An Explainable Artificial Intelligence-Driven Minimal Net for Tea Leaf Disease Detection

By
Faruk Ahmed
212-15-4205

FINAL YEAR DESIGN PROJECT REPORT

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

Supervised by

Dr. Md. Taimur Ahad
Associate Professor
Department of Computer Science and Engineering Daffodil International University

Co-Supervised by

Mr. Asraf Ullah Rahat
Lecturer
Department of Computer Science and Engineering Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

May 14, 2025

APPROVAL

This Project titled **"An Explainable Artificial Intelligence-Driven Minimal Net for Tea Leaf Disease Detection"**, submitted by Faruk Ahmed, ID No: **212-15-4205** 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. S.M Aminul Haque — **Chairman**
Professor & Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Sharmin Akter — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Ms. Syada Tasmia Alvi — **Internal Examiner**
Sr. Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

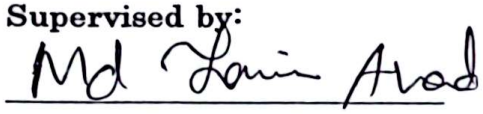
14.05.25

Dr. Md. Arshad Ali — **External Examiner**
Professor
Department of Computer Science and Engineering
Hajee Mohammad Danesh Science & Technology University

DECLARATION

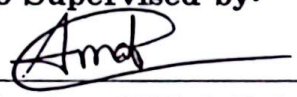
We hereby declare that this project has been done by us under the supervision of **Dr. Md. Taimur Ahad, Associate Professor**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this projectnor any part of this project has been submitted elsewhere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

Supervised by:



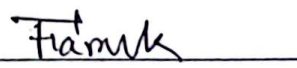
Dr. Md. Taimur Ahad
Associate Professor
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised by:



Mr. Asraf Ullah Rahat
Lecturer
Department of Computer Science and Engineering
Daffodil International University

Submitted by:



Faruk Ahmed
Student ID: 212-15-4205
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 **Dr. Md. Taimur Ahad**, **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 them at 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

Tea leaf diseases pose a significant threat to crop yield and quality, especially in regions where agriculture is a key economic sector. Accurate and real-time detection of tea leaf diseases remain a significant challenge due to limitations in existing automated approaches, including heavy model architectures, a lack of interpretability, and poor adaptability in low-resource environments. This study introduces a comprehensive and scalable framework for the automated detection of tea leaf diseases, integrating deep learning, explainable AI, and lightweight deployment mechanisms. Initially, five fine-tuned Convolutional Neural Networks (CNNs) and five transfer learning models were evaluated, and MobileNetV2 and VGG16 achieved the highest accuracies of 99.52% and 99.68%, respectively. However, VGG16 in the CNN stage and Xception in the transfer learning setting had relatively lower accuracy rates of 88.94% and 73.50%, respectively, which indicates the need for a more balanced and efficient approach. To address this, a lightweight yet high-performing CNN model, M-Net, was developed. With only 49,292 parameters, M-Net achieved a competitive accuracy of 97.29%, along with 97.12% precision and recall, and a low false alarm rate of 0.54%, making it ideal for real-time field deployment. The model's transparency was enhanced through the integration of LIME and SHAP that offer clear visual justifications for predictions. Finally, M-Net was successfully deployed as both a mobile application and a Streamlit-based web app, each capable of detecting all disease classes with high confidence. This work demonstrates the feasibility of deploying intelligent, explainable, and resource-efficient AI solutions in agricultural diagnostics to support smart farming and timely disease management.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter provides an overview of the thesis by introducing the background and significance of the study. It outlines the key motivation for addressing tea leaf diseases using deep learning. The adopted methodology is briefly described, which covers data collection, model development, explainability integration, and deployment. The chapter also presents the expected outcomes of the project and concludes with an outline of the overall structure of the report.

1.1 Introduction

Tea stands as one of the most popular non-alcoholic beverages globally and second only to freshwater in consumption [1], [2]. Besides its cultural significance, tea is increasingly recognized for its medicinal properties that make it a valued 'health drink' [3]. Moreover, tea cultivation plays a crucial role in the economies of several countries, including Bangladesh, which ranks 8th in cultivation area, 10th in production, and 12th in exports [4]. In 2020, Bangladesh produced 86.39 million kgs of processed tea from 167 estates with 2.17 million kgs exported [5]. Nevertheless, the tea industry is confronted with enormous threats from diseases like Tea Algae Leaf Spot (TALS), Tea Bud Blight (TBB), Tea White Scab (TWS), and Tea Leaf Blight (TLB) [6]. To note, these diseases can cause losses up to 20% of the annual yield, and thus seriously impoverish the producers [7]. Moreover, these diseases influence the quality, nutritional value, and the total volume of crops produced [8], a fact that may undermine the state of the world tea production chain and impede sustainable development.

Precision in the detection of tea leaf diseases is accompanied by various challenges arising from complex plantation settings, wide variability in the leaf textures, as well as the minimalistic signs in the initial stages of leaf infection [10]. Most of the traditional detection methods rely upon visual inspection, which is resource-intensive, subjective and error-prone [11]. Additionally, the conditions of tea plantations as conceptually dense, heterogeneous in terms of light and background involvement, often underestimate the application of the traditional methods of identification [12]. In addition, delayed identification causes more crop loss and economic problems for tea farmers. To solve these problems deep learning-based automated detection systems have emerged as a

promising approach that provides accurate identification of diseases [11]. Such systems can reduce crop losses; pesticide use and encourage sustainable farming. Recently, image analysis and pattern recognition based state-of-the-art (SOTA) convolutional neural network (CNN) have considerably progressed the tea leaf disease detection field [13]. These models can automatically learn hierarchical feature representations for raw image data that enables them to detect complex patterns for different tea leaf diseases [14], [15]. By use of layers of convolution and pooling layers and activation functions SOTA CNNs learn from scratch basic features such as color, texture, and shape important for distinguishing between healthy and diseased leaves [12], [16]. Therefore, SOTA CNNs have been very effective in coping with complex backgrounds and different leaf patterns and can therefore be applied effectively in the tea factory in field [17].

Based on this foundation, transfer learning has developed an effective method in tea leaf disease diagnosis to train already trained models on tea datasets [7], [18]. This technique makes the training process much shorter; moreover, it improves performance especially when there are not many annotated tea leaf images to work on. These models can be dedicated to fine-tuning task of tea leaf disease detection by transferring knowledge from generalized visual features learnt from large scale datasets [19]. Transfer learning's capacity to adjust the existing models to particular detection tasks increases their preciseness and robustness. Additionally, this method addresses the issue of overfitting especially when working with smaller domain-specific datasets typical of agricultural research [16], [20]. Motivated by the success of SOTA CNNs and transfer learning, lightweight CNN architectures have lately emerged as an attractive possibility to mitigate the shortcomings of conventional deep CNNs in the areas of plant disease detection. These models are built with fewer layers and with fewer parameters, thus reducing computationally complexity and resources consumption yet being accurate without trading off performance [21]. This makes lightweight CNNs easily deployable in such resource-strapped systems as mobiles and embedded systems used to support agriculture [22]. Through the exploitation of methods such as pruning, quantization and efficient architectural design, lightweight CNNs offer high performance without holding the inference speed or memory usage back [23]. Their adoption helps to connect the dots between some SOTA CNNs that are quite powerful, but expensive to build, and the need to have on field disease detection tools that are practical to use, thereby making it possible for farmers to access accurate and timely disease identification even with the shortage of computational resources on their end [14], [24].

Further development of CNN models into Explainable Artificial Intelligence (XAI), has been of integral importance in the identification of tea leaf disease. An example would be XAI techniques using approximation of an interpretable

model over the CNN's process of decision-making with the help of Local Interpretable Model Agnostic Explanations (LIME) to provide transparency [22]. In a similar way, Shapley Additive Explanations (SHAP) allocates significance scores to single features which give both local and global explanation to what the model is doing [25]. Furthermore, Class Activation Mapping (CAM) detects the spatial areas in the images the model finds useful for classification as well as highlighting the areas of interest [26]. In the case of tea leaf disease detection, the combination of XAI with CNNs not only enhances the trustworthiness of the system but also helps the researchers understand and elevate the system's pursuit of the relevant disease features.

Despite advancements in tea leaf disease detection using CNN models, several research gaps persist, limiting the reliability and broader applicability of existing solutions.

- Many studies, like Rahman et al. [27], highlight issues with imbalanced datasets, leading to weak generalization and unreliable predictions. Similar challenges exist in tea leaf disease research, where reliance on small, secondary datasets increases overfitting risks and reduces model reliability, limiting generalization across diverse tea plant varieties and conditions.
- Existing research often focuses on conventional CNNs like VGG16 or ResNet, overlooking SOTA architectures and transfer learning for better feature extraction [28]. Additionally, diverse ensemble strategies within a single dataset remain underexplored, limiting potential improvements in model robustness [22].
- The absence of thorough validation strategies, such as k-fold cross-validation, weakens the credibility of many tea leaf disease detection models. Custom CNN models proposed by researchers like Liang et al. [13] and Ding et al. [28], often lack robust testing mechanisms to ensure model stability and generalization across varied datasets. This weakens the credibility of model performance and limits their practical application in real-world scenarios.
- While lightweight CNN architectures have been proposed by Albanese et al. [26] and Fu et al. [32] to tackle computational complexity and deployment challenges, these studies rarely integrate Explainable AI (XAI) techniques. This lack of interpretability limits user trust in AI predictions, hindering broader adoption in agricultural applications.

1.2 Motivation

The primary motivation for undertaking this research stems from the convergence of three pressing needs: ensuring agricultural sustainability, enabling practical deployment of machine learning solutions, and addressing the

rising demand for interpretable and transparent AI systems. From a socio-economic perspective, tea cultivation is a cornerstone of Bangladesh's agricultural sector. The Sylhet and Moulvibazar regions collectively contribute over 90% of the country's total tea production, supporting more than 300,000 workers directly and indirectly [29][30]. According to the Bangladesh Tea Board, Bangladesh producing 86.39 million kilograms and exporting 2.17 million kilograms, valued at 347.14 million Bangladesh Taka [31]. However, these gains are increasingly threatened by preventable leaf diseases, which have been reported to reduce yield by 20–30% annually if not identified and controlled early [32]. Therefore, protecting the tea crop from such diseases is critical for maintaining food security, ensuring farmer livelihoods, and sustaining national economic contributions.

From a technical standpoint, while Convolutional Neural Networks (CNNs) have revolutionized image classification tasks including medical imaging, remote sensing, and agricultural diagnostics, their large parameter size and high computational demands pose a serious barrier to their deployment in field-based, low-power environments. Models like ResNet50 or InceptionV3, for instance, require hundreds of megabytes in storage and billions of floating-point operations, making them impractical for mobile deployment or integration into embedded systems such as Raspberry Pi or smartphones commonly used by farmers. This highlights the need for lightweight, memory-efficient CNN architecture that can be practically utilized in real-time scenarios without sacrificing classification accuracy. Moreover, the urgent call for trustworthy AI systems is a major barrier to AI adoption in agriculture. At present, the so-called "black box" nature of deep learning models provides predictions without any insight into their decision-making processes. In high-stakes applications such as crop disease detection where decisions can directly impact yield and income, users must be able to understand and trust the system's outputs. This study, therefore, incorporates XAI methods such as LIME, SHAP, and Grad-CAM to bring transparency and interpretability to the prediction process.

1.3 Objectives

The primary aim of this study is to develop a resource efficient and interpretable deep learning system for detecting tea leaf diseases. This study intends to validate that the proposed method not only performs with high accuracy but also proves to be feasible when put into use in real-world settings, particularly in resource-scarce rural areas like the Bangladeshi tea plantations. This study seeks to bridge the performance-interpretability gap by addressing the full pipeline from data collection and preprocessing to model development, explanation generation, and web deployment. The following detailed objectives guide the entire research process:

- To build a robust and diverse tea leaf image dataset with six classes, Algal Leaf, Gray Blight, Healthy, Helopeltis, Looper Infested, and Red Spider, by capturing high-resolution images under varied conditions, followed by preprocessing steps like resizing, normalization, and noise reduction, and applying augmentation techniques such as rotation, flipping, brightness adjustment, and zooming to enhance data diversity and model generalization.
- To assess the effectiveness of deep learning in detecting tea leaf diseases, six popular CNN architectures, VGG19, DenseNet201, InceptionV3, SeResNet152, ResNet152V2, and MobileNetV2 were fine-tuned on the custom tea leaf dataset. Each model's performance was evaluated using classification metrics like accuracy, precision, recall, and F1-score, along with inference time to measure efficiency.
- To investigate the application of transfer learning in the tea leaf dataset. Utilizing pre-trained weights obtained from ImageNet with domain-specific knowledge, this research seeks to analyze both the benefits and limitations of knowledge transfer. It seeks particularly to examine whether the models show signs of domain shift or negative transfer.
- To develop a custom lightweight CNN architecture, Minimal Net (M-Net), for high classification accuracy while maintaining a low parameter count and reduced computational cost. M-Net integrates efficient design elements such as depthwise separable convolutions, batch normalization, and dropout regularization, enabling it to run on edge devices like smartphones, tablets, or Raspberry Pi modules
- To enhance trust and transparency, this objective involves integrating three state-of-the-art XAI methods, LIME, SHAP, and Grad-CAM into the prediction pipeline of the M-Net model. These techniques provide both visual and numerical explanations of how and why certain predictions were made, thereby addressing the "black box" nature of CNNs.
- The final objective is to implement the M-Net into a user-friendly web application using the Streamlit framework. After the testing in Streamlit web, this study will build a mobile application using flutter that allows users to upload tea leaf images and receive real-time disease classification.

1.4 Methodology

The methodology of this study follows a systematic and structured approach, starting with data acquisition and ending with the deployment of an end-to-end explainable disease detection system for tea leaves. The first phase included extensive data collection from tea gardens in Sreemangal in the Sylhet district of Moulvibazar division in Bangladesh. The images were taken at high resolution

using an iPhone and a Samsung camera under different types of conditions to mimic demeanor in the real world. The dataset had initially 2113 images which included six different classes of images: Algal leaf, Gray blight, Healthy, Helopeltis, Looper infested, Red spider. Each image was personally confirmed and labeled in consultation with agricultural designers to ascertain the accuracy and consistency of class annotations. After collecting data, this study proceeds to image preprocessing and augmentation stage to preprocess the data set for deep learning model training. Preprocessing involved resizing all images to the 720 x 640 dimension for CNN input stage, normalizing to normalize pixel values and reduction of noise for enhanced visual clarity. To counter class imbalance and to increase the generalization ability of the model, Synthetic Minority Oversampling Technique (SMOTE) was used. This ensures that across all the classes data is balanced and takes care of overfitting problem and makes the model robust for varied visual conditions.

In the next stage, ten CNN architectures, including five fine-tuned models and five fine-tuned transfer learning models were trained and tested. Among the selected models were VGG16, DenseNet121, InceptionV3, Xception and MobileNetV2. For every architecture, a fine-tuning strategy was applied by adding a classification head that is lightweight. The described models were trained and tested with 80:20 data split and performance were measured by calculating accuracy, precision, recall, F1-score, false alarm rate and confusion matrix. In addition, a comprehensive study into transfer learning was undertaken to see if there is a domain shift or negative transfer. Another extension, lightweight CNN is well structured with an appropriate number of convolutional layers, batch normalization, ReLU activation functions, maxpooling, and dropout layers. Following evaluation of CNNs and transfer learning models a custom lightweight CNN model, M-Net, was developed from scratch to solve the problem of efficient model deployment in edge devices.

Furthermore, XAI techniques have been added to the final model of the M-Net to introduce more transparency and interpretability in the prediction of the model. Concretely, LIME and SHAP were used to find out what pixel regions have contributed the most toward each prediction. On the other side, Grad-CAM generates class-specific heatmaps that visually identify the most important image areas the network utilized for its classification. Finally, a web application based on Streamlit, and an android mobile application based on flutter were developed to deploy the whole system in a user friendly interface. These two applications enable users to upload an image of a tea leaf and to see the results of classification instantly. Designed to be responsive and useable on common devices in rural areas to increase accessibility to the app and make it applicable to non-technical users.

1.5 Project Outcome

This study presented a series of interesting results that support the case for application of artificial intelligence in the field of agriculture, focusing particularly on the detection of diseases in the leaves of tea. Drawing from deep learning approaches that ensure interpretability and accessibility of the results, this project targets important challenges in informatics in agriculture especially in developing countries such as Bangladesh. As a result of the study, there was the creation of an extensively curated and well balanced dataset consisting of 2113 images of leaves of tea. This dataset has a total of six classes of both diseased and healthy specimens. Algal Leaf, Gray Blight, Healthy, Helopeltis, Looper Infested and Red Spider, with equal labels and abundant data augmentation. This dataset plays both the important role of integral component of the deep learning analyses presented in this thesis and the important role of a future plant pathologist's and machine learning expert's resource.

The study conducted a rigorous and comparative evaluation of ten SOTA CNN architectures, including VGG16, DenseNet121, InceptionV3, Xception, and MobileNetV2. The findings offer a benchmark performance overview on tea leaf disease classification using accuracy, precision, recall, and F1-score as evaluation metrics. The comparative results are significant for guiding future research on model selection and transfer learning suitability in domain-specific applications. A custom lightweight CNN model, named M-Net, was developed to meet the needs of low-power, real-time applications. M-Net demonstrated high classification accuracy with substantially fewer parameters and reduced inference time compared to heavier models. The study integrated three leading XAI techniques, LIME, SHAP, and Grad-CAM, to provide both global and local interpretability of M-Net's predictions. These methods offered visual and quantitative insights into how the model made decisions, thereby improving transparency, debugging ability, and trust for end-users. The inclusion of XAI made the model not only accurate but also understandable for non-technical stakeholders like farmers and agricultural extension officers. A practical web application was developed using the Streamlit framework, allowing users to upload tea leaf images and receive instant disease classification. The application was designed to be responsive, lightweight, and accessible even on modest hardware and low-bandwidth internet, increasing its utility in field scenarios.

1.6 Organization of the Report

This report is structured into six comprehensive chapters, each designed to systematically present the research problem, review relevant literature, describe the adopted methodology, discuss implementation results, and offer final insights and recommendations. The organization aims to ensure a logical flow of information that reflects the technical depth and practical relevance of the study.

Chapter 1: Introduction
This opening chapter provides the foundational context for the research. It outlines the motivation behind the work, the societal and technical relevance of detecting tea leaf diseases, and the importance of integrating Explainable AI with lightweight deep learning models. It also clearly defines the research objectives, methodology overview, and expected outcomes.

Chapter 2: Background
This chapter presents a detailed literature review on existing methods of plant disease detection using image classification, with an emphasis on CNNs, transfer learning, and lightweight deep models. It also explores the emerging role of Explainable AI (XAI) in increasing the transparency and interpretability of machine learning decisions.

Chapter 3: Research Methodology
This chapter begins with the overview of the methodology followed by context diagram, data flow diagram, UI design and detailed methodology. The detailed methodology includes the process of image data collection, followed by a detailed description of preprocessing techniques. It then discusses the two experimental phases: (i) model evaluation using CNN architectures, (ii) integrating XAI tools to interpret the model predictions and deployment strategy using a Streamlit web application and android mobile app.

Chapter 4: Implementation and Results
This chapter presents the implementation details of the proposed system. It provides comparative analysis of all CNN models in terms of performance matrices. This chapter also visualizes the interpretability results from XAI tools, and the testing of the Streamlit-based web application and android application.

Chapter 5: Engineering Standards and Design Challenges
This chapter discusses the engineering standards followed during the development of the system, including software and hardware compliance. It also covers the ethical aspects, societal and environmental impacts, sustainability considerations, and the financial analysis of the project. The section on complex engineering problems outlines the challenges faced and the solutions.

Chapter 6: Conclusion
The final chapter summarizes the entire research, highlighting the key findings and outcomes. It outlines the limitations encountered during the study and provides directions for future research, particularly in enhancing model generalization, improving dataset diversity, and extending mobile application functionalities for broader field use.

Chapter 2

Background

This chapter provides an overview of the key concepts and technologies related to automated tea leaf disease detection. It begins with an introduction to deep learning and its applications in plant disease detection, focusing on CNNs, transfer learning, and XAI. A detailed literature review examines the current state of plant disease detection using CNNs and highlighting their successes and limitations. The chapter also identifies gaps in existing research, such as issues with model interpretability, the need for lightweight models, and deployment to varying conditions.

2.1 Introduction

The main objective of this current study is to create an automated framework for diagnosing diseases in tea leaves, which poses a significant challenge in crop cultivation, particularly in the production of tea. Tea plants are very sensitive to many diseases, which can have very significant impacts on the quality and yield of the production of teas. Early diagnosis of such diseases is critical to enable prompt interventions and minimize possible crop loss. Over the past few decades, deep learning techniques have emerged as very effective tools for plant disease diagnosis, yielding accurate and automated methods that can support farmers in identifying and managing diseases more effectively. Significantly, Convolutional Neural Network architectures have shown very good performance in image-based plant disease diagnosis because they can automatically extract features from images with minimal manual input. The addition of transfer learning further enhanced the performance of CNNs with the use of pre-trained models. Despite the great progress made in this field, many challenges remain, particularly with model interpretability, scalability, and adaptability to variable environmental conditions.

To address these issues, XAI methods have been incorporated into models for detecting diseases. XAI-related techniques improve clarity through increased insight into the decision processes of machine learning models. XAI incorporates transparency of the models that promotes trust among stakeholders, such as farmers and agricultural experts. Through the addition of XAI into systems for detecting diseases, the model's decision-making process becomes more

transparent, allowing users to understand which image features contributed towards decision making. This chapter presents an overview of the state of the art in detecting diseases from tea leaves, summarizes relevant literature regarding architectures of machine learning, such as CNNs and transfer learning, and assesses the growing significance of XAI in maximizing model transparency and dependability. The aim of this chapter is to provide an exhaustive overview of technological advancements and challenges in this area, hence serving as an introduction to the methodologies and experimental techniques described in the following chapters.

2.2 Literature Review

In recent years, CNNs have shown significant advancements in plant disease detection, particularly for tea leaves. Several SOTA models have been proposed to enhance detection accuracy, feature extraction, and computational efficiency. Bhuyan et al. [37] introduced the Res4net-CBAM architecture, which combines residual blocks with a Convolutional Block Attention Module (CBAM) to enhance feature learning. Their model demonstrated remarkable classification performance, achieving an accuracy of 98.27% on a self-curated dataset. While the results surpass traditional CNN architectures such as AlexNet, VGG16, ResNet50, DenseNet121, and InceptionV3, the model's dependency on handcrafted feature extraction and the Adagrad optimizer limits its scalability to more complex real-world scenarios. Moreover, its performance on datasets with varying lighting conditions or natural backgrounds remains unexplored.

Datta and Gupta [38] developed a deep CNN for multiclass classification of tea leaf diseases, achieving a high accuracy of 96.56% across six classes. Although the model performed well with an extensive dataset of 5,867 images, its reliance on standard CNN layers without attention mechanisms or multi-scale feature extraction potentially limits its effectiveness in capturing more nuanced disease patterns, especially in uncontrolled environmental conditions.

Bao et al. [6] proposed the use of an AX-RetinaNet, which employs both multiscale feature fusion and an adaptive channel attention module for the purpose of detecting tea leaf diseases. This model successfully overcame the challenge of feature extraction at different scales, with a mAP of 93.83% and F1-score of 0.954. Despite presenting improved detection performance compared with models like YOLO-v3, YOLO-v4, as well as the EfficientNet, the use of elaborate data augmentations suggests high demand for computing resources, which may make it inapplicable to real-time or edge-device applications.

Balasundaram et al. [39] proposed a CNN-based framework which detects and segments infected regions of a tea leaf. The advanced preprocessing incorporated the

use of OpenCV to identify, bound and crop-out the leaf to capture the diseased area through extracting the region of interest. In addition to that, the Segment Anything Model was used for zero-shot segmentation, to cut off the infected regions. Features were extracted and classified with the aid of models such as MLP, SVM and the Decision Tree classifiers. Despite the remarkable accuracy of 95.06%, model's performance is largely affected by appropriate preprocessing and segmentation accuracy.

In addition, Xu et al [40] compared RGB and hyperspectral image techniques for the analysis of the damage caused by the tea green leafhopper. For their study they used multiple different architectures: ResNet18, VGG16 and AlexNet, and their wavelet-transformed analogue prototypes, and in the end they found out that WT-VGG16 had a better performance boasting 80.0% accuracy. For hyperspectral image analysis, models UVE-SVM, SPA-SVM, and SPA-LSTM were compared and SPA-LSTM outperformed with an accuracy of 95.6%.

Transfer learning has been seen as an effective method to overcome issues of limited data availability and computationally intensive tasks in plant disease diagnosis, focusing particularly on tea leaves and crop plants. Yao et al. [41] aimed to detect Tea Leaf Blight (TLB) using transfer learning combined with data augmenting strategies. For increased detection of faint symptoms, the study integrates decoupling head detector along with a Triplet Attention mechanism embedded in the E-ELAN backbone, further boosting the model's power to detect faint symptoms of the disease. Moreover, optimization of the model's loss function with the use of Wasserstein distance significantly increases the model's performance in detecting faint targets. Compared to the standard YOLOv7 tiny model, their method showed great improvements, with accuracy improving by 6.5% to reach 92.2%, recall improving by 4.5% to reach 86.6%, and mean precision improving by 5.8% to reach 91.5%.

Ramdan et al. [42] applied transfer learning using pre-trained models such as VGGNet, ResNet, and Xception trained on ImageNet. Their study highlighted the limitations of directly applying transfer learning due to task differences between ImageNet and plant disease classification. To address this, they implemented fine-tuning techniques by modifying parts of the pre-trained networks, finding that models with fine-tuning consistently outperformed both vanilla transfer learning and models trained from scratch. This approach led to better feature extraction for plant disease detection tasks, demonstrating that tailored adjustments of pre-trained networks are essential for specific domain applications.

Sivaraman et al. [35] utilized the VGG16 network along with transfer learning to develop an efficient model for tea leaf disease prediction. Fine-tuning the current network on a dedicated dataset, the model managed to extract not only the disease-

specific features but also the general knowledge learnt from ImageNet. The model obtained up to 92% estimation on training; while applying it to an independent test data set, the accuracy remains to a rate of 86.17% in test measurement.

Chen et al. [14] tested the performance of transfer learning by using pre-trained models of VGGNet and the Inception module in plant leaf disease identification. The model weight initialization started with features from ImageNet, as opposed to the use of random weight initialization. This technique led to improved performance with at least validation accuracy of 91.83% when tested with publicly available datasets and with 92% accuracy in detecting diseases in rice plants even under unfavorable background conditions.

Table 2.1: Research matrix for tea leaf disease detection techniques

Authors	Models	Accuracy	Key Findings
Bhuyan et al. [37]	Res4net-CBAM	98.27%	Combines residual blocks and CBAM; surpasses traditional CNNs; scalability and performance under natural conditions are limitations.
Datta & Gupta [38]	Deep CNN	96.56%	Effective on large datasets; lacks attention mechanisms and multiscale feature extraction, limiting real-world generalization.
Bao et al. [6]	AX-RetinaNet + multiscale feature fusion + adaptive channel attention	mAP: 93.83%, F1-score: 0.954	Outperforms YOLOv3/YOLOv4/EfficientNet; high detection accuracy, but complex augmentation demands large computational resources.
Balasundaram et al. [39]	CNN + OpenCV preprocessing + SAM + MLP/SVM/Decision Tree	95.06%	Use ROI extraction and zero-shot segmentation; heavily depends on preprocessing, which makes it sensitive to noise and dataset variability.
Xu et al. [40]	WT-VGG16, SPA-LSTM	RGB: 80% (WT-VGG16), HSI: 95.6% (SPA-LSTM)	Combines RGB and hyperspectral imaging; hyperspectral models are accurate but not feasible for wide deployment due to cost and complexity.
Yao et al. [41]	YOLOv7 +	Precision:	Integrates advanced detection

	TSCODE + Triplet Attention + Wasserstein loss	92.2%, Recall: 86.6%, AP: 91.5%	and attention mechanisms with transfer learning; strong for small-target detection.
Ramdan et al. [42]	Transfer learning (VGGNet, ResNet, Xception) + fine-tuning	Not specified (outperforms baselines)	Fine-tuned transfer learning significantly outperforms vanilla TL and training from scratch for plant disease detection.
Sivaraman et al. [35]	VGG16 + Transfer learning	Train: 92%, Test: 86.17%	Robust performance with eight classes; practical, low-cost solution leveraging fine-tuned pre-trained models.
Jiang et al. [43]	Improved VGG16 + Multi-task transfer learning	Rice: 97.22%, Wheat: 98.75%	Multi-task learning allows shared representation learning across different crops; superior to ResNet50 and DenseNet121.
Chen et al. [14]	VGGNet + Inception + Transfer learning	≥91.83%, Rice: 92%	ImageNet pre-training boosts performance on plant datasets, especially in complex backgrounds.

2.2.1 Similar Applications

Recent developments in mobile technology and deep learning have significantly accelerated the advancement of effective systems for diagnosing tealeaf disease. Researchers have explored ensemble learning, real-time processing, and mobile-oriented methods to enhance the effectiveness, precision, and accessibility of plant disease diagnosis. Ensemble learning has proven to be an effective method for enhancing the efficacy and robustness of systems for diagnosing plant diseases. Utilizing several models, ensemble methods aim to combine the strengths of single models and avoid their weaknesses. Hang et al. [44] proposed an original fuzzy ensemble method that combines five pre-trained architectures, namely, VGG16, VGG19, Xception, InceptionV3, and InceptionResNetV2, which use Choquet integral fuzzy integrals for tomato disease classification. The method proved to have high efficacy with an accuracy rate of 99.80% when tested with the PlantVillage dataset, outperforming the best single model which achieved an accuracy of 98.63%. The model also proved to have an impressive accuracy of 97% in real-world environments, demonstrating power in real-world applications under conditions of variation in light and environmental factors. With the increasing need for accessible diagnostic tools, mobile-based solutions for plant disease detection have seen notable progress. Manasa [53] introduced a mobile-based system powered by machine learning to automate the

identification of plant leaf diseases. This system utilizes a Convolutional Neural Network (CNN) for both feature extraction and classification across five disease categories. The user interface, developed as a mobile app, enables farmers to take a photo of diseased tea leaves. Once uploaded, the image is processed by the system using its path, and the disease type, along with symptoms and confidence score, is presented to the user.

Obadaarachchi et al. [54] developed a customized CNN architecture tailored for real-time plant disease detection and treatment recommendations. This model surpassed the performance of the ResNet50V2 architecture, achieving a 94% accuracy rate. Li et al. [55] proposed an improved version of the MobileNetV3 architecture for recognizing tea leaf diseases and insect damage. The enhanced model incorporates a Coordinate Attention module to strengthen spatial awareness and uses a transfer learning strategy to accelerate convergence. On the constructed dataset, the standard MobileNetV3 achieved 94.45% accuracy with a 94.12% F1-score. The improved MobileNetV3-CA reached 94.58% accuracy without transfer learning, and 95.88% with it. Comparative experiments with models such as ResNet18, AlexNet, VGG16, SqueezeNet, and ShuffleNetV2 confirmed the superiority of the proposed model, which was later integrated into an application for intelligent diagnosis of tea-related diseases and pests. Lanjewar et al. [17] developed a real-time disease prediction system using CNN deployed on a Platform-as-a-Service (PaaS) cloud environment. Their CNN model achieved perfect accuracy across training, validation, and test datasets. Additionally, the dataset was evaluated using deep CNN models like ResNet50, Xception, and NASNetMobile. Comparative analysis using confusion matrices and K-fold cross-validation revealed the superior performance of their custom CNN model. The final model was deployed to the cloud, accessible via smartphone to allow users to capture and upload images for instant disease prediction on mobile devices. Tembhurne et al. [56] introduced an enhanced MobileNet-based model, fine-tuned using the Keras tuner, and trained on a comprehensive dataset of 12,318 images. This model demonstrated strong generalization capabilities on a unified, large-scale dataset compiled from several smaller datasets. It classifies 64 different categories across 22 crop types, achieving a classification accuracy of 95.94%. Ahmed & Reddy [57] presented a mobile-based deep learning system using CNNs to detect plant leaf diseases across 38 categories. Their model was trained and evaluated using a substantial dataset of 96,206 images, including both healthy and infected leaves from 14 crop types. The Android application allows farmers to take and upload leaf images, returning the disease name and confidence level. With an overall classification accuracy of 94%, this system is expected to help farmers maintain crop health and prevent the misuse of fertilizers, ultimately contributing to smarter agricultural practices.

Table 2.2: Similar applications in tea leaf disease detection

Authors	Model	Accuracy	Key Contributions
Manasa [53]	CNN (mobile-based)	Not specified	Developed a mobile app for detecting five plant leaf diseases with real-time image capture and classification.
Obadaarachchi et al. [54]	Customized CNN	94%	Designed for real-time detection with treatment suggestions; outperformed ResNet50V2.
Li et al. [55]	MobileNetV3-CA (with Coordinate Attention)	95.88% (with TL)	Improved MobileNetV3 with CA module and transfer learning; deployed in a mobile app for tea disease recognition.
Lanjewar et al. [17]	Custom CNN (cloud-based)	100%	Achieving perfect accuracy across all sets; cloud deployment enables real-time mobile predictions.
Tembhurne et al. [56]	Enhanced MobileNet (Keras tuner optimized)	95.94%	Trained on a unified dataset of 12,318 images across 22 crops; deployed as Plantscape Android app.
Ahmed & Reddy [57]	CNN (mobile-based)	94%	Trained on 96,206 images across 38 disease categories; app returns disease and confidence from uploaded image.

2.2.2 Related Research

Recent advancements in lightweight CNN architectures and XAI have significantly contributed to enhancing the real-world applicability of plant disease detection systems, including those focused on tea and other crops. Wu et al. [48] proposed the LBPAttNet model that combines coordinate attention mechanism into ResNet18 to enhance the disease localization and reduce the background interference. Moreover, the use of the Local Binary Patterns algorithm improves the extraction of local structural and textural features of tea leaf diseases. To compensate for imbalanced classes and different levels of

classification difficulty, the model uses a focal loss function, providing an improvement of recognition accuracy. Experimental results demonstrate that LBPAttNet obtains an accuracy of 92.78% and 98.13% on two publicly available Tea Disease datasets overpassing ResNet18 by 3.84% and 2.59% respectively. With regards to traditional models such as AlexNet, GoogleNet, MobileNet, and VGG16, LBPAttNet shows more robustness and accuracy. Similarly, Liang et al. [13] introduced the Lightweight Tea Diseases Detection Network, a model designed for high accuracy in complex backgrounds while remaining computationally efficient. It incorporates a Channel Focus Attention Mechanism to highlight key disease features, a Feature Reuse Module to reduce parameters and computational costs, and a Feature Enhancement Mobile Inverted Bottleneck Convolution Module to preserve detailed features during down sampling. LTDDN is significantly lighter, with parameters reduced to 1/30 of ResNet50 and 1/50 of ViTB16, making it suitable for edge devices. Tested on custom tea disease datasets, Mini-ImageNet, and PlantVillage, it outperformed conventional CNNs, lightweight CNNs, and ViT models. An Android app was also developed for real-time, offline tea disease detection enhanced its practical applicability in the field.

Expanding the application of lightweight CNNs beyond tea leaves, Ding et al. [31] employed machine vision technology to assess the fermentation degree of black tea. They proposed a lightweight CNN that utilizes knowledge distillation, with Shufflenet_v2_x1.0 serving as the student model and EfficientNet_v2 as the teacher model. Incorporating Focal Loss and testing various knowledge distillation techniques, the study found that using the Masked Generative Distillation (MGD) method with a distillation loss ratio of 0.8 yielded the best performance. The final model achieved precision, recall, and F1 scores of 0.9208, 0.9190, and 0.9192, respectively, ensuring accurate classification of fermentation levels without increasing computational complexity. Fu et al. [9] focused on pepper leaf disease detection and introduced a lightweight CNN using the GGM-VGG16 architecture. This model incorporates Ghost modules, global average pooling, and multi-scale convolution to enhance detection accuracy while maintaining computational efficiency. Trained on a dataset featuring images of healthy and diseased leaves, including images captured against human palm backgrounds (a common scenario in field conditions), the model achieved 100% accuracy on palm-background images and 87.38% on other backgrounds. The developed Android application, with a compact size of 12.84 MB, demonstrated strong performance in real-time pepper leaf disease recognition.

Moreover, Thakur et al. [49] developed VGG-ICNN, a lightweight CNN with approximately 6 million parameters, significantly fewer than most high-performing deep learning models. Tested on five diverse public datasets, PlantVillage, Embrapa, Apple, Maize, and Rice, the model consistently

outperformed existing approaches. Achieving a remarkable 99.16% accuracy on the PlantVillage dataset, VGG-ICNN also maintained competitive performance across all datasets, highlighting its versatility and efficiency in crop disease identification. Parallel to lightweight CNN development, there is growing emphasis on integrating XAI to improve interpretability and trust in model predictions. Several recent studies have explored the integration of XAI techniques with CNNs to enhance model transparency and interpretability in plant disease detection. Finally, Kurniawan et al. [52] highlighted the importance of explainability in machine learning-based plant disease detection systems, which often function as opaque 'black boxes.' To address this issue, they employed a DenseNet-based classification model, achieving 82.54% accuracy on initial test data and 87.5% on field data. Among various XAI methods tested, including LIME, SHAP, and Grad-CAM, the LIME framework proved to be the most effective, balancing interpretability, computational efficiency, and ease of understanding.

Table 2.3 Related research in tea leaf disease detection

Authors	Model Used	Accuracy	Key Contributions
Wu et al. [48]	LBPAttNet (ResNet18 + Coord Attention + LBP)	92.78%, 98.13%	Improved disease localization and robustness; outperforming ResNet18, MobileNet, VGG16.
Ding et al. [31]	Shufflenet_v2_x1.0 + EfficientNet_v2 (Knowledge Distillation)	Precision: 0.9208, Recall: 0.9190, F1: 0.9192	Masked Generative Distillation yielded best performance for fermentation detection.
Assaduzzaman et al. [33]	XSE-TomatoNet (EfficientNetB0 + SE blocks)	99.11%	Used LIME, SHAP, Grad-CAM++; web-based deployment for tomato disease detection.
Bhandari et al. [50]	EfficientNetB5	Train: 99.84%, Val: 98.28%, Test: 99.07%	Used Grad-CAM and LIME; high performance without segmentation.
Das et al. [51]	XLTLDisNet	97.24%	Lightweight model with Grad-CAM and LIME for transparency.
Kurniawan et al. [52]	DenseNet	82.54% (test), 87.5% (field)	LIME was most effective among SHAP, Grad-CAM for real-world XAI.

2.3 Gap Analysis

Despite substantial progress in automated tea leaf disease detection using deep learning, several critical gaps and limitations remain, particularly in the areas of model interpretability, computational efficiency, real-time deployment, and dataset localization. The following analysis highlights these research gaps across six key areas:

Many existing studies have utilized Convolutional Neural Networks (CNNs) for plant disease detection, but they often face the challenge of being "black box" models, making it difficult to interpret the decision-making process. Studies by Wu et al. [48], Liang et al. [13], and Fu et al. [9] focused on improving disease detection accuracy but did not emphasize model transparency. This lack of interpretability limits the trust that end-users, such as farmers, can place in the predictions made by these models.

Most of the studies reviewed rely on publicly available datasets, such as PlantVillage or PlantDoc, which may not accurately represent the plant diseases found in specific regions or under specific environmental conditions. For example, Liang et al. [13] and Thakur et al. [49] used publicly available datasets that may not capture the full diversity of plant diseases. This data shortcoming may cause the models to lack robustness when executed in real-world farming environments.

Deep learning models, especially expansive models like ResNet and EfficientNet, traditionally involve high computational needs. Even though these models often produce improved accuracy, they are poorly suited for real-time execution in edge devices that have limited processing power. Studies by Ding et al. [31], Bhandari et al. [50], and Kurniawan et al. [52] demonstrated high-performing models but did not offer efficient alternatives for edge deployment. This study overcomes this challenge by developing a custom lightweight CNN model (M-Net) designed specifically for real-time deployment in environments with limited computational resources.

While many existing models excel in terms of accuracy, they often lack accessibility and real-time usability. Several studies, such as those by Nigar et al. [32] and Bhandari et al. [50], provide high-accuracy models but do not integrate them into systems that are immediately useful for farmers. The lack of accessible deployment options restricts the practical impact of these models. In response, this study integrates the developed model into a real-time web-based application using Streamlit, allowing farmers to easily upload images of tea leaves and receive immediate predictions regarding disease presence and severity.

Even though various studies have concentrated on the issue of lightweight CNN models for plant disease detection, for example, those of Das et al. [51] and Fu et al. [9], no work has incorporated these XAI techniques in custom lightweight CNN models for disease categorization. The inclusion of XAI methods in lightweight models is critical for improving model transparency and fostering trust among end-users. This study uniquely contributes to this area by incorporating multi-method XAI techniques (Grad-CAM, LIME, and SHAP) into the custom lightweight CNN (M-Net). This combination ensures that the model not only performs efficiently but also provides interpretable predictions.

Most studies reviewed in the literature focus on individual aspects of the plant disease detection pipeline, such as data collection, model development, or deployment, but rarely provide a comprehensive, end-to-end solution. For example, some studies focus on model development without considering real-world deployment (Wu et al. [48], Liang et al. [13]), while others focus on deployment but lack a robust model (Thakur et al. [49]). This study addresses this gap by offering deployment via a Streamlit web application. This end-to-end approach ensures that the model is not only accurate but also practical and accessible to farmers.

Table 2.4 Gap analysis summary table

Gap Identified	Observed In	This Study's Contribution
Limited interpretability in CNN models used for detection	Wu et al. [48], Liang et al. [13], Fu et al. [9], Thakur et al. [49]	Integrated Grad-CAM, LIME, and SHAP to enhance model transparency and build user trust
Use of public/non-local datasets	Wu et al. [48], Liang et al. [13], Thakur et al. [49], Das et al. [51]	Collected and curated a real-world tea leaf disease dataset from Sreemangal, Bangladesh
High computational load in deep models	Wu et al. [48], Ding et al. [31], Bhandari et al. [50], Kurniawan et al. [52]	Designed and trained a custom lightweight CNN (M-Net) suitable for edge deployment
Lack of real-time deployment and accessibility	Nigar et al. [32], Bhandari et al. [50], Kurniawan et al. [52]	Built a Streamlit-based web app for practical, real-time tea leaf disease detection
Absence of XAI in custom lightweight CNNs	Das et al. [51], Fu et al. [9], Liang et al. [13]	First to combine a custom lightweight CNN with multi-method XAI integration
Fragmented research pipeline (no end-to-end system)	Most studies reviewed	Delivered a full pipeline: data collection → model training → XAI → deployment via web UI

2.4 Summary

This section reviewed the current state of tea leaf disease detection, focusing on the application of CNNs, transfer learning, and lightweight models, along with the incorporation of XAI techniques. The literature explored various approaches to improving the accuracy, efficiency, and scalability of disease detection systems for tea cultivation. CNNs have proven to be highly effective in plant disease detection, with models such as Res4net-CBAM and AX-RetinaNet showcasing impressive results. However, challenges like reliance on handcrafted features, sensitivity to environmental conditions, and computational limitations for real-time deployment remain. Transfer learning has emerged as a promising technique to overcome the challenge of limited labeled data, with models like VGG, ResNet, and Xception being fine-tuned to improve domain-specific performance. Despite its success, the direct application of pre-trained models often suffers from domain shift issues. Additionally, XAI is increasingly important for increasing transparency of deep learning model. However, even though such techniques as LIME, SHAP, and Grad-CAM provide valuable insights into how the model predicts and misclassifies, the technology is still in its early days when applied to tea leaf disease detection. These gaps in literature emphasize the need for more scalable, interpretable and efficient solutions. In a nutshell, although substantial progress has been made in the detection of tea leaf diseases using deep learning approaches, other challenges, such as model complexity, interpretability, and real time deployment remain.

Chapter 3

Research Methodology

This paper introduces a systematic approach to develop an explainable, lightweight CNN based system for tea leaf disease detection. It begins with the methodology and goes through the design, implementation and evaluation phases. Each phase is sequentially planned to enable reliable, accurate and interpretable results performed with the help of conventional and modern deep learning methods.

3.1 Methodology

3.1.1 Overview

The research design employed in this study aligns with an organized framework for the development of a lightweight and precise, interpretable deep learning system for detection of diseases in tea leaves. The data acquisition stage is the first phase of the methodology, which is achieved through direct acquisition from selected tea plantations at Sreemangal, Sylhet, and Moulvibazar. The acquired data is validated to ensure relevance to the context, variation of lighting and environmental conditions, as well as actual-world challenges from the fields. Following data acquisition, the data preprocessing and data augmentation make up the next phase. Preprocessing in this step makes the input images properly cleaned, resized, normalized, and standardized, hence making them suitable for feeding into CNNs. To improve the models' generalizability, SMOTE is applied to balance the instances across all the classes of the dataset. The core experimental section includes training and evaluation of six well-established CNN architectures, fine-tuned on the preprocessed dataset. This helps determine which architecture performs best in terms of accuracy, training efficiency, and robustness. After this, transfer learning models pretrained on large-scale datasets are evaluated to explore improvements in accuracy and learning efficiency when domain-specific fine-tuning is applied. Building on these results, this study develops a custom lightweight CNN model, M-Net. M-Net is designed to maintain high classification accuracy while significantly reducing the number of parameters and memory footprint. To enhance the interpretability of the M-Net model, XAI techniques such as LIME, SHAP, and Grad-CAM are integrated. Finally, a mobile application is developed as a practical deployment tool. This app enables users, particularly farmers and agricultural

officers to upload or capture images of tea leaves and receive immediate diagnostic feedback.

3.1.2 Proposed Methodology

This research aims to develop a CNN-based efficient tea leaf diseases detection system, and it is a structured research design. The first step in the process is data collection, in which images of tea leaves are collected from the real world. The prepared data is preprocessed, which includes resizing up to the standard resolution, and SMOTE for better model performance. The following are the division of the dataset into three subsets: training, validation, and testing. A CNN model will then be designed and compared with pre-trained networks used for feature extraction and classification. The model is trained with hyperparameter tuning practices and methods like early stopping and learning rate scheduling to avoid overfitting. Performance is measured using acronyms like accuracy, precision, recall, F1-score and false alarm rate which is done using XAI methods like LIME and SHAP besides providing interpretability through highlighting of important regions of the leaf for classification. Finally, a prototype android application is developed and tested on unseen images to ensure robustness and applicability in real world of tea agriculture.

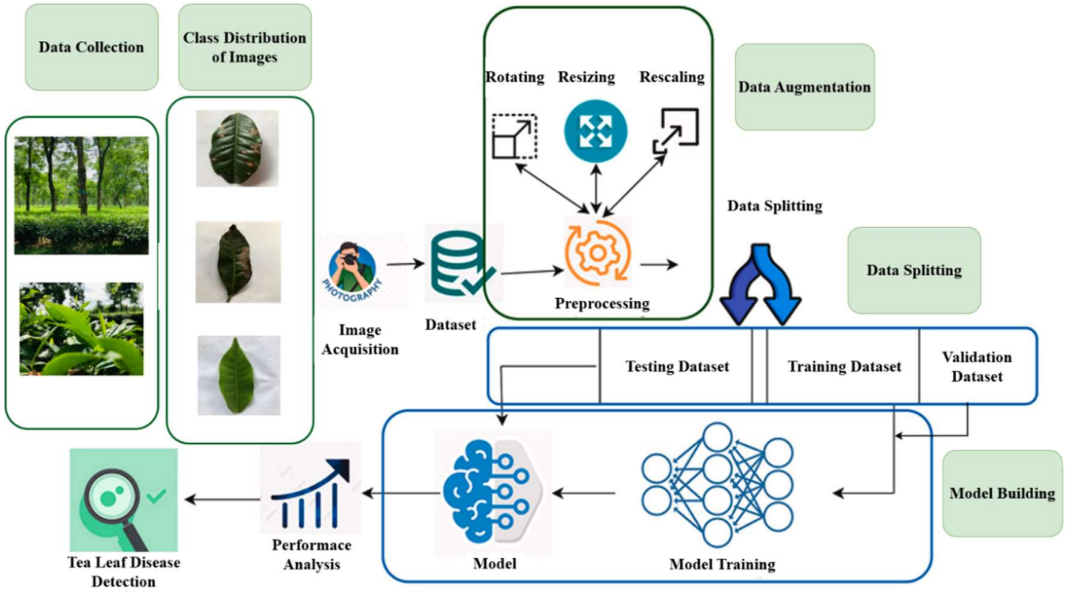


Figure 3.1: The methodological flowchart

3.1.3 Functional and Nonfunctional Requirements

The proposed framework supports key functional and non-functional requirements necessary to guarantee consistent performance and real world usability. The former outlines the core operations of the system, and the later supports standards as regards performance, usability, and security in case of practical application of the system.

Functional Requirements

Functional requirements define the basic operations and feature that the tea leaf disease detection system should support to achieve its mission. These requirements guarantee the way the system is supposed to action and react to varied user interactions.

- Image Upload: The system allows users to upload images of tea leaves either by capturing photos using a device camera or by selecting from the device gallery.
- Image Preprocessing: The system shall automatically resize, normalize, and enhance the uploaded image to prepare it for disease detection.
- Disease Classification: The system shall classify the preprocessed image into one of the predefined categories such as Blister Blight, Red Rust, Gray Blight, or Healthy using trained CNN models.
- Model Selection and Inference: Initially, the system shall support running multiple CNN models (fine-tuned and transfer learning-based) and later deploy a lightweight custom CNN model (M-Net) for final inference.
- Prediction Output: The system shall display the predicted disease class along with a confidence score (e.g., "Red Spider – 92.6% Confidence").
- Feedback Module: The system shall allow users to rate the accuracy of predictions and optionally submit comments to contribute toward continuous model improvement.
- Offline Prediction Capability: The mobile application shall support offline inference using the optimized lightweight CNN model (M-Net), enabling users to detect diseases without internet access.
- User Notifications: The system shall notify users of successful uploads, processing status, or errors through visual or text feedback.

Nonfunctional Requirements

Nonfunctional requirements define the quality characteristics and operational constraints of the system, ensuring its usability, performance, and reliability in real-world scenarios.

- Performance: The system must provide inference results within 3–5 seconds on both mobile and web platforms.
- Accuracy: The lightweight CNN model should achieve at least 90% classification accuracy, with high precision and sensitivity.
- Usability: The system interface must be automatic, responsive, and simple, so that the users with minimal technical expertise can operate this.
- Scalability: The system should be scalable to support future updates, such as integration of more disease classes, use of advanced ensemble methods, and incorporation of other plant species.
- Maintainability: The application's modular architecture must accommodate ease of upgrades, debug as well as versioning controls without hampering

serious downtime.

- Reliability: The system should operate consistently in various environments and the results under normal operating condition must be reliable.
- Security and Privacy: During storage and during transmission, user-uploaded picture and personal data must be encrypted to adhere to standard privacy policy.
- Resource Efficiency: The lightweight version of the model should be fine-tuned to run smoothly on low-end mobile type devices such that it has a low CPU as well as memory resource requirement.

3.1.4 Context Diagram

The context diagram provides a big picture for the tea leaf disease prediction system as a single process and its interactions with external players. The diagram demarcates system boundaries and highlights information exchange between system and its users or other external parties. The main external stakeholder under investigation in this context is the User who interacts with the system by providing images of tea leaves and looks forward to predictions about disease occurrence alongside visual explanations. The system aggregates all internal processes that include preprocessing and classification methods such as CNN, Transfer Learning, or M-Net as well as modules for explainability including XAI.

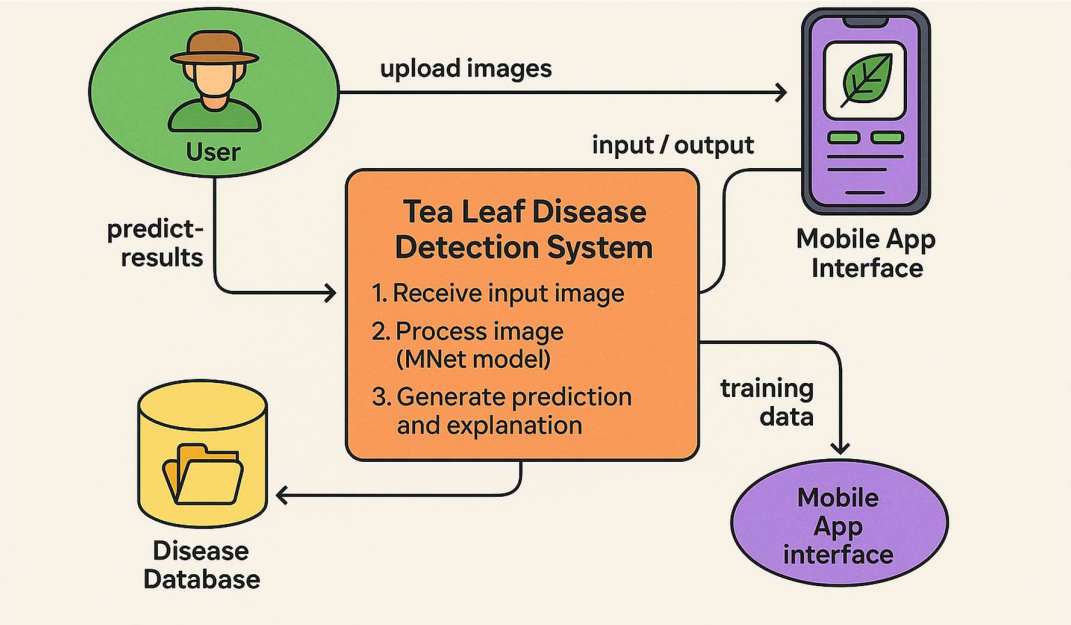


Figure 3.2: Context diagram of the system

3.1.5 Data Flow Diagram Level 1

The internal workings of the tea leaf disease detection system are illustrated in the Level 1 Data Flow Diagram (DFD). The objects to be identified, a tea leaf image, are

first uploaded by the user, received by the image input module. This image is then passed to the preprocessing module, where it is resized, normalized, and augmented. Next, the image is forwarded to the model inference module that initially evaluates the image using M-Net. This model is stored in the trained model repository. The prediction result is then sent to the explanation module, which generates interpretability outputs using LIME and SHAP. These visual explanations are temporarily stored in the explanation outputs store. The result; disease name predicted, and confidence score are presented to the user by the result display module.

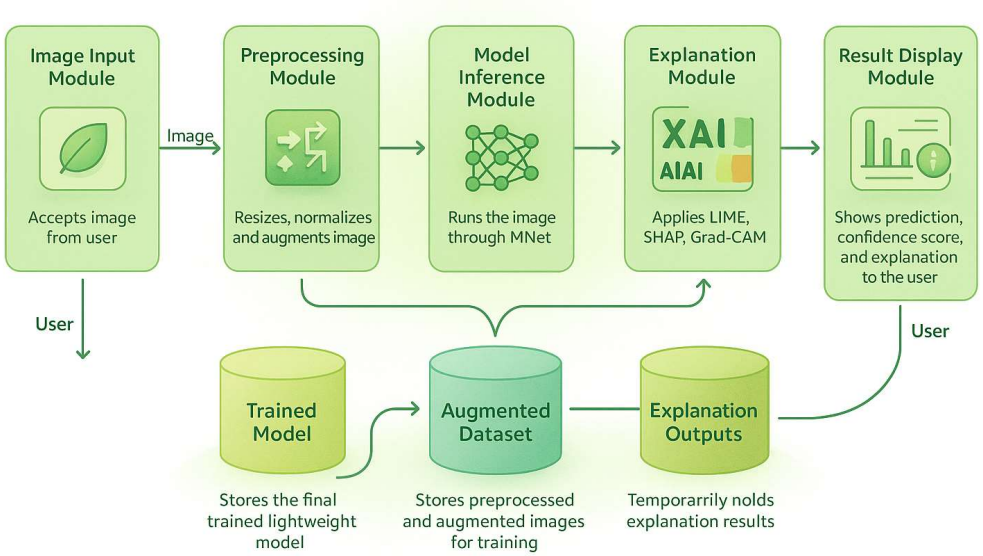


Figure 3.3: Data flow diagram of the system

3.1.6 UI Design

The User interface (UI) design of the proposed brain cancer detection system is engineered to deliver an intuitive and user friendly experience to end users; especially medical practitioners and researchers. The UI is intended to facilitate uploading of tea leaf images and classification result visualization through them, using the resulting model. It is based on two-tier architecture, with Flutter used for it to be platform independent.

1. Application Setup

The application setup consists of rolling out the trained M-Net model fully into a mobile application. The M-Net model weight is exported in TensorFlow Lite (. tflite) file format and then embedded in the application to allow for smooth processing of tea leaf images in mobile devices. The major steps in the application setup include:

- User Interface (UI) Components: The UI is created using Flutter to create an aesthetic and easily navigated interface. It presents vital components including an input button for uploading images, classification output

display. The classification result is returned by simply pushing a "Detect" button to trigger the model inference.

- Model Integration: The M-Net is converted to .tflite version for mobile deployment. The application loads this model into memory to process efficiency tea leaf pictures without the need for an active internet capability.
- Performance Optimization: The Tflite model is optimized for speed, ensuring minimal latency during classification.
- Cross-Platform Compatibility: Due to the development of the application with the help of Flutter, the application supports both Android and iOS platforms, hence will have a wider reach and accessibility available.

2. Two-Tier Architecture

The system has a two tier architecture, which appropriately separates the application functionality into two major layers:

- Presentation Layer (Frontend - User Interface and Interaction)
  - The presentation layer deals with all types of user interactivity and the visual interface items of the mobile application.
  - Created using Flutter, this layer covers UI components such as buttons, image uploading functions, results showing classification with confidence score.
  - It ensures that the user's experience is smooth, responsive, and intuitive, with a minimal learning curve for new users.
  - The frontend communicates with the processing layer to send input images and receive classification results.
- Processing Layer (Backend - Model Inference and Computation)
  - The processing layer contains the pre-trained Tflite model that performs feature extraction and classification.
  - It processes tea leaf images locally, extracting the most relevant features and classifying diseases into one of the six categories (algal_leaf, gray_blight, healthy,, Helopeltis, looper_infested, and red_spider).
  - Once the model completes classification, the results are sent back to the frontend for display.

3. Folder Structure:

The folder structure of the brain cancer detection application is organized to ensure modularity, maintainability, and efficient integration of machine learning components. The structure consists of key directories and files that facilitate model deployment, user interface development, and backend processing. The primary components are as follows:

- Root Directory: tea_leaf_disease_detector/

This is the main project directory containing all essential files and subdirectories

required for the mobile application.

- assets/ – This folder contains necessary resources such as labels and the trained TensorFlow Lite model.
  - labels.txt – A text file that stores the class labels (algal_leaf, gray_blight, healthy,, Helopeltis, looper_infested, red_spider).
  - model.tflite – The optimized TensorFlow Lite model that performs tea leaf disease classification.
- main.dart – The entry point of the Flutter application, handling UI rendering and interaction with the model

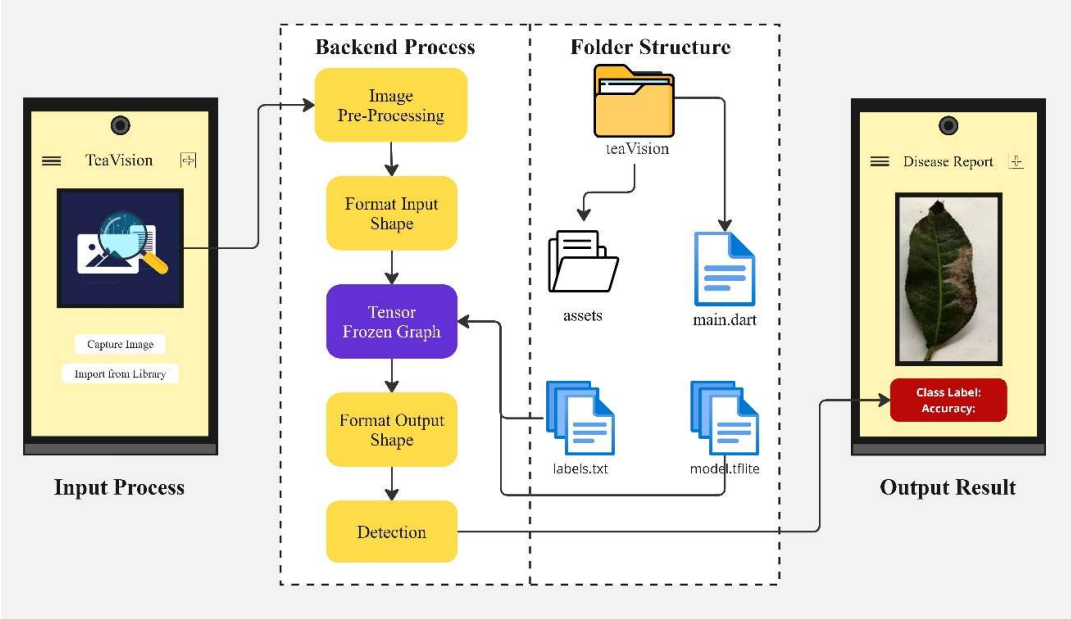


Figure 3.4: Two-tier architecture and UI design of the application

3.2 Detailed Methodology and Design

The experimental workflow for tea leaf disease classification is systematically designed and executed through multiple phases to ensure optimal performance, interpretability, and real-world applicability. The process begins with dataset acquisition, followed by image preprocessing to enhance data quality and variability. The study is then divided into three major experimental phases, concluding with model interpretation and deployment. In the first phase, five custom fine-tuned CNN models are trained and evaluated on the preprocessed tea leaf dataset. The second phase involves applying transfer learning using pre-trained models such as VGG19, DenseNet201, InceptionV3, ResNet152V2, SeResNet152, and MobileNetV2. These models, originally trained on the ImageNet dataset, are fine-tuned on the domain-specific tea leaf dataset to evaluate whether leveraging learned representations improves classification performance compared to custom fine-tuned CNNs.

In the third phase, a lightweight CNN architecture called M-Net is proposed and developed. The M-Net model is designed to significantly reduce the number of parameters and computational load, making it suitable for deployment in resource-constrained environments such as mobile devices. The fourth phase integrates XAI techniques, specifically LIME and SHAP into M-Net architecture. To validate the model's practicality and usability, the final trained M-Net model is deployed in two interfaces. First, a web-based application is developed to demonstrate real-time functionality and ensure ease of access for users via browsers. Subsequently, a mobile application is created to serve the needs of end-users. This mobile deployment emphasizes scalability, accessibility, and direct societal impact.

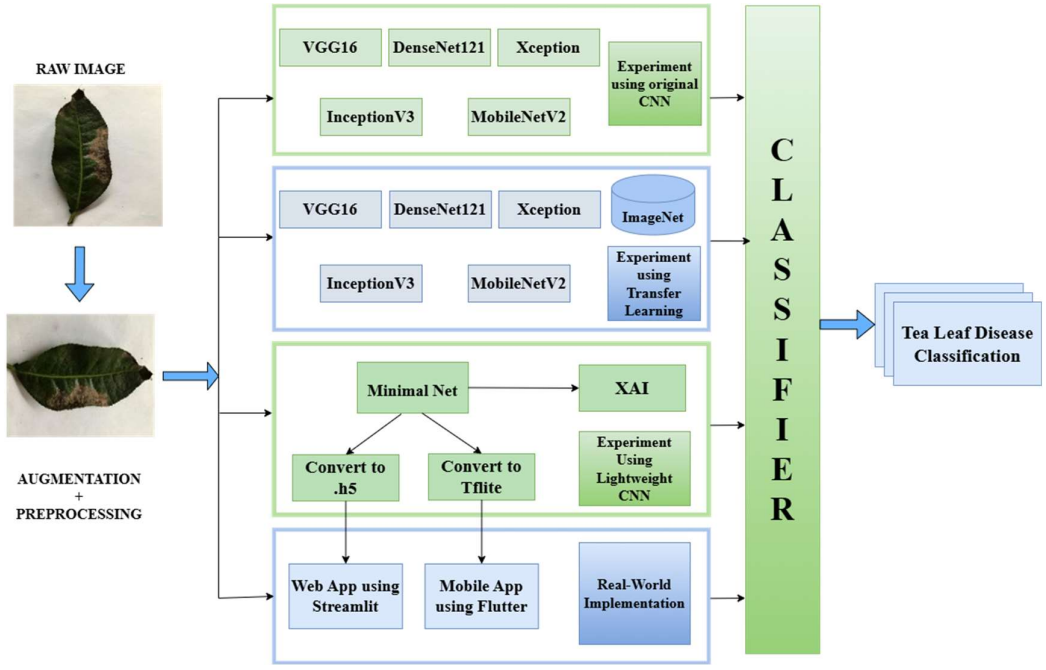


Figure 3.5: Experimental workflow of tea leaf disease classification

3.2.1 Dataset

3.2.1.1 Data Collection

To collect high-quality and representative data for tea leaf disease classification, this study comprised detailed review of existing research works to understand the key visual features related to common tea leaf diseases. These diseases included algal leaf, gray blight, helopeltis, looper infestation, and red spider, as well as healthy leaves. The dataset was collected from two major tea estates in Sreemangal, Moulvibazar, Bangladesh:

- M.R. Khan Tea Estate at coordinates: 24.27257, 91.75938
- Finlay Tea Estate at coordinates: 24.30334, 91.74249

Data collection was carried out over a week-long period from December 9 to

December 16, 2024, under real-world field conditions. Two smartphones, iPhone 7 and Samsung A21s were used to capture the images. This dual-device strategy introduced diversity in image resolution, color processing, and perspective, which helps in improving model generalization. Images were taken at various times of the day and from multiple angles to include natural variations in lighting and orientation. The images were saved in JPG format. The specifications of each device are detailed in table 1. Expert support was provided throughout the collection phase. On-site agronomists assisted in identifying visible symptoms of diseases, while final label verification was conducted in consultation with a professional from the Bangladesh Tea Research Institute (BTRI).

Table 3.1: Specification of the data collection devices

Feature	iPhone 7	Samsung A21s
Camera Manufacturer	Apple	Samsung
Camera Model	iPhone 7	SM-A21F
Aperture Value	f/1.8	f/2.0
Resolution	3024 x 4032 pixels	3000 x 4000 pixels
DPI	72	72
Bit Depth	24	24

3.2.1.2 Dataset Description

The final dataset used in this study comprises a total of 2,113 images of tea leaves, covering six distinct classes: Algal Leaf, Grey Blight, Healthy Leaf, Helopeltis, Looper Infested, and Red Spider. The images were saved in JPG format to ensure compatibility and efficient storage without significant loss in quality. The class-wise image distribution is provided below:

Table 3.2: Distribution of images among the classes in the dataset

No.	Name of class	Number of images
1	Algal leaf	312
2	Grey blight	302
3	Healthy leaf	461
4	Helopeltis	354
5	Looper infested	351
6	Red spider	333
	Total	2113

The pie chart titled "Class Distribution" illustrates the relative proportions of six categories within the dataset. Among these, the healthy class is the most prevalent, comprising 461 samples (21.8%), helopeltis and looper_infested classes follow closely, with 354 (16.8%) and 351 (16.6%) samples respectively, red_spider class includes 333 samples (15.8%), while algal_leaf and gray_blight represent 312 (14.8%) and 302 (14.3%) samples respectively. Overall, the distribution is moderately balanced, though the higher proportion of healthy samples may introduce slight bias in model training.

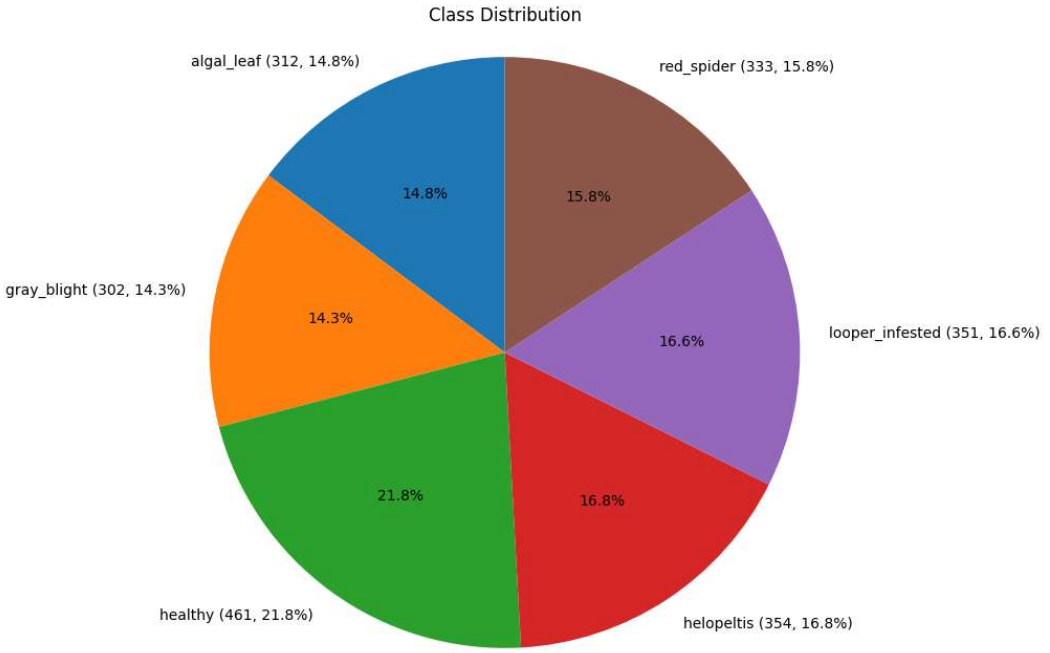


Figure 3.6: Class distribution in each class

Algal leaf | Gray blight | Healthy leaf

Helopeltis Looper infested Red Spider

Figure 3.7: Representative images from each classes

Figure 3.7 illustrates representative sample images from each of the six classes, highlighting the visual differences in disease symptoms, texture, and coloration. These examples reflect the diversity and complexity present in the dataset.

3.2.2 Data Preprocessing

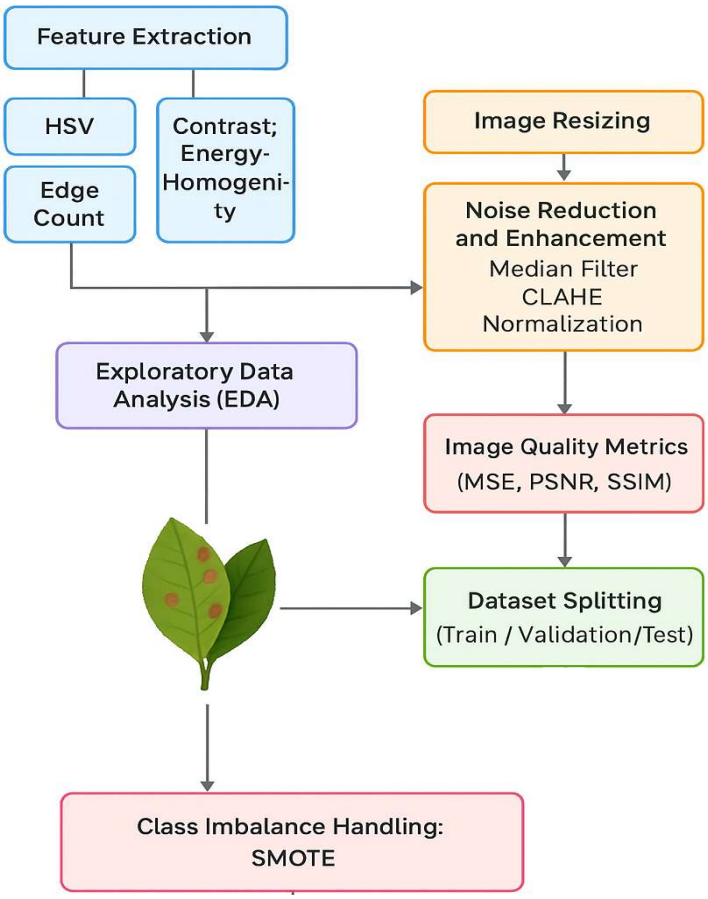


Figure 3.8: End-to-End Data Preprocessing and Analysis Workflow

Data preprocessing is a critical step in developing a robust and accurate tea leaf disease detection system. The collected raw dataset underwent several stages of preprocessing to improve model performance, ensure consistency, reduce noise, and balance the class distribution. In addition, a thorough descriptive and exploratory data analysis was conducted to understand the underlying patterns and statistical characteristics of the dataset. The complete preprocessing pipeline is illustrated in Figure 3.8.

3.2.2.1 Exploratory Data Analysis (EDA)

Understanding the underlying distribution of the dataset is a prerequisite for effective model training, and thus the EDA phase commenced with extracting meaningful handcrafted features to evaluate class-wise visual and statistical trends. A comprehensive set of features was derived from each image to support both quantitative analysis and intuitive interpretation. These included color features like Hue, Saturation, and Value (HSV) to represent chromatic composition; texture features such as Contrast, Energy, and Homogeneity, calculated from the Gray-Level Co-occurrence Matrix (GLCM) to capture spatial texture variations; and Edge Count, obtained using Canny edge detection, to quantify structural complexity present in diseased and healthy leaves. These features helped distinguish the visual uniqueness of each disease class and laid the groundwork for accurate classification. A sample of these extracted values is presented in Table 3.3, highlighting the diversity across different disease categories. To further interpret the data, several visualization techniques were employed. Boxplots (Figure 3.9) revealed the spread and central tendency of each feature, while Heatmaps (Figure 3.10) illustrated correlations to detect redundancy. For dimensionality reduction and pattern discovery, Principal Component Analysis (PCA) (Figure 3.11) projected the high-dimensional data into 2D space, showcasing inter-class clusters, and t-Distributed Stochastic Neighbor Embedding (t-SNE) (Figure 3.12) provided non-linear visualizations to uncover more intricate patterns and separability among classes. Together, these analyses confirmed that the handcrafted features were discriminative and suitable for training robust classification models.

Table 3.3: Sample Extracted Feature Values

Class	Hue	Saturation	Contrast	Energy	Homogeneity	EdgeCount
algal_leaf	26.080	48.363	136.752	0.0414	0.3503	2208.60
gray_blight	25.998	47.370	143.658	0.0399	0.3309	2470.89
healthy	67.418	68.435	69.414	0.0451	0.4294	901.63
helopeltis	95.772	65.625	53.237	0.0462	0.4749	1226.107
looper_infested	63.116	51.205	107.190	0.0592	0.4241	1759.653
red_spider	74.178	48.368	97.091	0.0613	0.5512	828.990

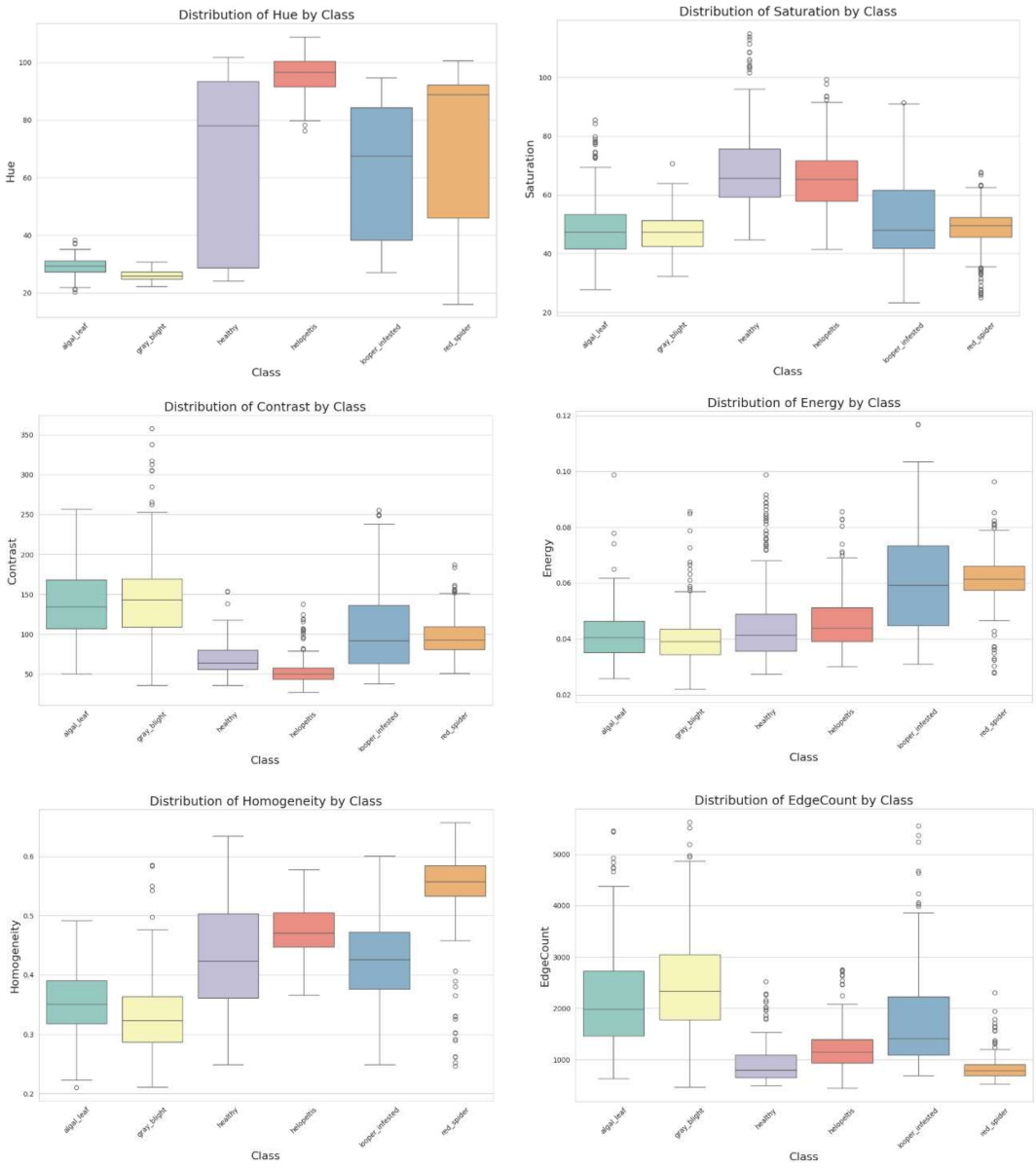


Figure 3.9: Boxplots of Extracted Features by Class

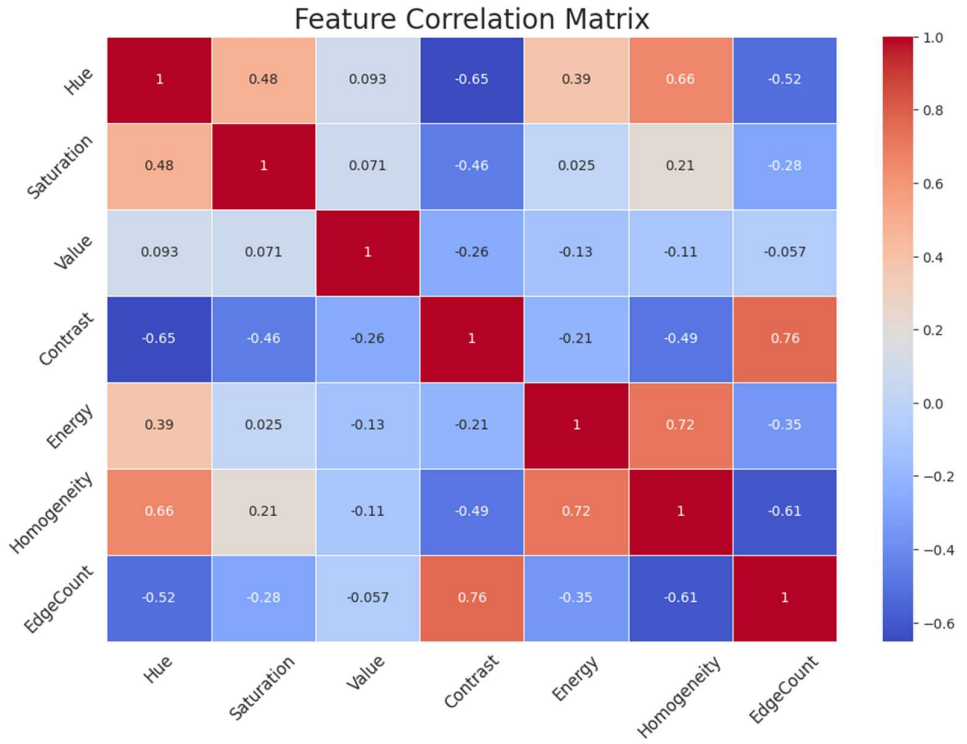


Figure 3.10: Feature Correlation Heatmap

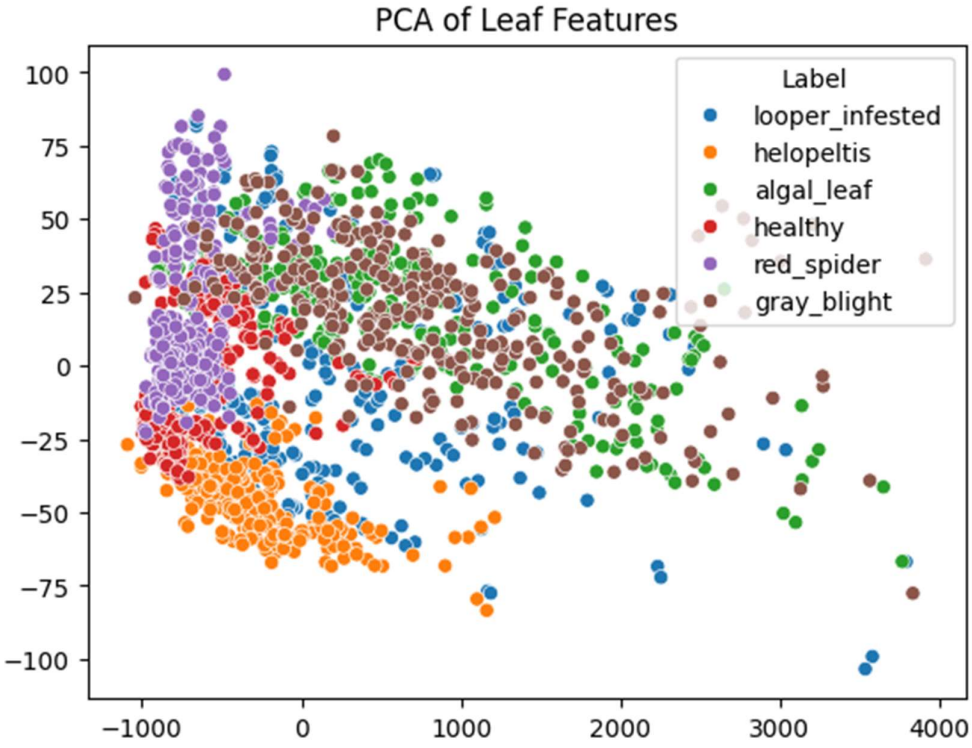


Figure 3.11: PCA Projection of Features

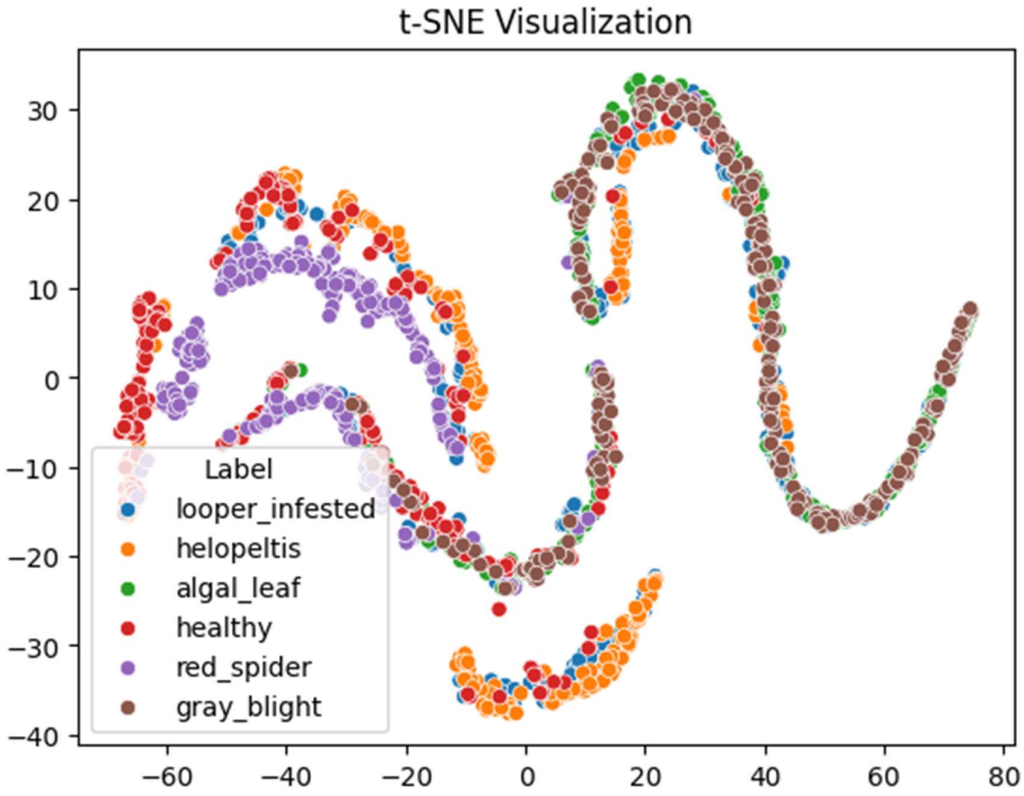


Figure 3.12: t-SNE Visualization of Feature Distribution

3.2.2.2 Image Preprocessing

To ensure consistency and enhance the quality of raw tea leaf images, which initially varied in resolution, lighting, and noise, several image preprocessing techniques were systematically applied. First, all images were resized to a fixed resolution of 720 × 640 pixels to maintain uniformity and ensure compatibility with convolutional neural network (CNN) architectures; this resizing was performed efficiently using Microsoft PowerToys for batch processing. To further refine image clarity, a two-step enhancement process was employed. A Median Filter was applied to suppress salt-and-pepper noise while preserving crucial edge features, thereby maintaining structural integrity. Following this, Contrast Limited Adaptive Histogram Equalization (CLAHE) was used to enhance contrast in regions with subtle variations, such as early disease symptoms that might otherwise be overlooked. Lastly, pixel values were normalized to a [0, 1] range to standardize input scales, stabilize the model training process, and accelerate convergence. The sequential impact of these preprocessing steps is visually depicted in Figure 3.13, which illustrates the transformation from raw input to the final enhanced image ready for analysis.

Figure 3.13: Image Enhancement Stages

3.2.2.3 Image Quality Metrics

To validate that the preprocessing steps effectively enhanced image quality without compromising essential visual details or introducing artifacts, several quantitative image quality metrics were computed. Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) were used to measure pixel-wise differences between the original and enhanced images, providing a baseline assessment of the extent of modification. Peak Signal-to-Noise Ratio (PSNR) was calculated to quantify overall image clarity, where higher PSNR values indicated improved visual quality with minimal distortion. Additionally, the Structural Similarity Index Measure (SSIM) was employed to assess the perceptual and structural similarity between the original and processed images, ensuring that crucial disease-related features remained intact. The results of these evaluations are summarized in Table 3.4 and Table 3.5, which presents the descriptive statistics of all metrics across the dataset. The consistently high SSIM values and acceptable PSNR levels confirmed that the preprocessing pipeline not only preserved but also enhanced key structural and

textural features critical for accurate disease classification.

Table 3.4: Image quality metrics per class

Class	Avg MSE	Avg RMSE	Avg PSNR	Avg SSIM
algal_leaf	2965.6386	52.6637	13.5336	0.6527
gray_blight	2316.2834	49.4868	13.4489	0.6483
healthy	2709.6397	50.4211	13.9311	0.5649
helopeltis	3659.4043	59.0551	12.7201	0.4763
looper_infested	3913.9250	60.7656	12.5762	0.4658
red_spider	2908.4894	52.2065	13.2605	0.6009

Table 3.4: Descriptive statistics for all classes

	Avg MSE	Avg RMSE	Avg PSNR	Avg SSIM
Mean	3128.3968	54.0998	13.2450	0.5682
std	531.8970	4.6786	0.5136	0.0819
Min	2613.2834	49.4868	12.5761	0.4658
25%	2759.3521	50.8674	12.8552	0.4985
50%	2937.0640	52.4351	13.3547	0.5829
75%	3485.9629	57.4573	13.5124	0.6365
max	3913.9250	60.7656	13.9311	0.6527

3.2.2.4 Dataset Splitting

After preprocessing, the dataset was partitioned into training (70%), validation (15%), and testing (15%) subsets using stratified sampling. This ensured that the distribution of each disease class was proportionally maintained across all subsets (Table 3.3). Stratification is crucial in imbalanced datasets to prevent the model from becoming biased toward dominant classes during training. The training set was used to optimize model parameters, while the validation set guided hyperparameter tuning and model selection. The testing set remained untouched until final evaluation to provide an unbiased estimate of model performance. This three-way split established a strong foundation for model generalization and reliable benchmarking.

3.2.2.5 Handling Class Imbalance with SMOTE

Due to the imbalanced nature of the dataset, Synthetic Minority Oversampling Technique (SMOTE) was employed on the feature space extracted after the CNN layers. SMOTE generates synthetic examples for minority classes by interpolating between existing samples, rather than duplicating data. This technique helped to balance the training set and reduce model bias toward dominant classes. By integrating SMOTE after deep feature extraction, the approach preserved complex spatial representations while effectively addressing imbalance, resulting in a more

equitable and accurate classification model. Initially, the training subset was imbalanced. To address this, SMOTE was applied to increase the training data and provide an equal number of images (277) across all the classes. This technique generated synthetic examples for minority classes by interpolating existing samples and effectively balanced the dataset. Class distribution before SMOTE is shown in Figure 3.14. The number of synthetic samples added for each minority class is summarized in Figure 3.15.

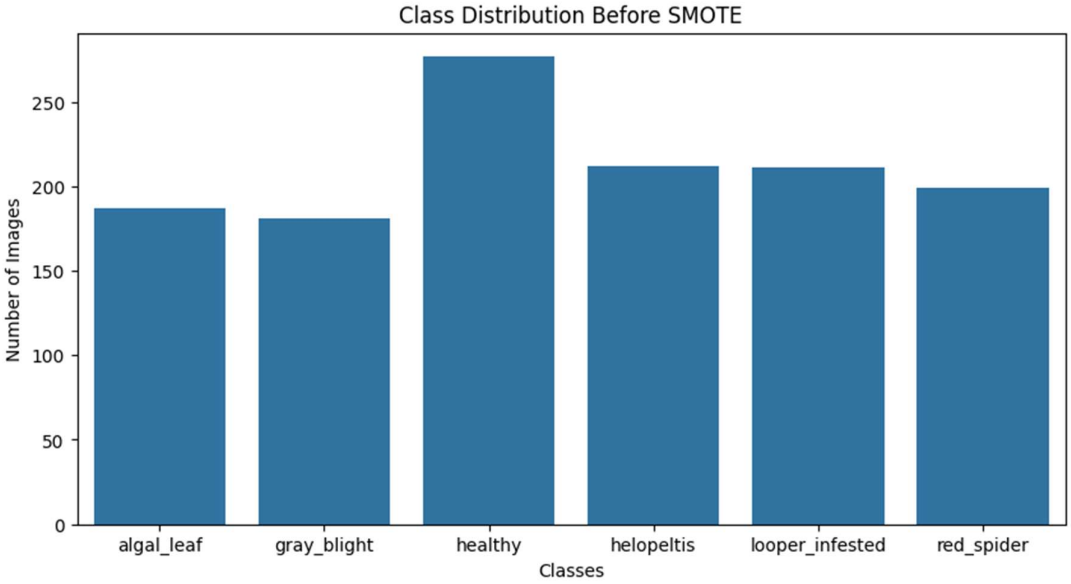


Figure 3.14: Class distribution of training data before SMOTE

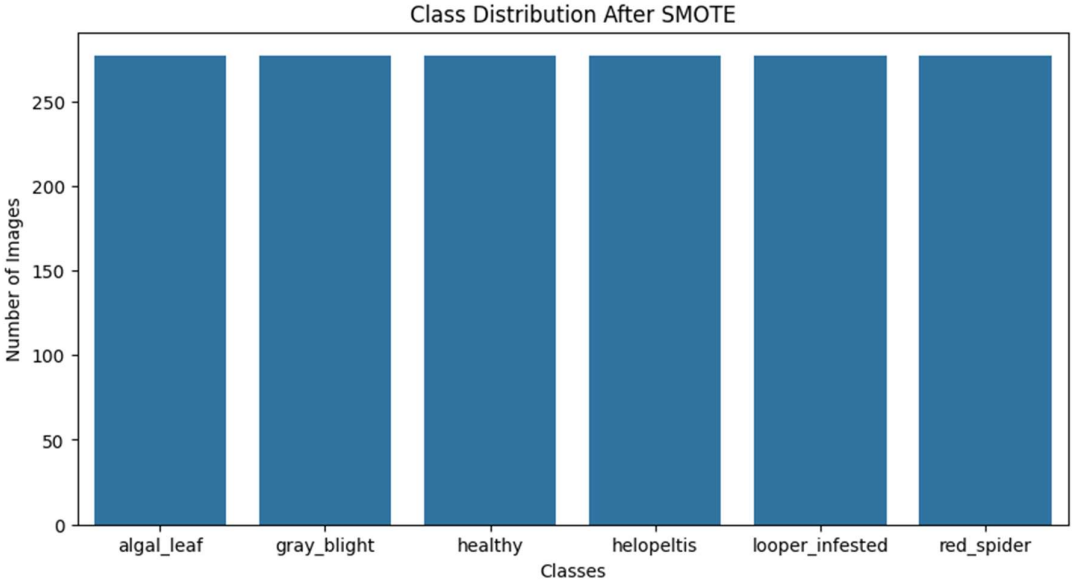


Figure 3.15: Class distribution of training data after SMOTE

3.2.3 Model Selection

In this study, a diverse set of CNN architectures was selected to evaluate their performance in classifying tea leaf diseases. The chosen models represent multiple architectural categories within the broader taxonomy of deep CNNs (Figure 3.16). VGG16 falls under the Spatial Exploitation based CNNs category, ResNet50 and DenseNet121 belong to the Depth based CNNs, InceptionV3 represents the Multi-Path based CNNs, Xception falls under the Width based Multi-Connection CNNs, and MobileNetV2 is classified under the Channel(input) Exploitation based CNNs. The selection of these models was driven by a dual objective: (i) to benchmark widely used deep learning architectures from different design philosophies, and (ii) to evaluate their comparative performance against a proposed lightweight, shallow CNN model.

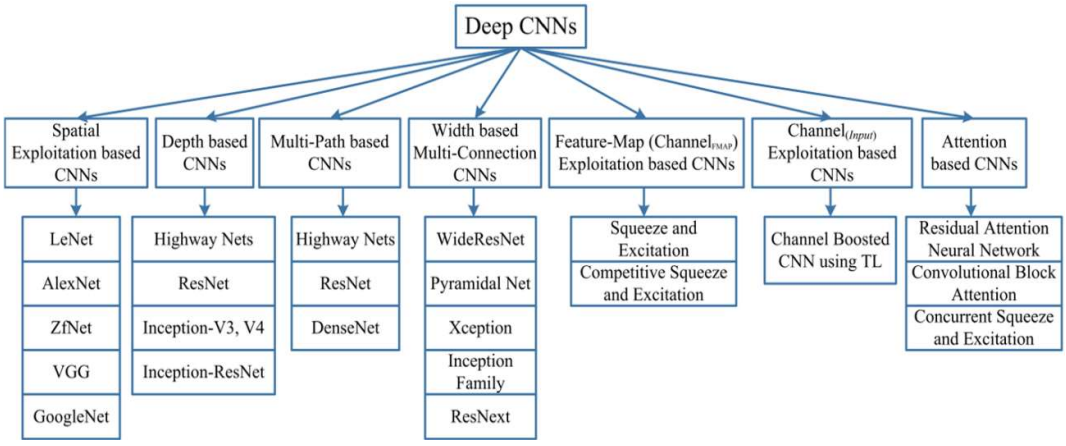


Figure 3.16: Taxonomy of deep CNN architectures

a) VGG16

VGG16 consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. The network is characterized by its uniform architecture, using 3×3 convolutional kernels with stride 1 and padding to preserve spatial resolution. Max pooling layers of size 2×2 are used to progressively reduce the spatial dimensions of the feature maps. ReLU activation functions follow each convolutional layer to introduce non-linearity. The architecture of VGG16 is shown in figure 3.17.

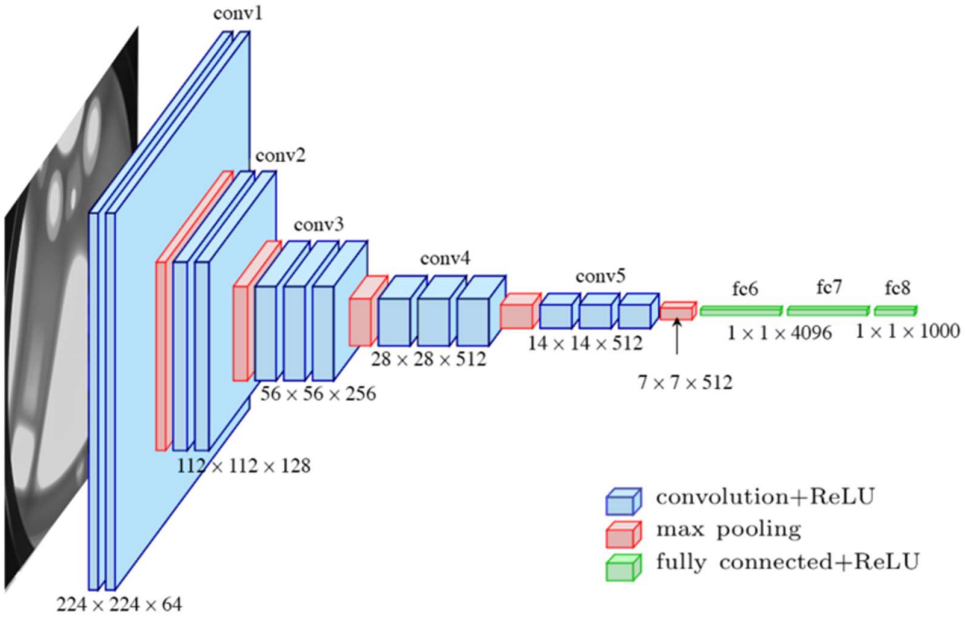


Figure 3.17: Architecture of VGG16 (source: Google)

b) DenseNet121

DenseNet121 introduces an innovative connectivity pattern in which each layer receives input from all preceding layers and passes its output to all subsequent layers as shown in figure 3.18. This dense connectivity improves feature propagation, encourages feature reuse, and significantly reduces the number of parameters compared to traditional CNNs. The architecture is composed of dense blocks separated by transition layers that perform convolution and pooling operations to reduce spatial dimensions. All the dense block contains convolutional layers, batch normalization, and ReLU activations.

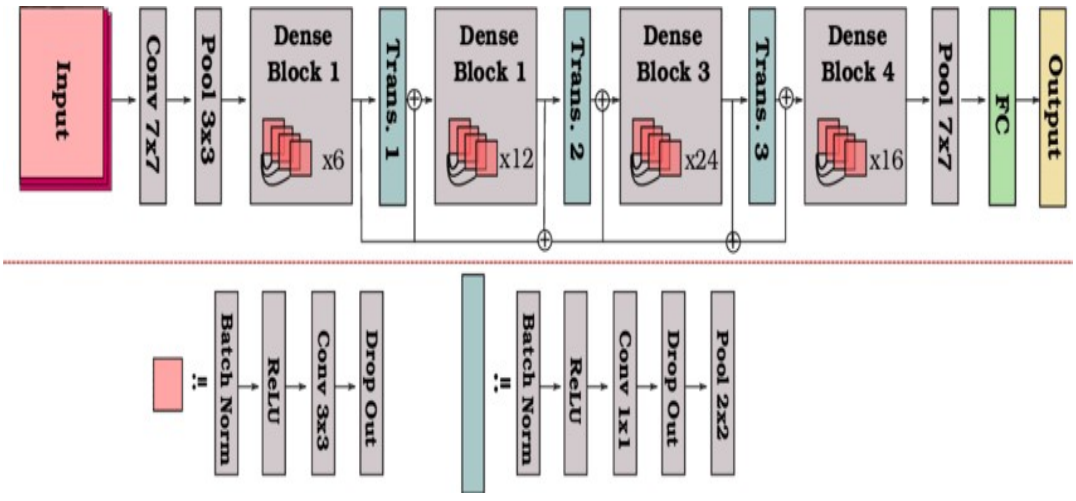


Figure 3.18: Architecture of DenseNet121 (source: Google)

c) MobileNetV2

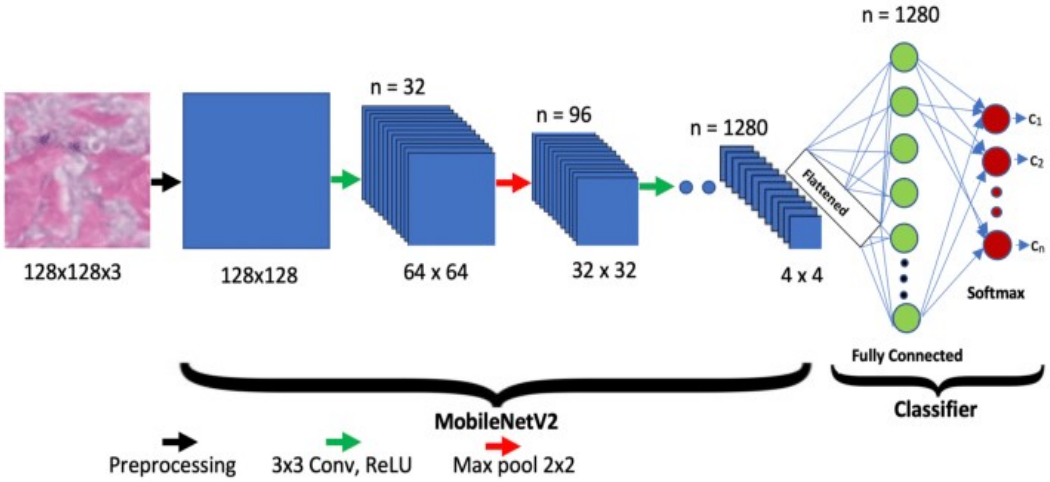


Figure 3.19: Architecture of MobileNetV2 (source: Google)

The key innovation in MobileNetV2 is inverted residual blocks with linear bottlenecks which first expand the input to a larger dimensional space, process this using depthwise separable convolutions and then reduce this back to a lower dimensional space. The design reduces the computational cost and model size by a great bit with the same level of performance. This model uses ReLU6 activation functions and batch normalization for improved stability. The architecture of MobileNetV2 is shown in figure 3.19.

d) Xception

Xception divides the convolution into two stages: a depthwise convolution that operates channel-wise, with a pointwise (1×1) convolution for channel concatenation. The architecture is structured into three flows, entry, middle, and exit, each consisting of multiple convolutional blocks with residual connections to maintain gradient flow. By decoupling spatial and cross-channel correlations, Xception improves representational efficiency and learning capacity. The architecture of Xception is shown in figure 3.20.

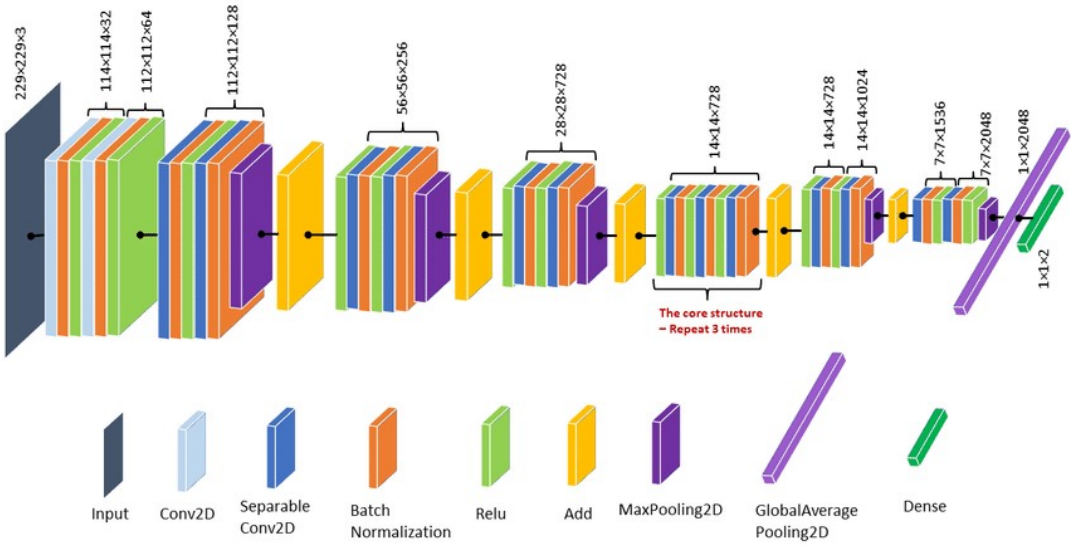


Figure 3.20: Architecture of Xception (source: Google)

e) InceptionV3

InceptionV3 is a deep CNN architecture that extends the original Inception model by optimizing network performance through architectural innovations such as factorized convolutions, asymmetric convolutions, and auxiliary classifiers. These techniques allow for efficient multi-scale feature extraction while reducing computational complexity. The use of grid size reduction modules further minimizes the number of parameters without sacrificing accuracy. InceptionV3's design facilitates the learning of rich hierarchical representations, which is beneficial for classifying multiple disease categories in tea leaves. Its strong generalization capability makes it a reliable choice for real-world plant disease recognition tasks. The architecture of InceptionV3 is shown in figure 3.21.

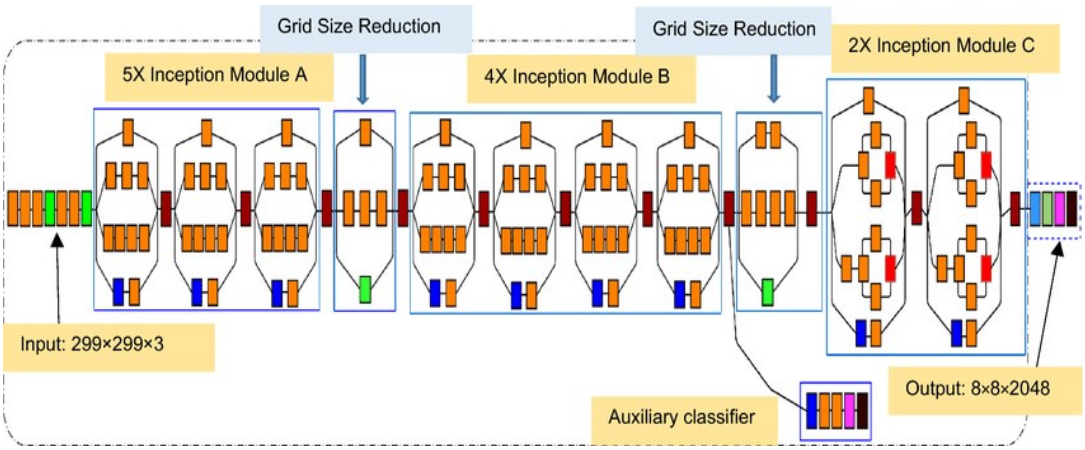


Figure 3.21: Architecture of InceptionV3 (source: Google)

3.2.4 Transfer Learning

To enhance model performance and reduce training time, transfer learning was employed using six pre-trained CNN architectures: VGG16, ResNet50, DenseNet121, MobileNetV2, Xception, and InceptionV3. Instead of training the networks from scratch, which would require extensive computational resources and a large dataset, the pre-trained models serve as powerful feature extractors, leveraging their learned representations of edges, textures, and object parts. In this study, the convolutional base of each pre-trained model was retained, and the final classification layers were replaced with custom dense layers tailored to the number of tea leaf disease classes. These added layers typically included a Global Average Pooling (GAP) layer followed by one or more fully connected (dense) layers and a softmax activation function for multi-class classification. The models were fine-tuned in two phases: initially, only the custom classification layers were trained while keeping the convolutional base frozen; later, selective layers of the base model were unfrozen to fine-tune the entire network, allowing it to adapt more specifically to the characteristics of the tea leaf images. This approach significantly improved learning efficiency and generalization, especially given the limited size and class imbalance of the dataset.

3.2.5 Fine Tuning Strategies

In this study, transfer learning was adopted by fine-tuning well-known CNNs such as VGG16, DenseNet121, MobileNetV2, InceptionV3, and Xception. These models were initially trained on the ImageNet dataset and have learned generic visual features, making them suitable for downstream tasks such as tea leaf disease classification. The final classifier head of each network was replaced with a custom architecture trained on the target dataset, enabling the model to adapt learned features to the specific characteristics of tea leaf disease images.

a) Feature Extraction

The input image $X \in R^{H \times W \times C}$ is passed through the base CNN's convolutional layers to extract deep feature maps. A GAP layer is then applied to reduce each feature map to a single scalar value, transforming a 3D tensor into a 1D vector while preserving semantic information.

Global Average Pooling (GAP)

$$x_{\text{GAP},c} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{i,j,c} \quad \forall c \in \{1, \dots, C\} \qquad (1)$$

- $x_{i,j,c}$:Activation at spatial position (i, j) in channel c.
- $x_{\text{GAP},c}$:GAP output for channel cx
- H, W:Height and width of the feature map

The output of the final convolutional block is a 3D tensor that later reduced to a 1D feature vector using GAP.

$$F_{\text{feat}}(X) = \text{GAP}\left(f_{\text{base}}\left(f_{\text{conv}}(X)\right)\right) \quad (2)$$

Where:

- f_{conv}:are the initial convolutional layers.
- f_{base}:are the deeper layers of the pre-trained
- GAP:denotes global average pooling.
- $F_{\text{feat}}(X) \in R^d$:is the final feature vector.

b) Fine-Tuning Classifier Head

Following feature extraction, a customized classifying head is attached and fine-tuned to classify the image into a tea leaf disease class. The process includes Dense Layers, Dropout, ReLU Activation, Batch Normalization, and Softmax.

Dense Layer (Fully Connected)

$$z = Wx_{\text{GAP}} + b \quad \text{and} \quad z_{\text{ReLU}} = \text{ReLU}(z) \quad (3)$$

- W: Weight matrix of the dense layer.
- b: Bias term.
- ReLU activation introduces non-linearity.
- Dropout (with dropout rate p)

$$z_drop = 0,\text{with probability } p; \frac{1}{1-p}z,\text{with probability } 1-p$$

- Dropout randomly sets activations to zero during training to prevent overfitting.

Softmax Output for Classification

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \quad (4)$$

- $\hat{y}_i$: Probability of input belonging to class i.
- C: Total number of classes.

c) Complete Fine-Tuning Architecture

The final classifier function combining all elements can be expressed as:

$$F_{\text{tuned}}(X) = \text{Softmax}\left(W_3 \cdot \sigma\left(W_2 \cdot \text{Dropout}\left(\sigma\left(W_1 \cdot \text{Dropout}\left(\text{BN}(F_{\text{feat}}(X))\right) + b_1\right)\right) + b_2\right) + b_3\right) \quad (5)$$

Where:

- $F_{\text{feat}}(X) = \text{GAP}\left(f_{\text{base}}\left(f_{\text{conv}}(X)\right)\right)$
- σ:ReLU activation
- BN:Batch Normalization
- W_1, W_2, W_3:Dense layer weights
- b_1, b_2, b_3:Bias terms

This formulation describes the general architecture that was used to fine-tune every CNN backbone.

3.2.6 Proposed M-net Model

Seeing an opportunity in designing a lightweight convolutional neural network to solve the need for a low computation and memory-efficient model deployable on resource-starved environments, this study designed a custom-lightweight convolutional neural network, M-Net. The M-Net architecture comprises a total of 49,292 parameters which consume about 192.55 KB in memory. It combines standard convolution, depthwise separable convolution, batch normalization, ReLU activation, and pooling operations to learn rich feature representations under low complexity. The architecture starts with the first convolutional block by a 2D convolutional layer (Conv2D) containing 32 filter of shape (3×3), stride 2, and with padding = 'same' followed by Batch Normalization and a ReLU activation. Finally, a MaxPooling2D layer is performed to down sample the feature map. Subsequently, the model presents 2 depthwise separable convolutional blocks that combine DepthwiseConv2D and pointwise Conv2D layers. The entry block deforms the channel dimension from 32 to 64 and the second one increases it from 64 to 128. To further reduce spatial dimensions while preserving semantic information, a Global Average Pooling layer is employed. This is followed by a fully connected dense layer with 256 units and ReLU activation, coupled with Dropout (rate = 0.3) to mitigate overfitting. The final output layer uses a Dense layer with a softmax activation function for multi-class classification. Table 3.5 shows the Model summary of the M-Net architecture. Table 3.6 presents the Parameter summary of the M-Net architecture.

Table 3.5: Model summary of the M-Net architecture

Layer Type	Output Shape	Parameters
Conv2D	(112, 112, 32)	896
BatchNormalization	(112, 112, 32)	128
DepthwiseConv2D	(56, 56, 32)	320
Conv2D (pointwise)	(56, 56, 64)	2,112
BatchNormalization	(56, 56, 64)	256
DepthwiseConv2D	(28, 28, 64)	640
Conv2D (pointwise)	(28, 28, 128)	8,320
BatchNormalization	(28, 28, 128)	512
Dense (256 units)	(256)	33,024
Dropout	(256)	0
Output Dense (6 classes)	(6)	3,084

Table 3.6: Parameter summary of the M-Net architecture

Parameter Type	Count	Size
Total Parameters	49,292	192.55 KB
Trainable Parameters	48,844	190.80 KB
Non-trainable Parameters	448	1.75 KB

Figure 3.22 illustrates the overall architectural design of the proposed M-Net model, highlighting each layer's structure and data flow.

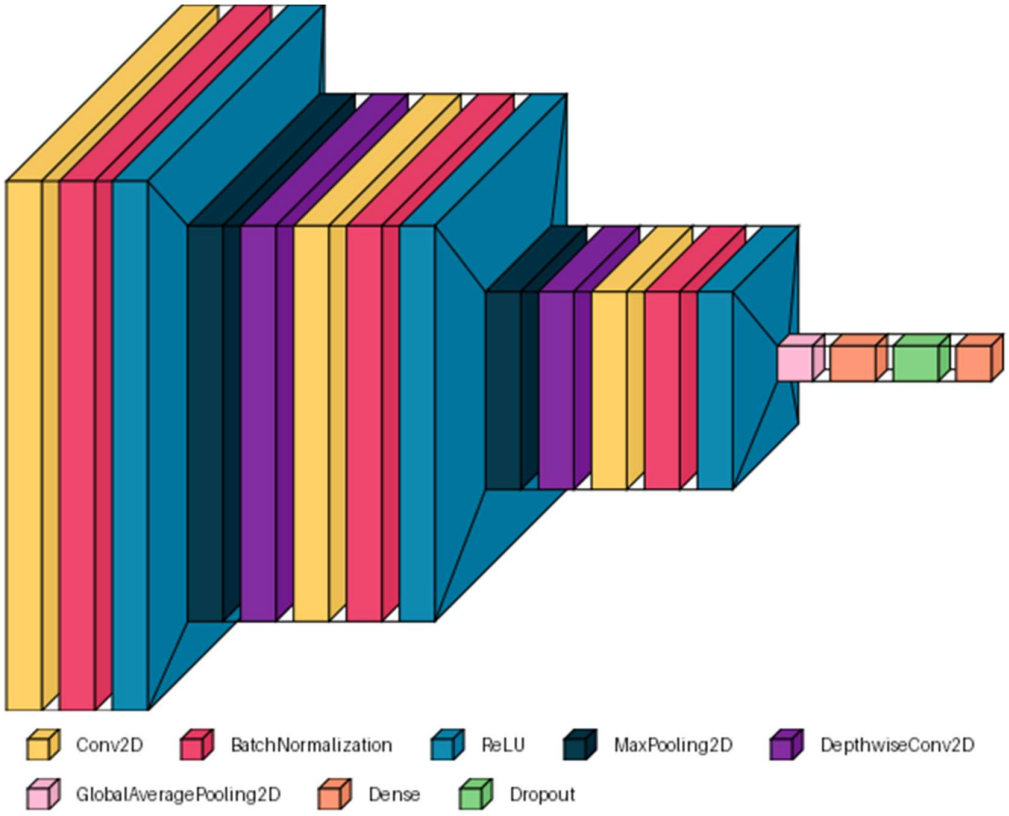


Figure 3.22: Layered view of the proposed M-Net architecture

3.2.7 Explainable AI Integration

LIME: LIME generates image segmentations that highlight the most influential regions contributing to a classification decision. LIME is especially useful for explaining individual class predictions. To create these explanations, LIME transforms the input into a more interpretable format such as marking significant pixels in images or highlighting key words in text data. It works by perturbing the input samples and analyzing how these changes affect the model's output. Through this process, LIME identifies which features are most responsible for specific

predictions. The goal of LIME is to provide clear and trustworthy explanations, which it achieves by minimizing a predefined loss function between the complex model and its interpretable approximation.:

$$\varepsilon(x) = argmin_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g) \quad (6)$$

Here, f is the original model, when g is the interpretable model, x represents the original observation, π_x indicates all possible combinations are to original size, $\mathcal{L}(f, g, \pi_x)$ component is an approach to measure the reliability of g calculating f in the range defined by π, and Ω(g) is an approach to measure the difficulty of a model.

SHAP: SHAP is employed to identify the most influential features contributing to the classification of cancer samples. This method constructs interpretable linear models for individual predictions, aiming to quantify the impact of each feature on the model's output for a specific instance. SHAP values are computed by evaluating how the model's prediction changes when features are included versus excluded across all possible combinations of input features As a result, the model involves retraining on every subset F of the entire set S of features (F ⊆ S). The SHAP value for the jth feature of the instance x is computed by organizing it across all possible subsets Equation:

$$\theta j(x) = \sum_{F \subseteq S - \{j\}} \frac{|F|!\,(|S| - |F| - 1)!}{|S|!} \; [\mathrm{fx}(\mathrm{F} \cup \mathrm{j}) - \mathrm{fx}(\mathrm{F})] \quad (7)$$

where |F|! refers to the permutations of features within the subset F, (|S| - |F| - 1)! The configurations of attributes in the subset S- (F ⊆ {j}) and |S|! stands for the overall count of feature configurations.

GRAD CAM: Grad-CAM is a widely used technique for producing class-discriminative localization maps that offer visual explanations for predictions made by CNNs. Grad-CAM functions by leveraging the gradients of a target class flowing into the final convolutional layer to produce a coarse localization map that highlights the important regions in the input image influencing the prediction. The class biased localization map requires to be identified Grad-CAM $L^c_{Grad-CAM} \in$ of width u and height v for any class c, First we calculate the gradient of the score for class c, y^c((before the SoftMax), as it comes to feature maps A^k of a convolutional layer, i.e. $\frac{\partial y^c}{\partial A^k}$. The neuron importance weights are determined by combining the global-average-pooled gradients that are returning $\propto^c_k$:

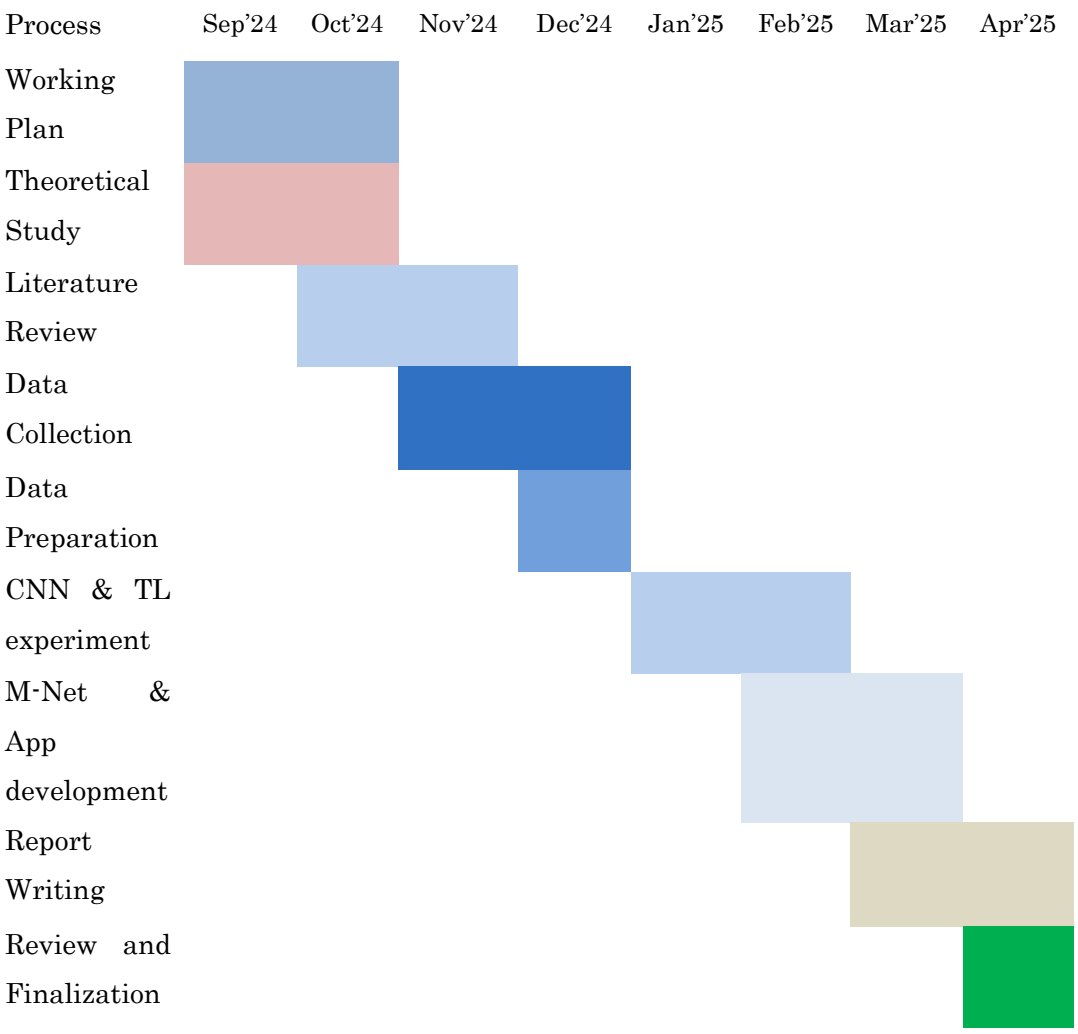
$$\propto^c_k = \sum_i \sum_j \frac{\partial y^c}{\partial A^k_{ij}} \quad (8)$$

In this equation, $\propto_k^c = \sum_i \Sigma_j$ is a global average pooling and $\frac{\partial y^c}{\partial y^c}$ is a gradient via backprop. This weight $\alpha^c{}_k$ is a partial linearization of the deep network next from A, and includes the 'importance' of feature map k for a target class c.

3.3 Project Plan

The project plan of this study is designed to follow a systematic and goal-driven workflow, progressing through several key phases to develop an accurate, explainable, and deployable tea leaf disease detection system. The study begins with data acquisition, where a comprehensive tea leaf image dataset is collected directly from the tea gardens of Sreemangal, Moulvibazar. Following this, the data preprocessing phase includes resizing, normalization, and label encoding. To address class imbalance, SMOTE is applied exclusively to the training data to ensure fair learning across all disease categories. Next, five well-established fine-tuned CNN architectures are implemented and evaluated to determine which performs best in the context of tea leaf disease classification. Following this, transfer learning is applied to the same set of models by freezing their base layers and retraining the top classification layers. Building on these results, a custom lightweight CNN model (M-Net) is designed and trained from scratch. The goal is to assess whether simpler, computationally efficient architecture can achieve comparable or superior accuracy with significantly fewer parameters. To ensure transparency and interpretability in predictions, the best-performing lightweight model is enhanced with XAI techniques. These methods are integrated to provide insight into the decision-making process of the model by highlighting important image regions and feature contributions for each prediction. Once the model is verified and its predictions are explainable, a Streamlit-based web application is developed to demonstrate its functionality. After successful testing, the system is further optimized and converted into a mobile application for on-the-go tea leaf disease detection. Table 3.7 describes the Gantt chart of project timeline.

Table 3.7: Gantt chart of project timeline



3.4 Task Allocation

All these tasks in this thesis were systematized in an organized sequence for streamlined research workflow. The first step comprised collection of tea leaf images from Sreemangal tea gardens in Sylhet for both healthy and disease samples. The dataset then proceeded for its preprocessing in terms of resizing, normalization, and splitting for training sets, validation sets, and testing sets. Balancing of training data was achieved through SMOTE. The experiment commenced using six CNN models that were fine-tuned and tested for evaluation purposes. Transfer learning was then applied in these models for increased accuracy in classifications. The final custom-built lightweight CNN model was called M-Net and was designed for keeping high performance while minimizing computational complexity. To make it interpretable, explanations by methods of XAI including LIME, SHAP, and Grad-CAM were incorporated within the M-

Net model. The first implementation of the model was for testing and visualizing on a web app based on an interface via Streamlit. After successfully deploying it into a web app, the model was converted into TensorFlow Lite and put into a mobile app for in-time tea leaf disease determination. The final step was final evaluation and comparison of all the results and documentation in detail.

3.5 Summary

The methodology which was pursued for the development of an explainable and optimal CNN-based system for tea leaf disease detection was explained in this chapter. The process involved collecting data of actual tea gardens and the preprocessing and data-augmentation methods for increasing the robustness of the dataset. The experimental phases recorded were testing optimized CNNs as well the transfer learning based methods, for a custom light weight model, M-Net to be developed and deployed in mobile platforms. For end-users to understand the system, explainability was discussed with the help of XAI tools like LIME, SHAP and Grad-CAM. The chapter also included system-level design details such as the context diagram, data flow diagram (Level 1), and an intuitive user interface plan. A two-tier architecture was described clearly separating user interaction from backend computation. Additionally, the project plan and task allocation demonstrated the structured and organized nature of the research, enabling the successful execution of each milestone. Through this detailed and methodical approach, the groundwork has been laid for implementing a functional, efficient, and interpretable tea leaf disease detection solution suitable for real-world agricultural applications.

Chapter 4

Implementation and Results

This chapter outlines the performance of the proposed methodological framework established for the diagnosis of tea leaf diseases with the help of deep learning methods anchored in XAI. The first part of the chapter presents an exhaustive investigation of experimental setup, both hardware and software characterizations that were utilized while training and testing the model. The following part reviews the performance of different thoroughly optimized CNN and transfer learning architectures. Apart from this, a custom CNN architecture, known as M-Net, is also presented to improve performance with minimal use of computations. The chapter also employs XAI methodologies to improve model prediction interpretability.

4.1 Environment Setup

The experiments in this study were performed with the use of Python version 3.10, with TensorFlow and Keras forming the core development frameworks. The computing resources utilized in this study were Google Colab Pro, which provided GPU support through NVIDIA Tesla T4 GPUs, hence allowing for improved training and testing processes inherent in the respective deep-learning frameworks. Integration of various libraries relating to both deep learning and machine learning, such as NumPy, Pandas, Matplotlib, scikit-learn, OpenCV, and imbalanced-learn for the use of SMOTE, were needed to implement these frameworks. The framework was developed according to set world standards in engineering to ensure compatibility, adaptability, data privacy, and usability in various applications in software. Table 4.1 provides an overview of the principal software and libraries used throughout the study.

Table 4.1: Software and libraries used

Tool/Library	Purpose
Python 3.10	Core programming language
TensorFlow 2.x	Deep learning framework
Keras	High-level neural network API

NumPy	Numerical operations and array manipulation
Pandas	Data handling and manipulation
Matplotlib, Seaborn	Visualization and plotting
OpenCV	Image preprocessing and manipulation
imbalanced-learn	Implementing SMOTE for class balancing
SHAP, LIME	Explainable AI tools
Streamlit	Web interface for initial model deployment
Flutter	Mobile application development and deployment

The data acquired from the regional plantations went through preprocessing, where all the images were resized to a uniform size of 224×224 pixels. Then, data augmentation and normalization methods were applied to enhance the quality of the dataset. The dataset was further divided into training, validation, and testing subsets with percentage splits of 70:10:20. The training dataset alone was treated with the SMOTE for handling class imbalance, so that an improved balanced dataset can be created for training the model. Additionally, EarlyStopping, ReduceLROnPlateau, and ModelCheckpoint features were incorporated to improve model training performance and prevent overfitting. These callbacks were created to check performance metrics while adjusting parameters adaptively. Based on trends in validation loss, training would halt or stop. Ultimately, the best performing model was selected for an interpretability study with Grad-CAM, LIME, and SHAP methods. The model was then transformed into TensorFlow Lite compatible formats and .h5 files for the purpose of deployment.

Table 4.2: Model training parameters

Parameter	Value
Input Image Size	224 × 224 × 3
Batch Size	32
Epochs	15
Optimizer	Adam
Learning Rate	0.0001
Loss Function	Categorical Crossentropy

4.2 Testing and Evaluation

In the evaluation of the accuracy of models trained for the classification of tea leaf diseases, several essential metrics have been used to ensure that the assessment is comprehensive and reliable. Such metrics are not only important to understand the models' overall accuracy but also their robustness, types of errors, and generalizability, which are especially important in the application of plant disease classification, where misclassifications can cause significant implications in agricultural practice.

Accuracy: Accuracy measures the proportion of correctly identified images of tea leaves (both the total number of images, including diseased and healthy cases).

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \quad (9)$$

Recall (Sensitivity): Recall is the ratio of correctly identified positive cases (e.g., leaves showing disease) to all the true positives. High recall means that in most cases of illness, the model picks up these cases correctly.

$$Recall = \frac{TP}{(FN + TP)} \quad (10)$$

Precision: Precision is the proportion of predictions of positive cases which are accurate among all positive predictions. In this sense high precision represents the ability of the model not to mislabel healthy leaves as diseased.

$$Precision = \frac{TP}{(FP + TP)} \quad (11)$$

F1-Score: The F1 score gives an appropriate balance of assessment by combining measurements of precision and recall into one measurement. It is especially relevant to this study because it considers both types of errors (false positive and false negative) during work on multiclass leaf disease detection tasks.

$$F1\ Score = 2 \times \frac{Precision + Recall}{(Precision \times Recall)} \quad (12)$$

False Alarm Rate (FAR): In terms of False Alarm Rate, it refers to the prevalence of the actual negative cases like the healthy leaves (negative examples), that, incorrect manner, were predicted as diseased (positive examples). Under this study a more desirable FAR is that farmers will not be lured by unnecessary interventions.

$$FAR = \frac{FP}{(FP + TN)} \quad (13)$$

Confusion Matrix: The confusion matrix is an interesting tool where one can summarize the number of correct and wrong classification for all classes. When applied to the dataset of tea leaf diseases it can help analyze which diseases are most often mislabeled therefore, showing model strengths and weaknesses and possibly areas for improvement.

Loss and Accuracy Curves: The Loss and Accuracy plots demonstrate the model's learning behavior during each training epoch. These plots are important for diagnosing underfitting/in overfitting problem since the visitors are being compared, the performance metric of training set with validation set.

- Accuracy Curve: The graph shows improvement in the model's classification accuracy over the training phase. A significant difference in the training accuracy and validation accuracy can indicate the presence of overfitting.
- Loss Curve: The graph shows the loss values for both the training and validation datasets. A downward trend generally indicates successful learning, while an increasing validation loss can be an indicator of the onset of overfitting or poor generalization capacity.

4.3 Results and Discussion

This section clarifies the diverse outcomes and findings obtained through experimental evaluations in this study. The performance of five systematically optimized CNN models, which have embedded various transfer learning methods with the novel light-weight M-Net architecture, is evaluated regarding an exhaustive set of classification measures. Then, techniques for XAI are applied to the developed M-Net model to further enhance the model's interpretability. The performance of the formulated system is further confirmed through the application of the proposed framework as both an Android application and an interactive Streamlit web application.

4.3.1 Results of Original CNN Models

Among the original CNN models tested, MobileNetV2 achieved the highest accuracy at 98.40%, followed by DenseNet121 (97.44%) and InceptionV3 (96.96%). Xception also performed well with 96.01%, while VGG16 had the lowest accuracy at 88.94%. These results highlight the superior performance of modern, lightweight architectures like MobileNetV2 for tea leaf disease classification.

Table 4.3: Accuracy of fine-tuned CNN models

Model	Accuracy
DenseNet121	97.44%
VGG16	88.94%
InceptionV3	96.96%
MobileNetV2	98.40%
Xception	96.01%

The performance of five CNN architectures, DenseNet121, VGG16, InceptionV3, MobileNetV2, and Xception were evaluated for tea leaf disease detection.

MobileNetV2 emerged as the top performer, achieving near-perfect precision, recall, and F1-scores across all disease classes, particularly excelling in detecting healthy, helopeltis, and red_spider leaves, all with extremely low false alarm rates (as low as 0.0). DenseNet121 also demonstrated strong and balanced performance, with high F1-scores across all categories and excellent detection of helopeltis and red_spider diseases (F1-scores of 0.99 and 0.97 respectively). However, it showed a slightly higher false alarm rate for healthy leaves (0.0102). InceptionV3 maintained robust classification performance with high F1-scores (≥0.96) in most classes and perfect precision for algal_leaf and helopeltis. Its low false alarm rates confirm its reliability, although it was marginally behind MobileNetV2 in overall accuracy. Xception produced consistent results, especially in helopeltis and healthy detection, but exhibited a slight dip in precision for algal_leaf (0.92) and looper_infested (0.95), along with relatively higher false alarm rates in those classes.

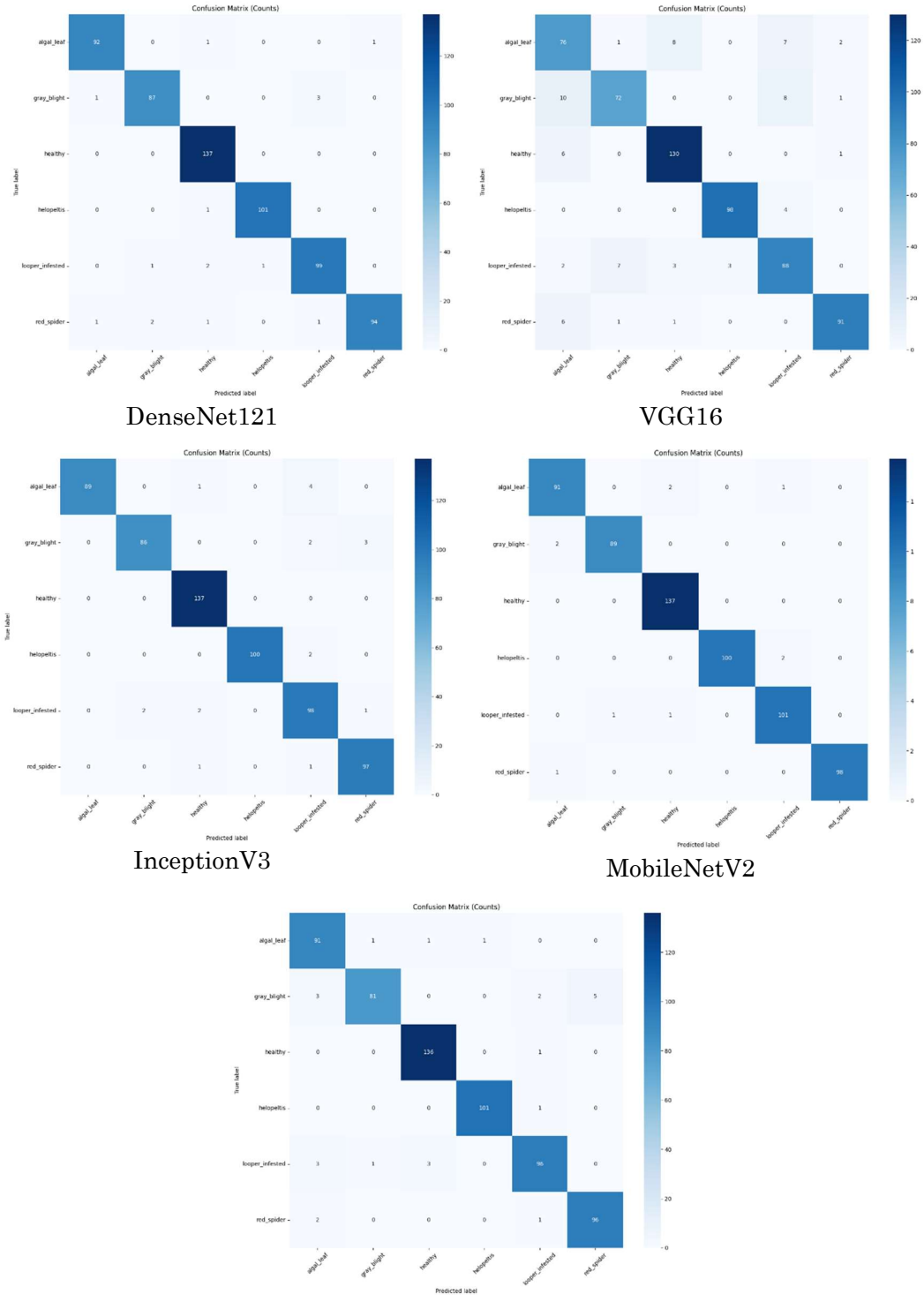
Table 4.4: Classification report of the fine-tuned CNN models

Model	Class	Precision	Recall	F1-score	False Alram Rate
DenseNet121	algal_leaf	0.98	0.98	0.98	0.0038
	gray_blight	0.97	0.96	0.96	0.0056
	healthy	0.96	1.00	0.98	0.0102
	helopeltis	0.99	0.99	0.99	0.0019
	looper_infested	0.96	0.96	0.96	0.0076
	red_spider	0.99	0.95	0.97	0.0019
VGG16	algal_leaf	0.76	0.81	0.78	0.0451
	gray_blight	0.89	0.79	0.84	0.0168
	healthy	0.92	0.95	0.93	0.0245
	helopeltis	0.97	0.96	0.97	0.0057
	looper_infested	0.82	0.85	0.84	0.0363
	red_spider	0.96	0.92	0.94	0..0076
InceptionV3	algal_leaf	1.00	0.95	0.97	0.0000
	gray_blight	0.98	0.95	0.96	0.0037
	healthy	0.97	1.00	0.99	0.0082
	helopeltis	1.00	0.98	0.99	0.0000
	looper_infested	0.92	0.95	0.93	0.0172
	red_spider	0.96	0.98	0.97	0.0076
MobileNetV2	algal_leaf	0.97	0.97	0.97	0.0056
	gray_blight	0.99	0.98	0.98	0.0019

	healthy	0.98	1.00	0.99	0.0061
	helopeltis	1.00	0.98	0.99	0.0000
	looper_infested	0.97	0.98	0.98	0.0057
	red_spider	1.00	0.99	0.99	0.0000
Xception	algal_leaf	0.92	0.97	0.94	0.0150
	gray_blight	0.98	0.89	0.93	0.0037
	healthy	0.97	0.99	0.98	0.0082
	helopeltis	0.99	0.99	0.99	0.0019
	looper_infested	0.95	0.93	0.94	0.0096
	red_spider	0.95	0.97	0.96	0.0095

Figure 4.1 shows the confusion matrices of five original CNN models, DenseNet121, VGG16, InceptionV3, MobileNetV2, and Xception used for tea leaf disease classification across six categories. MobileNetV2 and DenseNet121 exhibit strong diagonal dominance, indicating high classification accuracy. For example, DenseNet121 correctly classifies 138 healthy, 105 helopeltis, and 98 red_spider samples, with very few misclassifications such as one algal_leaf predicted as healthy. InceptionV3 also achieves high accuracy, correctly identifying nearly all instances in most classes, including 138 healthy and 105 helopeltis, with only one or two misclassifications per class. Xception performs similarly, though it shows slightly more off-diagonal values compared to MobileNetV2. In contrast, VGG16 shows more confusion among classes, with only 86 correct predictions for gray_blight and more frequent misclassifications such as algal_leaf and looper_infested being wrongly predicted.

The training and validation loss and accuracy curves reveal key performance differences among the CNN models. DenseNet121, InceptionV3, and Xception show smooth convergence with minimal gaps between training and validation, indicating strong generalization and stable learning. MobileNetV2 also demonstrates good alignment, though slightly less pronounced. In contrast, VGG16 shows signs of overfitting, with a significant gap between training and validation accuracy and early plateauing of validation loss. Overall, DenseNet121 and InceptionV3 stand out for their consistent and reliable performance.

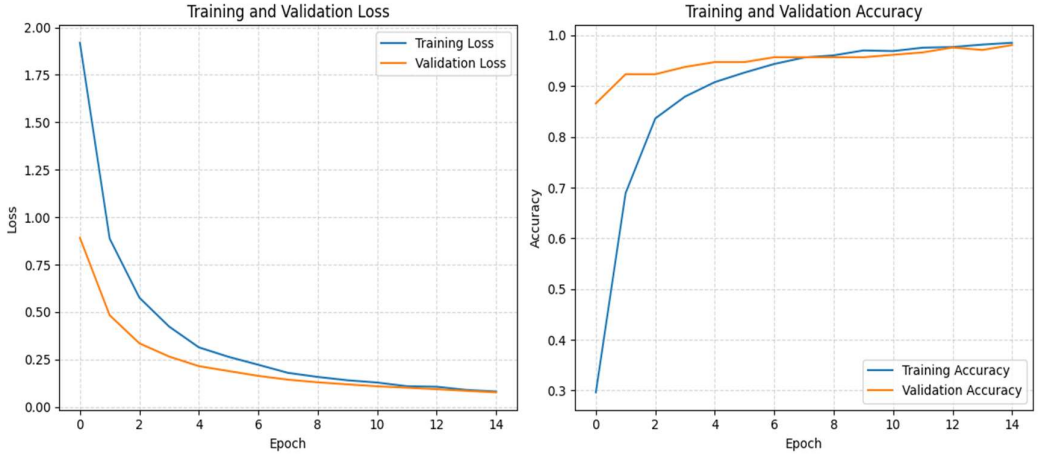


DenseNet121 VGG16

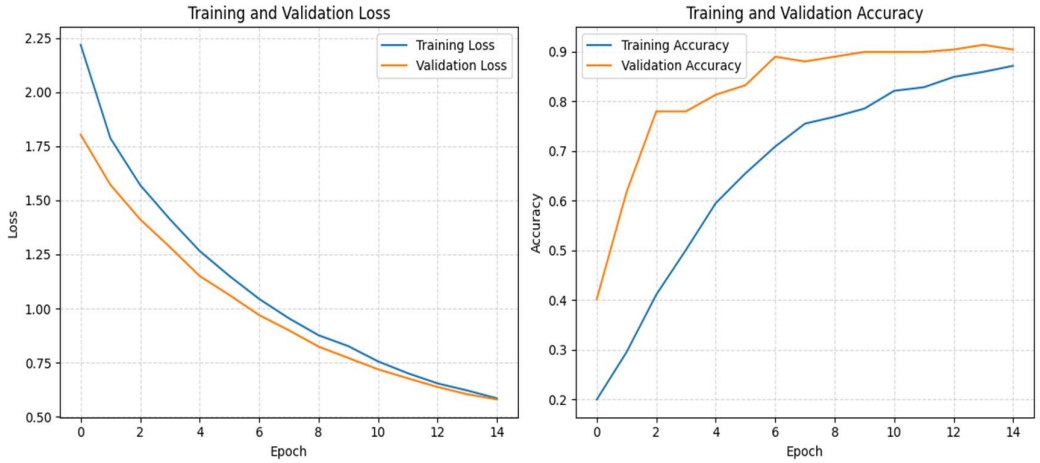
InceptionV3 MobileNetV2

Xception

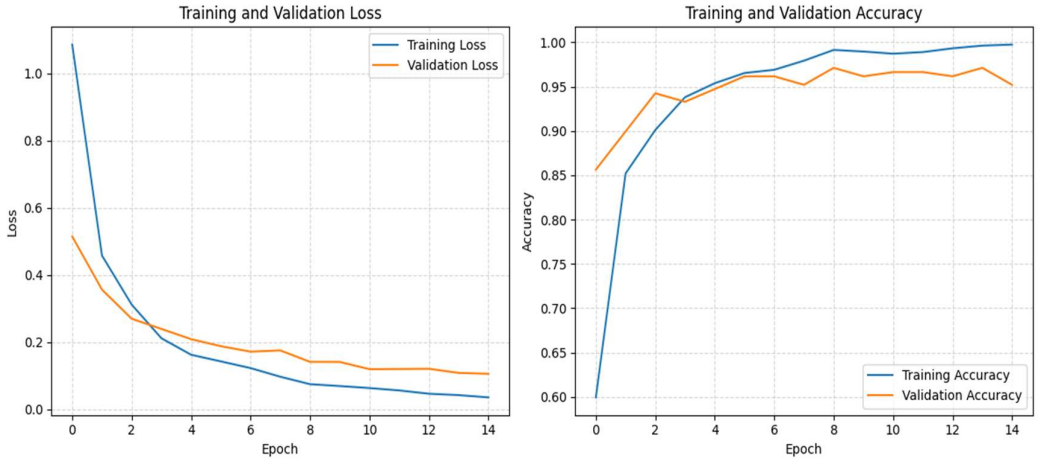
Figure 4.1: Confusion matrices of fine-tuned CNN models



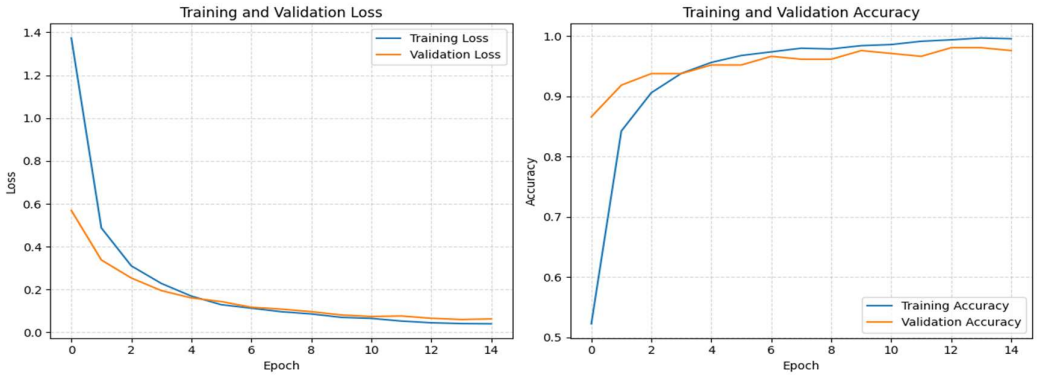
DenseNet121



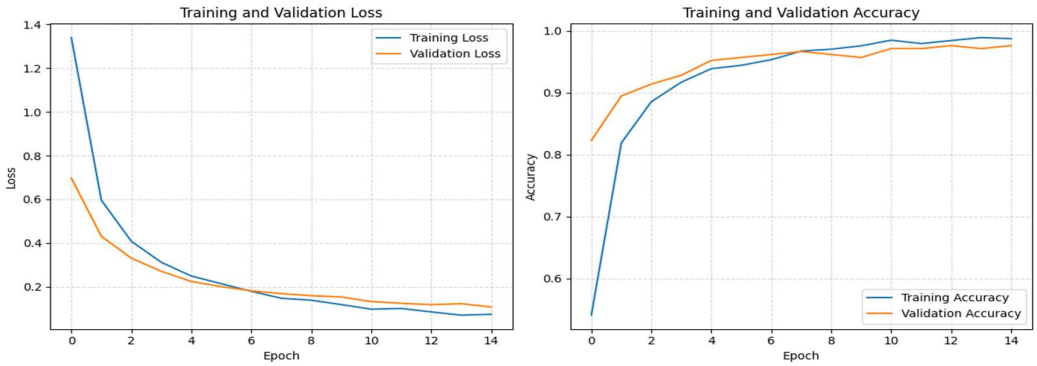
VGG16



InceptionV3



MobileNetV2



Xception

Figure 4.2: Training and validation curves (loss and accuracy) for each fine-tuned CNN models

4.3.2 Results of Fine Tuned Transfer Learning models

After fine-tuning the transfer learning models for tea leaf disease detection, a significant improvement in overall accuracy was observed in most models. VGG16 and MobileNetV2 showed the most substantial gains, with VGG16 achieving the highest accuracy at 99.68%, up from 88.94% in its original form. MobileNetV2 also improved from 98.40% to 99.52%, confirming that fine-tuning enhanced its feature representation for the specific dataset. Similarly, DenseNet121 and InceptionV3 maintained high accuracy at 97.79%, comparable to or slightly better than their original versions (97.44% and 96.96%, respectively). However, Xception exhibited a notable negative transfer, with accuracy dropping from 96.01% in the original model to 73.50% after fine-tuning.

Table 4.5: Accuracy of fine-tuned transfer learning models

Model	Accuracy
DenseNet121	97.79%
VGG16	99.68%
InceptionV3	97.79%
MobileNetV2	99.52%
Xception	73.50%

Table 4.2 displays the classification performance results of several transfer learning models: DenseNet121, VGG16, InceptionV3, MobileNetV2, and Xception; for various leaf conditions. The metrics are Precision, Recall, F1-score and Support for classes like; algal_leaf, gray_blight, healthy, helopeltis, looper_infested and red_spider. MobileNetV2 shows the best performance and stateless scoring (1.00) for several classes: "helopeltis", "looper_infested", "red_spider". DenseNet121, VGG16, and InceptionV3 also perform well across most classes, whereas Xception struggles with minority classes like "algal_leaf" and "gray_blight".

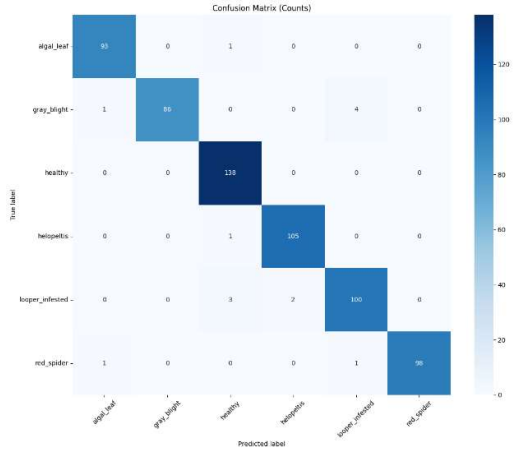
Table 4.6: Classification report of fine-tuned transfer learning models

Model	Class	Precision	Recall	F1-score	False Alram Rate
DenseNet121	algal_leaf	0.98	0.99	0.98	0.0037
	gray_blight	1.00	0.95	0.97	0.0000
	healthy	0.97	1.00	0.98	0.0101
	helopeltis	0.98	0.99	0.99	0.0038
	looper_infested	0.95	0.95	0.95	0.0095
	red_spider	1.00	0.98	0.99	0.0000
VGG16	algal_leaf	0.98	0.99	0.98	0.0000
	gray_blight	1.00	0.95	0.97	0.0018
	healthy	0.97	1.00	0.98	0.0000
	helopeltis	0.98	0.99	0.99	0.0019
	looper_infested	0.95	0.95	0.95	0.0000
	red_spider	1.00	0.98	0.99	0.0000
InceptionV3	algal_leaf	0.98	0.99	0.98	0.0037
	gray_blight	1.00	0.95	0.97	0.0000
	healthy	0.97	1.00	0.98	0.0101
	helopeltis	0.98	0.99	0.99	0.0038
	looper_infested	0.95	0.95	0.95	0.0095
	red_spider	1.00	0.98	0.99	0.0000
MobileNetV2	algal_leaf	0.99	0.99	0.99	0.0019
	gray_blight	0.99	0.99	0.99	0.0018
	healthy	0.99	1.00	0.99	0.0000
	helopeltis	1.00	1.00	1.00	0.0019
	looper_infested	0.99	0.99	0.99	0.0000
	red_spider	1.00	1.00	1.00	0.0000

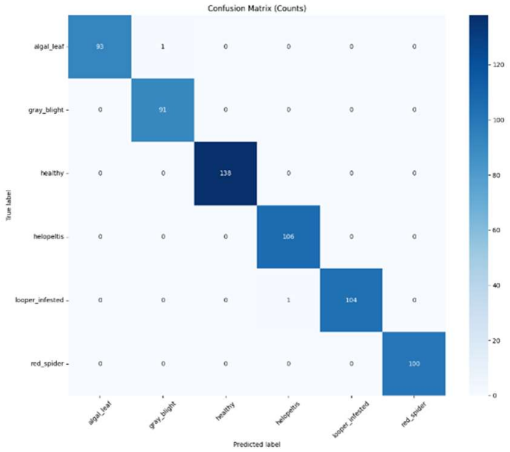
Xception	algal_leaf	1.00	0.15	0.26	0.0000
	gray_blight	1.00	0.31	0.47	0.0000
	healthy	0.93	0.99	0.96	0.0202
	helopeltis	1.00	0.78	0.88	0.0000
	looper_infested	0.54	0.99	0.70	0.1682
	red_spider	0.59	1.00	0.74	0.1292

The confusion matrices quantitatively assess the performance of each transfer learning model. DenseNet121 shows strong performance, correctly classifying 158 healthy, 105 looper_infested, and 98 red_spider samples, with very few misclassifications, typically one or two per class. InceptionV3 performs similarly well, accurately predicting 150 healthy and 105 looper_infested samples, with only two misclassifications in most classes. MobileNetV2 maintains good accuracy, correctly identifying 150 healthy and 98 red_spider cases, but with slightly more confusion, particularly between algal_leaf and helopeltis. Xception achieves 148 correct predictions for healthy but shows notable confusion among looper_infested, red_spider, and helopeltis, with around 7 misclassifications. VGG16 records the lowest performance, with only 88 correct predictions for red_spider and frequent misclassifications between similar classes such as algal_leaf and looper_infested.

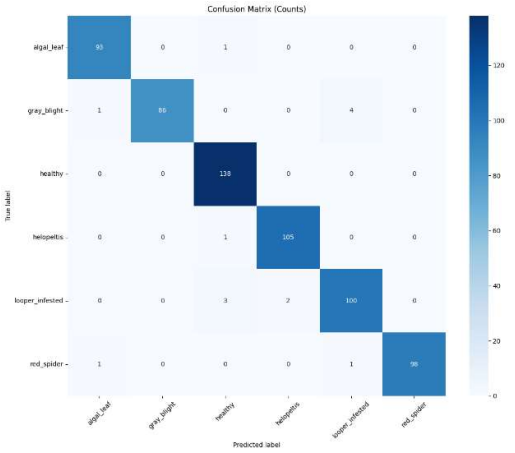
The training and validation curves provide key insights into the learning dynamics of each transfer learning model over the training epochs. DenseNet121, MobileNetV2, and InceptionV3 converge smoothly within the first 10–15 epochs, with steadily decreasing loss and rising accuracy, indicating efficient learning and good generalization. VGG16 also converges early but shows a noticeable gap between training and validation accuracy, hinting at overfitting despite a stable loss curve. Xception, however, struggles to stabilize—its validation loss fluctuates significantly, and accuracy remains inconsistent throughout training, suggesting sensitivity to the learning process or data.



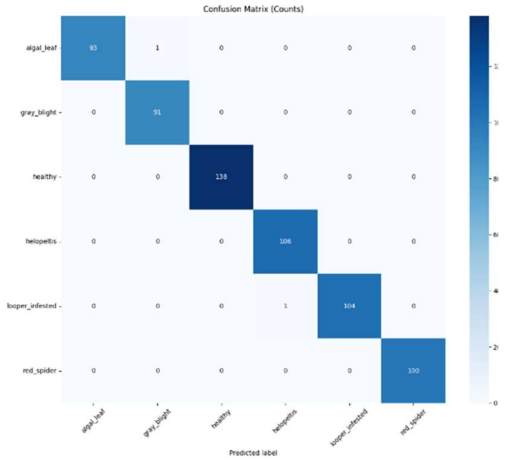
DenseNet121



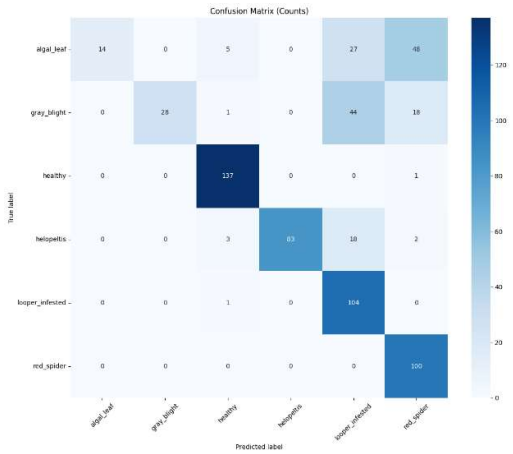
VGG16



InceptionV3

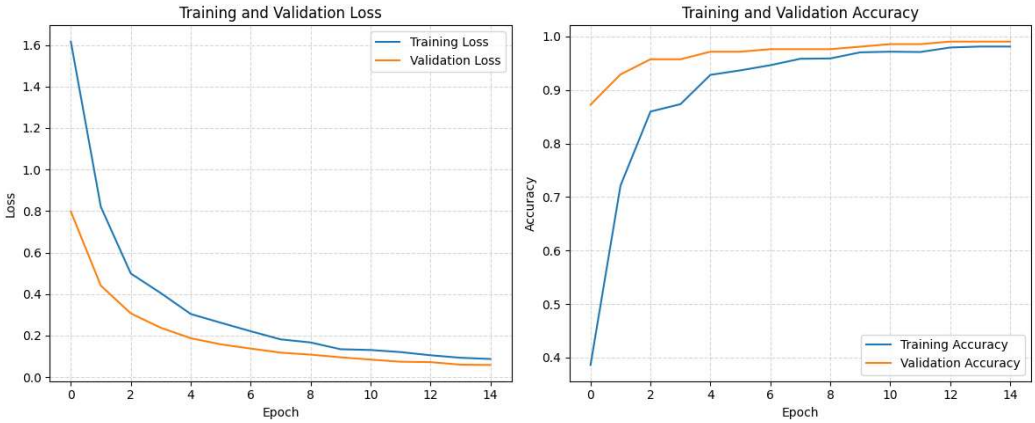


MobileNetV2

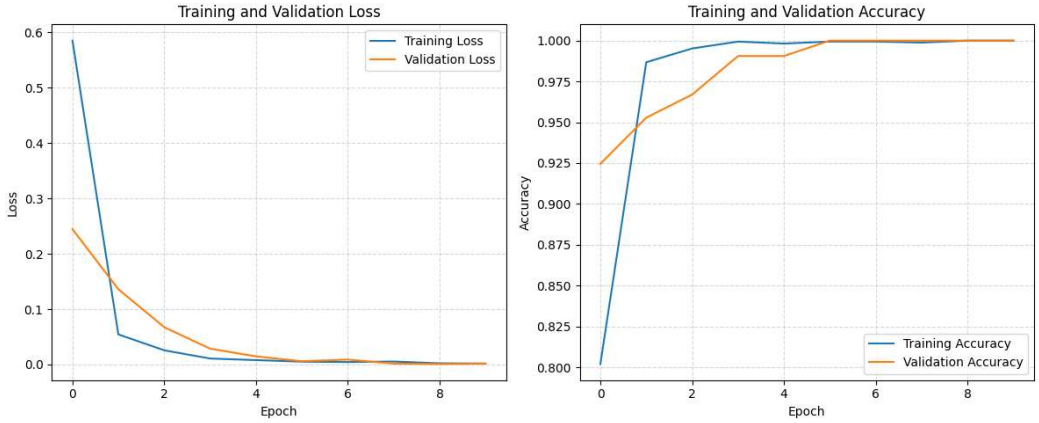


Xception

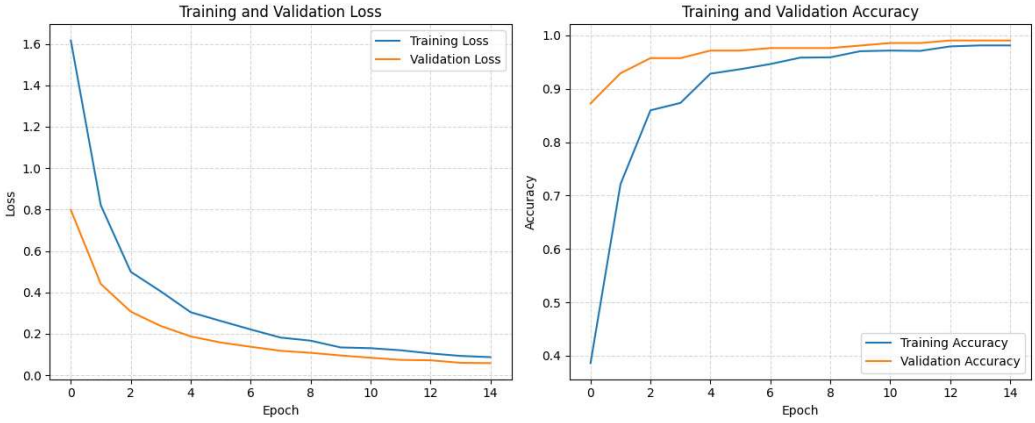
Figure 4.3: Confusion matrices of fine-tuned transfer learning models



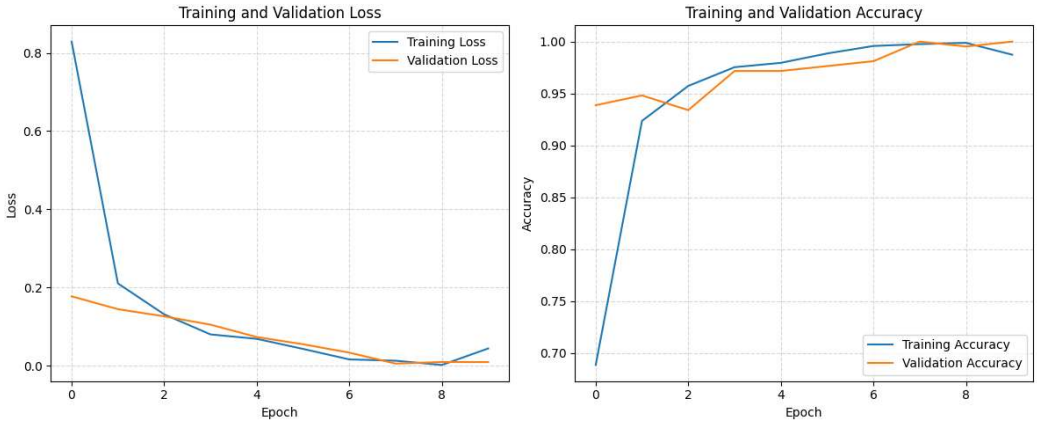
DenseNet121



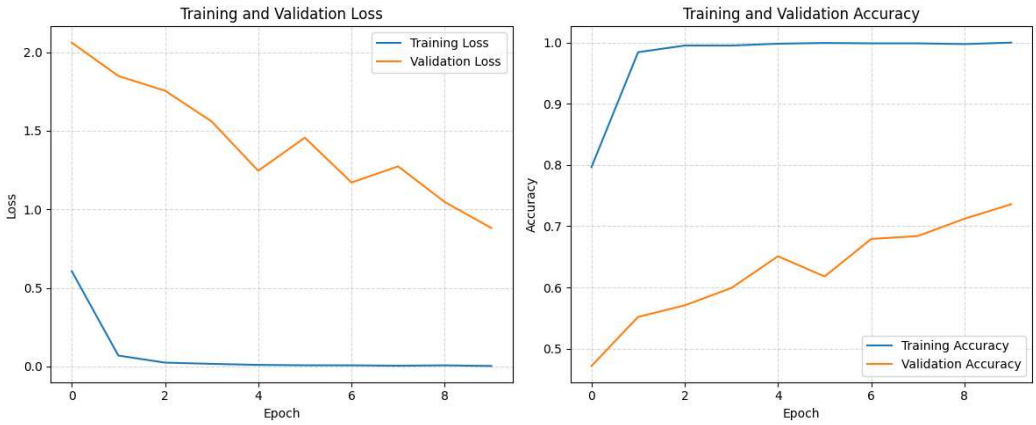
VGG16



InceptionV3



MobileNetV2



Xception

Figure 4.4: Training and validation curves (loss and accuracy) for each fine-tuned transfer learning models

4.3.3 Results of M-Net

The model's performance metrics show strong results, with an Accuracy of 97.29%. Both Precision and Recall are 97.12%, indicating a high rate of correct positive identifications and minimal false negatives. The F1 score is 97.00%, reflecting a balanced performance. Additionally, the False Alarm Rate is very low at 0.54%, suggesting a few misclassifications of negative instances. Overall, the model demonstrates excellent accuracy and reliability.

Table 4.7: Performance of the M-Net model

Matrices	Values
Accuracy	97.29%
Precision	97.12%
Recall	97.12%
F1-score	97.00%
False Alarm Rate	0.54%

Table 4.8: Classification report of the M-Net model

Class	Precision	Recall	F1-score	False Alram Rate
algal_leaf	1.00	0.87	0.93	0.0000
gray_blight	0.97	0.99	0.98	0.0056
healthy	1.00	0.99	0.99	0.0000
helopeltis	0.99	1.00	1.00	0.0019
looper_infested	0.89	0.99	0.94	0.0229
red_spider	0.99	0.99	0.99	0.0019

Table 4.8 shows the performances of M-Net model's classification with different leaf conditions of plants. The model performs superbly with values of Precision and Recall close to 1.00 for most classes; for healthy and helopeltis, which are perfect as well. F1-scores are high too from 0.93 to 1.00 (for algal_leaf to helopeltis) and thus shows a good balance between Precision and Recall. The False Alarm Rate is low in all classes, highest of 0.0229 for looper_infested, which implies that the model rarely confuses negatives.

The performance of M-Net model confusion matrix for the detection of tea leaf disease is high throughout six classes. The model correctly classified 82 out of 94 algal_leaf, 3 of them as gray_blight, 8 of them as helopeltis and 1 of them as a red_spider. It scored 90 out of 91 correct gray_blight predictions but misclassified 1 correctly as helopeltis. For the healthy class 135 of 137 were correctly classified but 2 were erroneously labeled helopeltis. The model classified perfectly for helopeltis as it correctly classified all 102 instance. In the looper_infested category, there were 102 positives of the 103 that were classified correctly; 1 was classified as helopeltis. Likewise for red_spider the model correctly classified 98 out of 99 and only one was incorrectly classified as looper_infested.

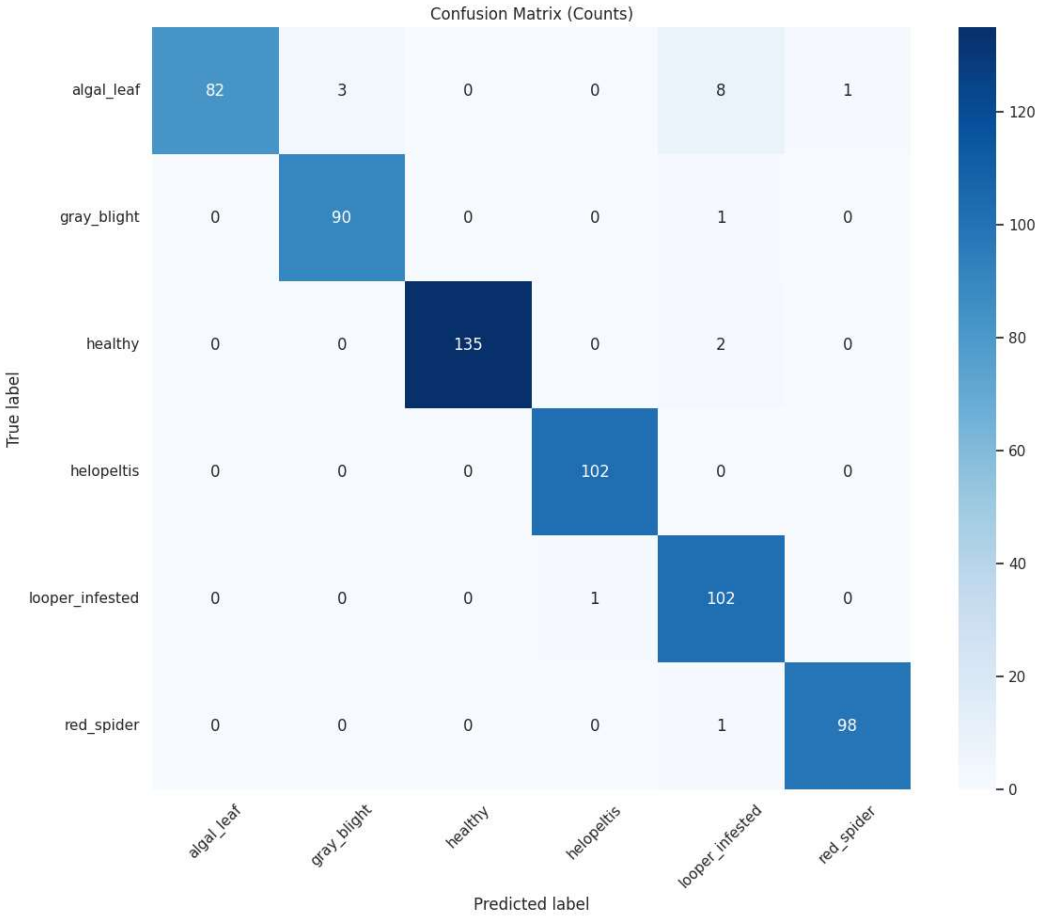


Figure 4.5: Confusion matrix of the M-Net model

4.3.4 Results of XAI

The LIME visualizations give a clear view on how the M-Net classifier evaluates different regions on tea leaf images in classifying them. Each of the visualizations points out regions that affect the model's decision positively (green) or negatively (red). In several examples, the green regions suggest that the model attends specific, texturally rich portions of the leaf, frequently around the mid veins or outstanding spots, indicating that these portions are influential to predictions. However, a pattern continues to emerge, in that there are large red areas on some leaves, indicating that the model is assigning negative weight to areas of the image that may contain some of the relevant disease characteristics. This inverse attention suggests that the model might be unaware of critical cues or can be captivated by background textures or leaf shapes. The regularity of green activations in selected patterns also displays the model's dependence on specific morphological characters. In the meantime, the prevalence of red areas in other cases indicates a lack of confidence or confusion in identifying subtle disease markers. Figure 4.6 presents the LIME-based interpretability for the M-Net model's prediction.

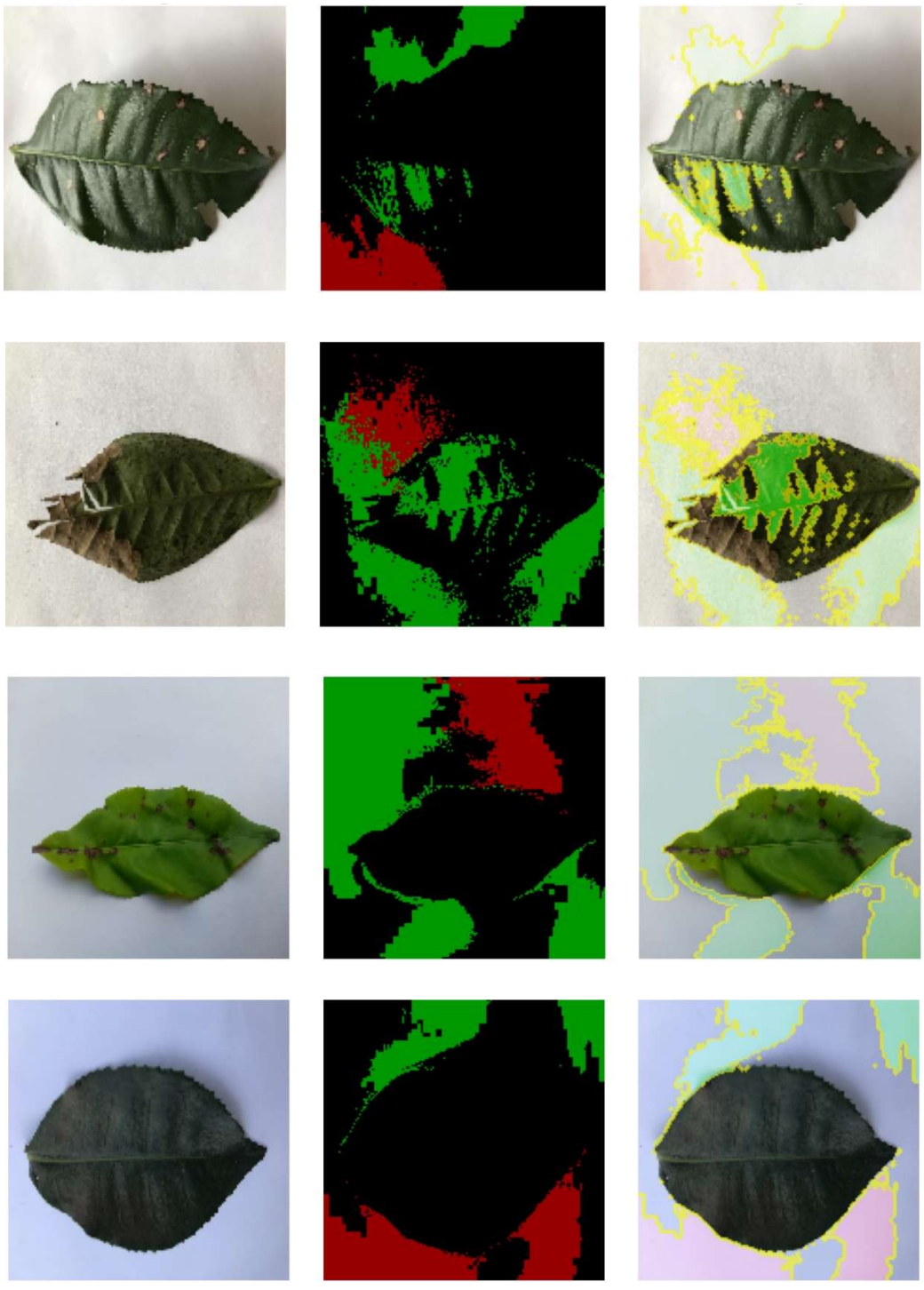


Figure 4.6: LIME-based interpretability for the M-Net model's prediction

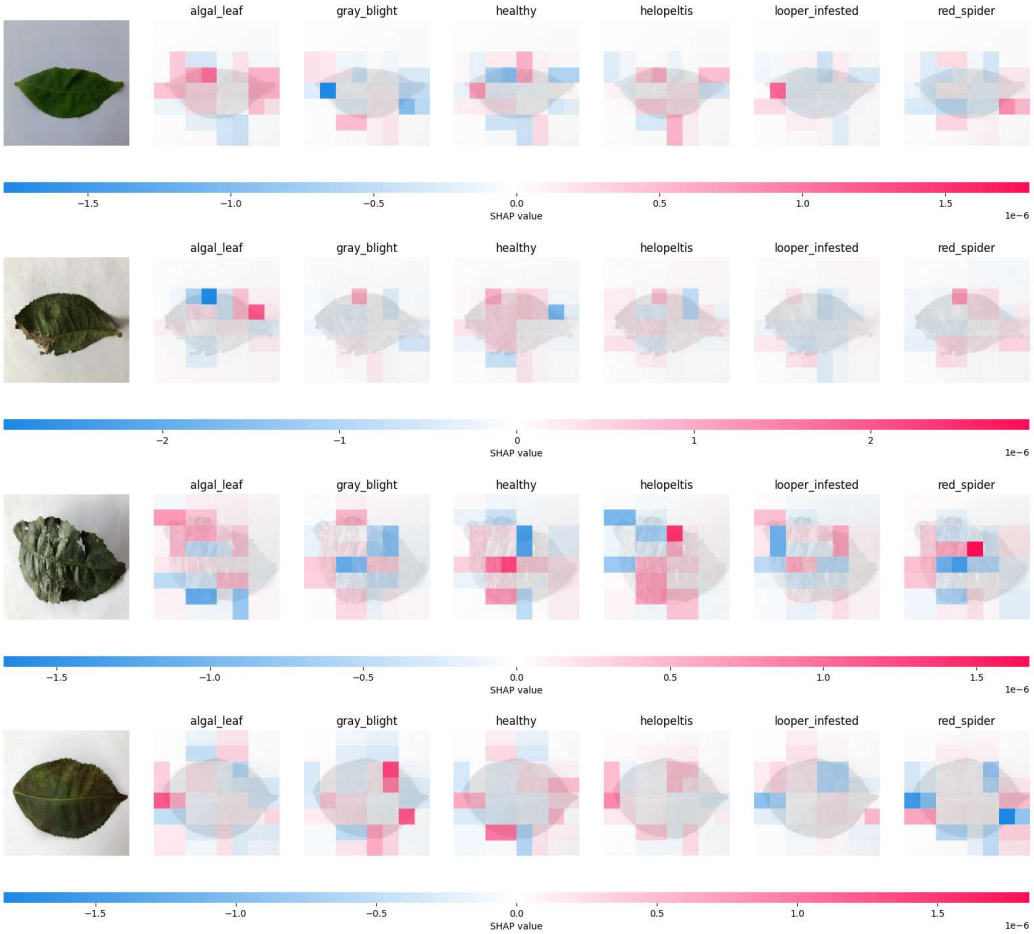


Figure 4.7: SHAP-based interpretability for the M-Net model's prediction

SHAP visualizations in figure 4.7 shows the degree of impact different parts of the tea leaf images have on the predictions for different disease categories as output by the M-Net model. Each row has an original leaf image followed by multiple SHAP heatmaps for each category; in these graphical displays, pink areas indicate a positive effect on the prediction for a given class while blue areas represent a negative effect. In all cases, SHAP maps demonstrate that the model specializes in targeting certain regions of the leaf by either boosting or reducing predictions for certain classes. For instance, in most cases, large pink areas tend to be in regions showing discoloration, changes in texture, or spotting, thus implying that these areas have an important effect on the model's accuracy in disease prediction. On the contrary, blue regions are generally found in more attractive or milder scenes of interest, implying that these parts of the image lower the possibility of certain class predictions. The presence of pink and blue areas for different classes in one image demonstrates a trade-off in the evidence being considered in course of the final classification process.

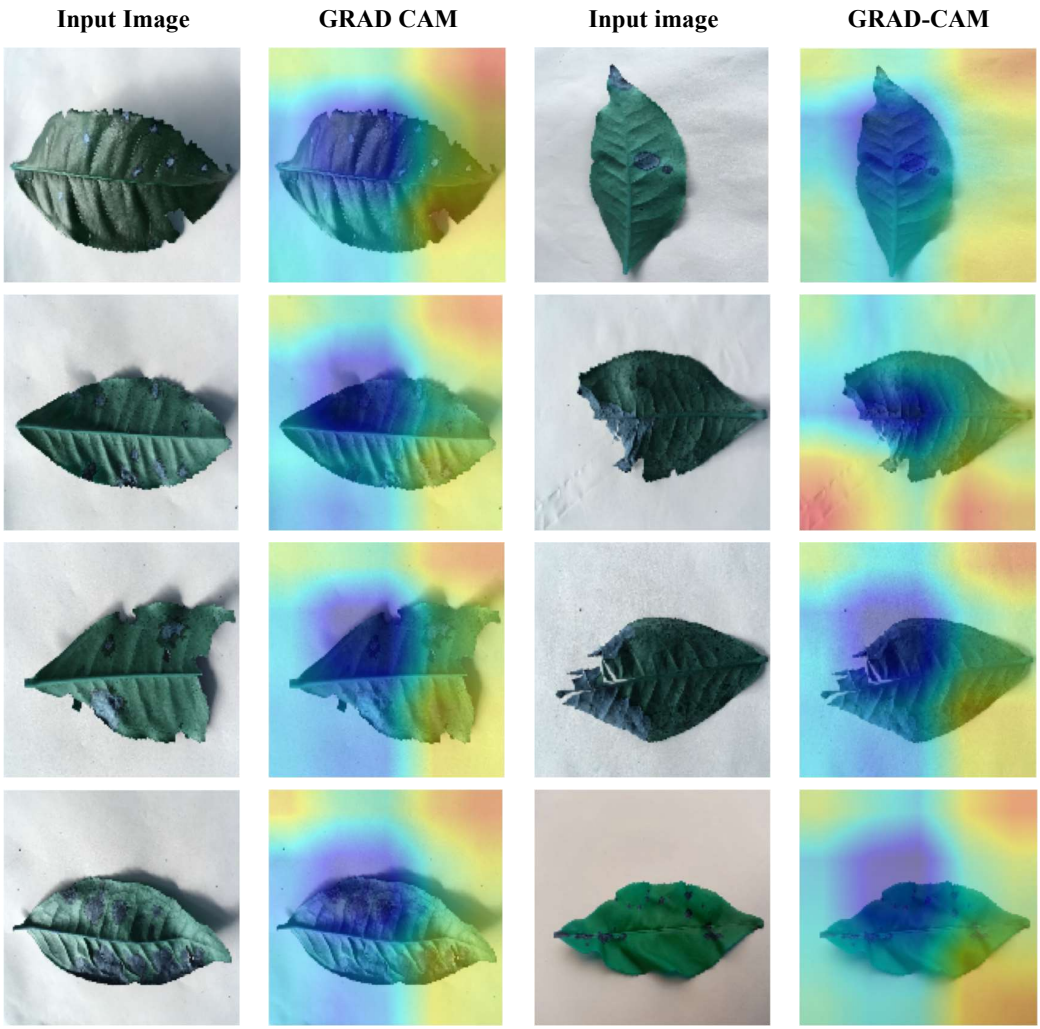


Figure 4.8: Grad-CAM interpretability for the M-Net model's prediction

4.3.5 Deployment Results: Web and Mobile Applications

This section describes the deployment of a tea leaf disease detection system on two different platforms: a Streamlit-based web application and a dedicated mobile app for Android devices. The Android app, named TeaVision, was designed for real-time detection of tea leaf diseases on the device of the user. Users can either take an image using the device camera or use an available image from the device gallery. Upon presentation of an image, it is processed through a light convolutional neural network model (M-Net) embedded in the app, returning an instantaneous diagnosis of the disease category and its confidence measure. As shown in Figure 4.8, the app successfully detects some of the conditions in tea leaves. As shown in Figure 4.8, the application accurately classifies various tea leaf conditions. During testing, the app demonstrated high prediction accuracy across various classes: algal_leaf (92%), gray_blight (96%), healthy (99%), helopeltis (99%), looper_infested (96%), and red_spider (97%). These results highlight the reliability and practicality of the deployed model in a mobile environment.

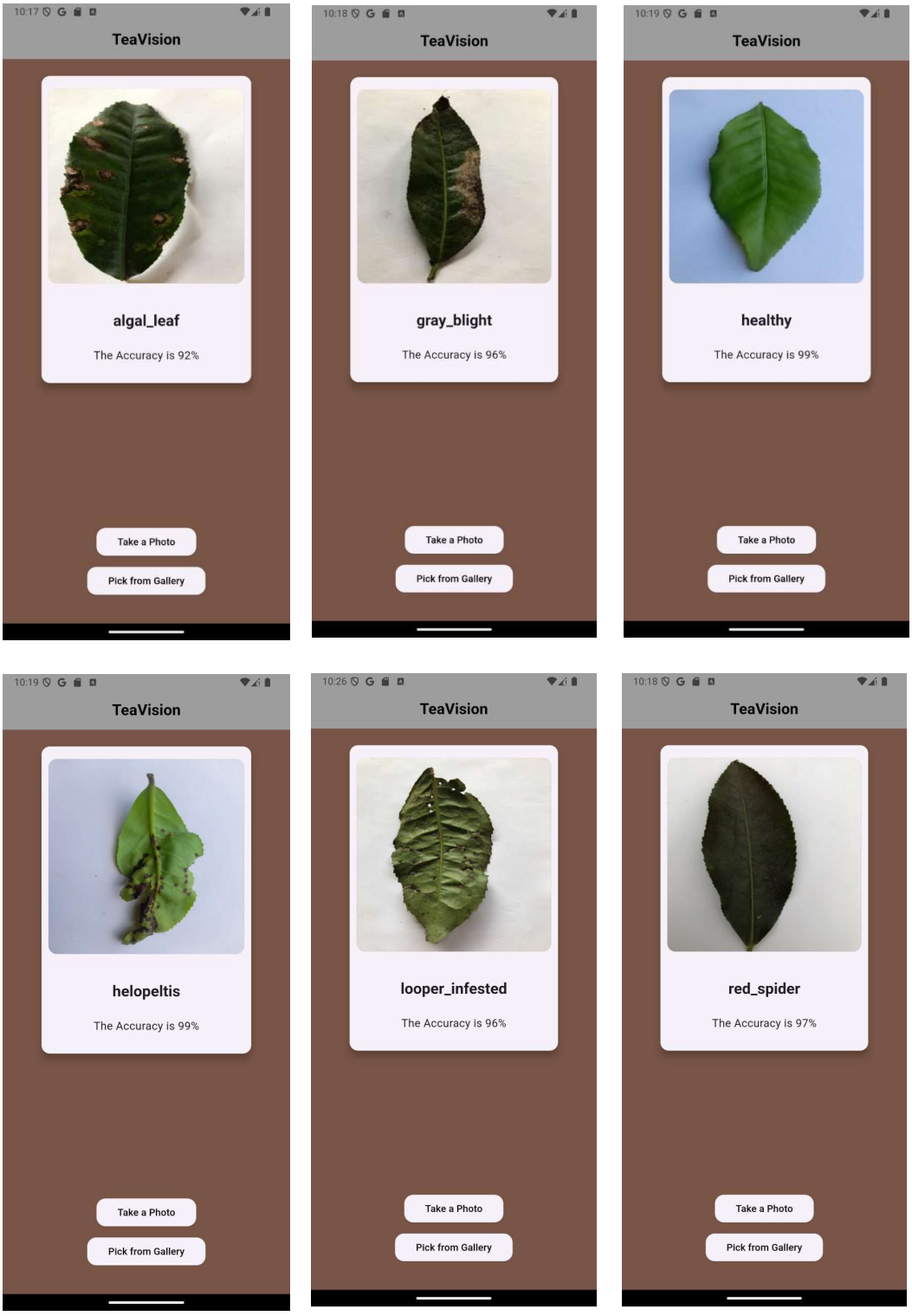


Figure 4.8: Mobile app interface showing tea leaf disease classification results

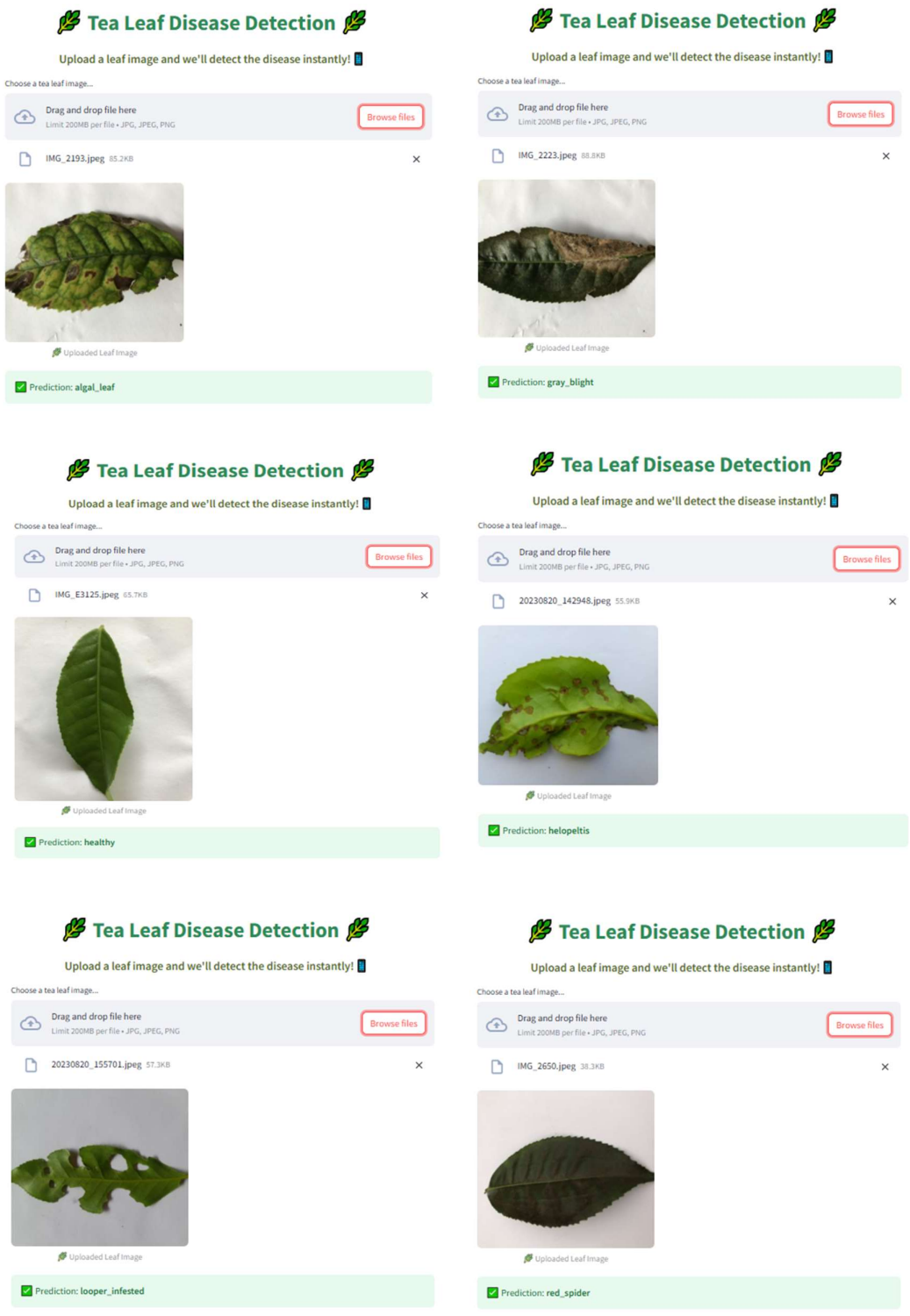


Figure 4.9: Web app interface showing tea leaf disease classification results

A Streamlit-based web application was also developed to showcase the capability of the proposed M-Net model in a browser-accessible environment. The web interface allows users to upload images of tea leaves, after which the model processes the input

and returns the predicted class along with the confidence score. The web app successfully detected all six classes, algal_leaf, gray_blight, healthy, helopeltis, looper_infested, and red_spider, with high accuracy and consistent predictions. This platform-independent deployment demonstrates the adaptability and responsiveness of the model, making it suitable for research demonstrations, expert evaluations, and remote disease monitoring.

4.4 Summary

This chapter has provided a comprehensive evaluation of the experimental outcomes of the proposed methodology for tea leaf disease detection. Initially, six fine-tuned CNN models were assessed, among which MobileNetV2 achieved the highest accuracy of 98.40%, followed by DenseNet121 with 97.44%, and InceptionV3 at 96.96%. This comparative analysis established a baseline for subsequent evaluation using transfer learning techniques. Further experimentation using fine-tuned transfer learning models demonstrated substantial improvements, with VGG16 achieving the highest accuracy of 99.68%, closely followed by MobileNetV2 with 99.52%. In addressing the need for an efficient and lean model for real-time processing in particular, an M-Net model was specially designed. With its streamlined structure that contains only 49,292 parameters (192.55 KB), M-Net proved its performance equivalent to other models with an accuracy rate of 97.29%, including precision and recall rates of 97.12%, and an F1-score of 97.00%, all while having an exceptionally low false alarm rate of 0.54%. These performance indicators prove the effectiveness of the model in providing high effectiveness while having minimal computational requirements, hence making it viable for use on mobile and edge platforms. To promote interpretability and explainability. XAI methods were incorporated into the M-Net model. The effectiveness and real-time capability of the M-Net model were additionally proved by implementing both a mobile app and Streamlit app. The two platforms adequately classified all six disease classes in demonstrating the reliability of the model in practical use. The mobile app had an intuitive interface showing confidence values for every prediction, while for its part, the Streamlit app offered similar functionalities in a web-based environment.

Chapter 5

Engineering Standards and Design Challenges

This chapter explains the engineering standards that were applied in developing the tea leaf disease detection system and discusses the broader impacts of the project on society, the environment, and sustainability. It also discloses ethical issues and compliance with relevant software, hardware, and communications standards.

5.1 Compliance with the Standards

This project applies globally accepted engineering ideas to link hardware, software, and communication dimensions to compatibility, robustness, data security, and practical viability. The points of focus were the use of available and open source platforms and technology, including use of smartphones for data collection as well as Microsoft PowerToys for image resizing, and Google Colab Pro for the validation of the model. This well informed decision ensured efficient system development, testing, and rollout of the explainable AI-driven tea leaf disease diagnostic system.

5.1.1 Software Standards

The software requirements of this project complied with standards established by ISO/IEC 25010:2011, which recognizes prominent software quality characteristics such as usability, maintainability, performance efficiency, and portability. The construction of the main model and experimentation were conducted on Google Colab Pro, a web-based Jupyter notebook platform with GPU/TPU acceleration and complying with typical web-based software runtime environments.

- Model Development: Python was utilized to code the machine learning codebase with PEP 8 style conventions for readability and ease of maintenance. TensorFlow, Keras, NumPy, Pandas, and OpenCV libraries were used for data preprocessing, augmentation, training, and testing. Google Colab Pro facilitated the use of high-end GPUs and long runtime sessions, which were critical in model tuning and iterative experimentation.

- Preprocessing Tools: Initial image resizing and formatting were carried out using Microsoft PowerToys Image Resizer, ensuring consistent dimensions across thousands of images while complying with standard Windows file system protocols. This preprocessing step allowed seamless integration with the training pipeline on Colab.
- Mobile Application Development: The mobile frontend was built using Android Studio, aligning with Google's Android development standards. These standards ensure consistent user experience, memory efficiency, and compatibility across diverse Android devices, especially important for field use by non-technical users such as farmers.

Several alternatives were considered during the development process. Java was evaluated for backend development due to its robust object-oriented structure and extensive ecosystem; however, its verbose syntax and slower pace of experimentation made it less suitable, especially for deep learning integration. Kotlin was considered for the mobile application because of its modern features and concise syntax ideal for Android development, but it posed a steeper learning curve and had limited community support for machine learning-based applications. An iOS application using Swift offered better device performance and a smoother user interface; nonetheless, it was deemed less accessible in rural areas of Bangladesh due to the higher cost of Apple devices and their relatively low market penetration.

Python was selected for backend model development due to its extensive support for machine learning libraries (TensorFlow, Keras, OpenCV), ease of debugging, and fast prototyping. Google Colab Pro was chosen for experimentation because it offered cost-effective access to GPUs and pre-configured environments. Android Studio was chosen for app development due to the dominant use of Android devices among farmers in the targeted regions, making the system both accessible and practical for real-world deployment.

5.1.2 Hardware Standards

The system was specifically designed to function on standard consumer-grade hardware, eliminating the need for expensive GPUs or specialized AI components. It complies with key hardware standards to ensure reliability and efficiency. For instance, IEEE 802.11 (Wi-Fi) enables optional internet connectivity, facilitating tasks such as model updates and data submission in connected environments. Additionally, adherence to IEEE 1725 supports responsible battery management in mobile devices, which is essential for prolonged use in field conditions. The system also ensures compatibility with ARM architecture, as the custom lightweight convolutional neural network model (M-Net) was optimized to run efficiently on Android smartphones powered by ARM Cortex-A processors.

For image acquisition, high-resolution images were captured using a smartphone

(iPhone 12), a practical and cost-effective choice. This method aligns with IEC 60065 for electronic safety and ISO 12233 for image quality, ensuring that data collected during the dataset development phase was both safe and of high quality. By relying on readily available smartphone technology, the system maintained consistent data quality while avoiding the need for expensive DSLR cameras, making the solution more accessible and scalable in resource-constrained environments.

Several hardware alternatives were considered for image capture and model deployment. A DSLR camera was initially evaluated due to its superior image quality and customizable settings; however, its high cost, bulkiness, and impracticality for frequent field use made it less suitable for this project. Raspberry Pi-based deployment offered an open-source, low-power, and customizable platform, but it posed challenges in terms of portability, setup complexity, and limited processing capabilities. The NVIDIA Jetson Nano was also considered for its edge machine learning potential and GPU acceleration, yet its higher cost and maintenance difficulty for non-technical users made it less feasible, particularly in rural or low-resource settings.

Smartphones were chosen for both image acquisition and app deployment due to their affordability, portability, and ubiquity in rural communities. The M-Net model was optimized specifically to run on devices with limited memory and processing power, such as low to mid-range Android smartphones, eliminating the need for specialized AI hardware. This ensured real-world practicality and cost-effectiveness for end users.

5.1.3 Communication Standards

The system supports a hybrid communication model, functioning both offline and online to ensure flexibility in areas with unreliable internet access. When connectivity is available, it uses HTTP/HTTPS protocols for secure and standardized web communication. A RESTful API architecture facilitates structured and scalable interactions between the client and server, enabling efficient data exchange. To protect user privacy and maintain data integrity, TLS/SSL encryption is employed during the transmission of sensitive information such as feedback or image uploads to cloud services.

Several communication alternatives were evaluated for the system. The MQTT protocol was considered due to its lightweight nature, fast performance, and low bandwidth usage, making it ideal for IoT scenarios; however, it was less suitable for transmitting large payloads like images and required a more complex, custom setup. FTP (File Transfer Protocol) offered a simple and easy-to-implement solution, but it lacked encryption and was deemed insecure by modern standards. WebSockets provided real-time bidirectional communication, but its added complexity was unnecessary for this use case, particularly given the system's need to operate

effectively in environments with intermittent internet connectivity.

HTTP/HTTPS was selected due to its wide compatibility with mobile and web services, ease of integration with REST APIs, and built-in encryption via TLS. It offers a secure and efficient way to enable optional cloud features while maintaining offline usability. RESTful architecture ensures modularity and simplifies future expansion, such as adding cloud-based analytics or farmer feedback loops.

5.2 Impact on Society, Environment and Sustainability

This section surveys how the proposed tea leaf disease detection system fits with the larger societal goals and ethical responsibilities, while taking ongoing sustainability further in mind. The growth of an explainable, mobile-based AI solution is an exciting prospect for both farmers and agricultural actors, but it is also appealing about environmental preservation and editorially responsible technological advancement. The system was developed not only for its core technical abilities, but also as inclusive, available and effective beyond its technical scope.

5.2.1 Impact on Life

The system has the potential to significantly enhance the daily lives of small and medium-scale tea farmers, particularly in remote or underserved areas of Bangladesh. Early and accurate identification of diseases like Red Rust or Grey Blight allows for timely intervention, preventing disease progression and minimizing yield loss. This leads directly to increased productivity, better-quality harvests, and improved livelihoods. The inclusion of explainable AI (XAI) techniques such as LIME, SHAP, and Grad-CAM ensures that predictions are not only accurate but also transparent and understandable. Farmers and agricultural extension officers can visualize which parts of the leaf were influential in the model's decision-making process. This not only builds trust in AI but also facilitates learning, enabling farmers to better understand disease symptoms and patterns. Furthermore, the mobile-friendly design ensures that even those with limited technical expertise can operate the system intuitively. By reducing dependency on experts or lab-based diagnoses, the system empowers users at the grassroots level and enhances agricultural self-reliance.

5.2.2 Impact on Society & Environment

Societally, the project contributes to the digitalization of agriculture and promotes inclusive technological adoption. By providing a scalable and affordable AI solution tailored to the local context, it fosters greater equity in access to digital tools. This democratization of technology helps bridge the urban-rural divide and supports community resilience. Environmentally, the system encourages precision agriculture. Rather than blanket application of fungicides or pesticides, farmers can apply treatments only where necessary based on specific disease detection. This reduces chemical runoff, conserves soil health, and protects nearby water sources

and ecosystems from contamination. Additionally, the system's ability to operate offline ensures that energy-intensive cloud computation is not continuously required, which contributes to reduced carbon emissions. By deploying the lightweight M-Net model directly on smartphones, the need for data transfer and server usage is minimized, lowering the system's overall environmental footprint. In alignment with the United Nations Sustainable Development Goals (SDGs), particularly Goals 2 (Zero Hunger), 9 (Industry, Innovation, and Infrastructure), and 12 (Responsible Consumption and Production), this system promotes sustainable agricultural practices and digital empowerment.

5.2.3 Ethical Aspects

Ethical responsibility was a core design principle throughout the project. All data collection was performed transparently using smartphone cameras in real-world tea gardens, with prior permission and informed consent from farm owners. No personal or sensitive data was collected, and all images were anonymized. The model was rigorously tested for fairness and bias mitigation, especially in relation to lighting conditions, background clutter, and leaf orientations. This helps prevent misdiagnosis and ensures that farmers can trust the results regardless of how the image is captured. Transparency is further upheld through the use of XAI methods. Rather than functioning as an opaque "black box", the model highlights what it sees and why, giving users visibility into the AI's reasoning process. This strengthens ethical alignment with principles of autonomy, accountability, and non-maleficence, ensuring that users can understand and challenge model outputs when necessary.
Moreover, no user data is stored or transmitted without consent, and optional cloud connectivity is protected via secure HTTPS protocols. This ensures adherence to data privacy and digital rights.

5.2.4 Sustainability Plan

To ensure that the system remains effective, relevant, and usable over time, a multi-dimensional sustainability plan has been adopted:

- Technical Sustainability: The M-Net model is designed with minimal parameters to run efficiently on low-spec Android phones. The use of Python, TensorFlow/Keras, and standard ML libraries ensures ease of maintenance and future extensibility. The modular structure allows component upgrades such as adding new disease classes or refining the model without requiring complete system redesign.
- Economic Sustainability: The solution uses widely available resources, smartphones and open-source software to keep implementation costs low. This makes it viable for deployment at scale in resource-constrained regions. The offline functionality further reduces recurring costs associated with data usage or server maintenance.
- Environmental Sustainability: The system supports localized, low-energy processing and encourages responsible pesticide use by enabling accurate

disease identification. This contributes to environmentally sound farming practices and minimizes the ecological footprint of agriculture.

- Community and Knowledge Sustainability: The integration of XAI contributes to farmer education, making the app not just a tool but also a learning aid. Extension workers and educators can use the model outputs to teach best practices in disease management. The intention to release future versions as open-source fosters community-driven improvement and long-term viability.

5.3 Project Management and Financial Analysis

This section presents the financial planning, budget estimation, resource management, and potential revenue model of the developed system. Although the research project was carried out independently without commercial funding, realistic cost modeling is crucial for future deployment, scalability, and market readiness. A dual-budget scenario is analyzed based on open-source/self-supported development (as conducted in this project) versus a commercial/enterprise-level implementation to reflect the possible pathways for real-world adaptation.

5.3.1 Project Planning and Task Management

Effective project management was critical to the successful execution of this research. Given the individual nature of the study, tasks were systematically planned, scheduled, and monitored to ensure timely completion while maintaining quality and precision. The project was divided into logical, goal-oriented phases, enabling smooth progress and flexible adjustments where needed. The workflow was designed to follow a milestone-based structure, beginning with documentation and background research, moving through data collection, experimentation, model development, explainability integration, and ultimately leading to deployment. Each phase was interconnected, ensuring that outputs from earlier stages informed decisions in the subsequent ones. Agile principles were adopted to accommodate changes and refinements based on performance feedback and results. All experiments and model training were carried out using Google Colab Pro, which provided GPU acceleration and seamless integration with Python-based deep learning frameworks such as TensorFlow and Keras. Planning tools like Google Calendar and Trello were utilized to track tasks, while GitHub ensured version control for code development and reproducibility.

Table 5.1: Project Timeline

Phase	Start Date	End Date	Duration
Initial Documentation (Intro, Literature Review, Gap Analysis)	Nov 7, 2024	Dec 7, 2024	4 weeks
Data Collection & Preprocessing	Dec 8, 2024	Jan 5, 2025	4 weeks
Methodology Design	Jan 6, 2025	Jan 30, 2025	3 weeks
Experimentation with SOTA CNN Models	Feb 1, 2025	Feb 14, 2025	2 weeks
Experimentation with Transfer Learning Models	Feb 15, 2025	Feb 28, 2025	2 weeks
Development of M-Net Model	Mar 1, 2025	Mar 20, 2025	3 weeks
XAI Integration & Model Evaluation	Mar 21, 2025	Apr 10, 2025	3 weeks
Mobile Application Deployment	Apr 11, 2025	Apr 20, 2025	1.5 weeks
Final Documentation & Printing	Apr 21, 2025	Apr 28, 2025	1 weeks

The Mobile Application Deployment phase was crucial for delivering the final output in a practical and accessible form. During this phase, the trained and validated lightweight M-Net model was integrated into a user-friendly Android application. The app was initially prototyped using Streamlit for web-based demonstrations and later ported to Android Studio for native mobile performance. Special attention was given to optimizing inference time, user interface design, offline functionality, and compatibility with low-end smartphones commonly used in tea-growing regions. Testing was performed on multiple devices to validate usability and performance in real-world conditions.

5.3.2 Financial Analysis

The financial analysis is used to determine both the actual and hypothetical costs involved in developing and deploying the proposed tea leaf detection system. Since this research was carried out on an independent basis and out of an academic intent, a cost effective strategy was used for implementing personal resource, free open source software and buy low cost cloud hosting services such as Google Colab Pro. Such choices greatly simplified the financial aspect yet significantly maintained the quality experimentation and deployment. However, while looking at the financials of large scale implementation or commercialization, an alternate budget scenario was also illustrated.

Table 5.2: Actual Research Budget (Self-Supported, Research-Based)

Component	Estimated Cost (BDT)	Remarks
Smartphone Camera for Image Capture	0 (Personal Device)	iPhone 12 used for dataset collection
Internet and Cloud Resources	2,000/month × 4 = 8,000	Google Colab Pro for model training and testing
Software Tools	0	Python, Keras, OpenCV, and other libraries are open source
Image Preprocessing Tool	0	Microsoft PowerToys (free image resizer)
Local Travel for Field Visits	4,000	Visit to Sreemangal tea gardens
Thesis Printing and Binding	2,000	For hardcopy submission
Miscellaneous (USB, Cables, etc.)	1,000	Small accessories
Total Cost	15,000 BDT	

Table 5.3: Alternate Budget (Enterprise-Scale Deployment)

Component	Estimated Cost (BDT)	Remarks
Commercial Smartphone (for app testing)	25,000	Mid-range Android device for target testing
Dedicated DSLR/Smartphone for Data Capture	30,000	Higher quality image dataset with consistent lighting
Data Collection & Labeling (Labor Cost)	20,000	Payment for field workers/agricultural experts
Cloud Compute Credits (GCP or AWS)	15,000	Model training, fine-tuning on high-performance GPUs
Android Developer License	2,000	For app publishing on Google Play Store
App UI/UX Development (Freelance/UI Expert)	10,000	Professional UI design and mobile optimization
Marketing & Awareness Campaigns	15,000	Promotion among farmers and tea estates
Maintenance, Updates & Customer Support	10,000/year	Optional support and versioning
Total Estimated Budget	127,000 BDT	

5.4 Complex Engineering Problem

This section brings out the complication of the engineering problem discussed in this thesis. Attention to such a design challenge as building an accurate, interpretable and mobile-compatible tea leaf disease detection system precluded a cross-disciplinary combination of areas of knowledge and addressed practical design constraints and restraints to generate realistic applicability. This work involved computer vision, deep learning, transfer learning, XAI, mobile deployment, and understanding the agricultural domain. Further discussions on resource constraints, concerns on system expandability, and user access were critical towards influencing design and implementation. The following subsections explain how the study fits complex engineering problem-solving as well as knowledge profiles and engineering activities.

5.4.1 Complex Problem Solving

To successfully undertake this research task, different aspects pertaining to complex engineering problem-solving were employed, including algorithm selection, model integration, optimization for mobile platforms, and system deployment in real time. The following mapping is an indication of an alignment of this project against the Engineering Problem (EP) framework.:

Table 5.4: Mapping with complex problem solving.

EP1 Dept of Knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stake-holder Involvement	EP7 Interdependence
✓	✓	✓	✓			✓

Justifications:

- EP1 – Depth of Knowledge: The project demanded in-depth knowledge across multiple areas, such as convolutional neural networks (CNNs), transfer learning, data augmentation, TensorFlow/Keras implementation, and interpretability methods (Grad-CAM, LIME, SHAP). Additionally, knowledge in web/mobile app development using Streamlit was essential for final deployment.
- EP2 – Conflicting Requirements: This study addressed multiple conflicting requirements throughout the system development lifecycle. On one hand, it aimed to achieve high classification accuracy through advanced deep learning

techniques; on the other hand, it had to maintain model simplicity and reduce computational overhead to ensure compatibility with mobile devices.

- EP3 – Depth of Analysis: A comprehensive experimental pipeline was implemented to compare six CNN models and six transfer learning models. Evaluation metrics such as accuracy, precision, recall, and confusion matrix were used to deeply analyze model performance. The best-performing model was further optimized into a lightweight CNN.
- EP4 – Familiarity of Issues: Technical challenges like class imbalance, data noise, limited training samples, and the need for explainable AI were anticipated due to prior coursework, academic experience, and domain research. Familiarity helped in designing effective preprocessing and augmentation strategies.
- EP7 – Interdependence: The architecture was designed in a modular manner, allowing integration of different CNN backbones, XAI modules, and deployment frameworks. The lightweight CNN (M-Net) works as a core model and can be easily updated or adapted for similar crop disease detection tasks.

Mapping with Knowledge Profile for EP1

Table 5.5: Mapping with knowledge Profile.

K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K8 Research Literature
✓		✓	✓	✓

Justifications:

- K3 – Engineering Fundamentals: This research applied machine learning principles such as classification, optimization, data normalization, and evaluation metrics. It also utilized some foundational image processing techniques such as resizing, color normalization, and augmentation.
- K5 – Engineering Design: The system design involved selecting suitable CNNs, building a lightweight custom model (M-Net), integrating XAI tools, and deploying the model via Streamlit as a functional web app for field testing. The trade-offs between model size, performance, and interpretability were addressed systematically.
- K6 – Engineering Practice: This study followed standard engineering

procedures from data collection to mobile deployment. Each step, data preprocessing, model evaluation, XAI integration, and app development was carried out methodically. Evaluation metrics and model comparison ensured reliable results, aligning with real-world engineering rigor and best practices.

- K8 – Research Literature: A thorough review of existing studies on tea and plant disease detection informed model selection, dataset preparation, and the need for explainability. Gaps in literature such as lack of lightweight models and XAI integration were identified and addressed in this thesis.

5.4.2 Engineering Activities

The project lifecycle encompassed multiple complex engineering activities, including dataset collection, model training, performance evaluation, visualization through XAI, and building a deployable application. These activities reflect the real-world complexity of delivering a working agriculture AI system under practical constraints.

Table 5.6: Mapping with complex engineering activities.

EA1 Range of re-sources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
✓		✓	✓	✓

Justifications:

- EA1 – Range of Resources: A variety of tools and platforms were used: smartphones (for dataset collection), Python (for preprocessing), Keras/TensorFlow (for model development), Grad-CAM/LIME/SHAP (for explainability), and Streamlit (for web deployment). GPU-based training was performed using Google Colab.
- EA3 – Innovation: This study proposes a novel pipeline: from field data collection to a lightweight CNN model with integrated XAI explanations, deployed as a web tool. While many existing works focus on high-performance models, this work emphasizes usability and interpretability under practical constraints.
- EA4 – Societal & Environmental Consequences: The system contributes to the agricultural sector by aiding early detection of tea leaf diseases. This

helps farmers take preventive action, reducing chemical usage and increasing yield—positively impacting both society and the environment.

- EA5 – Familiarity: The techniques and tools used, such as CNNs, Grad-CAM, data augmentation, and Python-based development, were familiar due to prior academic training and projects, enabling rapid prototyping and experimentation.

5.1 Summary

This chapter gives a recapitulation of the major engineering principles and more complex design challenges in designing the proposed disease detection for tea leaves. The project needed a multi-dimensional approach that brought together computer vision, deep learning, XAI with mobile-compatible deployment. From dataset collection in real tea gardens to model deployment, each stage was aligned with real-world agricultural applications. A lightweight CNN was custom designed to maintain high accuracy with fewer parameters, optimizing performance for mobile use. The project addressed societal needs by enabling early disease detection in low-resource environments and ensured ethical and environmental sustainability through on-device processing, privacy preservation, and energy efficiency. All complex engineering activities were mapped using recognized frameworks, showing the project's alignment with engineering problem-solving attributes (e.g., depth of knowledge, conflicting requirements, interdependence) and engineering activities (e.g., innovation, societal impact). Engineering knowledge, design thinking, and research insights were applied throughout the development lifecycle.

Chapter 6

Conclusion

This chapter summarizes the research outcomes, outlines the limitations encountered during the study, and highlights potential directions for future enhancements. The findings from the extensive experimentation, explainability integration, and real-time deployment are consolidated to demonstrate the practical effectiveness of the proposed lightweight CNN model.

6.1 Summary

This study focused on designing an efficient and interpretable deep learning-based system for the automatic detection of six types of tea leaf conditions, including both diseased and healthy classes. The core contribution was the development of a lightweight, yet high-performing convolutional neural network named M-Net, specifically optimized for deployment in resource-limited environments like smartphones and web applications. Initially, a set of benchmark experiments was conducted using widely recognized fine-tuned CNN architectures, including DenseNet121, VGG16, InceptionV3, MobileNetV2, and Xception. These were also evaluated under a transfer learning framework to assess improvements in generalization. Among them, models like VGG16 and MobileNetV2 in the transfer learning setting yielded very high classification accuracies, achieving 99.68% and 99.52% respectively. However, these models typically involve larger parameter counts and require more memory and computational power, which may not be suitable for deployment on edge devices.

To overcome this limitation, the proposed M-Net model was designed from the ground up to be lightweight while still delivering competitive results. It achieved an overall accuracy of 97.29%, with precision and recall of 97.12%, and an F1-score of 97.00%, indicating consistent and balanced performance across classes. Also, it maintained a very low false alarm rate of 0.54 % to minimize unnecessary alerts in practical cases. Crucially, the total parameter count of this model was only 49,292, which took just 192.55 KB of storage, supporting its appropriateness for low resource environments. In order to create interpretability and trust, the study has included explainable artificial intelligence (XAI) methods such as LIME and SHAP which gave visual as well as feature based explanations of the model's prediction. This facilitated demystifying the process of decision making and the need for

transparency was supported which is of paramount importance in sensitive agricultural applications where actionable decisions rely on accurate and justifiable outputs. In addition, the M-Net model was usefully deployed in real time environments via both Streamlit-based web application and an Android-based mobile application. Both platforms correctly identified all six tea leaf categories in actual world test images at a high degree of confidence, thus confirming the robustness and utility of the model beyond a test environment. This deployment is a milestone toward practical agricultural intelligence tools for empowering farmers and agronomists in disease detection at the early stage and better crop management.

6.2 Limitation

Although the results from the proposed system were promising, this study has had several limitations, which should be noted and improved in future studies. First, the dataset applied was region-specific and sourced only from tea gardens in Sreemangal, Moulvibazar, Sylhet. Even though the dataset contained variability in lighting, the angles and backgrounds, its geographic as well as climatic homogeneity may not allow the model to generalize over different environments, including communities at high altitude tea plantations or areas with varying cultivation practices. Also, the amount of the dataset is large enough for validation of experiments, but on a relatively small scale. An increased and more diversified dataset would reduce the chances of biases surfacing and enhance robustness of the model.

Secondly, while the proposed M-Net model was optimized to be lightweight and efficient for real-time deployment, its simplified architecture may lead to slight underperformance in complex edge cases, such as severely occluded leaves, overlapping disease symptoms, or background clutter. In such cases, deeper and more complex models might offer marginally better classification accuracy at the expense of computational cost. Thirdly, although efforts were made to include a variety of environmental conditions in the training data (e.g., different times of day and lighting conditions), real-world variability in background, leaf orientation, motion blur, and partial occlusions was not comprehensively addressed. These factors could potentially affect model predictions in uncontrolled field settings. Finally, the deployed web and mobile applications currently lack integration with XAI (Explainable AI) visualizations due to platform and resource constraints. While LIME and SHAP were successfully used during model development for interpretability, their absence in the deployment phase limits the user's ability to visually understand or validate the model's decisions. This reduces transparency and may impact trust among end-users such as farmers and agricultural consultants.

6.3 Future Work

Building upon the outcomes of this study, several avenues can be pursued to enhance the reliability, scalability, and interpretability of the proposed tea leaf disease detection system. A primary direction for future research involves expanding and diversifying the dataset. Although the current dataset was carefully collected and preprocessed, it remains geographically constrained to Sreemangal, Sylhet. Collecting images from tea plantations across different regions, seasons, and climatic conditions would increase the model's generalizability and reduce regional bias. Additionally, increasing the overall dataset size with more images per class and including rare or early-stage disease samples would further boost classification performance, especially in real-world scenarios. Another critical advancement would be the integration of explainable AI (XAI) directly into the deployed applications. While LIME and SHAP were used effectively during model evaluation to interpret predictions, the deployed Streamlit web application and Android mobile app currently lack built-in visual explanations due to platform constraints. Embedding lightweight, mobile-compatible XAI techniques such as Grad-CAM or simplified LIME variants could significantly enhance user trust, especially for non-technical users like farmers and agronomists who would benefit from understanding *why* a specific disease is predicted.

Further optimization of the M-Net architecture is also a promising direction. Although M-Net is already lightweight, techniques such as model pruning, quantization, and knowledge distillation could be applied to make it even more efficient in terms of speed, memory usage, and energy consumption. These improvements are particularly relevant for deploying the model on low-power edge devices and in rural areas with limited computational resources. Moreover, expanding the scope of the system to support multimodal learning can lead to more comprehensive health assessments. Combining visual data with environmental sensor inputs (e.g., temperature, humidity, soil moisture) or chemical analysis could lead to more robust and context-aware predictions. Such an approach would not only enhance classification accuracy but also help detect environmental stress factors contributing to disease. Finally, future versions of the system could move beyond binary or multiclass classification to include disease severity estimation. Estimating how severe a detected disease is, or tracking the progression of symptoms over time through temporal image analysis, would enable more proactive and informed crop management. Coupling this with real-time feedback from users in the field would create a continuous improvement loop, ensuring the tool evolves in line with practical needs and user experiences.

References

[1] Hajiboland, R. (2017). Environmental and nutritional requirements for tea cultivation. Folia horticulturae, 29(2), 199-220.

[2] Mamun, M. S. A. (2019). Tea production in Bangladesh: From bush to mug. Agronomic Crops: Volume 1: Production Technologies, 441-505.

[3] Nasir, T., & Shamsuddoha, M. (2011). Tea productions, consumptions and exports: Bangladesh perspective. International Journal of Educational Research and Technology, 2(1), 68-73.

[4] ITC 2015, Annual Bulletin of Statistics (September 2015), International Tea Committee (ITC), London, UK, p. 158.

[5] BTB 2021, Monthly Bulletin of Statistics on tea. January 2021. Bangladesh Tea Board, Nasirabad, Chittagong.

[6] Bao, W., Fan, T., Hu, G., Liang, D., & Li, H. (2022). Detection and identification of tea leaf diseases based on AX-RetinaNet. Scientific reports, 12(1), 2183.

[7] Hu, G., Yang, X., Zhang, Y., & Wan, M. (2019). Identification of tea leaf diseases by using an improved deep convolutional neural network. Sustainable Computing: Informatics and Systems, 24, 100353.

[8] Kalaydjian, C. T. (2023). An application of vision transformer (VIT) for image-based plant disease classification. University of California, Los Angeles.

[9] Fu, X., Ma, Q., Yang, F., Zhang, C., Zhao, X., Chang, F., & Han, L. (2024). Crop pest image recognition based on the improved ViT method. Information Processing in Agriculture, 11(2), 249-259.

[10] Zhang, J., Guo, H., Guo, J., & Zhang, J. (2023). An information entropy masked vision transformer (iem-vit) model for recognition of tea diseases. Agronomy, 13(4), 1156.

[11] Ahmed, F., Emon, Y. R., Ahad, M. T., Munna, M. H., & Mamun, S. B. (2023, November). A Fuzzy-Based Vision Transformer Model for Tea Leaf Disease Detection. In International Conference on Trends in Computational and Cognitive Engineering (pp. 229-242). Singapore: Springer Nature Singapore.

[12] Wang, Y., Xu, R., Bai, D., & Lin, H. (2023). Integrated learning-based pest and disease detection method for tea leaves. Forests, 14(5), 1012.

[13] Liang, J., Liang, R., & Wang, D. (2025). A novel lightweight model for tea disease classification based on feature reuse and channel focus attention mechanism. Engineering Science and Technology, an International Journal, 61, 101940.

[14] Chen, J., Chen, J., Zhang, D., Sun, Y., & Nanehkaran, Y. A. (2020). Using deep transfer learning for image-based plant disease identification. Computers and Electronics in Agriculture, 173, 105393.

[15] Xue, Z., Xu, R., Bai, D., & Lin, H. (2023). YOLO-tea: A tea disease detection model improved by YOLOv5. Forests, 14(2), 415.

[16] Panchbhai, K. G., & Lanjewar, M. G. (2024). Enhancement of tea leaf diseases identification using modified SOTA models. Neural Computing and Applications, 1-19.

[17] Lanjewar, M. G., & Panchbhai, K. G. (2023). Convolutional neural network based tea leaf disease prediction system on smart phone using paas cloud. Neural Computing and Applications, 35(3), 2755-2771.

[18] Dhaka, V. S., Meena, S. V., Rani, G., Sinwar, D., Ijaz, M. F., & Woźniak, M. (2021). A survey of deep convolutional neural networks applied for prediction of plant leaf diseases. Sensors, 21(14), 4749.

[19] Hossain, S., Tanzim Reza, M., Chakrabarty, A., & Jung, Y. J. (2023). Aggregating different scales of attention on feature variants for tomato leaf disease diagnosis from

image data: a transformer driven study. Sensors, 23(7), 3751.
[20]Fan, C., Sun, Y., Zhao, Y., Song, M., & Wang, J. (2019). Deep learning-based feature engineering methods for improved building energy prediction. Applied energy, 240, 35-45.
[21]He, Y., Luo, Z., Lin, L., & Li, S. (2023, November). A Two-Stage Transfer Learning Method for Tea Disease Identification. In 2023 13th International Conference on Information Technology in Medicine and Education (ITME) (pp. 800-804). IEEE.
[22]Uskaner Hepsağ, P. (2024). Efficient plant disease identification using few-shot learning: a transfer learning approach. Multimedia Tools and Applications, 83(20), 58293-58308.
[23]Joshi, A., Pradhan, B., Chakraborty, S., Varatharajoo, R., Gite, S., & Alamri, A. (2024). Deep-Transfer-Learning Strategies for Crop Yield Prediction Using Climate Records and Satellite Image Time-Series Data. Remote Sensing, 16(24), 4804.
[24]Hossen, M. I., Awrangjeb, M., Pan, S., & Mamun, A. A. (2025). Transfer learning in agriculture: a review. Artificial Intelligence Review, 58(4), 97.
[25]Albanese, A., Nardello, M., & Brunelli, D. (2022). Low-power deep learning edge computing platform for resource constrained lightweight compact UAVs. Sustainable Computing: Informatics and Systems, 34, 100725.
[26]Rahman, H., Ahmad, I., Jon, P. H., Salam, A., & Rabbi, M. F. (2024). Automated detection of selected tea leaf diseases in Bangladesh with convolutional neural network. Scientific Reports, 14(1), 14097.
[27]Ding, Z., Yang, C., Hu, B., Guo, M., Li, J., Wang, M., ... & Dong, C. (2024). Lightweight CNN combined with knowledge distillation for the accurate determination of black tea fermentation degree. Food Research International, 194, 114929.
[28]Nigar, N., Faisal, H. M., Umer, M., Oki, O., & Lukose, J. (2024). Improving plant disease classification with deep learning based prediction model using explainable artificial intelligence. IEEE Access.
[29]Assaduzzaman, M., Bishshash, P., Nirob, M. A. S., Al Marouf, A., Rokne, J. G., & Alhajj, R. (2025). XSE-TomatoNet: An explainable AI based tomato leaf disease classification method using EfficientNetB0 with squeeze-and-excitation blocks and multi-scale feature fusion. MethodsX, 14, 103159.
[30]Sivaraman, R., Praveena, S., & Naresh Kumar, H. (2025). Sustainable Agriculture Through Advanced Crop Management: VGG16-Based Tea Leaf Disease Recognition. Generative Artificial Intelligence for Biomedical and Smart Health Informatics, 121-133.
[31]Fu, Y., Guo, L., & Huang, F. (2024). A lightweight CNN model for pepper leaf disease recognition in a human palm background. Heliyon, 10(12).
[32]Islam, Z. (2018). A regional pattern of tea garden and tea production in Bangladesh (Doctoral dissertation, University of Dhaka).
[33]Amin, S. B. (2021). Economy of tourism in Bangladesh. Springer International Publishing.
[34]Koiry, S., Kairi, B., & Pooja, P. (2024). Impact of income diversification on multidimensional poverty: Household level evidence from tea estates in Bangladesh. Heliyon, 10(5).
[35]Pandey, A. K., Sinniah, G. D., Babu, A., & Tanti, A. (2021). How the global tea industry copes with fungal diseases–challenges and opportunities. Plant Disease, 105(7), 1868-1879.
[36]Bhuyan, P., Singh, P. K., & Das, S. K. (2024). Res4net-CBAM: A deep cnn with convolution block attention module for tea leaf disease diagnosis. Multimedia Tools and Applications, 83(16), 48925-48947.
[37]Datta, S., & Gupta, N. (2023). A novel approach for the detection of tea leaf disease

using deep neural network. Procedia Computer Science, 218, 2273-2286.
[38]Balasundaram, A., Sundaresan, P., Bhavsar, A., Mattu, M., Kavitha, M. S., & Shaik, A. (2025). Tea leaf disease detection using segment anything model and deep convolutional neural networks. Results in Engineering, 25, 103784.
[39]Xu, Y., Mao, Y., Li, H., Shen, J., Xu, X., Wang, S., ... & Wang, Y. (2025). A Deep Learning Model Based on RGB and Hyperspectral Images for Efficiently Detecting Tea Green Leafhopper Damage Symptoms. Smart Agricultural Technology, 100817.
[40]Yao, X., Lin, H., Bai, D., & Zhou, H. (2024). A Small Target Tea Leaf Disease Detection Model Combined with Transfer Learning. Forests, 15(4), 591.
[41]Ramdan, A., Heryana, A., Arisal, A., Kusumo, R. B. S., & Pardede, H. F. (2020, November). Transfer learning and fine-tuning for deep learning-based tea diseases detection on small datasets. In 2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET) (pp. 206-211). IEEE.
[42]Shovon, M. S. H., Mozumder, S. J., Pal, O. K., Mridha, M. F., Asai, N., & Shin, J. (2023). Plantdet: A robust multi-model ensemble method based on deep learning for plant disease detection. IEEE Access, 11, 34846-34859.
[43]Wu, P., Liu, J., Jiang, M., Zhang, L., Ding, S., & Zhang, K. (2025). Tea leaf disease recognition using attention convolutional neural network and handcrafted features. Crop Protection, 107118.
[44]Thakur, P. S., Sheorey, T., & Ojha, A. (2023). VGG-ICNN: A Lightweight CNN model for crop disease identification. Multimedia Tools and Applications, 82(1), 497-520.
[45]Bhandari, M., Shahi, T. B., Neupane, A., & Walsh, K. B. (2023). Botanicx-ai: Identification of tomato leaf diseases using an explanation-driven deep-learning model. Journal of Imaging, 9(2), 53.
[46]Das, A., Pathan, F., Jim, J. R., Ouishy, M. R., Kabir, M. M., & Mridha, M. F. (2025). XLTLDisNet: A Novel and Lightweight Approach to Identify Tomato Leaf Diseases with Transparency. Heliyon.
[47]Kurniawan, W. M., Gambetta, I. W., & Alfianny, I. R. (2024, October). Explainable Artificial Intelligence (XAI) Application on Classification of Diseases and Symptoms of Pest Infestation on Plants (Case Study on Rice Plant Leaves). In 2024 IEEE International Conference on Data and Software Engineering (ICoDSE) (pp. 137-142). IEEE.
[48]Manasa, J. P. (2023, February). Mobile Based Application For Tea Leaf Disease Detection. In 2023 IEEE 3rd International Conference on Technology, Engineering, Management for Societal impact using Marketing, Entrepreneurship and Talent (TEMSMET) (pp. 1-5). IEEE.
[49]Obadaarachchi, E., Saputhanthri, D., Vithanarachchi, P., Zuhry, Y., & Rathnayake, R. (2024, December). GreenGuard: Mobile Application for Real-Time Tea Leaves Disease Detection and Treatment Guidance. In 2024 6th International Conference on Advancements in Computing (ICAC) (pp. 1-6). IEEE.
[50]Li, Y., Lu, Y., Liu, H., Bai, J., Yang, C., Yuan, H., ... & Xiao, Q. (2024). Tea leaf disease and insect identification based on improved MobileNetV3. Frontiers in Plant Science, 15, 1459292.
[51]Tembhurne, J. V., Gajbhiye, S. M., Gannarpwar, V. R., Khandait, H. R., Goydani, P. R., & Diwan, T. (2023). Plant disease detection using deep learning based Mobile application. Multimedia Tools and Applications, 82(18), 27365-27390.
[52]Ahmed, A. A., & Reddy, G. H. (2021). A mobile-based system for detecting plant leaf diseases using deep learning. AgriEngineering, 3(3), 478-493.

212-15-4205

ORIGINALITY REPORT

22%	11%	17%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	3%
2	dspace.daffodilvarsity.edu.bd:8080 Internet Source	2%
3	"Proceedings of the Fifth International Conference on Trends in Computational and Cognitive Engineering", Springer Science and Business Media LLC, 2024 Publication	1%
4	export.arxiv.org Internet Source	1%
5	www.mdpi.com Internet Source	1%
6	Peng Wu, Jinlan Liu, Mingfu Jiang, Li Zhang, Shining Ding, Kewang Zhang. "Tea leaf disease recognition using attention convolutional neural network and handcrafted features", Crop Protection, 2025 Publication	<1%
7	Submitted to United International University Student Paper	<1%
8	Pierfrancesco Novielli, Donato Romano, Michele Magarelli, Pierpaolo Di Bitonto et al. "Explainable artificial intelligence for microbiome data analysis in colorectal cancer	<1%