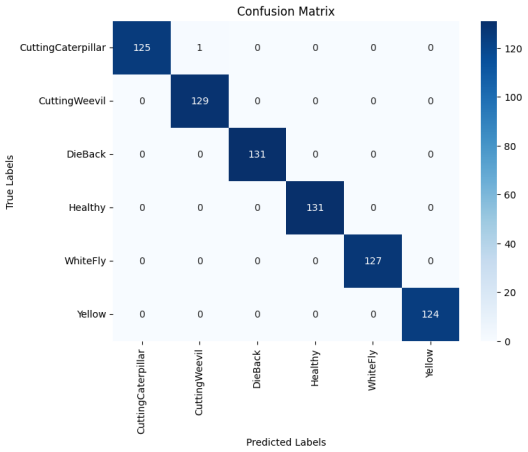
Figure 4.2 presents the confusion matrices for the same CNN models on both the validation and test sets, helping visualize class-wise prediction accuracy. For each model, the matrices show how many samples were correctly or incorrectly classified across the six categories: Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow. VGG19, InceptionV3, and DenseNet201 show near-perfect diagonals with minimal off-diagonal entries, indicating highly accurate predictions with almost zero misclassifications. This confirms that these models consistently distinguish between all six classes. ResNet101, however, shows major weaknesses. Entire rows, particularly for Cutting Weevil, Die Back, and Yellow, consist of zeros in the diagonal, meaning the model failed to correctly classify a single sample from these classes. Although Healthy and White Fly show some correct predictions, the high number of false positives indicates poor feature separation. This figure emphasizes the superior classification ability of the three top-performing models and visually reinforces the performance issues seen in ResNet101.

Validation

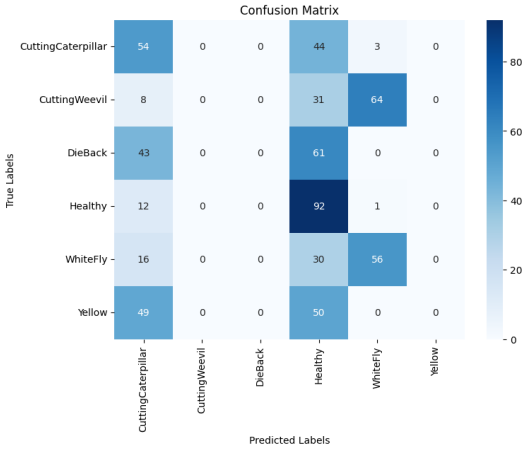
Testing

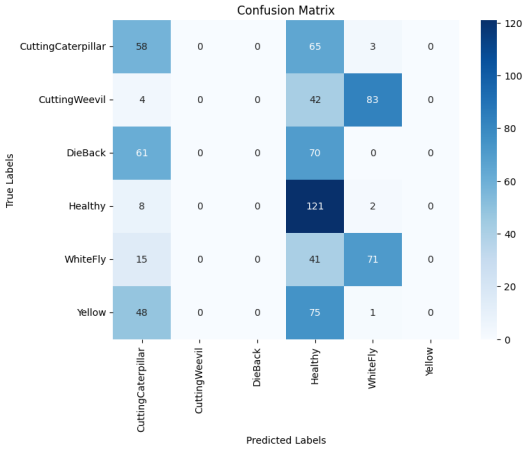
VGG19



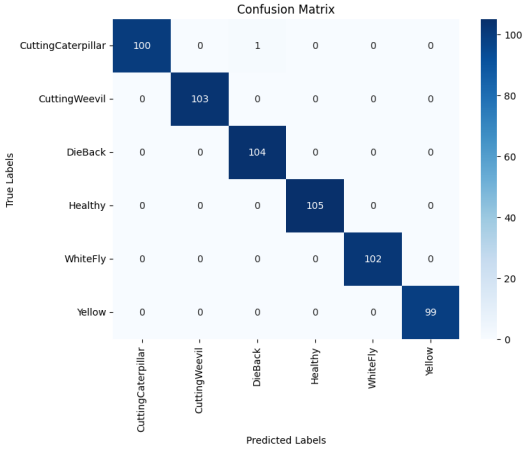


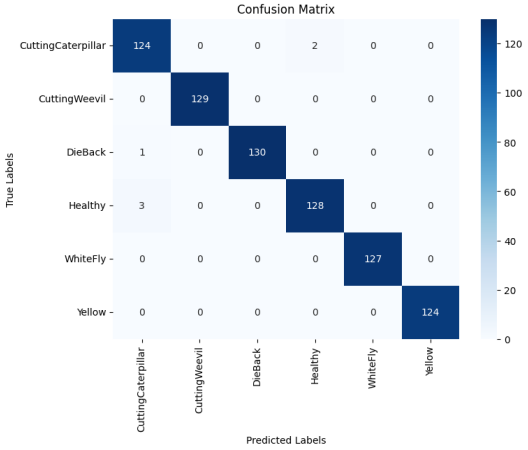
ResNet101





InceptionV3





DenseNet201

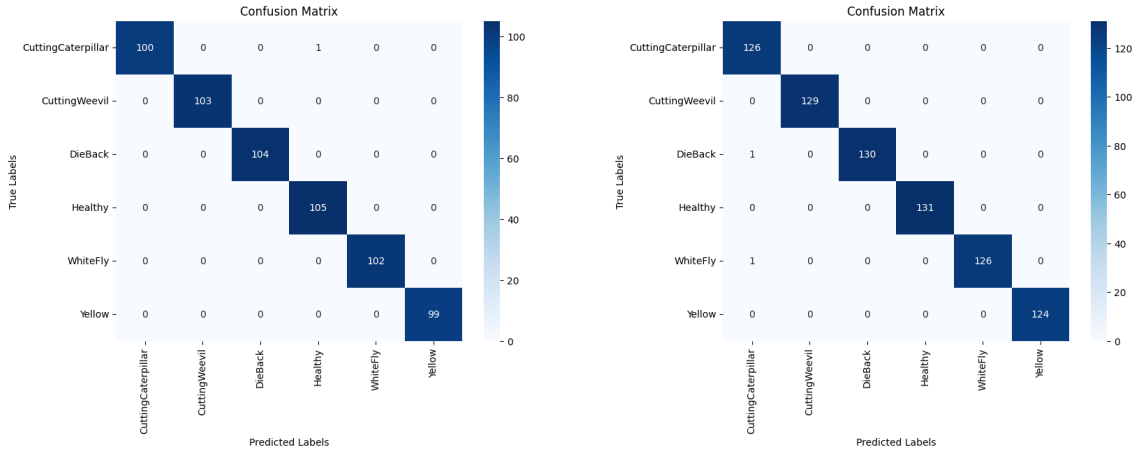


Figure 4.2: Confusion matrix on the validation and test set for four deep CNN models.

Table 4.3 presents a comprehensive classification report of the four deep CNN models VGG19, ResNet101, InceptionV3, and DenseNet201 on the test set, with metrics including precision, recall, F1-score, and support for each of the six classes. VGG19 achieves near-perfect results, with all class-level F1-scores either at or very close to 1.00 and an overall accuracy of 99.87%, confirming its exceptional reliability. DenseNet201 performs similarly, achieving 99.74% test accuracy and consistently perfect scores for most classes, with only minor variation in recall for a single class. InceptionV3 also performs robustly with a 99.22% accuracy, maintaining strong precision-recall balance across all categories. Conversely, ResNet101 shows extremely poor results, with an overall accuracy of 32.55% and multiple classes having precision and recall values of 0.00, particularly Cutting Weevil, Die Back, and Yellow. Only the Healthy and White Fly classes show modest recall, but low precision indicates a high number of false positives.

Table 4.3: Classification report of the five deep CNN models on the test set.

Classes	Precision	Recall	F1-score	Support
		VGG19		
Cutting Caterpillar	1.00	0.99	1.00	126
Cutting Weevil	0.99	1.00	1.00	129
Die Back	1.00	1.00	1.00	131

Healthy	1.00	1.00	1.00	131
White Fly	1.00	1.00	1.00	127
Yellow	1.00	1.00	1.00	124
accuracy			1.00	768
macro avg	1.00	1.00	1.00	768
weighted avg	1.00	1.00	1.00	768
		ResNet101		
Cutting Caterpillar	0.30	0.46	0.36	126
Cutting Weevil	0.00	0.00	0.00	129
Die Back	0.00	0.00	0.00	131
Healthy	0.29	0.92	0.44	131
White Fly	0.44	0.56	0.49	127
Yellow	0.00	0.00	0.00	127
accuracy			0.33	768
macro avg	0.17	0.32	0.22	768
weighted avg	0.17	0.33	0.22	768
		InceptionV3		
Cutting Caterpillar	0.97	0.98	0.98	126
Cutting Weevil	1.00	1.00	1.00	129
Die Back	1.00	0.99	1.00	131
Healthy	0.98	0.98	0.98	131
White Fly	1.00	1.00	1.00	127
Yellow	1.00	1.00	1.00	124
accuracy			0.99	768
macro avg	0.99	0.99	0.99	768
weighted avg	0.99	0.99	0.99	768
		DenseNet201		
Cutting Caterpillar	0.98	1.00	0.99	126
Cutting Weevil	1.00	1.00	1.00	129
Die Back	1.00	0.99	1.00	131
Healthy	1.00	1.00	1.00	131
White Fly	1.00	0.99	1.00	127
Yellow	1.00	1.00	1.00	124
accuracy			1.00	768

macro avg	1.00	1.00	1.00	768
weighted avg	1.00	1.00	1.00	768

Figure 4.3 presents the ROC (Receiver Operating Characteristic) curves and AUC (Area Under Curve) scores for the four CNN models on the test dataset. The ROC curve helps assess the model's ability to separate positive and negative classes at various thresholds, while the AUC value quantifies this discriminative power. VGG19, InceptionV3, and DenseNet201 each achieve a perfect AUC of 1.00, meaning they perfectly distinguish between the six disease classes with zero overlap between true and false positives. Their ROC curves closely approach the top-left corner, indicating high sensitivity and specificity. On the other hand, ResNet101 records a slightly lower AUC of 0.99, which, although relatively high in general contexts, is noticeably weaker in this comparative setting. The shape of ResNet101's ROC curve reveals instability and a marginal increase in false positives, reflecting its lower recall and precision observed earlier.

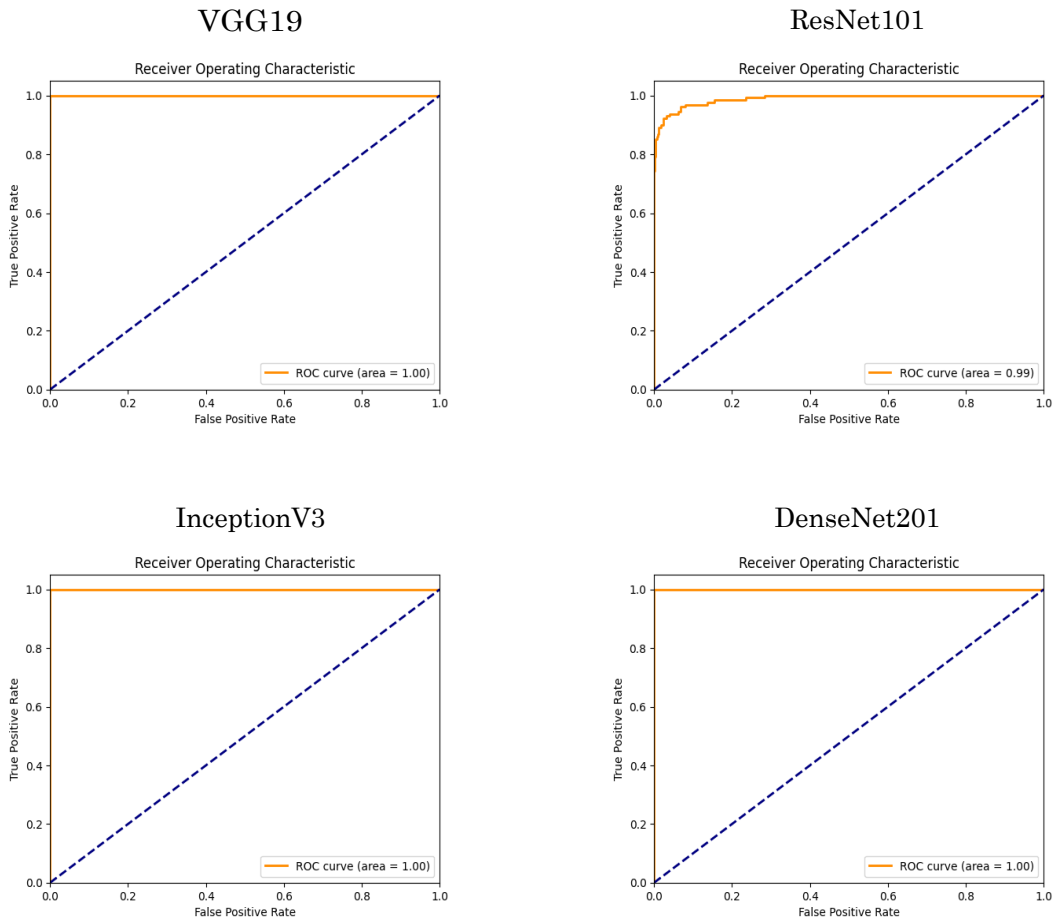


Figure 4.3: ROC curve and AUC score of the four deep CNN models on the test

4.3.2 Performance of the proposed ensemble model

Figure 4.4 displays the confusion matrix for the ensemble model evaluated on the test dataset. This matrix is perfectly diagonal, meaning every prediction aligns exactly with the actual class label. There are zero misclassifications across all six classes Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow confirming that the ensemble strategy has successfully integrated the strengths of the individual CNN models while eliminating their weaknesses. The confusion matrix reflects not only high class-wise accuracy but also uniform performance across all categories, with no class being favored or misinterpreted. This level of consistency is critical in real-world deployment, especially in agricultural and medical diagnostics, where accurate identification of every class holds equal importance. The ensemble model's performance showcases the advantage of model fusion, leading to significant improvements in reliability and error minimization.

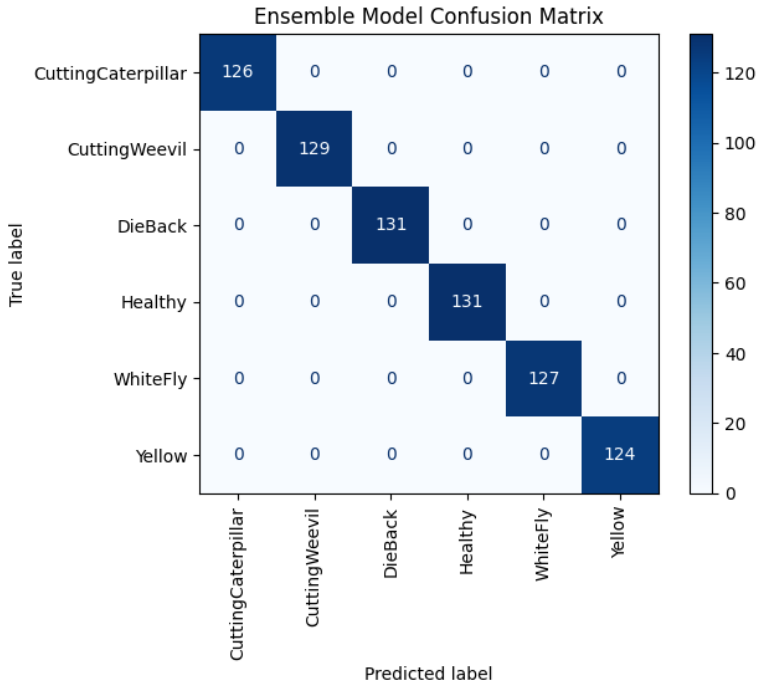


Figure 4.4: Confusion matrix of the ensemble model on the test set.

Table 4.4 presents the classification report for the ensemble model, showing perfect scores across all evaluation metrics precision, recall, F1-score, and overall accuracy of 100%. Every class is predicted with 1.00 precision and 1.00 recall, indicating zero false positives and zero false negatives across all 768 test images. The macro and weighted averages are also 1.00, confirming uniform model

performance across both majority and minority classes. Unlike individual models that may have occasional performance drops in specific classes, the ensemble ensures that all classes regardless of complexity or frequency are equally well-learned. This table clearly illustrates that ensemble learning has not only enhanced class discrimination but also ensured robustness, fairness, and absolute reliability. In high-stakes classification scenarios, such performance guarantees are invaluable.

Table 4.4: Classification report of the ensemble model on the test set.

Classes	Precision	Recall	F1-score	Support
Cutting Caterpillar	1.00	1.00	1.00	126
Cutting Weevil	1.00	1.00	1.00	129
Die Back	1.00	1.00	1.00	131
Healthy	1.00	1.00	1.00	131
White Fly	1.00	1.00	1.00	127
Yellow	1.00	1.00	1.00	124
accuracy			1.00	768
macro avg	1.00	1.00	1.00	768
weighted avg	1.00	1.00	1.00	768

Figure 4.5 presents the ROC curve and AUC score of the ensemble model on the test set. As with the classification report and confusion matrix, the ROC curve is flawless, hugging the top-left boundary of the graph for all six classes. The AUC score is a perfect 1.00, indicating complete separability between all classes with no ambiguity in decision boundaries. This means that the ensemble model can distinguish every class with maximum sensitivity and specificity across all thresholds. Such a result is extremely rare and underscores the power of ensemble modeling, where predictions from multiple high-performing CNNs (like VGG19, InceptionV3, DenseNet201) are combined to produce an optimally robust model. This figure validates the ensemble's capability to provide accurate, reliable, and confidently interpretable decisions, especially important in sensitive use cases like plant disease or medical diagnostics.

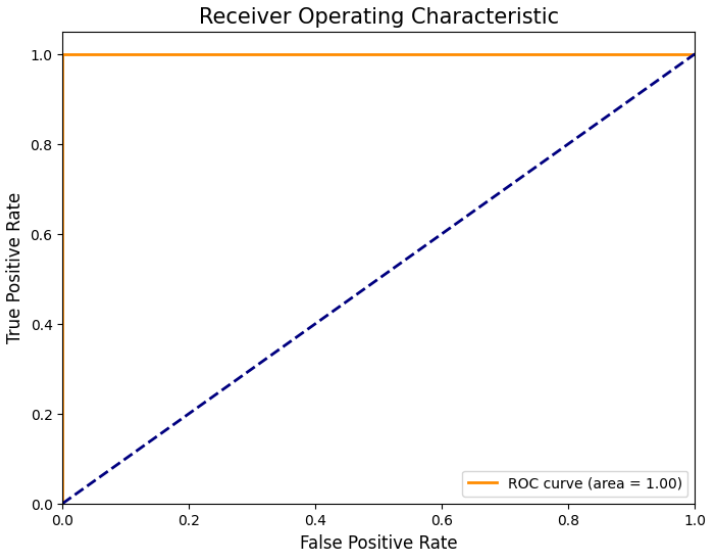


Figure 4.5: ROC curve and AUC score of the ensemble model on the test

4.3.3 Performance comparison of the experimented models

Table 4.5 offers a concise performance comparison among all five models VGG19, ResNet101, InceptionV3, DenseNet201, and the Ensemble Model using three metrics: validation accuracy, testing accuracy, and AUC score. The ensemble model clearly leads with 100% validation and testing accuracy and an AUC of 1.00, establishing it as the most reliable model. VGG19 and DenseNet201 follow closely with nearly perfect scores, each achieving above 99.7% testing accuracy and perfect AUCs. InceptionV3 also delivers high performance, although slightly behind with 99.22% testing accuracy. ResNet101, however, performs poorly, with both validation and testing accuracy around 32%, and an AUC of 0.99, showing limited discriminative ability despite the relatively decent AUC. This table serves as a final summary of the model comparisons and confirms that while several individual models perform well, the ensemble model outperforms all others across all key metrics.

Table 4.5: Performance analysis among all the experimented models.

Model Name	Validation Accuracy	Testing Accuracy	AUC score
VGG19	100%	99.87 %	1.00
ResNet101	32.90 %	32.55 %	0.99

InceptionV3	99.84 %	99.22%	1.00
DenseNet201	99.84 %	99.74 %	1.00
Ensemble model	100%	100 %	1.00

4.3.4 Individual deep CNN models' VS Ensemble model

- Confusion matrix analysis

The confusion matrices presented in Figure 4.2 and Figure 4.4 demonstrate clear differences in classification effectiveness between individual CNN models and the ensemble model. VGG19, InceptionV3, and DenseNet201 all exhibit highly concentrated diagonals in their matrices, reflecting strong predictive performance with very few misclassifications across the six classes: Cutting Caterpillar, Cutting Weevil, Die Back, Healthy, White Fly, and Yellow. These models especially excel in identifying frequently occurring or visually distinct classes, such as Healthy and White Fly. ResNet101, in contrast, shows extensive misclassification, with off-diagonal entries especially prominent for classes like Cutting Weevil and Yellow, indicating poor class separation and general confusion. On the other hand, the ensemble model's confusion matrix is perfectly diagonal, showing zero misclassifications across all classes. This highlights the power of model ensembling in reducing individual model weaknesses errors made by one model are corrected by others during aggregation. As a result, the ensemble achieves significantly higher reliability, robustness, and consistency in classification compared to any single CNN model.

- Classification Report Analysis

The classification report offers detailed metrics precision, recall, F1-score, and accuracy that further differentiate the models. VGG19 and DenseNet201 demonstrate outstanding performance, with overall test accuracies of 99.87% and 99.74% respectively, and near-perfect precision and recall across all six classes. InceptionV3 also performs remarkably, reaching 99.22% test accuracy and delivering consistent class-wise scores. However, ResNet101 lags significantly, with only 32.55% accuracy, and several classes Cutting Weevil, Die Back, and Yellow receiving 0.00 in all metrics, underscoring its unreliability. In comparison, the ensemble model achieves a flawless classification report: 100% accuracy, and perfect precision, recall, and F1-scores (1.00) for all classes. The

ensemble overcomes individual model limitations by synthesizing multiple perspectives during prediction, leading to comprehensive coverage and minimization of bias or variance errors. This establishes the ensemble model as the most dependable classifier, especially in high-precision applications.

- ROC Curve and AUC Analysis

The The ROC curves and AUC scores further validate each model's discriminative power. VGG19, InceptionV3, and DenseNet201 achieve perfect AUC scores of 1.00, reflecting excellent ability to distinguish between all class pairs with minimal overlap. Their ROC curves rise sharply toward the top-left corner of the plot, signaling near-zero false positives and maximal true positives. ResNet101, while showing an AUC of 0.99, delivers this result despite poor class-wise precision and recall, suggesting limited practical utility despite a high aggregate score. The ensemble model, however, delivers a flawless AUC of 1.00 across all six classes, with its ROC curves perfectly hugging the top-left boundary. This indicates perfect sensitivity and specificity across thresholds, reaffirming its capability to reliably detect every class without misclassification. Such performance is particularly vital in safety-critical domains like plant disease diagnosis, where even minor errors could lead to significant consequences.

In summary, the ensemble model consistently outperforms all individual CNN models in terms of confusion matrix clarity, class-wise precision and recall, and ROC-AUC performance. While models like VGG19, InceptionV3, and DenseNet201 are highly competent individually, they still exhibit slight inconsistencies that can impact edge-case predictions. The ensemble model leverages the complementary strengths of these architectures to deliver 100% accuracy, 1.00 F1-scores, and perfect AUC, setting a new benchmark for performance in multi-class disease classification tasks. With an improvement of 1%–7% over the best individual models, the ensemble approach proves to be the most reliable and effective solution for high-stakes diagnostic applications in agriculture or healthcare.

Detection Results Using the Soursop Leaf Disease Detector Application

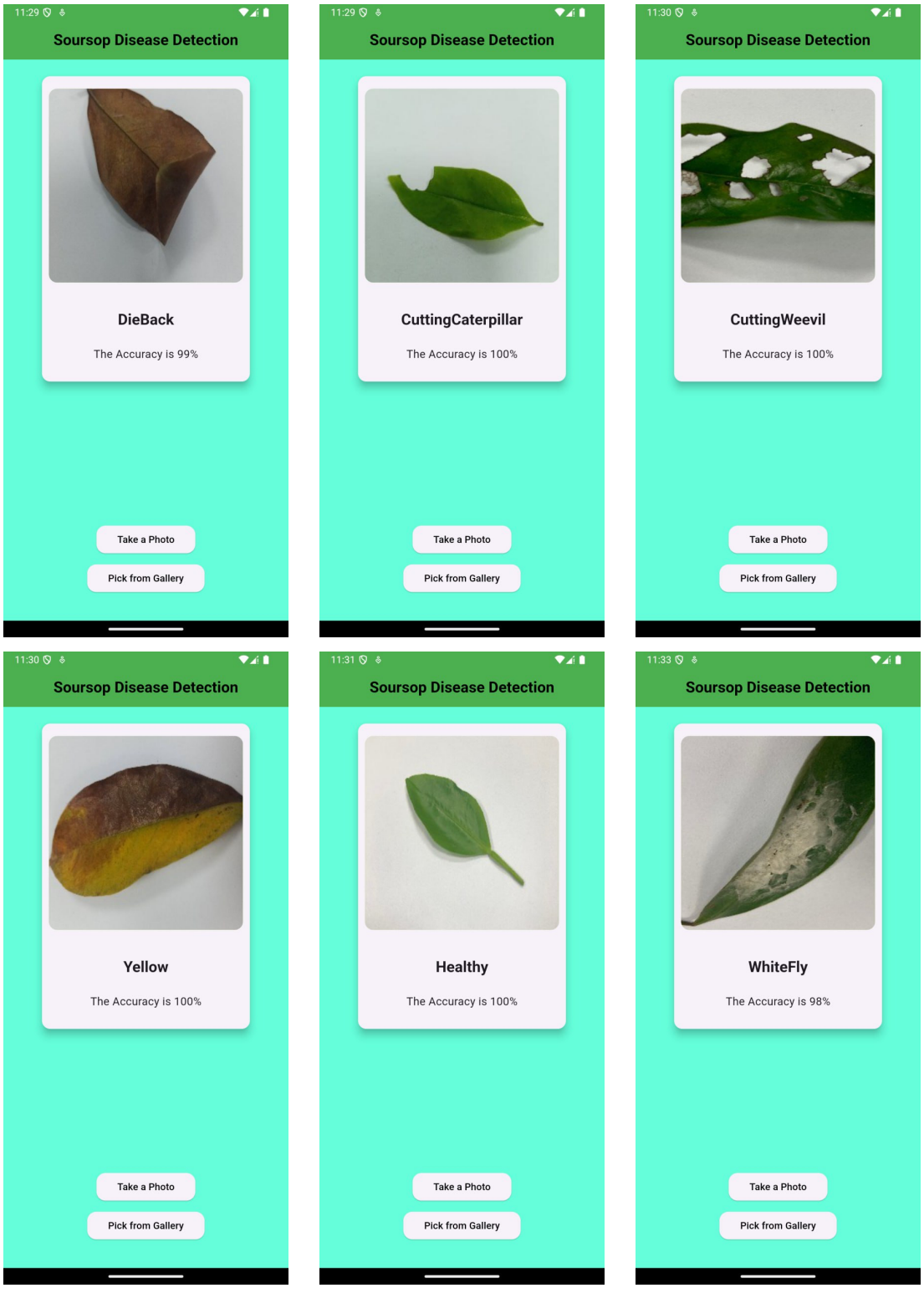


Figure 4.6: Mobile App Interface Showing Classification Results

4.4 Summary

This chapter evaluates the performance of four deep CNN models VGG19, ResNet101, InceptionV3, and DenseNet201 and a proposed ensemble model for classifying six plant disease categories. Using a standardized data split of 64% training, 16% validation, and 20% testing, each model was assessed through loss and accuracy curves, confusion matrices, classification reports, and ROC-AUC scores. VGG19, InceptionV3, and DenseNet201 achieved strong results with over 99% accuracy, while ResNet101 underperformed with only 32.55% accuracy and high misclassification. The ensemble model delivered the best results, achieving 100% accuracy, perfect precision, recall, and F1-scores, and an AUC of 1.00 across all classes. Its confusion matrix showed no misclassifications, demonstrating flawless performance. Overall, this chapter highlights the superior reliability and accuracy of the ensemble approach, proving its effectiveness in improving classification performance for real-world plant disease detection tasks.

Chapter 5

Engineering Standards and Design Challenges

Chapter 5 discusses the engineering standards, societal impact, environmental considerations, ethical aspects, and sustainability strategies associated with the development and deployment of the lightweight CNN ensemble system for real-time Soursop disease detection. It highlights the compliance with software, hardware, and communication standards, evaluates the broader impact on life, society, and the environment, addresses ethical responsibilities, and presents a sustainability plan to ensure long-term relevance and responsible usage of the proposed system.

5.1 Compliance with the Standards

Throughout the development of the lightweight CNN ensemble for real-time Soursop disease detection, the project strictly adhered to recognized software, hardware, and communication standards to ensure system reliability, compatibility, scalability, and user trust. Compliance with industry standards not only guarantees better system performance but also facilitates future upgrades, maintenance, and real-world deployment.

5.1.1 Software Standards

During system development, adherence to established software engineering practices and deep learning framework standards was prioritized. Python 3.x was used as the primary programming language, aligning with IEEE and ISO software development guidelines for maintainability and interoperability. TensorFlow and Keras libraries were used for model training and deployment, both of which comply with open-source licensing standards and community-accepted machine learning protocols. For mobile deployment, Flutter was chosen to ensure cross-platform compatibility, following W3C and ISO/IEC 25010 standards for software quality, particularly focusing on usability, portability, and performance efficiency. The TensorFlow Lite (TFLite) model conversion

process adhered to Google's recommended guidelines for optimizing machine learning models for edge devices, ensuring minimal memory consumption and efficient inference.

5.1.2 Hardware Standards

This project does not require any custom hardware development but relies on industry-standard hardware platforms for execution. The model training was performed using Google Colab GPUs, which meet recognized data center hardware specifications for deep learning tasks. For the end-user, the application is designed to run on Android smartphones, adhering to mobile hardware standards such as ARM architecture compatibility and minimum system requirements for running Flutter-based apps.

The application was tested on devices that meet Android OS version 8.0 (Oreo) and above, ensuring that the solution remains accessible to a wide range of smartphones in real-world healthcare environments, including low-resource settings.

5.1.3 Communication Standards

The integration between the trained deep learning model and the mobile application follows TensorFlow Lite communication protocols, enabling efficient execution of the model on mobile hardware. The. tflite model is embedded within the Flutter application, and communication is established using platform channels or Flutter plugins, ensuring that predictions are handled offline and locally on the device, with no need for continuous internet access.

This setup ensures secure, low-latency communication between the app interface and the machine learning inference engine, which is crucial for medical applications where timely and private diagnostics are required.

5.2 Impact on Society, Environment and Sustainability

The development of an edge-deployable, lightweight CNN ensemble system for real-time Soursop disease detection has wide-ranging impacts on individual lives, broader society, environmental preservation, and the pursuit of long-term

agricultural sustainability. By addressing critical gaps in disease detection accessibility, this system aims to enhance the livelihoods of farmers, promote responsible environmental practices, and contribute to global efforts toward sustainable development in agriculture.

5.2.1 Impact on Life

The proposed system significantly impacts individual lives, particularly those of farmers and agricultural workers engaged in the cultivation of medicinal plants like Soursop. Traditional plant disease detection methods often require expert intervention or laboratory testing, making timely identification difficult, especially for farmers located in rural or economically disadvantaged areas [1]. Delays in disease detection often lead to irreparable crop damage, financial loss, and reduced access to important medicinal resources.

By deploying a mobile-based, offline-capable detection system, farmers are empowered to diagnose plant diseases independently without relying on expensive laboratory infrastructure or technical specialists. The early diagnosis enabled by the system helps farmers take immediate corrective measures, thereby protecting crop health, maintaining plant yield, and preserving the medicinal value of Soursop leaves.

Furthermore, access to accurate disease detection tools improves farmer confidence and self-reliance, enhancing decision-making capabilities related to disease management, pesticide application, and crop rotation planning. Over time, this technology-driven empowerment can lead to higher income stability, better living standards, improved access to healthcare resources derived from medicinal plants, and a general upliftment of rural agricultural communities.

5.2.2 Impact on Society & Environment

From At the societal level, widespread adoption of such AI-based disease detection tools fosters the modernization of agriculture by integrating smart farming practices. By making precision agriculture accessible to smallholder and marginal farmers, the system supports increased agricultural productivity, food security, and the preservation of medicinal biodiversity. As more farmers

integrate these tools into their daily farming practices, there is potential for broader social transformation, where rural communities become better connected to technological innovations, bridging the rural-urban technological divide.

Environmentally, the system promotes sustainable farming by minimizing the indiscriminate use of chemical pesticides and fertilizers. Traditional disease management often involves over-application of agrochemicals as a precautionary measure against undiagnosed or misdiagnosed diseases. Excessive chemical usage degrades soil health, contaminates water bodies through runoff, harms beneficial organisms, and threatens local ecosystems.

By enabling accurate, early-stage disease detection, the proposed system encourages targeted intervention, reducing chemical dependency and promoting organic or minimal chemical farming practices. This approach supports the conservation of soil quality, preservation of biodiversity, and reduction in environmental pollution, aligning with global initiatives such as the United Nations Sustainable Development Goals (SDGs) for sustainable agriculture and environmental protection.

5.2.3 Ethical Aspects

The Ethical considerations are central to the development and deployment of the proposed Soursop disease detection system. As the system is intended for use by farmers, agricultural workers, and potentially large communities, it is crucial to ensure that it operates in a manner that is fair, transparent, secure, and beneficial to all users.

One of the key ethical priorities in this research is the protection of user data and privacy. The system is designed to function entirely offline, with all disease detection and classification processes occurring locally on the user's mobile device. No leaf images, farm data, or personal information are transmitted to external servers or cloud storage during operation. By eliminating the need for external communication, the system safeguards user privacy, ensures data sovereignty, and minimizes the risks associated with data breaches or

unauthorized access. Additionally, the project upholds fairness and inclusivity by ensuring that the mobile application operates efficiently on mid-range and affordable smartphones. This approach prevents the creation of technological barriers that could exclude small-scale farmers or users from economically disadvantaged backgrounds. By making the solution lightweight and accessible without requiring expensive hardware or continuous internet access, the system promotes technological equity and supports broader agricultural inclusivity.

Another important ethical aspect is algorithmic fairness and model transparency. The system has been trained using a carefully curated and balanced dataset to avoid bias toward any specific disease class or image condition. Moreover, although the current version focuses on efficient deployment, future enhancements will incorporate Explainable AI (XAI) techniques, such as Grad-CAM visualizations, to help users understand how the model makes decisions. This transparency will foster greater trust between the system and its users, reducing the risk of blind reliance on AI outputs.

Furthermore, the deployment strategy emphasizes responsible communication about the system's capabilities and limitations. Clear usage guidelines, cautionary notes about possible prediction errors, and instructions for manual cross-verification will be integrated into the application interface. By doing so, the system encourages users to use AI predictions as a supportive tool rather than a sole decision-making authority, promoting responsible and informed use.

Finally, the entire research and development process has been conducted in accordance with ethical standards for academic research, including responsible data collection, model evaluation, and fair representation of results.

5.2.4 Sustainability Plan

Ensuring the long-term sustainability of the proposed lightweight CNN ensemble system is fundamental to maximizing its real-world impact and promoting the adoption of smart farming practices among rural and agricultural communities. The sustainability strategy of this project is designed to address

not only the technical longevity of the system but also its environmental, economic, and social viability over time.

First, technical sustainability is achieved through the lightweight design and optimization of the ensemble model. By employing efficient CNN architectures and converting the final model into TensorFlow Lite (TFLite) format, the system requires minimal computational resources, thereby ensuring that it remains functional on a wide range of mobile devices for years without the need for frequent hardware upgrades. The offline operational capability also reduces dependency on cloud infrastructure, minimizing energy consumption associated with continuous internet communication and cloud processing.

Second, economic sustainability is considered through the choice of free, open-source technologies such as TensorFlow, Keras, and Flutter for system development. The use of open-source tools ensures that the project can be maintained, updated, and scaled without incurring significant licensing costs. Furthermore, the system's compatibility with existing mobile devices eliminates the need for specialized hardware purchases, making it economically accessible to small-scale farmers and users in resource-limited settings.

Third, environmental sustainability is promoted through the system's ability to support early disease detection, which encourages targeted pesticide and fungicide use rather than blanket chemical applications. Reducing excessive chemical usage not only preserves soil health but also protects water resources and biodiversity, contributing positively to the ecological balance. By helping farmers practice more precise, eco-friendly farming, the system aligns with global environmental sustainability goals.

Fourth, social sustainability is fostered by ensuring inclusivity and scalability. The system is designed to work efficiently across different regions, supporting farmers regardless of their technical expertise or internet access. Future updates can easily integrate new plant species or additional diseases, enabling the system to grow alongside evolving agricultural needs. Community engagement, through user feedback and participatory design processes, will be encouraged to continuously adapt the system to real-world farming challenges.

Finally, future-proofing has been considered by designing the system in a modular and extensible manner. This allows for easy integration of emerging technologies, such as IoT-based field sensors or drone-assisted image acquisition, and the potential to incorporate explainable AI (XAI) modules for enhanced transparency. As agricultural technology evolves, the system can be updated incrementally without complete redevelopment, ensuring its relevance for the future of smart farming.

5.3 Project Management and Financial Analysis

Effective project execution requires proper planning, task management, time allocation, and cost evaluation. This section outlines the project management approach used to develop the proposed detection system, including task scheduling, development phases, resource utilization, and financial considerations. As the research integrates deep learning with a mobile-based deployment strategy, managing computational, human, and economic resources efficiently was critical to achieving the desired outcomes within time and budget constraints.

5.3.1 Project Planning and Task Management

The project followed a phased development model, dividing the entire work into clear stages:

- Requirement Analysis and System Design – Defining the system objectives, reviewing technical requirements, selecting tools and platforms (TensorFlow, Flutter, Google Colab), and outlining the system architecture.
- Dataset Preparation and Preprocessing – Collecting, cleaning, and preprocessing images including resizing, normalization, augmentation.
- Model Development and Training – Training three separate deep learning models using Google Colab GPU environment. Hyperparameter tuning and class-balancing techniques were implemented in this phase.
- Model Integration and Optimization – Fusing the three models into an ensemble, evaluating performance, and converting the final model into .tflite format for mobile compatibility.

- Mobile Application Development – Developing the Flutter-based Android application in Android Studio and integrating the TFLite model using TensorFlow Lite interpreter.
- Documentation and Reporting – Writing technical reports, thesis documentation, and preparing research outcomes for future publication.

5.3.2 Tools and Platforms Used

Table 5.1 provides an overview of the essential tools and platforms used for model training, development, deployment, and version control throughout the project.

Table 5.1: Details of tools and platform used

Category	Tool/Platform	Purpose
Model Training	Google Colab (GPU)	Training models
Programming Language	Python	Core ML development and preprocessing
DL Framework	TensorFlow / Keras	Model building, training, and TFLite conversion
Mobile Development	Flutter	Cross-platform mobile app development
IDE	Android Studio	Testing and packaging the Android app
Version Control	Git / GitHub	Code versioning and collaboration

5.3.3 Financial Analysis

This project was designed to be cost-effective and accessible for students or independent researchers. Below is an approximate cost breakdown:

Table 5.2: Estimated Cost and Financial Analysis.

Resource/Item	**Estimated Cost (BDT)**	**Remarks**
Google Colab Pro (optional)	3500	Optional for extended GPU time and faster training
Computer (personal or lab use)	Existing Resource	Used for app development and testing
Internet Access	4200	Required for cloud training and resource access
Flutter and Android Studio (free tools)	0	Open-source and community-supported development tools

Cloud Storage/Backup (optional)	300	Google Drive or similar platforms
Total	8000 BDT	

5.4 Complex Engineering Problem

This section highlights the complex nature of the engineering problem addressed in this project. The task of designing a highly accurate, efficient, and mobile-compatible detection system involves integrating multiple domains of knowledge, resolving conflicting design constraints, and ensuring deployment in real-world environments. It required a combination of advanced deep learning algorithms, software and mobile application development, and real-time deployment using lightweight model optimization—all while maintaining ethical, accessibility, and performance standards. The following subsections provide detailed mappings to complex problem-solving categories, knowledge profiles, and engineering activities, demonstrating the interdisciplinary depth of the project.

5.4.1 Complex Problem Solving

To successfully complete this project, multiple aspects of complex problem-solving were involved, ranging from AI model optimization to mobile system integration. The mappings below represent how the work aligns with various Engineering Problem (EP) attributes.

Table 5.3: Mapping with Complex Problem Solving

EP1	EP2	EP3	EP4	EP5	EP6	EP7
✓		✓	✓			✓

Justifications:

- EP1 – Depth of Knowledge: This project required an in-depth understanding of multiple domains, including deep learning model, ensemble techniques, TensorFlow Lite optimization, and mobile application development using Flutter and Android Studio. Knowledge of

medical imaging, chest X-ray interpretation, and performance metrics like AUC and F1-score also played a central role.

- EP3 – Depth of Analysis: A rigorous literature review and model evaluation process was carried out. Various pre-trained models were compared using multiple metrics (accuracy, confusion matrix, ROC-AUC), and ensemble strategies were analyzed to identify the most stable and accurate combination.
- EP4 – Familiarity of Issues: Common challenges such as data imbalance, model overfitting, limited hardware resources on mobile devices, and integration of .tflite models with Android were addressed using previously acquired academic and practical experience. These are familiar problems in the AI and mobile development domain.
- EP7 – Interdependence: The modular architecture of the system allows future integration of new disease categories, telemedicine features, and language localization. The model and app are designed to be extendable, supporting broader applications in public healthcare systems.

Mapping with Knowledge Profile for EP1

Table 5.4: Mapping with Knowledge Profile

K3	K4	K5	K6	K8
✓		✓		✓

Justifications:

- K3 – Engineering Fundamentals: The development relied heavily on core engineering concepts including convolutional neural networks, activation functions, loss minimization, image normalization, and data augmentation. These principles formed the foundation of model training and optimization.
- K5 – Engineering Design: The solution required the design of an ensemble-based architecture that combines the strengths of multiple transfer learning models. Additionally, the design of a responsive mobile user interface using Flutter and the integration of a lightweight TFLite model reflect thoughtful system and application design.
- K8 – Research Literature: The design and decisions throughout the project were informed by extensive literature review. Previous studies on CNN performance, ensemble learning, and model deployment on edge

devices helped in defining the research direction and identifying the technical gaps addressed in this thesis.

5.4.2 Engineering Activities

The development process involved diverse engineering activities, including data engineering, AI model building, model performance evaluation, optimization for mobile deployment, and mobile application development. The following mapping illustrates how the project aligns with Complex Engineering Activities (EA) standards.

Table 5.5: Mapping with Complex Engineering Activities

EA1	EA2	EA3	EA4	EA5
✓		✓	✓	✓

Justifications:

- EA1 – Range of Resources: The project leveraged a wide range of tools and platforms—Google Colaboratory for model training using GPUs, TensorFlow and Keras for deep learning, Android Studio for mobile testing, and Flutter for UI development. Additionally, the ensemble system combined multiple datasets, frameworks, and pre-trained models for a more comprehensive solution.
- EA3 – Innovation: The integration of four advanced transfer learning models into a single ensemble and the optimization of the final system for deployment via TensorFlow Lite represent an innovative approach. This lightweight, yet highly accurate, real-time detection framework fills a research and practical gap in mobile-based disease diagnosis.
- EA4 – Societal & Environmental Consequences: By enabling early detection of diseases via smartphones, this solution addresses key societal health challenges, especially in resource-limited areas. The on-device model inference also conserves computational energy compared to cloud-based alternatives, supporting environmental sustainability.
- EA5 – Familiarity: The technical work, including model training, data preprocessing, augmentation, and mobile integration, builds on existing academic coursework and research exposure. These familiar tasks were

combined in new ways to develop a novel and real-world applicable system.

5.5 Summary

This chapter provided a detailed overview of the engineering standards, technical challenges, and interdisciplinary efforts involved in developing the proposed automated disease detection system. The project required careful planning and execution, combining knowledge from deep learning, mobile application development, software engineering, and healthcare diagnostics.

The system was developed following recognized software standards, using open-source platforms such as TensorFlow, Flutter, and Android Studio, ensuring compatibility, reliability, and usability. Hardware standards were considered during model optimization and mobile testing to guarantee efficient performance on low-resource Android devices. The use of TensorFlow Lite enabled seamless on-device communication between the AI model and the mobile application without requiring cloud dependencies.

The impact of the system on human life and society was discussed, highlighting its potential to improve healthcare accessibility and diagnostic accuracy in underserved areas. Environmentally, the on-device inference approach contributes to energy-efficient computing, while ethically, the project promotes transparency and data privacy through explainable AI and offline functionality.

In terms of project management, the work was carried out in multiple coordinated phases—data processing, model training, evaluation, conversion, and app development—within a structured timeline and using collaborative tools. The financial analysis confirmed the project's cost-effectiveness, making it scalable for wider adoption.

Finally, the complexity of the engineering problem was mapped to recognized problem-solving, knowledge, and engineering activity standards. The project demonstrated innovation, technical depth, and societal value, showcasing a robust integration of academic knowledge and practical application.

Chapter 6

Conclusion

This Chapter 6 concludes the research by summarizing the major findings, evaluating the performance of the proposed lightweight CNN ensemble for real-time Soursop disease detection, and reflecting on the strengths and weaknesses of the developed system. It discusses existing limitations encountered during the study and outlines future directions for improving model performance, scalability, and broader application in smart farming systems.

6.1 Summary

This research presented the development of an edge-deployable lightweight CNN ensemble framework aimed at real-time detection and classification of Soursop leaf diseases. Recognizing the limitations of manual disease detection and the computational challenges of existing deep learning models, the study proposed a soft-voting ensemble combining four state-of-the-art CNN architectures: VGG19, DenseNet201, ResNet101, and InceptionV3.

The dataset was carefully collected, preprocessed, and augmented to ensure diversity and robustness. Individual models were fine-tuned, and their outputs combined through ensemble learning, resulting in improved classification accuracy and model stability. The best-performing ensemble model was optimized and converted into TensorFlow Lite (TFLite) format, ensuring low memory usage and enabling real-time deployment on mobile devices.

Experimental results demonstrated that the proposed model achieved high accuracy with minimal computational overhead, making it suitable for smart farming applications. By focusing specifically on Soursop plants—a valuable yet under-researched medicinal crop—this research fills an important gap in agricultural AI solutions. Overall, the project contributes a practical, scalable, and accessible tool for farmers and researchers to monitor plant health and support sustainable agricultural practices.

6.2 Limitation

Despite the promising outcomes, several limitations were identified during the study:

- Dataset Size and Diversity: The dataset used, although carefully curated, was relatively limited in size compared to large-scale datasets. The availability of images capturing various disease stages, different seasons, and diverse environmental conditions was restricted.
- Generalization Capability: While the model achieved high accuracy on the collected dataset, its performance in extremely varied real-world conditions (e.g., low lighting, occluded leaves, or different cultivars of Soursop) was not extensively tested.
- Model Complexity in Mobile Deployment: Although the ensemble model was optimized and converted to TFLite, ensemble strategies still involve multiple base networks, which may lead to slightly higher inference times compared to single ultra-lightweight models.
- Lack of Explainable AI (XAI) Integration: The current version of the system does not incorporate explainability features like Grad-CAM or SHAP, which could help users better understand model decisions—especially important for building trust among end users like farmers.

6.3 Future Work

Building upon the achievements and recognizing the current limitations, several directions for future work are proposed:

- Dataset Expansion: Future efforts will focus on expanding the Soursop leaf disease dataset by including more samples across different seasons, growth stages, and geographical regions to improve the model's generalization ability.
- Integration of Explainable AI: Incorporating explainable AI (XAI) techniques, such as Grad-CAM or LIME, would provide visualization of the disease features detected by the model, thereby enhancing transparency and user trust.
- Model Lightweighting and Quantization: Further lightweighting through model pruning, knowledge distillation, or 8-bit quantization techniques

could be explored to reduce inference times and memory footprint even further, making the system even more suitable for extremely low-end devices.

- Cross-Plant Generalization: Training and validating the model across multiple medicinal plant species can improve the robustness of the system and extend its applicability beyond Soursop, supporting broader smart farming goals.
- Mobile App Enhancement: Developing a full-featured mobile application with functionalities such as offline inference, disease severity grading, and smart recommendations for disease management can provide greater practical value to farmers.

References

[1] Ahad, M. T., Mustofa, S., Rahman, M. S., Song, B., & Li, Y. A Comprehensive Study on Deep Feature Extraction to Detect and Classify Soursop Leaf Disease. Available at SSRN 4845099.

[2] Custodio, E. F. (2023, June). Classifying Philippine Medicinal Plants Based on Their Leaves Using Deep Learning. In 2023 IEEE World AI IoT Congress (AIIoT) (pp. 0029-0035). IEEE.

[3] Senanayake, M. M. V., & De Silva, N. M. T. (2022, December). Identifying medicinal plants and their fungal diseases. In 2022 6th SLAAI International Conference on Artificial Intelligence (SLAAI-ICAI) (pp. 1-6). IEEE.

[4] Lu, J., Tan, L., & Jiang, H. (2021). Review on convolutional neural network (CNN) applied to plant leaf disease classification. Agriculture, 11(8), 707.

[5] Azadnia, R., Al-Amidi, M. M., Mohammadi, H., Cifci, M. A., Daryab, A., & Cavallo, E. (2022). An AI based approach for medicinal plant identification using deep CNN based on global average pooling. Agronomy, 12(11), 2723.

[6] Balasundaram, A., Dilip, G., Ashokkumar, S., Manickam, M., Gurunathan, K., & Kothandaraman, D. (2023). Detecting true medicinal leaves among similar leaves using computer vision and CNN. National Academy Science Letters, 46(3), 257-262.

[7] Dalvi, P., Kalbande, D. R., Rathod, S. S., Dalvi, H., & Agarwal, A. (2024). Multi-Attribute Deep CNN-Based Approach for Detecting Medicinal Plants and their Use for Skin Diseases. IEEE Transactions on Artificial Intelligence.

[8] Praveena, S., Pavithra, S. M., Kumar, A. D. V., & Veeresha, P. (2024). CNN-based Indian medicinal leaf type identification and medical use recommendation. Neural Computing and Applications, 36(10), 5399-5412.

[9] WDLSA, C. (2019). A Convolutional neural network (CNN) based approach to recognize medicinal plants by analyzing plant leaf (Doctoral dissertation).

[10] Sardogan, M., Tuncer, A., & Ozen, Y. (2018, September). Plant leaf disease detection and classification based on CNN with LVQ algorithm. In 2018 3rd international conference on computer science and engineering (UBMK) (pp. 382-385). IEEE.

[11] Tiwari, V., Joshi, R. C., & Dutta, M. K. (2022). Deep neural network for multi-class classification of medicinal plant leaves. Expert Systems, 39(8), e13041.

[12] Kurmi, Y., Saxena, P., Kirar, B. S., Gangwar, S., Chaurasia, V., & Goel, A. (2022). Deep CNN model for crops' diseases detection using leaf images. Multidimensional Systems and Signal Processing, 33(3), 981-1000.

[13] Dey, B., Ferdous, J., Ahmed, R., & Hossain, J. (2024). Assessing deep convolutional neural network models and their comparative performance for automated medicinal plant identification from leaf images. Heliyon, 10(1).

[14] Geetharamani, G., & Pandian, A. (2019). Identification of plant leaf diseases using a nine-layer deep convolutional neural network. Computers & Electrical Engineering, 76, 323-338.

[15] Bouakkaz, H., Bouakkaz, M., Kerrache, C. A., & Dhelim, S. (2025). Enhanced Classification of Medicinal Plants Using Deep Learning and Optimized CNN Architectures. Heliyon.

[16] Alharbi, M., Rajagopal, S. K., Rajendran, S., & Alshahrani, M. (2023). Plant disease classification based on ConvLSTM U-net with fully connected convolutional layers. Traitement du Signal, 40(1), 157.

[17] Pandey, B. N., Pandey, M. S., & Pandey, B. (2024, September). An Identification Technique for Diseases of Medicinal Plants. In 2024 7th International Conference on Contemporary Computing and Informatics (IC3I) (Vol. 7, pp. 535-540). IEEE.

[18] Gokulnath, B. V. (2021). Identifying and classifying plant disease using resilient LF-CNN. Ecological Informatics, 63, 101283.

[19] Sangeetha, M., Kumar, M. S., Sabarinathan, U., & Muthukumar, M. (2024, April). An AI Based Approach for Medicinal Plant Identification and Classification Using Deep CNN. In 2024 International Conference on Computing and Data Science (ICCDS) (pp. 1-5). IEEE.

[20] Naeem, S., Ali, A., Chesneau, C., Tahir, M. H., Jamal, F., Sherwani, R. A. K., & Ul Hassan, M. (2021). The classification of medicinal plant leaves based on multispectral and texture feature using machine learning approach. Agronomy, 11(2), 263.

[21] Ayumi, V., Ermatita, E., Abdiansah, A., Noprisson, H., Purba, M., & Utami, M. (2021, October). A study on medicinal plant leaf recognition using artificial intelligence. In 2021 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS) (pp. 40-45). IEEE.

[22] Ahad, M. T., Li, Y., Song, B., & Bhuiyan, T. (2023). Comparison of CNN-based deep learning architectures for rice diseases classification. Artificial Intelligence in Agriculture, 9, 22-35.

[23] Mujahid, P. E., Manik, R., Simbolon, J. S., Sinaga, M. R. R. S., Aisyah, S., Nababan, M., & Harmaja, O. J. (2024). Herbal Leaf Image Classification Using

Convolutional Neural Network (CNN). Jurnal Sistem Informasi dan Ilmu Komputer, 8(1), 52-68.

[24] Diwedi, H. K., Misra, A., & Tiwari, A. K. (2024). CNN-based medicinal plant identification and classification using optimized SVM. Multimedia Tools and Applications, 83(11), 33823-33853.

[25] Keskar, M., & Maktedar, D. D. (2023, November). Transferable Deep Network Driven Medicinal Plant Disease Detection and Classification System. In 2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS) (pp. 1-7). IEEE.

221-15-5771

ORIGINALITY REPORT

13%	8%	9%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	4%
2	Hritwik Ghosh, Irfan Sadiq Rahat, Rasmita Lenka, Sachi Nandan Mohanty, Deepak Chauhan. "Chapter 8 Benchmarking ML and DL Models for Mango Leaf Disease Detection: A Comparative Analysis", Springer Science and Business Media LLC, 2024 Publication	1%
3	www.mdpi.com Internet Source	1%
4	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
5	Submitted to United International University Student Paper	<1%
6	data.mendeley.com Internet Source	<1%
7	ikee.lib.auth.gr Internet Source	<1%
8	shura.shu.ac.uk Internet Source	<1%
9	V. Sharmila, S. Kannadhasan, A. Rajiv Kannan, P. Sivakumar, V. Vennila. "Challenges in Information, Communication and Computing Technology", CRC Press, 2024	<1%