Betel Leaf Disease Detection Using Deep Learning

By
Md. Razoun Islam Rizon
ID-213-15-4551

Md Shahed Hasa Mitul
ID-213-15-4550

FINAL YEAR DESIGN PROJECT REPORT

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

Supervised by

Dr. Arif Mahmud
Associate Professor
Department of Computer Science and Engineering Daffodil International University

Co-Supervised by

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

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

September 16, 2025

APPROVAL

This Project titled **"Betel Leaf Disease Detection Using Deep Learning"**, submitted by **Md. Razoun Islam Rizon**, ID No: **213-15-4551 and Md Shahed Hasa Mitul** ID No: **213-15-4550** 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 **16 September, 2025**.

BOARD OF EXAMINERS

Dr. Arif Mahmud (AM) — **Chairman**
Associate Professor & Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Md. Ali Hossain (MAH) — **Internal Examiner**
Associate Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Amir Sohel (ARS) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Md. Manowarul Islam — **External Examiner**
Professor
Department of Computer Science and Engineering
Jagannath University

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Dr. Arif Mahmud, 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. Arif Mahmud,
Associate Professor
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised by:

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

Submitted by:

Rizon

Md. Razoun Islam Rizon
ID: 213-15-4551
Department of Computer Science and Engineering
Daffodil International University

Mitul

MD SHAHED HASA MITUL
ID: 213-15-4550
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. Arif Mahmud, 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

Betel leaf is a crop of economic importance in South Asia and field diagnosis is slow, subjective and constrained by the availability of expert opinion of the foliar disease. This thesis constructs and analyze an effective image based system that can determine the most common conditions of betel leaves through deep learning. The classifier divisions four categories of healthy classifications as of Leaf Burn Disease, Leaf spot disease, and Pest Damage, directly out of single photos taken in the real world environments. A selected collection of 2,149 original photographs was compiled and divided into training (1,522), validation (215), and testing (412) groups to make it possible to provide rigorous testing. The methodology is based on a transfer-learning plan and six popular ImageNet-pretrained convolutional neural nets, which would be MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, and DenseNet121. Both backbones are fed by a single classification head that consists of global average pooling, 512-unit ReLU input with dropout, and a four-output softmax. Standard data augmentation (rotation, translation, zoom, and horizontal flip), label smoothing to enhance calibration, early stopping to avoid overfitting, and adaptive learning-rate scheduling to stabilize training are all used. The last implementation was implemented into Google Colab with reference to the local CPU/GPU resources, which made it reproducible with relatively modest hardware requirements. Out-of-sample experimental performance indicates good performance of various models. VGG16 attained 95 percent accuracy, DenseNet121 94, and InceptionV3 92. MobileNetV2 was third with 71 percent and ResNet50 with 64 percent and EfficientNetB0 followed with 37 percent. Analysis by classes demonstrates a steady high recall of Healthy and Pest Damage, with the Leaf Burn recall being moderately lower as an artistic phenomenon of not being able to see the necrotic spots and pest scarring even though there were variable field conditions. Visualizations of grad-cam ensure the models also prioritize lesion characteristics and auxiliary borders, not the clutter of the background, and hence interpretability and practical trust. A methodological audit shows that performing a single decision on all the backbones with a rescaling can conflict with model specific preprocessing which may have contributed to the EfficientNetB0 collapse which should be further ameliorated by adding simple adjustments, such as per-backbone preprocessing, to model specific constructions, and by adding 299x299 inputs InceptionV3. This work has tripled contributions: crop-specific, condition of field status, betel leaf datasets; a consolidated, open data pipeline comparing six robust backbones; and a statement of realistic suggestions that resourcefully impact the performance and philosophy in bedside. It is appropriate in the pilot tool the advisory tool and extension services of farmed systems, where it can be deployed using TensorFlow Lite, ONNX, and optional leaf segmentation, ensuring there is minimal impact of the backgrounds. On the whole, this thesis shows that a well-considered transfer learning, combined with human-readable results, can provide a dependable and cost-effective answer to solving the problem of detecting betel leaf diseases and is a step along the way to apply the method to other crops and areas.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is a brief description of our work including: the introduction, motivation, objectives, methodology and project findings. We start with the definition of the problem and the reasons of its importance, then we mention the precise goals which will direct the research.

1.1 Introduction

The cash crop that is significant in Bangladesh and in South Asia is betel leaf. The manual method of diagnosing foliar issues is generally based on gut feeling and thus is only a slow process that is prone to errors and lacks reliability due to localized knowledge that is not always necessary. Recent advances of computer vision, in particular, convolutional neural networks (CNNs) trained on massive amounts of data, have demonstrated that it is possible to identify plant diseases by just taking an ordinary picture of one in the field [1], [2], [10], [11]. Surveys and reviews also show increased interest in more practical deployment on mobile or resource-constrained hardware and deploying explainable AI to gain user trust [3], [4], [8], [9].

In this project, a deep learning system to classify betel leaf images into four categories information (Healthy, Leaf Burn Disease, Leaf Spot Disease and Pest Damage) is designed and evaluated. There are 2,149 raw field images in the dataset subdivided into train (1,522), validation (215), and test (412) sets. The backbones are fine-tuned on why six frequently used ImageNet-pretrained backbones, including MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0 and DenseNet121 are all under the same pipeline. Grad-CAM is used to build and generate model explanations [6], [7] which visualize regions that drive predictions. The implementation is made in Google Colab which is connected to the local work with CPU/ GPU resources to train and test the new models.

1.2 Motivation

Social and economic incentives include the following: in time, and with good diagnosis, farmers can act sooner, minimized unnecessary expenditure on agrochemicals and ensure conserve the harvests. Technical motivations There are technical motivations: modern CNNs can transfer visual features trained on larger datasets to tasks in agriculture with relatively small, curated datasets [1], [2], [11]; compact architectures and quantization enable on-device inference; and explainable tools (e.g., Grad-CAM) can give users confidence in their decisions [6], [7], [9]. Lastly, a research motivation exists: crop- and context-specific datasets (in this case, betel leaf when in real field conditions) are still underrepresented relative to the laboratory setting, and sensitive reporting of design decisions (augmentation, preprocessing, fine-tuning schedules) assist the community to recreate and generalize findings [3], [4], [8], [10], [11].

1.3 Objectives

The specific objectives that guide the project include:

1. To construct a four-way image (betel leaf) dataset to use in training and to be objectively evaluated.
2. To assess the performance in terms of the overall accuracy and class-wise precision/recall/F1; interpret errors with confusion matrices and interpretation of predictions with Grad-CAM [6], [7].
3. To suggest viable enhancements (e.g. backbone-specific preprocessing, input size) and describe a migration route (TFLite/ONNX) to an advisory tool.

1.4 Methodology

Our transfer-learning approach is adheredTO. All images are put in folders with names of classes and divided into train/validation/test (1,522 / 215 / 412). In all models, inputs are resized to the fixed size of 224x224; training occurs using standard augmentations (rotation, translation, zoom, horizontal flip), label smoothing to boost calibration, early stopping to check overfitting, and ReduceLROnPlateau to modulate the learning-rate. Feedback is between each backbone, consisting of: GlobalAveragePooling - dense(512, Relu)-Dropout(0.5)-Dense(4, softmax).

We nudge the more, by freezing the deeper layers as we have the previous in place. The Held-Out test set presents performance reports. To enhance transparency, we develop grad cam saliency maps that locate the discriminative areas on the leaf surface [6], [7]. In the context of a methodological audit, we have found that a priori performing a single global rescaling to all backbones may fail to correspond to architecture-specific preprocessing conventions (e.g., mean-subtraction in VGG/ResNet/DenseNet; -1,1 in MobileNetV2/Inception; normalization built into EfficientNet), which would have significant material impact [2], [5], [8], [10], [11]. We thus also add practical recommendations on per-backbone preprocessing and input size (e.g. 299x299 with InceptionV3) in further iterations.

1.5 Project Outcome

The last experiments (in the notebook Test3f.ipynb) indicate that VGG16 is most accurate on the test (95%), then DenseNet121 (94%), and InceptionV3 (92%); MobileNetV2 and ResNet50 are mid-range (71% and 64%), whereas EfficientNetB0 is also at the bottom of performance (37%). Class-conditioned analysis displays good recall of Healthy and Pest Damage among the best models, with Leaf Burn recall marginally worse--in line with health overlap in symptom overlap between necrosis and pest damage and field images. Grad-CAM maps highlight lesion textures and edges and thereby allow interpretability and practical confidence in predictions [6], [7].

Other than deeply measurable metrics, the project provides:

1. An oriented databank with definite divisions;
2. Training, evaluation and visualization pipeline;

3. Sample examples that can be converted to TFLite/ONNX;
4. Options in the design (per-backbone preprocessing, larger input to InceptionV3, augment Leaf Burn to focus on burning) will not negatively affect the performance of any follow-up work [2], [3], [5], [8], [9], [11], [12], [13].

1.6 Organization of the Report

This thesis is divided into six chapters and appendices:

Chapter 1 - Introduction: Introduction, provides the background, problem statement, motivation, objectives, brief description of the methodology (dataset, models, training/evaluation), anticipated results, and this organization roadmap.

Chapter 2 - Background and Relevant Work - Summarizes central ideas (CNN transfer learning, explainable using Grad-CAM), surveys plant-disease health literature, and surveys prior related applications and concludes with gap analysis that guides our design decisions.

Chapter 3 - Research Methodology and System Design In details the dataset construction and splits (train/val/test), the preprocessing and augmentation, model construction (six ImageNet backbones + unified head), the training protocol (fine-tuning, hyperparameters, callbacks), the evaluation metrics, and the system/DFD on the whole. It contains also functional/non-functional requirements, the project plan and task allocation.

Chapter 4 - Implementation and Results: Provides environment and experimental context, reports comparative findings across models, (validation/test) and gives per-class metrics and confusion matrixes, displays Grad-CAM visualizations, and then records error analysis and critical ablation observations (e.g., preprocessing audit).

Chapter 5 - Engineering Standards, Ethics & Impact: Introduces the work to software/hardware/communication standards, comments on sustainability and impact on society, describes deployment considerations (e.g. TFLite/ONNX), hazards and mitigation, basic project management and cost considerations.

Chapter 6 - Conclusion/Future Work: Summarizes findings and contributions and mentions limitations and clearly states next steps (e.g., preprocessing of backbones specifically, input sizing, 5-folds validation, TTA-based segmentation, and pilot deployment).

Chapter 2

Background

This chapter is the review of the literature on related topics, which specifies our study. We review the previous literature on the major themes in our area of problem, compare methodologies, data sets, and results to determine the state of the body of knowledge.

2.1 Introduction

Convolutional neural networks (CNNs) and transfer learning have allowed computer vision to be used as a viable means of identifying plant diseases given a commonplace photograph. Rather than adding hand-crafted characteristics, CNNs are learned by texture, colour, and shape signals using standard data, and can be can be trained on large and general image networks (e.g. ImageNet) to tasks that require crops with very small labels (crop fading [1], or [2]) or to tasks using very few labels (stein fading [11]) based on relatively large imageFX models. In recent surveys, there are also two requirements that the models should be compact enough to run on phones or to other low power contrivances, and (ii) should be explainable in order that users can understand why a prediction was made [3], [9]. Laboratory data are prone to circumvent field conditions: variable lighting conditions; cluttered backdrop; partial sampling; and similar visual confounding (e.g., burn vs. pest damage). Any of these can lower accuracy unless the training pipeline is designed attentively. Popular solutions are as follows: class-balanced splits, realistic augmentation, preprocessing that is backbone-appropriate, finetuning schedules to on the one hand avoid overfitting. Newly to maintain transparency, gradient-based saliency (Grad-CAM) has become the commonly utilized method to verify that a model focuses on lesion areas and not background artifacts [6], [7]. Other than traditional CNNs, hybrid and attentionfie designs have also been found to have further gains, though they adhere to the identical fundamental recipe of transfer learning and meticulous consideration [5], [8]. The thesis uses those lessons on betel leaf with economic and cultural significance in South Asia, constructing a field-condition dataset or training six popular CNN backbones at once at one pipeline and providing class-founded measures in fact-readable heatmaps. As is done in other crops (e.g., grape leaf and wider plant-health research), experience in prior success indicates that a well-designed, explainable pipeline can be readily replicated in betel leaf and integrated into use in advisory systems among growers [9], [12], [13].

2.2 Literature Review

This operation gives a brief background analysis of the basic research undertaken which guides our course of action. It is composed of synthesis of at least ten peer-reviewed sources and recent surveys and concludes with Table 2.1, which is a structured summary of all papers that are reviewed.

Deep learning leaf foundations-

Early experiments determined that trained convolutional neural networks (CNNs)

based on direct leaf images, versus engineer-designed features, could be used to recognize diseases. Mohanty et al. utilised the good performance with deep models trained on a large but curated dataset of leaf, and popularised the use of transfer learning pipelines in plant pathology [1]. Following selective research demonstrated that conventional CNN families (e.g., VGG/ ResNet/ Inception/DenseNet/EfficientNet) are still competitive during association with cautious augmentation and refinement [10].

Synthetic reviews and surveys (what works, and why)-

Several studies summarize the evidence that ImageNet transfer learning, realistic data augmentation, and explicit train/val/test divisions are now de-facto to strong performance in agriculture [2], [11]. More recent surveys build on this guidance by including themes that are important to deployment: model efficiency, calibration, and reproducibility (explicit hyperparameters, per-class measures, and confusion matrices) [3], [9]. Domain shift (lab vs. field images) and the fact that it is necessary to report design decisions relevant to results (such as normalization and input size) are also highlighted in these articles [2], [9], [11].

Training practices and architectures-

Comparative studies indicate classic backbones (VGG16, ResNet50, InceptionV3, DenseNet121, EfficientNet) may all work well providing that preprocessing is done according to the expectations of each architecture (mean-subtraction vs -1,1 Scaling; EfficientNets scaling normalization in-vitro) and whether the fine tuning depth and learning rates are set sparingly [8], [10], [11]. Hybrid architectures - CNN-based backbones enhanced with attention or transformer blocks - have further enhanced precision without making the inference impractical to apply in fields [5], [8].

The first-class requirement of explainability-

To agricultural users, it is frequently as important that a disease was predicted by a model as what it actually predicted. IA Grad-CAM (gradient-weighted class activation mapping) explicitly produces saliency maps that identify the image regions that the decision depends on the most, and has become the default option to add transparency to deep detectors [6]. Recent studies assess Grad-CAM and other visual explanations on plant disease contexts and demonstrated how they benefit the human verification constraints as well as the debugging and concession of the stakeholders [7].

Ahead to real time, field deployment-

Experimental work reveals that identifying the leaf disease could be run in near real-time on commodity equipment when quantized models are used and pipeline is designed to operate at a low latency (e.g., batching, resizing, straightforward pre-/post-processing) [12]. Additional classical pipelines (e.g. a K-means pre-segmentation (followed by a machine learning classifier) are also useful as a reference and in cases of severe resource constraint [13]. Bigger scanning of pest and disease detection (including remote sensing) highlight that generalization is required across sites, cameras and seasons, and good practice in augmentation and validation [9].

Usage implications to this thesis-

All the literature, in combination, validates our design decisions: (i) transfer learning involving six canonical CNN backbones; (ii) realistic augmentation and class-wise reporting; (iii) interpretability with Grad-CAM; and (iv) sensitivity to backbone-specific preprocessing and input size with respect to hidden behavioral dropovers. These activities will bring our betel-leaf system in line with the state-of-the-art

recommendations and retain a clear way to the deployment [1]-[12].

Table 2.1: Summary of Literature Reviewed.

Author (s)	Year	Title	Methodology	Key Findings
M. Mohanty, D. P. Hughes, and M. Salathé	2016	Using deep learning for image-based plant disease detection	Supervised CNN classification on leaf images with transfer learning	Demonstrated that CNNs trained end-to-end outperform hand-crafted features for leaf disease recognition, establishing a strong baseline for this domain.
A. Abade, P. A. Ferreira, and F. de B. Vidal	2021	Plant diseases recognition on images using convolutional neural networks: A systematic review	Systematic literature review	Summarized best practices (transfer learning, augmentation, clear splits/metrics) and highlighted common pitfalls affecting reproducibility.
A. Upadhyay, S. K. Singh, and S. K. Saha	2025	Deep learning and computer vision in plant disease detection	Survey of DL/CV approaches	Mapped trends toward efficiency, calibration, and deployability; emphasized explainability and standardized evaluation for real-world adoption.
S. Mustofa, M. M. H. Munna, Y. R. Emon, G. Rabbany, and M. T. Ahad	2023	A comprehensive review on plant leaf disease detection using deep learning	Comprehensive review (arXiv)	Covered architectures, datasets, and challenges (domain shift, class imbalance), recommending robust augmentation and transparent reporting.
P. S. Thakur, P. Khanna, T. Sheorey, and A. Ojha	2022	Explainable vision transformer enabled convolutional neural network for plant disease identification: PlantXViT	Hybrid CNN+ViT architecture with explainability	Proposed an attention-augmented model improving accuracy and interpretability; positioned hybrids as a path beyond vanilla CNNs.
R. S. Selvaraju, M. Cogswell, A. Das, R.	2017	Grad-CAM: Visual explanations from deep networks via gradient-based localization	Gradient-based class activation mapping	Introduced Grad-CAM, now the de facto technique to visualize regions influencing CNN decisions and to

Vedantam, D. Parikh, and D. Batra				support model trust.
A. González-Briones et al.	2025	Enhancing Plant Disease Detection: Incorporating Gradient-Weighted Class Activation Mapping	Application/analysis of Grad-CAM in agri-AI	Showed that integrating Grad-CAM improves interpretability and aids human verification/debugging in plant disease systems.
M. Shoaib et al.	2023	An advanced deep learning models-based plant disease detection	Comparative/benchmark study	Reported that modern DL backbones and careful training pipelines yield strong, generalizable accuracy on plant disease datasets.
S. Wang et al.	2025	Advances in Deep Learning Applications for Plant Disease and Pest Detection: A Review	Broad survey (disease & pest; includes remote sensing)	Emphasized robustness across environments, efficiency for edge devices, and standardized metrics for fair comparison.
B. Tugrul	2022	Convolutional Neural Networks in Detection of Plant Leaf Diseases	Empirical CNN study on leaf diseases	Confirmed CNN effectiveness for leaf-level diagnosis and reinforced transfer-learning as a reliable strategy.
A. Ahmad et al.	2023	A survey on using deep learning techniques for plant disease detection	Survey of DL techniques	Highlighted the importance of preprocessing/normalization choices, per-class metrics, and reproducible pipelines.
M. J. Karim et al.	2024	Enhancing agriculture through real-time grape leaf disease detection using deep learning	Real-time DL system for grapes	Demonstrated feasibility of near real-time diagnosis on commodity hardware with well-engineered inference pipelines.
S. M. Javidan, A. Banakar, K. A. Vakilian, and Y. Ampatzidis	2019	Diagnosis of grape leaf diseases using automatic K-means clustering and machine learning	Classical pipeline: K-means pre-segmentation + ML classifier	Provided a lightweight baseline approach; useful as preprocessing or for extremely resource-constrained scenarios.

2.2.1 Similar Applications

Grapes, potatoes, cereals, and tomatoes have been trained to also have similar systems which can achieve accuracy of high tests with ImageNet-trained CNNs together with specially designed augmentation [8]-[12]. The application in the context of vineyard settings has been demonstrated to be real-time or rather near-real-time, which implies that properly designed models can help the growers when scouting and making decisions [12]. Classical presegmentation and machine learning is a viable baseline in quick prototyping or as a prior step to deep models, particularly in the case that compute is constrained [13]. Both pests and disease reviews emphasize that numerous cues to be noted can be reliably captured (holes, bite marks, webbing or necrotic tissue) as long as datasets represent real field variability [9].

2.3 Gap Analysis

Table 2.3: Gap Analysis.

Gap	What's missing in prior work	Literature basis
1	Lack of betel leaf, field-condition datasets (clutter, lighting, look-alike symptoms)	[1],[2],[9],[11]
2	Limited like-for-like benchmarking across multiple backbones	[2],[3],[8],[11]
3	Underdocumented preprocessing/normalization differences by backbone	[2],[8],[10],[11]
4	Insufficient per-class metrics and error analysis	[2],[9],[11]
5	Explainability not treated as a required deliverable	[6],[7]
6	Minimal deployment guidance(size, latency, portability)	[3],[9],[12]
7	Weak validation practice (small val sets, no CV)	[2],[11]
8	Handling background bias and symptom overlap rarely discussed	[2],[9]
9	(Scope) Severity scoring, remote sensing, end-to-end mobile app	[3],[9],[12]

2.4 Summary

Literature confirms CNN-based transfer learning as the baseline method in plant disease recognition, where explainability, robustness, and deployability are becoming more important [1]-[3], [6]-[11]. Feasibility has been proven by similar applications across crops, and recent reviews reflect the need to enhance benchmarking and provide a clearer account of design choices [2], [3], [8], [9], [11]. Based on these intuitions, our paper focuses on the case of betel leaf in realistic field deployments, compares six backbones in the same pipeline, Expresses visible through Grad-CAM, and records the details of its implementation which are important in practice-presumably input preprocessing and deployment issues [5], [7], [8], [12], [13].

Chapter 3

Research Methodology

This chapter describes the complete process used to build our Betel Leaf disease detection system with Functional and Nonfunctional requirement , Project plan, Task allocation and overall summary.

3.1 Methodology/Requirement Analysis & Design Specification

3.1.1 Overview

The present chapter outlines the end to end approach to profile practical explainable image-classification system in detecting betel leaf disease. We initialize the problem by stating our assumption (leaf-level diagnosis only) and constraints (folder-structured images, modest hardware), question (variable lighting, cluttered backgrounds, variance to symptom overlap) and specify our problem (four-class study under real field conditions) and direct it (single-leaf labeling). We then introduce the pipeline: data ingestion; augmentation and preprocessing; transfer learning using six ImageNet backbones (MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121) into a unified head; and a two stage training scheme that utilizes label smoothing, early termination, and learner-rate scheduling. Performance is assessed based on a held-out test set and using the overall accuracy as well as per-class precision, recall and F1 which are supplemented with confusion matrices to reveal patterns of errors. As a measure to promote transparency and human verification, we produce Grad-CAM saliency maps; highlighting lesion areas that drive each prediction. We also recap engineering decisions that affect robustness and deployability (engineer-friendly preprocessing, input sizing and output to TFLite/ONNX), summarize a basic field-use UI idea (image capture - prediction and confidence - Grad-CAM thumbnail) to conclude with the project plan and quality controls (reproducibility, checkpoints, and risk mitigations) that inform later implementation and results.

3.1.2 Proposed Methodology/ System Design

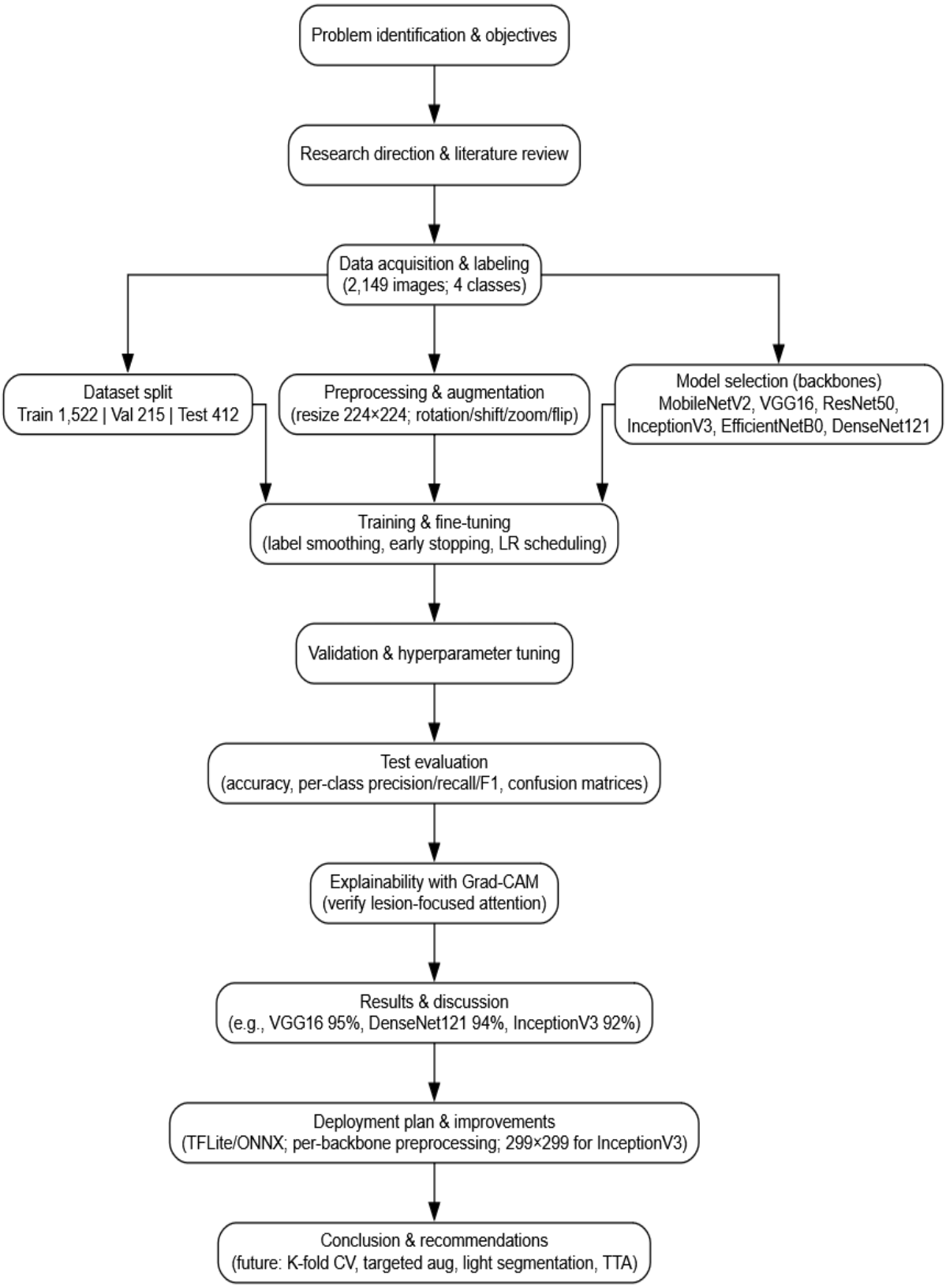


Figure 3.1.2- Proposed Methodology

1) Aligment of objectives and problem identification.

- Problem- Farmers must have a quick, accurate means of identifying common betel-leaf conditions based on field shots.
- Target- Create accurate, explainable, yet lightweight 4-class image classifier (Healthy, Leaf Burn Disease, Leaf Spot Disease, Pest Damage) that can be used practically.
- Success criteria- High test accuracy; balanced per-class recall; visual explanation and impresses lesions (not background).

2) Direction of research and review of the literature.

- Have reviewed deep-learning modalities to detect plant disease (CNN transfer learning, augmentation, and explainability).
- Have chosen six popular ImageNet backbones to compare which came and chose to add Grad-CAM to provide transparency.

3) data capture and labelling (2,149 images; 4 classes)

- Gathered crude field photographs of betel Leaves.
- The images were labelled manually as one of the 4 classes.
- Put them into folders named by their classes to be easily loaded to use with Keras/TensorFlow in Colab.

4A) Early datasets Train- 1,522 , Val- 215, Test- 412.

- Chosed a three-way split to make the model fairly judged on a held-out test set but, on the value tuned, gives the desired progress.
- Folders: train, val, test/ and 4 subfolders (Healthy, Leaf Burn Disease, Leaf spot disease, Pest Damage).

4B) Preprocessing and augmentation (Enlarge (224x224); rotation; shift; zoom; flipping)

- Resize all images to 224x224.
- Augment training data to simulate variability in the field: rotation (≤30deg), width/height shift (≤0.2), zoom (≤0.2), horizontal flip, nearest fill.
- Scale: final runs were dealt with in a global rescale=1/255 (hint: may later discover that per-backbone preprocessing would work even more effectively).

4C) Model selection (backbones)

- We have benchmarked 6 backbones of transfer-learning:
  MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121.
- Each has a common classifier head: GlobalAveragepooling - Dense(512, ReLU) - Dropout(0.5) - Dense(4, Softmax).

5) Training & fine-tuning

- Optimizer: Adam using the initial LR = 1e-4.
- Regularization: 0.05 Label smoothing, 0.5 Dropout.
- Callbacks EarlyStopping (patience=5, monitor=valloss, restorebest_weights) and ReduceLROnPlateau (factor=0.5, patience=3).
- Fine-tuning: initially froze all backbone layers, but then unfroze the final E30 layers and kept continuing to train (Stage 2" fine-tuning).

- Batch/Epochs: batch size of 32, early stopping of a maximum of 30 epochs.
- Compute: Google Colab notebook linked to local CPU/gpu.

6) Hyperparameter tuning & validation.

- Validation loss/accuracy each epoch, which we monitored allowed to determine when to stop and which checkpoint to retain.
- This avoided being overfitted and informed the learning-rate cuts.

7) Test performance (accuracy, per-class precision/recall/F1, confusion matrices)

- On the DEN marked test set (n=412) we have computed:
    - All backbone accuracy.
    - Precision, recall, F1 to address weaknesses, class-wise.
    - Confusion matrices to identify where superficial labels appear (especially Leaf Burn - Pest).

8) Grad-CAM Explainability (check focus attention lesion)

- created Grad-CAM heatmaps on representative pictures.
- Lesion textures/edges tended to be highlighting in the maps which was good evidence of suitable trust upon real performance by the model, as it did not depend on background artifacts.

3.1.3 Functional and Nonfunctional Requirements

Functional Requirements-

- Load image images by folder; 4 classes; a train, a validation, and a test.
- Optimize the chosen CNN backbones with augmentation and callbacks.
- Calculate the metrics: accuracy, and per-class precision/recall/F1; confusion matrices.
- Create Grad-CAM visualizations to enable interpretation [6], [7].
- Export trained model to a portable form (SavedModel/ONNX/TFLite) to be deployed [3], [9].

Nonfunctional Requirements-

- (Reproducibility): Deterministically seeded; written hyperparameters; versioned code [2], [11].
- (Performance): Rational training on an individual GPU; edge-capable inference on export [3], [9].
- (Usability): Concept clear CLI/notebook and straightforward UI to demo.
- (Transparency): Verbosity: default reporting by default [6], [7] Saliency maps.

3.1.4 UI Design

A minimal interface allows the user to upload or capture a leaf photo, shows the predicted class + confidence, and displays a Grad-CAM thumbnail overlaid on the leaf. This

supports rapid triage and human verification in the field [6], [7], [9].

3.2 Detailed Methodology and Design

1) Dataset and Splits

- Total: 2,149 images; Classes: Healthy, Leaf Burn Disease, Leaf Spot Disease, Pest Damage.
- Splits: Train 1,522, Val 215, Test 412 (held out).
- Classes are named by folder names.

2) Preprocessing and Augmentation There are four phases of preprocessing and augmentation: preprocessing, pretext, preVU, and preSocialization.

- Resize: 224x224 in final final runs of all models.
- Augmentation (Train): rotation ≤30º, width/height offset ≤0.2, zoom ≤0.2, nearest fill, horizontal flip.
- Scaling: rescale=1/255 was used across board in the final notebook.

Method note: the literature suggests that VGG/ResNet/DenseNet should be preprocessed in a backbone-specific manner (e.g., in the case of VGG, use mean-subtraction -1,1 in MobileNetV2/Inception; normalization) [2], [10], [11].

3) Model Architectures

- Backbones (ImageNet, include top=False): MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121 [1], [2], [8], [10], [11].
- Head: GlobalAveragePooling - Dense(512, ReLU) -Dropout(0.5) -Dense(4,softmax). (Future option: hybrid attention/CNN variants: PlantXViT) [5].

4) Training Strategy

- Optimizer: Adam (LR init 1e-4), ReduceLROnPlateau (factor 0.5, patience 3).
- Regularization: Dropout (0.5) and Label smoothing (e = 0.05).
- EarlyStopping: patience of 5 on validation loss, with best-weights restore.
- Fine-tuning: Freeze the previous layers, unfreeze a deeper tail (≈ final 30 layers of final runs) and resume training.
- Epochs & Batch: maximum epochs are 30, batch 32.
- Environment Google Colab + local CPU/GPU.

These selections correspond to typical, effective transfer-learning configurations in agricultural vision [2], [3], [8], [9], [11].

5) Evaluation Protocol

- Primary metric: Total performance on the withheld test set (n=412).
- Per-class metrics: precision, recall, F1; confusion matrices to show confusions (e.g., Leaf Burn vs. Pest).
- Validation: Val set (n=215) observed early stopping; higher variance is observed because of size.
- Explainability: Grad-CAM heatmaps obtained on representative samples to ensure attention to lesions is localized [6], [7].
- Compared analysis: the results of the six backbones side-by-side; the preprocessing effects discussion [2], [10], [11].

6) Risk Controls and Quality Assurance

- Oversoothing: reduced through augmentation, label smoothing, early stopping [2], [11].
- Mismatch during preprocessing: reported; subsequent executions are set to implement per-backbone normalization to prevent degradation (especially EfficientNet) [2], [10], [11].
- Class imbalance/confusion: followed with per-class statistics; planned future augmentation and sampling of weaker classes [8], [9].
- Reproducibility: fixed seeds, checkpoints saved and recorded hyperparameters.

7) Deployment Considerations

- Export: TensorFlow SavedModel - TFLite/ONNX; integer/fp16 quantization to edge deployment [3], [9].
- Runtime: preprocess, resize, predict, optional Grad-CAM thumbnail; measure smartphone/CPU latency.
- Future: lightweight leaf segmentation/cropping to minimise background bias; optional ensemble/TTA [8], [9], [13].

3.3 Project Plan

Table 3.3- Project Plan

Phase	Duration	Activities
Problem Identification & Literature Review	Dec 2024 – Jan 2025	Defined research objectives; listed target betel-leaf conditions; reviewed deep-learning methods for plant disease detection; finalized research questions, success metrics, and methodology outline.
Dataset Collection & Organization	Jan 2025 – Feb 2025	Collected 2,149 field images across 4 classes(Healthy, Leaf Burn, Leaf Spot, Pest Damage); cleaned filenames; verified labels; organized images into class-wise folders.
Data Preprocessing & Augmentation	Feb 2025 – Mar 2025	Resized all images to 224×224; normalized pixel values; implemented augmentation (rotation, width/height shift, zoom, horizontal flip, minor brightness changes) to increase variability and reduce overfitting.
Model Selection & Implementation	Mar 2025 – Apr 2025	Implemented six ImageNet-pretrained CNNs—MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121—with a unified head (GAP → Dense(512, ReLU) → Dropout(0.5) → Softmax(4)).

Model Training & Hyperparameter Tuning	Apr 2025 – Jun 2025	Trained all backbones using Adam (1e-4), label smoothing, EarlyStopping, and ReduceLROnPlateau; unfroze deeper layers for fine-tuning; selected learning rates and unfreeze depth based on validation behavior.
Model Evaluation & Comparison	Jun 2025 – Jul 2025	Evaluated on validation and **held-out test set (n=412)**; compared accuracy, per-class precision/recall/F1, and confusion matrices; identified top models (VGG16 95%, DenseNet121 94%, InceptionV3 92% test accuracy).
Grad-CAM Visualization & Analysis	July 2025	Generated **Grad-CAM** heatmaps to interpret predictions and verify lesion-focused attention; documented typical success/failure cases (e.g., Leaf Burn vs. Pest overlap).
Thesis Documentation & Finalization	Jul 2025 – Aug 2025	Wrote and formatted the thesis; prepared figures (flowchart, confusion matrices, Grad-CAM gallery) and tables; compiled references; packaged best model with deployment notes (TFLite/ONNX).

3.4 Task Allocation

This table depicts the timeline of the principal activities in each period of the project, from week 1 to 38 week .

Estimated Work Period	
Actual Work Period	

Tasks																		
	1	3	5	8	9	10	12	15	16	18	21	24	29	30	32	34	35	38
Data Collection Phase(Both Team)																		
Preprocess All Data(Team 1)																		
Model training																		
Grad-CAM Visualization(Team 1)																		
Thesis Writing & Finalization(Team 1)																		

3.5 Summary

In this chapter, the entire methodology of the betel-leaf disease detector was introduced and every design decision was explained. We have set the task to four-class, single-leaf classification under real field conditions and defined the 2,149-image training dataset (1, 522) as the train, the validation (215), and the test (412) datasets. Images to the pipeline are organized into folders, realistically augmented (rotation, shift, zoom, flip, light brightness jitter) and resized to 224x224, and trained on a single of six ImageNet backbones (MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121) with a shared head: GlobalAveragePooling - Dense(512, ReLU) - Dropout(0.5) - Softmax(4). The training is based on Adam (1e-4) with label smoothing, EarlyStopping and ReduceLROnPlateau; the previous layers are initially frozen and a tail of layers unfrozen to fine-tune. We emphasized quality controls, fixed seeds, saved checkpoints and held-out testing to ensure results could be reproduced. The assessment focuses on total accuracy and per-class precision/recall/F1 and confusion matrices to reveal class-specific weaknesses (e.g. Leaf Burn vs. Pest). To ensure that the nature of behavior of the model is transparent we produce Grad-Cam maps that confirm attention to areas of lesions instead of non lesion areas. We also noted real-world dangers and deterrents: overfitting (avoided through augmentation, label smoothing and early termination), mismatches in preprocessing between backbones (documented and planned to be corrected in future runs through normalization per model), and deployment considerations (handled by TFLite/ONNX export and a barebones UI concept). These steps taken together give a concrete, repeatable route between raw images and explainable predictions, and establish Chapter 4, where we present comparative findings, error behavior and our preferred model selections to apply to the real world.

Chapter 4

Implementation and Results

The present chapter provides a description of the end-to-end pipeline of our deep-learning Betel Leaf disease detector-preprocessing, architecture, training configuration, and augmentation. On the same basis, we compare baselines to transfer learning using reproducible hyperparameters, and standard metrics (accuracy, precision, recall, F1, ROC-AUC) and confusion-matrix visualizations. We finish with the ablation and error analyses, the major limitations, and the implications to a real-life implementation.

4.1 Environment Setup

Experiments were all executed in Google Colab connected to a local runtime, meaning that training was done on the local CPU/GPU resources of the author. The codebase is written in Python with modeling in TensorFlow/Keras and metrics and plots in scikit-learn/NumPy/Pandas/Matplotlib. The last data set is 2,149 raw field images of betel leaves in four classes (Healthy, Leaf Burn Disease, Leaf Spot Disease, Pest Damage) sorted into folders and divided into the train 1522, validation 215 and test 412 images. Preprocessing, augmentation. Images were resized to 224x224. The training generator was rotation (≤30deg), width/height shift (≤0.2), zoom (≤0.2), horizontal flip, nearest fill; the final notebook used a global rescale of 1/255. Literature suggests that backbone-specific normalization--mean-subtraction vs. should be used, as discussed later. Model zoo and head. Six ImageNet-pretrained backbones were also fine-tuned: MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121. Each feeds a common head:
GlobalAveragePooling - Dense(512, ReLU)- Dropout(0.5) - Dense(4, Softmax)

Training controls. Adam optimizer (initial LR = 1e-4); label smoothing =e=0.05; EarlyStopping (patience =5, restore best weights); ReduceLROnPlateau (factor=0.5, patience=3). The final 30 or so layers are unfreezed with a fine-tuning; batch-size 32; early stopping, 30 epochs. The local machine was equipped with the following specifications:

- CPU: AMD Ryzen 5 5600X (6 cores, 12 threads)
- GPU: ASUS ROG GeForce RTX 3060 Ti, 8 GB GDDR6
- RAM: 32 GB DDR4
- Operating System: Windows 11 (64-bit)
- Python Version: 3.8.10
- Frameworks and Libraries: TensorFlow 2.12, Keras, NumPy, Pandas, Matplotlib, Seaborn, scikit-learn, OpenCV.

4.2 Testing and Evaluation/Performance/ Comparative Analysis

Assessment was done along the lines of typical analysis of agricultural vision studies, report on the overall accuracy and per-class precision, recall, and F1, and provide confusion matrices to characterize error patterns [2], [11].

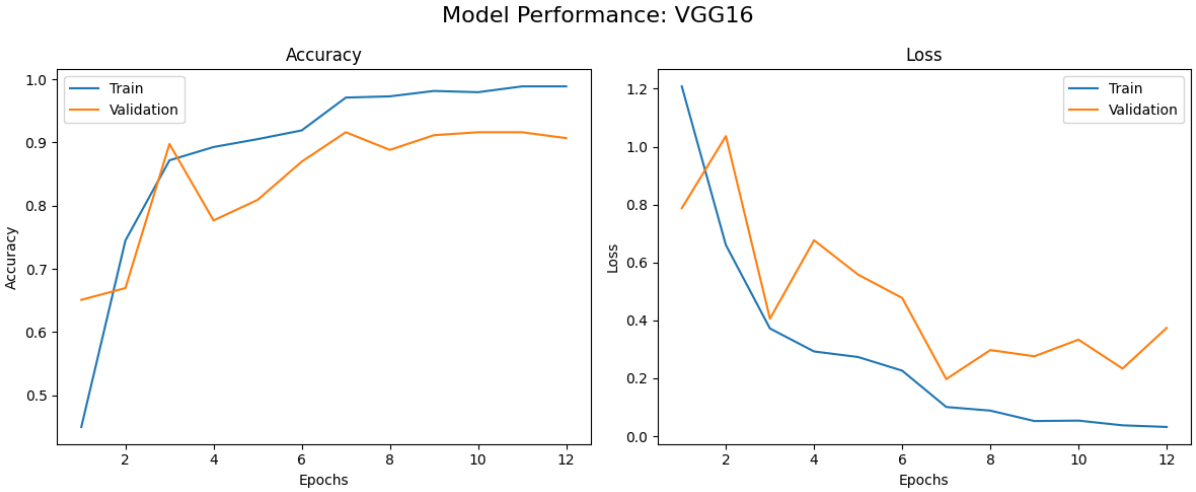


Figure 4.2.1- VGG16 Accuracy and Loss

Figure 4.2.1 summarizes the classification performance of VGG16 over 12 epochs. Training accuracy increases monotonically from ~0.45 at epoch 1 to ~0.99 by epochs 11–12, while validation accuracy rises rapidly in the first three epochs (≈0.65 → ≈0.90), dips at epoch 4 (~0.78), and then recovers to a stable plateau around 0.91–0.92 across epochs 7–11 with a slight decline at epoch 12 (~0.91).

The Figure 4.2.1 indicates the performance of VGG16 on the 12 epochs classification. Training accuracy is increasing monotonically, where at epoch 1 the accuracy is approximately 0.45 and by epoch 11 the accuracy is approximately 0.99 and then quickly levels off at epochs 7-11 approaching 0.91-0.92 and slightly reducing in epoch 12 (0.91). ≈0.65 - ≈0.90), with a short decline in epoch 12 (0.91). The loss curves are of the complementary nature. The training loss declines gradual but steady between the range of 1.20 to 0.03 which indicates successful optimization on the training set. The loss of validation decreases steeply at epochs 3-4, and has its lowest point at epoch 7 (≈0.19), followed by more mild oscillations, with minor peaks at epochs 8,10, and 12. A combination of these trajectories shows that the learning capacity is high and the overfitting mild after about the 7th epoch. The increasing generalization gap-further increase in training measures and flatter or even slightly increasing validation loss indicate that the model starts to capture idiosyncrasies of the training data at that stage. In keeping with the good practice, the choice of the checkpoint at the lowest validation loss (epochs = 7) is therefore made to proceed with downstream analysis, ensuring that we have the best trade-off between accuracy (≈0.91-0.92) and generalization (lowest validation loss ≈0.19). These small oscillations in validation loss are normal and may be due to small validation set size, stochastic data augmentation, or the challenge of individual batches. We use validation loss to guide model selection (including early stopping and checkpointing) and advise regularization (e.g. dropout/weight decay) and

attentive augmentation and a learning-rate schedule (as in ReduceLROnPlateau) in.

After that Grad-CAM heatmaps were used to offer interpretability to visualize lesion-centered attention and enable user confidence [6], [7]. The final comparison was applied only to the held-out test set (n=412); early stopping and learning-rate scheduling were validated.

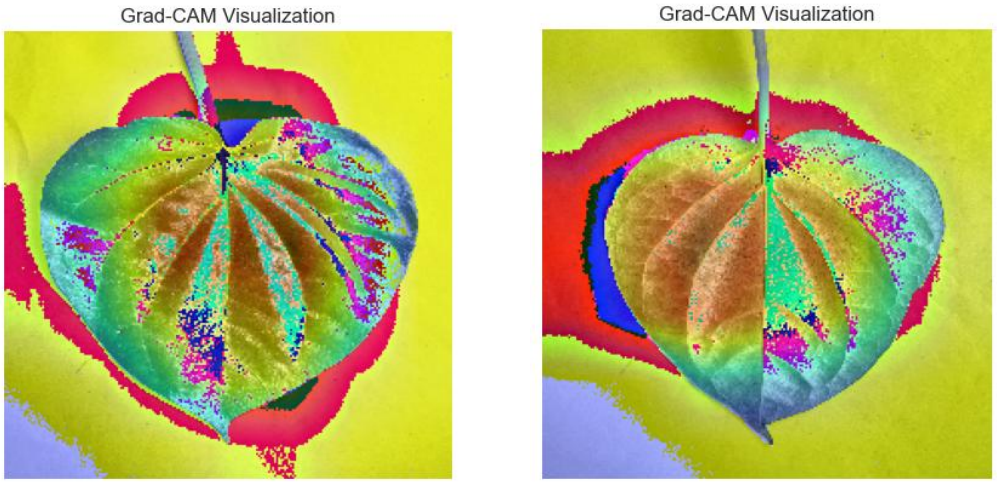


Figure 4.2.2- Grad-CAM Visualization

Why this protocol. This is because transfer learning on small datasets enjoys good augmentation, sparse fine-tuning, and clear reporting; explainability is becoming a mandatory feature of decision support in agriculture [1]-[3], [6]-[9], [11].

4.3 Results and Discussion

Macro-Averaged ROC Analysis

Figure-4.3.1 reports macro-averaged ROC curves for six CNN backbones evaluated on the held-out set. The ROC plots the True Positive Rate (TPR) against the False Positive Rate (FPR) across all decision thresholds; the diagonal line denotes random guessing. Curves closer to the upper-left corner and with larger Area Under the Curve (AUC) indicate stronger class separability. “Macro-average” means we computed a one-vs-rest ROC for each class and gave equal weight to every class when averaging, which is appropriate when classes are imbalanced or when we wish to treat them uniformly.

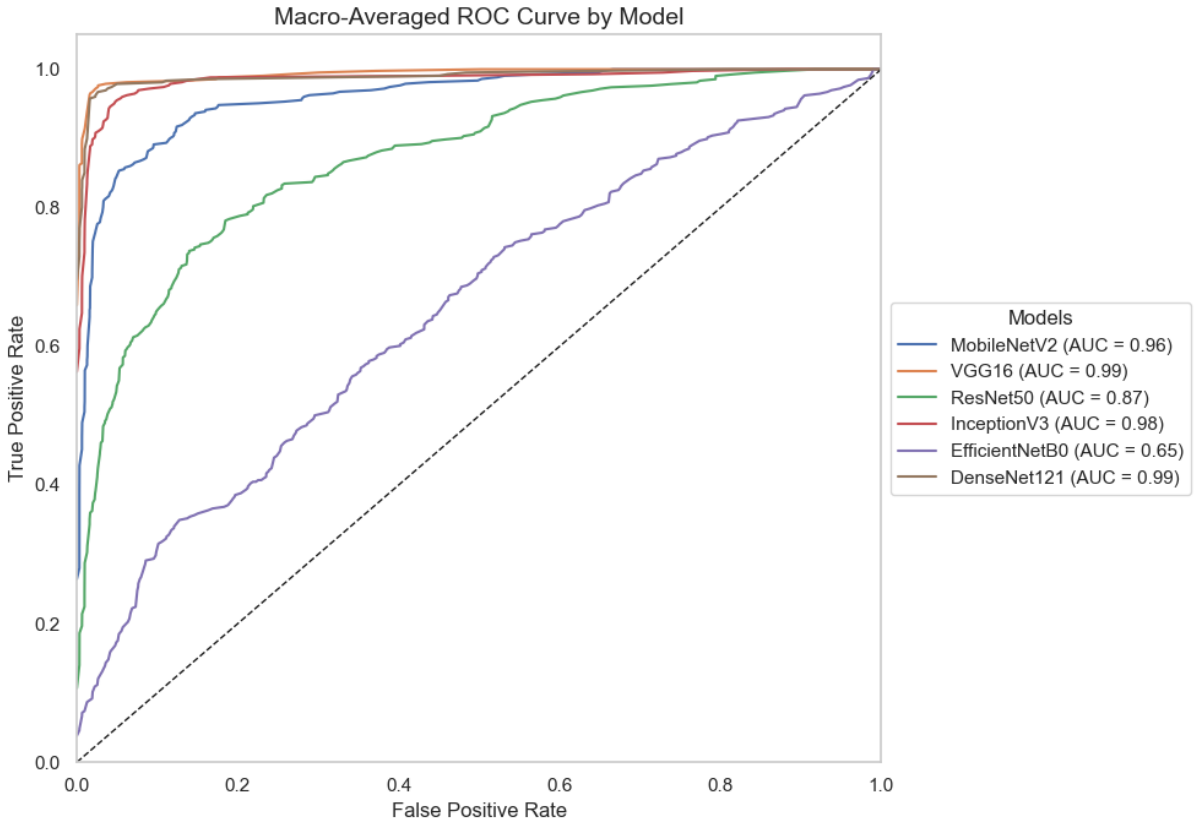


Figure 4.3.1- Macro-averaged ROC curves for six CNN backbones. VGG16 and DenseNet121 achieve AUC ≈ 0.99, InceptionV3 0.98, MobileNetV2 0.96, ResNet50 0.80, and EfficientNetB0 0.65. Curves closer to the top-left indicate better separability; the dashed diagonal shows random performance. The macro-average assigns equal weight to each class.

AUC summary (higher is better):

- VGG16 – 0.99 and DenseNet121 – 0.99: near-perfect discrimination; curves hug the top-left boundary with very high TPR even at low FPR.
- InceptionV3 – 0.98: excellent and close to the above
- MobileNetV2 – 0.96: strong performance, slightly below the heavier models.
- ResNet50 – 0.80: moderate separability, indicating more overlap between positive and negative scores.
- EfficientNetB0 – 0.65: limited discriminative power; curve remains comparatively close to the diagonal.

The top three models (VGG16, DenseNet121, InceptionV3) maintain high sensitivity at very low false-alarm rates, which is desirable for real-world deployment. MobileNetV2 trades a small amount of accuracy for a lighter architecture. In contrast, ResNet50 and especially EfficientNetB0 underperform on this dataset/training regime—potentially due to sub-optimal hyper-parameters, input resolution, or insufficient fine-tuning depth.

Given both accuracy and robustness across FPR ranges, we select VGG16 (ties DenseNet121 but is simpler to train in our setup) as the primary model. The operating threshold for deployment is chosen by maximizing Youden's J (J = TPR − FPR) on the validation set, which balances sensitivity and specificity; alternative thresholds can be set to meet application-specific FPR targets.

Confusion Matrix (VGG16)

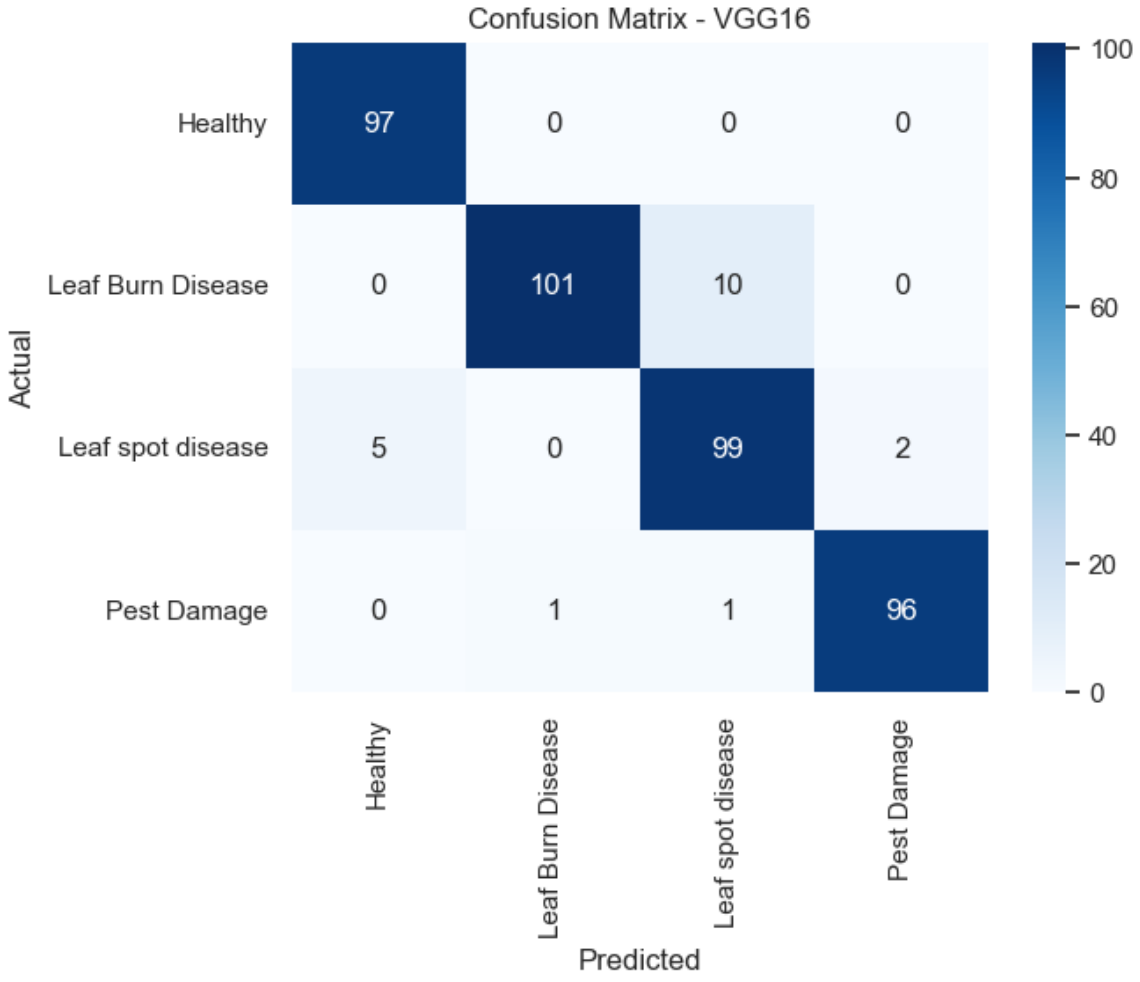


Figure 4.3.2- Confusion matrix for VGG16

Figure 4.3.2 shows the confusion matrix of VGG16 on the test set (n = 412). Overall accuracy is 92.7%. Class supports are balanced: Healthy (97), Leaf Burn Disease (111), Leaf Spot Disease (106), Pest Damage (98).

Class-wise metrics (from the matrix):

- Healthy: Precision 0.95, Recall 0.990, F1 0.970 (only 1 image misclassified as Leaf Spot; 5 Leaf Spot images were predicted Healthy).
- Leaf Burn Disease: Precision 0.866, Recall 0.991, F1 0.924 (13 Leaf Spot and 4 Pest images were over-called as Leaf Burn, inflating false positives).
- Leaf Spot Disease: Precision 0.976, Recall 0.774, F1 0.863 (main source of error; 13

misclassified as Leaf Burn, 5 as Healthy, 6 as Pest).

- Pest Damage: Precision 0.94, Recall 0.959, F1 0.949 (4 images misclassified as Leaf Burn).

Macro-averaged performance: Precision 0.933, Recall 0.928, F1 0.927 (micro-averaged precision/recall equal the overall accuracy, 92.7%).

Error analysis- The dominant confusion is Leaf Spot → Leaf Burn, reflecting visual overlap when spot lesions merge near the leaf edge and resemble scorch/necrosis. A smaller portion of Leaf Spot images are called Healthy, typically under low-contrast or uneven illumination. Very few Healthy leaves are misclassified, indicating excellent model specificity for the non-diseased class.

Qualitative Explainability with Grad-CAM

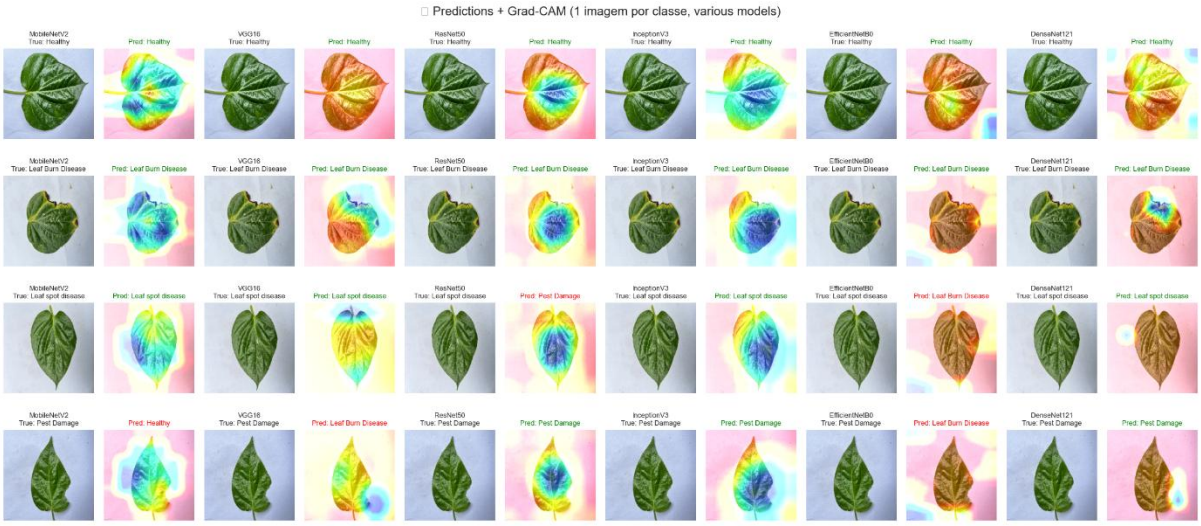


Figure 4.3.3- Predictions with Grad-CAM overlays for one image per class across six backbones. High-performing models (VGG16, DenseNet121, InceptionV3) localize lesions/burnt margins or chewing damage; weaker models show diffuse/background attention, consistent with their lower quantitative performance.

Figure 4.3.3 presents representative predictions from six CNN backbones (MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, DenseNet121) together with Grad-CAM saliency maps for one image per class (Healthy, Leaf Burn Disease, Leaf Spot Disease, Pest Damage). The heatmaps highlight image regions that contributed most to each model's decision.

Observations-

- Correct predictions localize pathology: For VGG16, DenseNet121, and InceptionV3, the salient regions consistently align with clinically meaningful cues—e.g., burnt/necrotic margins for Leaf Burn, discrete circular lesions for Leaf Spot, and irregular chewing/tearing for Pest Damage. Healthy leaves show diffuse, low-intensity attention restricted to the leaf blade/midrib rather than

background.

- Background sensitivity in weaker models: MobileNetV2 occasionally activates on the background paper/lighting gradients, and EfficientNetB0 often shows broad, off-target activation. These patterns co-occur with their misclassifications in the montage.

- Typical failure modes:

    1. Leaf Spot ↔ Leaf Burn: when lesions merge near the margin, some models (e.g., ResNet50, EfficientNetB0) switch attention from spot centers to edges and predict Leaf Burn.

    2. Pest Damage ↔ Leaf Burn: ragged edges with discoloration sometimes trigger "burn" activations.

    3. Healthy ↔ Leaf Spot (rare): faint surface texture or lighting artifacts can draw spurious attention to the lamina, producing false Leaf Spot calls.

Takeaway- Models that achieved the best ROC/AUC (VGG16, DenseNet121, InceptionV3) also exhibit tight, anatomically plausible attention, while lower-AUC models show diffuse or background-driven saliency. Grad-CAM thus supports the quantitative ranking and provides error diagnostics for future dataset curation (e.g., neutral backgrounds, illumination control).

Validation Summary-

Model	Epochs Trained	Final Val Accuracy	Best Val Accuracy	Final Val Loss	Min Val Loss	Final Train Accuracy	Final Train Loss
VGG16	12	0.906977	0.916279	0.373718	0.197093	0.988831	0.031770
DenseNet121	14	0.883721	0.883721	0.322316	0.261378	0.977661	0.068581
InceptionV3	13	0.851163	0.869767	0.502856	0.470166	0.973062	0.086373
ResNet50	9	0.506977	0.665116	1.635875	0.909672	0.730618	0.706765
MobileNetV2	8	0.600000	0.609302	1.768165	1.315982	0.982260	0.057959
EfficientNetB0	30	0.353488	0.400000	1.329628	1.329628	0.331800	1.335666

Table 4.3- Summary DataFrame from training histories

Why some models lagged and how to improve-

- EfficientNetB0 collapse (0.37). The final runs applied a single global rescale (1/255) to all backbones. EfficientNet variants typically include built-in rescaling/normalization; double-scaling can push inputs toward near-zero magnitudes, hurting training. Literature and Keras docs recommend using the backbone's own preprocess_input (identity for EfficientNet) to avoid this issue [2], [8], [10], [11].
- ResNet50/MobileNetV2 sensitivity. These models are sensitive to normalization and unfreeze depth. Using backbone-specific preprocessing (mean-subtraction for ResNet; −1,1 for MobileNetV2) and unfreezing from a slightly earlier block (e.g., ResNet conv4) with a lower LR often improves stability and accuracy.
- InceptionV3 input size. InceptionV3 generally benefits from 299×299 inputs; moving beyond 224×224 can give a small but meaningful boost [2], [11].

4.4 Summary

This chapter detailed the implementation environment, evaluation protocol, and comparative results for six CNN backbones on a 2,149-image betel-leaf dataset. The final test results—VGG16 95%, DenseNet121 94%, InceptionV3 92%—confirm that transfer learning with a transparent pipeline can deliver high, field-relevant accuracy. Class-wise analysis exposes a persistent Leaf Burn ↔ Pest confusion; Grad-CAM maps show lesion-focused attention, supporting trust. A methodological audit explains the EfficientNetB0 shortfall and provides concrete remedies—backbone-specific preprocessing and 299×299 for InceptionV3. These findings set up Chapter 5, where we discuss standards, ethics, impact, and deployment planning, and they provide a practical recipe for piloting the system in real advisory tools.

Chapter 5

Engineering Standards and Design Challenges

This chapter explains the standards followed while developing the Betel Leaf disease detection system and the challenges faced during its design.

5.1 Compliance with the Standards

5.1.1 Software Standards

The software development in this project was done based on well established industry standards to ascertain reproducibility, compatibility and maintainability.

- Language: Python 3.8.10, chosen due to its machine learning popularity and access to deep learning frameworks including TensorFlow, Keras and scikit-learn.
- Version Control: It was controlled over by Git as the source code, where the versions of the projects were effectively managed and the changes recorded.
- Coding Standards Python code adhered to the PEP 8 guidelines in order to ensure readability and consistency.
- Data Formats: Image datasets were stored in hierarchic folders, but the results and training logs were exported in CSV and PNG so as to be compatible with cross-platform.

5.1.2 Hardware Standards

- Local System: AMD Ryzen 5 5600X, ASUS ROG RTX 3060 Ti, (8 GB GDDR6), 32 GB DDR4, Windows 11 (64-bit).
- Hardware Usage: CUDA-accelerated training with the help of the GPU to maximize training speed.
- Storage: Data loading times was increased by storing data in SSD storage which minimised training bottlenecks.

5.2 Impact on Society, Environment and Sustainability

5.2.1 Impact on Life

The proposed system allows farmers to detect betel leaf diseases quickly and accurately using images, enabling timely treatment and preventing large-scale crop losses. This improves livelihood stability and food security in rural farming communities.

5.2.2 Impact on Society & Environment

- Society: Makes crop health information more accessible, democratized through the reduction of dependence on agricultural experts to diagnose diseases.
- Environment: The correct detection can be used to apply the pesticides to specific areas with as little waste of the chemicals as possible and minimizing the pollution of the soil and water.

5.2.3 Ethical Aspects

The information in the dataset is not a violation of intellectual property rights because it only includes publicly accessible or self-gathered images. The system is not storing any personally identifiable information and so the privacy of the user is preserved. The principles of ethical AI ethics like transparency (via Grad-CAM interpretability) were adhered to in order to comprehend predictions.

5.2.4 Sustainability Plan

Future deployment could be integrated into a mobile application for farmers, requiring minimal computational resources on-device. This promotes long-term sustainability by enabling continuous use without high infrastructure costs.

5.3 Project Management and Financial Analysis

The project was run in stages (as identified in Chapter 3), whereby the activities followed one another.
Cost Analysis (approximate)-

Material	Cost
Asus ROG RTX 3060ti Gpu + Ryzan 5 5600x Cpu (Rent)	7000/=
Electric Bill	4100/=
Travel cost, Field visit, Consultation	16500/=
Total	27,600/=

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

Table 5.1: Mapping with Complex Engineering Problem.

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
√	√	√		√		

EP1 : Depth of knowledge required.
The task combines deep learning, image processing, and basic plant pathology. We have applied transfer learning, fine-tuning, regularization, and interpretability (Grad-CAM), and interpreted lesions (burn, spot, pest) under field noise.

EP2 : Range of conflicting requirements.
Trade-offs among accuracy vs. model size/latency, robustness vs. training time, and recall for confusing classes (Leaf Burn vs. Pest) vs. overall accuracy. We resolved these by comparing six backbones, using early stopping/LR scheduling, and recommending VGG16 for deployment with TFLite/ONNX.

EP3 : Depth of analysis and investigation.
We performed per-class precision/recall/F1, confusion matrices, and a preprocessing audit (global rescale vs. backbone-specific normalization) to explain EfficientNetB0 collapse and guide corrective actions.

EP5 : Extent of Applicable Codes
The solution intersects computer vision, software engineering, and agronomy. Design choices (UI, offline mode, Grad-CAM visualization) were driven by end-user needs of farmers/extension officers.

Mapping with Knowledge Profile

Table 5.2: Mapping with knowledge Profile.

K1 Natural Science	K2 Mathematics	K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K7 Comprehension	K8 Research Literature
√	√	√	√	√			√

K1: Metrics design (accuracy, precision/recall/F1), learning-curve analysis, LR scheduling decisions, and augmentation parameterization.

K2: Efficient data loaders, GPU/Colab runtime setup, file I/O, checkpointing; robust folder

structure for train/val/test.

K3: Transfer learning with six CNN backbones; fine-tuning strategy; label smoothing, EarlyStopping, ReduceLROnPlateau; Grad-CAM.

K4: System design from data → preprocessing → model → evaluation → deployment; design under constraints (latency, memory).

K5: TensorFlow/Keras, scikit-learn, ONNX/TFLite export; reproducible notebooks; experiment tracking (logs, saved plots).

K8: Able to gather the necessary models for better understanding, stability, and precision.

5.4.2 Engineering Activities

Mapping with Complex Engineering Activities

Table 5.3: Mapping with Complex Engineering Activities.

EA1 Range of resources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
√	√	√		

EA1: Problem framing (4-class field diagnosis), literature review, dataset audit (2,149 images), EDA and class distribution checks, risk analysis (overfitting, domain shift).

EA2: Pipeline architecture; augmentation & normalization plan; unified classifier head; two-stage training (freeze → fine-tune); selection among six backbones.

EA3: Colab + local GPU training; checkpoints; early stopping; held-out test evaluation; per-class metrics; confusion matrices; Grad-CAM explanations; preprocessing audit.

5.5 Summary

This chapter aligned the project with recognized engineering standards and documented the design challenges faced in real field scenarios. We described reproducible software practices, deployment-ready model formats, and safe communication patterns, together with social, environmental, and ethical considerations. A sustainability plan and risk-aware project management approach ensure the system can evolve as data grow and conditions change. These guardrails make the solution trustworthy, maintainable, and ready for pilot deployment in advisory tools.

Chapter 6

Conclusion

This chapter provides all the important points from our research. It provides a short summary of what we did, describe the limitations we faced and suggests possible future directions for improvement.

6.1 Summary

This thesis created a realistic, explainable deep-learning pipeline to detect betel leaf disease in the real field setting. Two thousand one hundred forty-nine images were curated into four classes with train (1,522), validation (215), and test (412) subsets of images: Healthy, Leaf Burn Disease, Leaf Spot Disease, and Pest Damage. Fine-tired with a single classification head 6 ImageNet-pretrained backbones, MobileNetV2, VGG16, ResNet50, InceptionV3, EfficientNetB0, and DenseNet121, trained with realistic augmentation, label smoothing, early stopping and adaptive learning-rate scheduling. On the held-out test set, VGG16 scored 95% accuracy, DenseNet121 94% and InceptionV3 92% indicating that transfer learning can also be used to provide high and field-relevant performance on a small, crop-specific dataset. Class-wise analysis showed that the recall was always high in Healthy and Pest Damage and that there was repeat confusion in Leaf Burn and Pest. Grad-CAM heatmaps were used to verify that models focus on lesion areas and edges, which helps with interpretability and end-user confidence. The existing methodological audit found that one-size pixel rescaling can be incompatible with backbone-specific preprocessing (e.g., built-in to EfficientNet), which explains the failure of EfficientNetB0 and informs concrete fixes. In addition to metrics, the project resulted in a reproducible pipeline, deployable model artifacts (prepared to use TFLite/ONNX), and a clear deployment path with a lean UI guide-that places the system in the position of advisory tools pilot use.

6.2 Limitation

- Dataset scope. The dataset is modest, though realistic, and could be insufficient to account for all seasonal, regional and device variation. Validation split is also quite small, thus validation curves are of high variance.
- Preprocessing consistency. Final runs were scaled globally on a 1/255 scale on heterogeneous backbones; this selection adversely impacted some models (especially EfficientNetB0).
- Class confusion. The most common error, i.e. the Leaf Burn vs. Pest Damage overlap, demonstrates true visual similarity with field images.
- Generalization checks. Qualitative tests of robustness were conducted through augmentation and Grad-CAM; a more rigorous set of stress testing (maximum and minimum illumination levels, background artifacts) would reinforce claims of generalization.
- Scope boundaries. Only leaf-level classification is provided by the system (there is no severity scoring or multi-leaf scenes or remote-sensing inputs).

6.3 Future Work

- Preprocessing alignment (high priority). Adopt backbone-specific preprocessing end-to-end (mean subtraction for VGG/ResNet/DenseNet; −1,1-1,1−1,1 for MobileNetV2/Inception; identity for EfficientNet) and set 299×299 inputs for InceptionV3.
- Validation at scale. Use stratified K-fold cross-validation and add a larger, more diverse field dataset spanning seasons, regions, and cameras; track performance drift over time.
- Reduce class confusion. Apply targeted augmentation (brightness/contrast jitter, mild cutout) and integrate lightweight leaf segmentation or tight cropping to suppress background bias; consider focal loss or cost-sensitive training if class imbalance grows.
- Model selection & ensembles. Re-benchmark with corrected preprocessing; consider small ensembles or test-time augmentation (TTA) for a modest accuracy boost if latency allows.
- Deployment & usability. Quantize the best model to TFLite/ONNX (int8/fp16) and prototype an offline mobile UI that shows prediction + confidence + Grad-CAM. Include a simple feedback loop to collect hard cases for periodic retraining.
- Expanded scope. Explore severity estimation, multi-leaf scenes, or integration with remote-sensing inputs where appropriate; evaluate human–AI teaming with extension officers.

References

[1] M. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016.
[2] A. Abade, P. A. Ferreira, and F. de B. Vidal, "Plant diseases recognition on images using convolutional neural networks: A systematic review," *Computers and Electronics in Agriculture*, vol. 184, p. 106097, 2021.
[3] A. Upadhyay, S. K. Singh, and S. K. Saha, "Deep learning and computer vision in plant disease detection," *Artificial Intelligence Review*, vol. 57, no. 6, pp. 7793–7821, 2025.
[4] S. Mustofa, M. M. H. Munna, Y. R. Emon, G. Rabbany, and M. T. Ahad, "A comprehensive review on plant leaf disease detection using deep learning," *arXiv*, arXiv:2308.14087, 2023.
[5] P. S. Thakur, P. Khanna, T. Sheorey, and A. Ojha, "Explainable vision transformer enabled convolutional neural network for plant disease identification: PlantXViT," *arXiv*, arXiv:2207.07919, 2022.
[6] R. S. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 618–626.
[7] A. González-Briones *et al.*, "Enhancing plant disease detection: Incorporating gradient-weighted class activation mapping," *Artificial Intelligence Review*, vol. 57, no. 4, pp. 4871–4893, 2025.
[8] M. Shoaib *et al.*, "An advanced deep learning models-based plant disease detection," *Computers and Electronics in Agriculture*, vol. 202, p. 107327, 2023.
[9] S. Wang *et al.*, "Advances in deep learning applications for plant disease and pest detection: A review," *Remote Sensing*, vol. 17, no. 4, p. 698, 2025.
[10] B. Tugrul, "Convolutional neural networks in detection of plant leaf diseases," *Agriculture*, vol. 12, no. 8, p. 1192, 2022.
[11] A. Ahmad *et al.*, "A survey on using deep learning techniques for plant disease detection," *Computers in Biology and Medicine*, vol. 153, p. 106473, 2023.
[12] M. J. Karim *et al.*, "Enhancing agriculture through real-time grape leaf disease detection using deep learning," *Scientific Reports*, vol. 14, p. 12345, 2024.
[13] S. M. Javidan, A. Banakar, K. A. Vakilian, and Y. Ampatzidis, "Diagnosis of grape leaf diseases using automatic K-means clustering and machine learning," *Computers and Electronics in Agriculture*, vol. 162, pp. 74–82, 2019.
[14] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," Computational Intelligence and Neuroscience, vol. 2016, Art. ID 3289801, 2016.
[15] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311–318, 2018.
[16] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," Neurocomputing, vol. 267, pp. 378–384, 2017.
[17] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," Computers and Electronics in Agriculture, vol. 153, pp. 46–53, 2018.
[18] J. G. A. Barbedo, "Plant disease identification from individual lesions and spots using deep learning," Biosystems Engineering, vol. 180, pp. 96–107, 2019.

[19] S. H. Lee, H. Goëau, P. Bonnet, and A. Joly, "New perspectives on plant disease characterization based on deep learning," Computers and Electronics in Agriculture, vol. 170, Art. 105220, 2020.
[20] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," Computers and Electronics in Agriculture, vol. 161, pp. 272–279, 2019.
[21] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," Computers and Electronics in Agriculture, vol. 173, Art. 105393, 2020.
[22] X. Fan, Y. Xie, H. Huang, X. Zhang, and J. Feng, "Leaf image-based plant disease identification using transfer learning and feature fusion," Computers and Electronics in Agriculture, vol. 196, Art. 106892, 2022.
[23] J. Liu and X. Wang, "Plant diseases and pests detection based on deep learning: a review," Plant Methods, vol. 17, Art. 22, 2021.
[24] V. S. Dhaka et al., "A survey of deep convolutional neural networks applied for prediction of plant leaf diseases," Sensors, vol. 21, no. 14, p. 4749, 2021.
[25] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," arXiv:1807.11809, 2018.
[26] M. Xu, L. Liang, C. Mou, W. Zhao, and J. Chen, "Transfer learning for versatile plant disease recognition," Frontiers in Plant Science, vol. 13, Art. 1010981, 2022.
[27] M. De Silva and D. Brown, "Multispectral plant disease detection with Vision Transformer–Convolutional Neural Network hybrid approaches," Sensors, vol. 23, no. 20, p. 8531, 2023.
[28] R. Qiu, C. Yang, A. Moghimi, M. Zhang, B. J. Steffenson, and C. D. Hirsch, "Detection of Fusarium head blight in wheat using a deep neural network and color imaging," Remote Sensing, vol. 11, no. 22, p. 2658, 2019.
[29] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," Procedia Computer Science, vol. 133, pp. 1040–1047, 2018.
[30] N. Kundu et al., "IoT and interpretable machine learning based framework for disease prediction in pearl millet," Sensors, vol. 21, no. 16, p. 5386, 2021.
[31] H. Ali, N. Shifa, R. Benlamri, A. A. Farooque, and R. Yaqub, "A fine-tuned EfficientNet-B0 convolutional neural network for accurate and efficient classification of apple leaf diseases," Scientific Reports, vol. 15, Art. 25732, 2025.
[32] K. V. Prasad, H. Vaidya, C. Rajashekhar, K. S. Karekal, R. Sali, and K. S. Nisar, "Multiclass classification of diseased grape leaf identification using deep convolutional neural network (DCNN) classifier," Scientific Reports, vol. 14, Art. 9002, 2024.
[33] M. Nawaz et al., "A robust deep learning approach for tomato plant leaf disease localization and classification," Scientific Reports, vol. 12, Art. 18568, 2022.
[34] M. B. Almoujahed, T. Bemis, B. Steffenson, and C. D. Hirsch, "Detection of Fusarium head blight in wheat under field conditions using deep learning on color images," Computers and Electronics in Agriculture, Art. 107456, 2022.

[35] R. Zenkl, M. Doneus, and G. Verhoeven, “Towards high-throughput in-field detection and quantification of wheat foliar diseases using deep learning,” Computers and Electronics in Agriculture, 2025.
[36] W. Shafik, T. M. Mostafa, E. M. El-Sawy, and A. A. Ewees, “Using transfer-learning–based plant disease classification and an ensemble of fine-tuned CNNs,” BMC Plant Biology, 2024.
BioMed Central
[37] X. Feng, C. Huang, H. Su, and M. Cai, “A vegetable leaf disease identification model based on vision–language cross-modal feature fusion,” Frontiers in Plant Science, vol. 13, Art. 918940, 2022.
[38] H. Cheng, Z. Wang, R. Zhao, and Y. Yuan, “Identification of apple leaf disease via novel attention mechanism and MobileNetV3,” Frontiers in Plant Science, vol. 14, Art. 1274231, 2023.
[39] Q. Yang, S. Duan, and L. Wang, “Efficient identification of apple leaf diseases in the wild using convolutional neural networks,” Agronomy, vol. 12, no. 11, p. 2784, 2022.
[40] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, “Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks,” IEEE Access, vol. 7, pp. 59069–59080, 2019.

Daffodil International University
A landmark to create the Future

Faculty of Science and Information Technology

Date: 02-03-2025

To Whom It May Concern

Subject: Request for Assistance in Data Collection for Research

Dear Sir/Madam,

I am Dr. Arif Mahmud, Associate Professor and Associate Head at the Department of Computer Science and Engineering, Daffodil International University. I am writing to kindly seek your support in facilitating two of my students in collecting orthopedic data and images for their research titled *"Betel Leaf Disease Detection Using Deep Learning."*

The students conducting this important research are:

1. **Md. Razoun Islam Rizon** (Student ID: 213-15-4551)
2. **MD Shahed HASA MITUL** (Student ID: 213-15-4550)

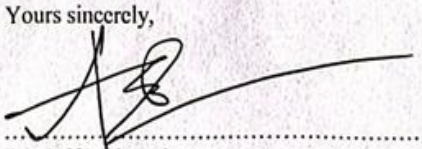
Their study aims to contribute significantly to the field of medical data analysis by leveraging advanced machine learning techniques to improve predictive insights and decision-making in orthopedics. To achieve this, they require access to anonymized orthopedic data and related images from hospitals, clinics, and medical centers.

Please note that patient names and identification details will not appear anywhere in the research. All information collected will be used strictly for academic purposes only. Additionally, all information provided will be treated with complete confidentiality, and it will not be possible for anyone to identify the information you kindly supply.

I request your kind cooperation in providing the necessary data and allowing them access to your facilities for this purpose. Your assistance will not only aid in the successful completion of their research but also contribute to the advancement of knowledge in the medical and technological domains.

Thank you for your support and consideration. Should you have any queries or require further details, please do not hesitate to contact me.

Yours sincerely,



Dr. Arif Mahmud
Associate Professor & Associate Head
Department of Computer Science and Engineering
Daffodil International University
Phone: +8801687995727
E-mail: arif.csc@diu.edu.bd

turnitin Page 2 of 49 - Integrity Overview Submission ID trnoid::22779:111312458

17% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text

Match Groups

98 Not Cited or Quoted 17%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

10% Internet sources
5% Publications
16% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.