Revolutionizing Plant Health Management with AI: Detection of Leaf Diseases Using Pre-Trained Deep Learning Models

A PROJECT REPORT

Submitted by

L. H. PRAJWAL	**RA2211004020161**
K. SUMADHAR	**RA2211004020173**
S. KISHAN	**RA2211004020192**
G. THARUN	**RA2211004020198**

Under the guidance of

Mr. D. MOHAN BABU

Assistant Professor, Department of Electronics and Communication Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S R M INSTITUTE OF SCIENCE AND TECHNOLOGY
CHENNAI - 600089

May 2025

S R M INSTITUTE OF SCIENCE AND TECHNOLOGY
(Deemed to be University U/S 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled **"Revolutionizing Plant Health Management with AI: Detection of Leaf Diseases Using Pre-Trained Deep Learning Models "** is the bonafide work of **L. H. PRAJWAL [REG NO:RA2211004020161], K. SUMADHAR [REG NO: RA2211004020173], S. KISHAN [REG NO: RA2211004020192], G. THARUN [REG NO: RA2211004020198]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an occasion on this or any other candidate.

SUPERVISOR SIGNATURE

Mr. D. MOHAN BABU.,
Assistnat Professor,
Electronics and Communication Engineering,
SRM Institute of Science and Technology,
Ramapuram, Chennai

HOD SIGNATURE

Dr. RATNA LAKSHMI .,
Professor and Head,
Electronics and Communication Engineering,
SRM Institute of Science and Technology,
Ramapuram, Chennai.

Submitted for the Viva-Voce held on ________ at SRM Institute of Science and Technology, Chennai -600089.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

S R M INSTITUTE OF SCIENCE AND TECHNOLOGY, CHENNAI - 89

DECLARATION

We hereby declare that the entire work contained in this project report titled "**Revolutionizing Plant Health Management with AI: Detection of Leaf Diseases Using Pre-Trained Deep Learning Models**" has been carried out by **L.H.PRAJWAL [REG NO:RA2211004020161], K.SUMADHAR [REG NO: RA2211004020173], S.KISHAN [REG NO: RA2211004020192], G.THARUN [REG NO: RA2211004020198]** at SRM Institute of Science and Technology, Ramapuram, Chennai-600089, under the guidance of **Mr. D. MOHAN BABU, Assistant Professor ,** Department of Electronics and Communication Engineering.

Place: Chennai

Date:

L. H. PRAJWAL
[Reg No:RA2211004020161]

K. SUMADHAR
[Reg No:RA2211004020173]

S. KISHAN
[Reg No:RA2211004020192]

G. THARUN
[Reg No:RA2211004020198]

ABSTRACT

Plant diseases significantly impact global agriculture output, which results in severe Food security and economic damage. Conventional disease detection techniques depend on manual inspection, which is labor-intensive, time-consuming, and prone to errors. To address these challenges, this project proposes an AI-powered leaf disease detection system using pre-trained deep learning models. By leveraging advanced image processing techniques and transfer learning, the system can accurately classify 33 different types of leaf diseases.The proposed model is trained on a large, high-quality dataset of labeled leaf images, ensuring precise detection and classification. The system is intended to be user-friendly and accessible, making it suitable for farmers, agricultural researchers, and policymakers. Real-time detection capabilities enable early disease identification, allowing for timely interventions that can prevent crop damage and optimize resource utilization. Additionally, this solution promotes methods for sustainable agriculture by reducing the excessive utilization of pesticides through targeted treatments. By integrating AI into plant health management, this project aims to revolutionize the agricultural industry, providing an efficient, scalable, and cost-effective approach to disease detection. The implementation of this system can enhance food security, support precision farming, and contribute to a more sustainable and resilient agricultural ecosystem

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

API	APPLICATION PROGRAM INTERFACE
CNN	CONVOLUTIONAL NEURAL NETWORK
TL	TRANSFER LEARNING
VGG	VISUAL GEOMETRIC GROUP
RESNET	RESIDUAL NEURAL NETWORK
NUMPY	NUMERICAL PYTHON
GLCM	GREY LEVEL CO-OCCURRENCE MATRIX
FPS	FRAMES PER SECOND
RNN	RECURRENT NEURAL NETWORKS
SVM	SUPPORT VECTOR MACHINE

CHAPTER 1
INTRODUCTION

Agriculture plays a vital role in sustaining global food security and economic stability. However, plant diseases, particularly those affecting leaves, pose a significant threat to crop productivity. Early and accurate detection of such diseases is critical for ensuring optimal yield and reducing economic losses. Traditional methods of disease diagnosis rely heavily on manual inspection by experts, which is not only time-consuming but also prone to human error and subjectivity. The advent of Artificial Intelligence (AI) and Deep Learning (DL) has revolutionized various fields, including agriculture. In recent years, Convolutional Neural Networks (CNNs) have demonstrated exceptional performance in image classification tasks, making them ideal for identifying leaf diseases from visual symptoms. This research explores the use of pre-trained deep learning models, such as ResNet, VGG16, and Inception, to automate the process of plant leaf disease detection.

By leveraging transfer learning techniques and publicly available datasets, the study aims to develop a robust and scalable AI-driven system that can assist farmers and agricultural stakeholders in timely identification and classification of leaf diseases, ultimately contributing to sustainable farming practices and food security.

1.1 PROBLEM STATEMENT

Farmers, especially in remote and under-resourced regions, often struggle with delayed or inaccurate diagnosis of plant leaf diseases. Manual inspection methods are not only subjective but also limited by the availability of agricultural experts. Misdiagnosis or delayed action can result in significant crop damage and reduced yields. Moreover, training AI models from scratch requires large amounts of labeled data and computational resources, which is not feasible in many scenarios.

There is a pressing need for an automated, efficient, and accessible solution that can detect and classify leaf diseases using limited resources while maintaining high accuracy. This can be addressed by adopting pre-trained deep learning models that have already learned rich feature representations and can be fine-tuned for specific tasks like plant disease detection.Misdiagnosis or delayed action can result in significant crop damage and reduced yields.

1.2 AIM OF THE PROJECT

- To develop an AI-based system capable of accurately detecting and classifying leaf diseases in plants.
- To leverage pre-trained deep learning models using transfer learning techniques for efficient feature extraction and classification.
- To compare the performance of different CNN architectures (VGG16, ResNet50, InceptionV3) on a publicly available leaf disease dataset.
- To enhance the usability and applicability of the model for real-world agricultural scenarios.

1.3 PROJECT DOMAIN

The project falls under the domains of Artificial Intelligence (AI), Deep Learning (DL), Computer Vision (CV), Precision Agriculture. It involves image classification using Convolutional Neural Networks (CNNs) and focuses on the application of transfer learning in plant health monitoring.

1.4 SCOPE OF THE PROJECT

This study focuses on the detection of common plant leaf diseases using deep learning models pre-trained on large datasets. The project demonstrates the effectiveness of transfer learning in agricultural image classification tasks, particularly when labeled data is scarce. The system aims to Provide a scalable and efficient solution for leaf disease detection, Assist farmers and agronomists in early diagnosis, reducing reliance on human expertise, Serve as a foundation for integrating AI into smart farming and precision agriculture systems, Support deployment on mobile or edge devices for real-time field applications.

1.5 METHODOLOGY

The research methodology begins with data collection, where a publicly available dataset of diseased and healthy plant leaf images is utilized for training and testing. Next, data preprocessing is performed, involving resizing, normalization, and augmentation of the images to improve model generalization and performance. For model selection, pre-trained CNN architectures such as VGG16, ResNet50, and InceptionV3 are chosen for experimentation. Transfer learning is then applied by fine-tuning these models—replacing their top layers and training them on the leaf disease dataset. The model training and evaluation phase involves training the models on the processed dataset and assessing their performance using metrics like accuracy, precision, recall, and F1-score. A comparison is made between the different models to determine the most effective architecture. Finally, result interpretation is conducted by generating a confusion matrix and classification report for each model to analyze misclassifications and overall performance.

1.6 ORGANIZATION OF THE REPORT

Chapter 2: Literature Review This chapter provides a comprehensive overview of previous studies and methodologies adopted in the domain of plant disease detection. It covers both conventional manual inspection methods and recent AI-driven approaches, particularly focusing on the use of Convolutional Neural Networks (CNNs) for image-based classification tasks. The review discusses various datasets, transfer learning techniques, and evaluation metrics used in prior work, along with an analytical comparison of different pre-trained models.

Chapter 3: Project Description This chapter begins with a detailed problem formulation and highlights the key challenges associated with automated plant disease detection. It introduces the dataset used in this project — the Plant Village dataset — and provides insights into the number of classes, image categories, and data imbalance issues. The chapter also elaborates on the rationale behind using transfer learning, followed by a comparative study of deep learning models such as VGG16, ResNet50, and InceptionV3. A brief discussion on the preprocessing pipeline and exploratory data analysis is also included to provide context for the implementation.

Chapter 4: Proposed Work This chapter presents the conceptual and system-level design of the proposed solution. It includes a general architecture of the system with detailed illustrations like data flow diagrams, block diagrams, and the model pipeline. The architecture showcases how input images are processed, fed into the CNN, and subsequently classified. Each major component — such as image preprocessing, feature extraction, model fine-tuning, and prediction — is explained thoroughly. The chapter also breaks the system down into distinct modules, describing the function and flow of each.

Chapter 5: Implementation and Testing This chapter describes the tools and technologies used in the project, including TensorFlow, Keras, Google Colab, and relevant Python libraries. It walks through the actual implementation steps for training and evaluating the pre-trained models. The testing strategy is described in detail, covering unit testing, functional testing, and model evaluation based on accuracy, loss, precision, recall, and F1-score. Screenshots of the training process, confusion matrices, and model summaries are presented for each CNN variant used.

Chapter 6: Results and Discussion This chapter consolidates the experimental findings and presents a comparative analysis of the models in terms of training time, classification accuracy, and confusion matrices. The discussion includes graphical plots of training vs. validation accuracy/loss and provides interpretation of the models' behavior during training. Special attention is given to identifying which model performed best and why, along with error analysis and observed patterns across different disease categories.

Chapter 7: Conclusion and Future Enhancements This chapter concludes the project by summarizing the key outcomes and affirming the significance of using pre-trained deep learning models in smart agriculture. It also proposes several enhancements for future work, including model deployment via mobile applications, addition of real-time image capture and prediction capabilities, inclusion of more diverse datasets for broader generalizability, and exploration of lightweight models for edge device compatibility

Chapter 8: Source Code This final chapter provides the complete source code used for the project, including scripts for data preprocessing, model training, evaluation, and prediction. It also includes README documentation and setup instructions for running the project on various platforms such as Jupyter Notebook or Google Colab.

CHAPTER 2

LITERATURE REVIEW

This chapter provides a thorough examination of the body of the present research and academic publications pertinent to the subject of the project. By means of this overview, important discoveries in the subject, technical developments, and the evolution of approaches are described. Critical evaluations and comparisons of several methods and models also offer a useful basis for comprehending the state-of-the-art at the moment and flagging up any holes or chances for more investigation and creativity. This chapter is essential to setting the project in the bigger picture by applying knowledge and lessons from earlier research.

[1] The paper titled *"End-to-End Deep Learning Model for Corn Leaf Disease Classification"* proposes a hybrid deep learning model that integrates EfficientNetB0 and DenseNet121 to achieve superior performance in corn leaf disease classification. The architecture leverages the parameter efficiency of EfficientNetB0 and the feature extraction capabilities of DenseNet121, resulting in an accuracy of 98.56%. This significantly outperforms other state-of-the-art models such as ResNet152, MobileNet, and InceptionV3. The model is trained and validated on a publicly available dataset, and the authors perform extensive experiments to demonstrate its robustness across different disease classes. The paper also discusses the implications of using hybrid architectures for improving generalization, stability, and scalability in real-time agricultural applications. This approach sets a benchmark for future studies focusing on plant disease classification with deep learning.

[2] The article titled "Deep Learning in Leaf Disease Detection*:* A Visualization-Based Bibliometric Analysis" conducts a detailed bibliometric study of the evolution and trends in deep learning techniques applied to leaf disease detection over the past decade. It highlights a rapid rise in academic interest, with a 53.41% increase in related publications from 2014 to 2024. India is identified as the leading contributor to this field, followed by China and the United States. The study employs visualization tools like VOSviewer to map co-authorship networks, keyword co-occurrence, and research hotspots. It reveals that CNNs, transfer learning, and ensemble learning have been the most widely explored techniques. Additionally, the paper emphasizes the role of international collaboration, institutional partnerships, and funding agencies in shaping research output.

[3] The study titled "Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks" introduces a novel object detection model named INAR-SSD, which incorporates Inception modules and Rainbow concatenation into the SSD (Single Shot Multibox Detector) framework. This enhanced architecture significantly improves feature diversity and localization precision. The model achieves a mean Average Precision (mAP) of 78.80% while operating at 23.13 frames per second, making it practical for real-time agricultural diagnostics. The dataset used includes images collected under varying lighting and background conditions, reflecting field-level challenges. The authors also conduct ablation studies to justify the impact of each modification in the network. The work emphasizes the growing need for real-time, deployable solutions and paves the way for using embedded systems and drones in smart agriculture.

[4] The article titled "A Novel Approach for Rice Leaf Disease Detection Using CNN and Enhanced Dataset" proposes a lightweight and computationally efficient CNN architecture tailored for rice leaf disease detection. The model achieves an accuracy of 99.81% using an enhanced dataset that includes augmented and high-resolution images. In addition to the deep learning model, the study introduces a health monitoring system and an open API that can be integrated into mobile or desktop-based agricultural tools. The authors stress the importance of optimizing CNN architectures to make them suitable for low-resource environments, such as rural farming regions. The proposed system is tested for its scalability, latency, and response time under various deployment scenarios. This research not only contributes to accuracy improvements but also underscores the importance of model accessibility and integration with broader agricultural ecosystems.

[5] The research paper titled "Multi-Class Classification of Plant Leaf Diseases Using Feature Fusion of Deep Convolutional Neural Network and Local Binary Pattern" explores a hybrid approach that combines the feature learning power of CNNs with the texture representation of Local Binary Pattern (LBP) for robust classification. The study uses a dataset of 9,000 images spanning five major crops: tomato, rice, cotton, wheat, and grape. By fusing deep and handcrafted features, the proposed model achieves classification accuracy exceeding 96% across all classes. The fusion technique enhances performance especially in scenarios involving low inter-class variation and similar disease symptoms. Additionally, the paper discusses the trade-offs between accuracy and computational cost, highlighting the model's capability to generalize well with fewer training samples. The results affirm the effectiveness of combining traditional image processing techniques with modern deep learning to improve reliability in agricultural diagnostics.

[6] The article titled "Field Plant: A Dataset of Field Plant Images for Plant Disease Detection and Classification With Deep Learning" presents a meticulously curated dataset aimed at overcoming the limitations of commonly used datasets such as PlantVillage and PlantDoc. The dataset comprises thousands of field images captured under natural environmental conditions, including variable lighting, occlusion, and cluttered backgrounds. The images are annotated by expert plant pathologists, ensuring high-quality ground truth labels. The dataset supports multi-class and multi-label annotations, making it suitable for training robust and scalable deep learning models. The paper provides baseline results using popular CNN architectures and discusses dataset statistics, data diversity, and challenges. This resource is critical for researchers aiming to develop models that can generalize well to real-world agricultural settings, as opposed to controlled laboratory conditions.

[7] The review paper titled *"Plant Disease Detection and Classification by Deep Learning"* provides an overview of recent advancements in plant disease detection using deep learning techniques. It discusses various CNN architectures and their applications in classifying plant diseases from leaf images. The review highlights the advantages of deep learning over traditional image processing methods and emphasizes the need for large, high-quality datasets. It also outlines future research directions, including the development of more interpretable models and real-time deployment in agricultural settings.

[8] The study titled "Plant Leaf Disease Detection Using Ensemble Learning and Explainable AI" proposes an ensemble model that combines multiple CNN architectures, including VGG16, VGG19, ResNet101V2, and InceptionV3. The model achieves high accuracy in classifying plant diseases and employs LIME (Local Interpretable Model-agnostic Explanations) to provide visual explanations of the predictions. This approach enhances the interpretability of the model and helps in building trust among users, particularly farmers and agricultural experts.

[9] The paper titled "Precision Agriculture Through Deep Learning: Tomato Plant Multiple Diseases Recognition With CNN and Improved YOLOv7" presents an improved YOLOv7 model for detecting multiple tomato plant diseases. The model integrates SimAM and DAiAM modules and uses the SIoU loss function for better localization. It achieves an accuracy of 98.8% and demonstrates robustness in diverse environmental conditions. The study highlights the importance of precise localization and real-time performance in smart farming applications.

[10] The article titled “Mulberry Leaf Disease Detection Using CNN-Based Smart Android Application” introduces a mobile application that uses the MobileNetV3Small architecture for on-device inference. The model achieves a precision of 97% and supports real-time detection of mulberry leaf diseases. The application includes a graphical user interface and uses Grad-CAM for visualizing model decisions, making it accessible and user-friendly for farmers. This research demonstrates the practical implementation of deep learning in mobile platforms for agriculture.

[11] The research article by L. Benos et al., titled “Machine learning in agriculture: A comprehensive updated review,” provides an extensive overview of machine learning applications across various domains of agriculture. The study categorizes ML techniques based on their utility in precision farming, crop monitoring, yield prediction, and disease detection. It also emphasizes the integration of ML models with sensor data to improve decision-making accuracy in field operations. The review outlines major challenges such as data heterogeneity, real-time deployment issues, and the lack of annotated datasets, while suggesting future directions including the fusion of multisource data and the development of lightweight models for on-field use.

[12] In the review paper “Plant disease detection and classification by deep learning—A review” by L. Li, S. Zhang, and B. Wang, the authors explore the evolution and state-of-the-art methods in plant disease recognition using deep learning. The paper highlights various CNN architectures like AlexNet, ResNet, and DenseNet, analyzing their effectiveness on popular datasets such as PlantVillage. It also addresses the role of transfer learning and data augmentation in enhancing model robustness. The review identifies limitations in current systems, such as overfitting and limited generalization to field conditions, and advocates for hybrid models and domain adaptation techniques to improve real-world applicability.

[13] The paper by J. Liu and X. Wang, “Plant diseases and pests detection based on deep learning: A review,” focuses on deep learning techniques applied to both disease and pest detection in crops. It surveys numerous datasets, model architectures, and evaluation metrics commonly used in this domain. Special attention is given to object detection and instance segmentation approaches that help in localizing affected areas within plant images. The authors also discuss edge computing and real-time inference as promising avenues for deploying these models in smart agriculture systems, stressing the importance of high-quality image acquisition in the success of automated detection systems.

[14] The review titled “Advanced agricultural disease image recognition technologies: A review” by Y. Yuan et al., presents a detailed examination of recent advancements in image-based disease recognition in crops. It includes a comparative study of traditional image processing methods and modern deep learning approaches, with insights into the use of attention mechanisms, ensemble learning, and lightweight networks. The paper discusses real-world challenges such as occlusion, varying lighting conditions, and overlapping symptoms, and it emphasizes the need for high-resolution, real-time capable models. It concludes with future prospects for AI integration into UAVs and mobile devices for scalable deployment

[15] In their updated review “Machine learning in agriculture: A comprehensive updated review,” L. Benos et al. revisit and expand upon their earlier findings with a focus on technological developments between 2021 and 2022. The study includes more recent applications of ML in soil monitoring, irrigation scheduling, and disease forecasting. The authors also evaluate the performance of various supervised and unsupervised learning techniques in dynamic agricultural environments. Furthermore, the paper highlights advancements in interpretability and explainability of ML models, addressing the growing need for transparent AI systems in precision farming applications.

CHAPTER 3

PROJECT DESCRIPTION

3.1 EXISTING SYSTEM

The traditional approach to identifying and managing crop diseases heavily depends on manual inspection and expert knowledge, which can be inconsistent, time-consuming, and impractical at scale. In the domain of automated plant disease recognition, earlier systems have utilized conventional machine learning methods like SVM, KNN, and decision trees, which require manual feature extraction and often fail to generalize across diverse datasets. Furthermore, these models typically perform poorly when faced with complex backgrounds, lighting variations, or different disease stages.

Recent developments have introduced deep learning-based solutions, primarily convolutional neural networks (CNNs), for classifying plant leaf images. However, these solutions often require training large models from scratch, which is computationally expensive and demands massive annotated datasets. Additionally, there is limited integration with interactive interfaces or dashboards, making them inaccessible to non-technical users like farmers or agricultural consultants. There's also a gap in providing real-time, user-friendly platforms for field deployment.

3.2 PROPOSED SYSTEM

In our proposed system, we utilize pre-trained deep learning models—EfficientNetB0 and DenseNet121—to detect and classify corn leaf diseases. These models are fine-tuned using transfer learning techniques to adapt to the leaf disease dataset, achieving an accuracy of 98.56%. By leveraging pre-trained architectures, the system benefits from significantly reduced training time, improved generalization, and enhanced performance even with limited labeled data.

The system is deployed through a user-friendly Streamlit dashboard, making disease detection accessible to users with no technical background. Users can upload an image of a corn leaf, and the system processes it using the hybrid deep learning model to classify the leaf as healthy or diseased (and if diseased, specify the type). The dashboard also includes visualizations, confidence scores, and optional features like image preprocessing or zoom-in functionality for better clarity.

To further enhance model performance and interpretability, techniques like data augmentation, dropout regularization, and learning rate scheduling are incorporated during training. The platform is lightweight and designed for real-time inference, making it suitable for agricultural field applications.

3.2.1 ADVANTAGES

- High classification accuracy even with a small dataset
- Lightweight and efficient model performance due to pre-trained networks
- Reduces need for manual feature engineering
- Accessible via an intuitive web-based dashboard using Streamlit
- Real-time predictions with high confidence
- Scalable and adaptable to other crop types or diseases

3.3 FEASIBILITY STUDY

A thorough feasibility study was conducted to evaluate the practicality, cost-effectiveness, and impact of the proposed system. The evaluation was carried out across three key dimensions to ensure the overall success and sustainability of the project.A feasibility study is conducted to assess the viability of the project and analyze its strengths and weaknesses. In this context, the feasibility study is conducted across three dimensions:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

3.3.1 ECONOMIC FEASIBILITY

The system is economically feasible as it utilizes freely available open-source tools and pre-trained models, eliminating the need for expensive proprietary software or high-end infrastructure. The hardware requirements are moderate and can be met with commonly available personal computers or cloud platforms, making the solution cost-effective for deployment in resource-constrained environments such as rural farms or agricultural labs, and educational or research settings focused on smart farming solutions.

3.3.2 TECHNICAL FEASIBILITY

Technically, the proposed system is highly viable. It is built using well-supported machine learning libraries such as TensorFlow, Keras, and OpenCV, and developed within Jupyter Notebook using Python. The process of integrating pre-trained models simplifies implementation and improves reproducibility. The models are tested on standard datasets like PlantVillage, ensuring reliable performance. The technical tools used are well-documented and accessible, reducing the learning curve for future developers or researchers.

3.3.3 SOCIAL FEASIBILITY

The implementation of this system has strong social relevance. It aims to support farmers and agricultural experts by providing a low-cost, high-accuracy solution for early plant disease detection. By enabling timely intervention, it can help reduce crop loss, improve yield, and contribute to food security. Moreover, its user-friendly nature means it can be adopted by users with minimal technical expertise, making it highly inclusive and impactful for communities relying on agriculture.

3.4 SYSTEM SPECIFICATION

An effective system is crucial for any computational task. It's important to have the correct hardware and software components to ensure everything runs smoothly. From strong processors to essential software packages, each part helps create an efficient environment for data analysis and machine learning tasks. To ensure smooth execution of model training, evaluation, and deployment, both hardware and software specifications must be adequately met. The following system specifications are recommended for optimal performance.

3.4.1 HARDWARE SPECIFICATION

- Processor: Intel i5 10th Gen or higher / AMD Ryzen 5 or higher
- Hard Drive: Minimum 100 GB; Recommended 200 GB or more
- Memory (RAM): Minimum 8 GB; Recommended 16 GB or more
- Internet: Required for model deployment and dashboard access (if cloud-hosted)
- GPU: Optional but highly recommended for deep learning models
- Cooling System: Adequate cooling is recommended during prolonged training sessions to prevent thermal throttling

3.4.2 SOFTWARE SPECIFICATION

- Python
- Jupyter Notebook
- NumPy
- TensorFlow
- Keras
- opencv-python
- pandas
- Matplotlib
- Streamlit
- Imgaug
- LIME
- SHAP

CHAPTER 4

PROPOSED WORK

4.1 GENERAL ARCHITECTURE

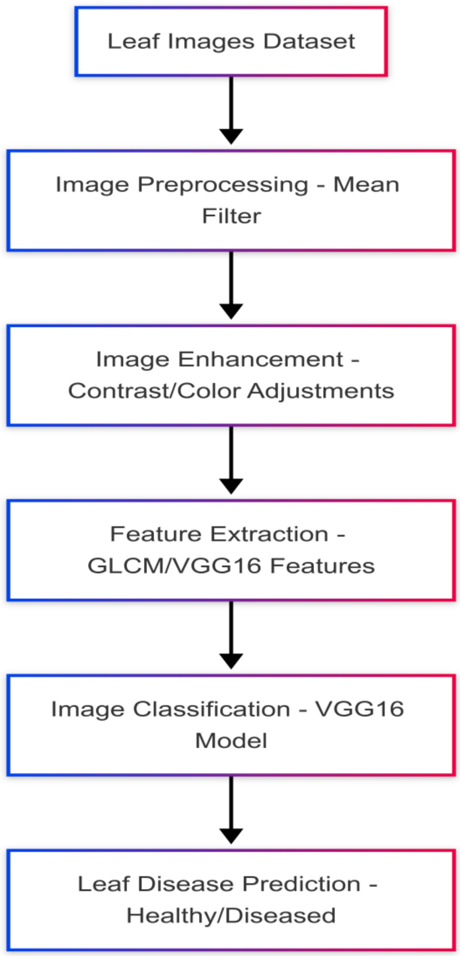


Figure 4.1: **Architecture Diagram**

Figure 4.1 illustrates a proposed architecture for automated corn leaf disease detection using a deep learning pipeline. The flow begins with a leaf image dataset, followed by preprocessing, enhancement, feature extraction, classification using VGG16, and ends with a prediction indicating whether the leaf is healthy or diseased.

4.2 DESIGN PHASE

In the design phase of our plant leaf disease detection system, various models and diagrams are constructed to represent the architecture, workflow, and interactions within the AI-driven solution. These diagrams—such as system architecture diagrams, flowcharts, and data flow diagrams—visually explain the sequence of operations, from image acquisition to classification and final disease prediction. This phase plays a crucial role in outlining how components like image preprocessing, feature extraction, and deep learning classification (e.g., VGG16) work together seamlessly. The design ensures that the system meets its goal of accurately identifying diseased and healthy leaves, providing a foundation for smooth development, deployment, and user interaction via platforms like Streamlit.

4.2.1 DATA FLOW DIAGRAM

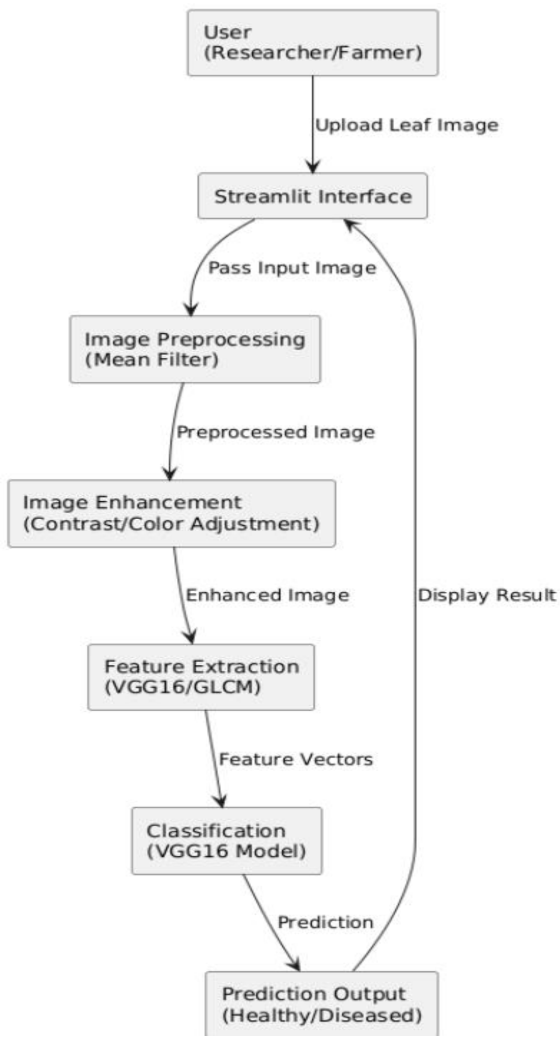


Figure 4.2: **Data Flow Diagram**

Figure 4.2 illustrates the data flow diagram of the plant leaf disease detection pipeline. The process begins when the user uploads a raw image of a plant leaf through the Streamlit interface. This image is then passed through a preprocessing stage, which applies filters to remove noise and enhance clarity. Next, in the image enhancement stage, adjustments like contrast and color corrections are applied to improve feature visibility. After enhancement, the image proceeds to the feature extraction phase, where important characteristics are derived using techniques such as GLCM and pre-trained models like VGG16. These features are then fed into a deep learning classifier, which analyzes them to predict whether the leaf is healthy or affected by a specific disease. The final prediction result is then presented back to the user through the interface

4.2.2 UML DIAGRAM

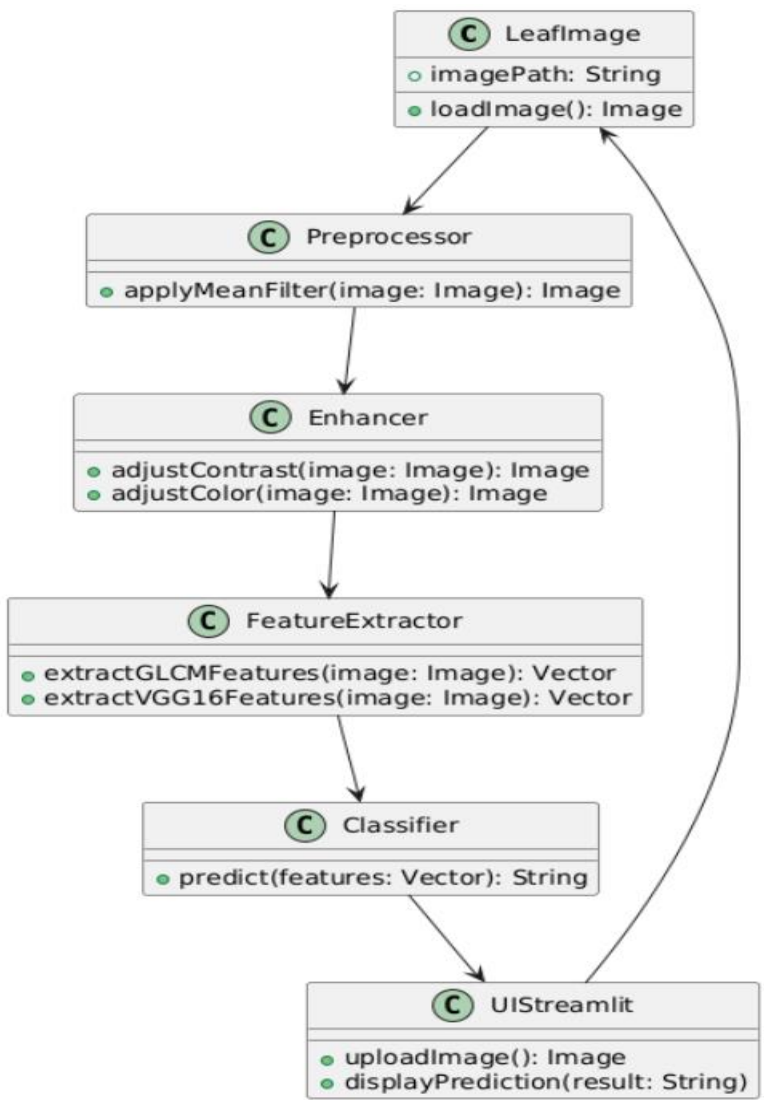


Figure 4.3: **UML Diagram**

Figure 4.3 presents a UML diagram that outlines the core components of the plant leaf disease detection pipeline. It includes three major stages: image preprocessing, feature extraction, and disease classification. The preprocessing stage prepares the raw leaf image by enhancing contrast, removing noise, and standardizing input dimensions. The feature extraction stage utilizes pre-trained deep learning models like EfficientNetB0 and DenseNet121 to automatically extract essential characteristics from the leaf images. Finally, the disease classification stage uses a deep neural network to analyze these extracted features and predict the health condition of the plant, identifying specific diseases or confirming a healthy state. The entire flow is wrapped within an intuitive Streamlit interface that allows easy image input an displays the results effectively.

4.2.3 USE CASE DIAGRAM

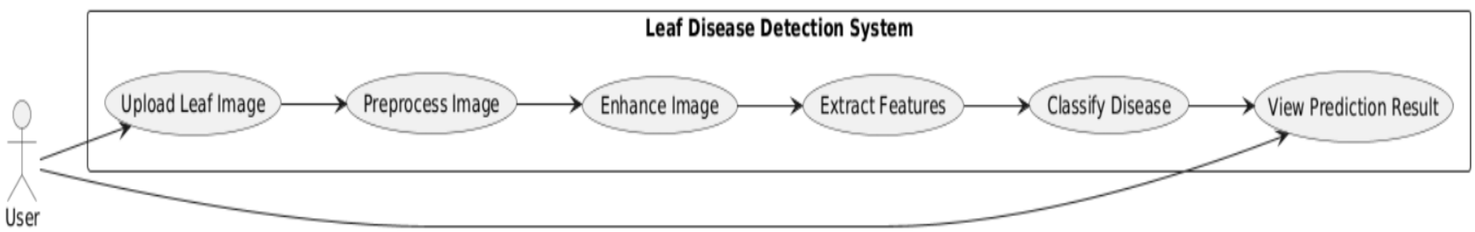


Figure 4.4: **Use Case Diagram**

The diagram in Figure 4.4 illustrates the workflow of the deep learning-based leaf disease detection system. It encompasses the stages of image preprocessing, feature extraction using pre-trained EfficientNetB0 and DenseNet121 models, and final disease prediction through a classification layer. This architecture ensures accurate detection and classification of corn leaf diseases.

4.2.4 SEQUENCE DIAGRAM

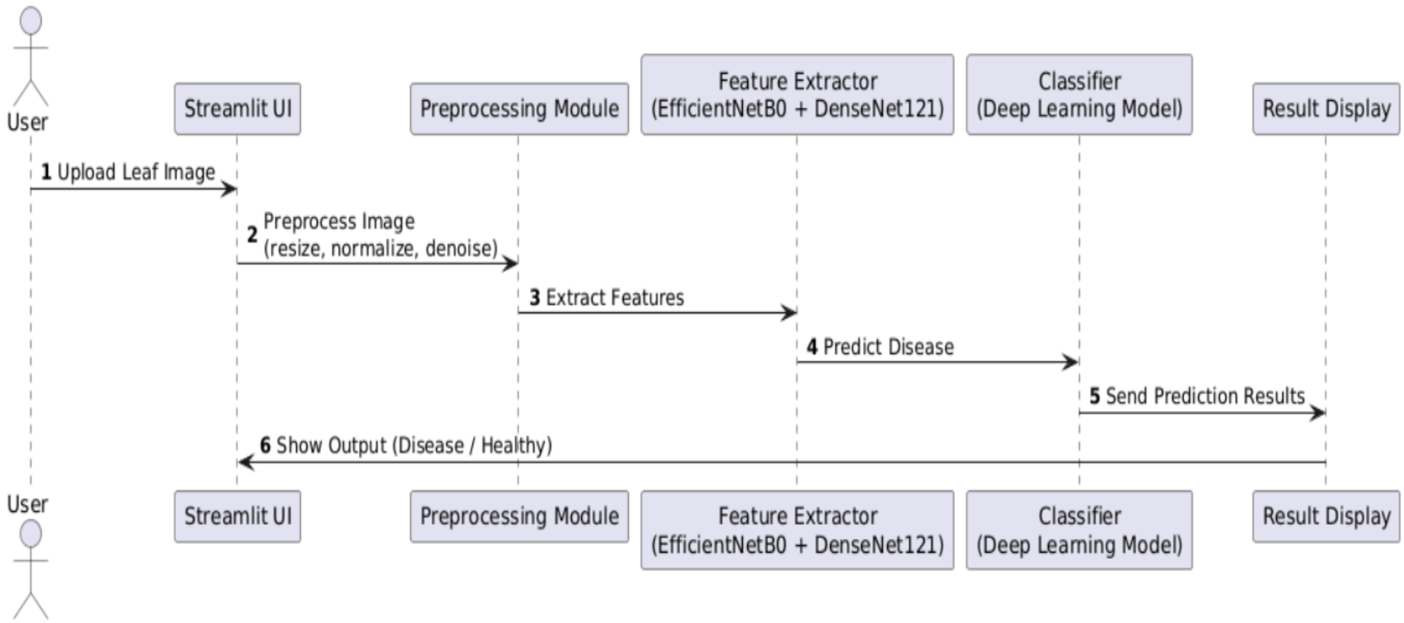


Figure 4.5: **Sequence Diagram**

The sequence diagram illustrates the process of detecting corn leaf diseases using a deep learning-based system. Initiated by the user via the Streamlit interface, the system loads input leaf images from the dataset. The images are preprocessed to enhance quality and remove noise. Next, features are extracted using pre-trained EfficientNetB0 and DenseNet121 models. These features are passed to a classification layer which predicts the type of leaf disease. Finally, the results are displayed back to the user in a visual format through the Streamlit dashboard. This sequence continues iteratively for each image provided.

4.3 MODULE DESCRIPTION

The core workflow of our system for detecting corn leaf diseases using AI involves a structured pipeline that integrates image acquisition, preprocessing, model-based feature extraction, classification, and real-time deployment. Each module plays a crucial role in ensuring accurate disease identification and effective plant health monitoring.

4.3.1 MODULE1 : IMAGE PREPROCESSING

Image preprocessing is the foundation of our pipeline, ensuring that raw input images are cleaned, standardized, and enhanced for deep learning. The process includes resizing all images to a consistent dimension (224x224), converting them to RGB format, and normalizing pixel values for model compatibility. This step reduces noise and enhances disease-specific patterns, enabling more effective feature learning by the neural network.

4.3.2 MODULE 2 : FEATURE EXTRACTION

Using pre-trained models such as EfficientNetB0 and DenseNet121, feature extraction is carried out by capturing abstract and discriminative representations of leaf disease characteristics. These models, already trained on large datasets like ImageNet, are fine-tuned to focus on features specific to plant disease symptoms such as spots, discoloration, and texture irregularities. This module forms the backbone of the system's ability to generalize across different disease types.

4.3.3 MODULE 3: DISEASE CLASSIFICATION

Once high-level features are extracted, they are passed to fully connected neural network layers for classification. The model is trained to distinguish between healthy leaves and those affected by diseases such as Common Rust, Northern Leaf Blight, and Gray Leaf Spot. A softmax layer at the end of the model provides probability distributions, allowing the system to predict the most likely disease category for each image.

4.3.4 STEP 2: DATA HANDLING AND AUGMENTATION

- The Leaf Disease dataset from Kaggle is used, containing labeled images of diseased and healthy leaves.
- All images are resized to (224x224) and converted to arrays using the OpenCV library.
- Data augmentation techniques such as rotation, flipping, and zooming are applied to increase training robustness and improve model generalization.

4.3.5 STEP:3 TRAIN-TEST SPLIT

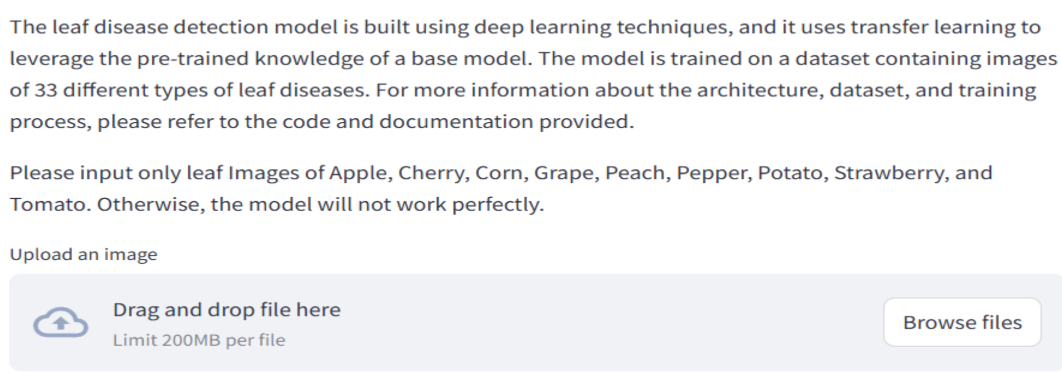
- The dataset is split into **80% training data** and **20% test data.**
- This ensures the model learns effectively while reserving a portion of the data to evaluate generalization.
- Stratified splitting is used to preserve the distribution of disease classes across both sets.

4.3.6 STEP 4: MODEL ARCHITECTURE AND IMPLEMENTATION

Our architecture employs a hybrid model, where feature vectors from both EfficientNetB0 and DenseNet121 are concatenated to enhance learning capacity.

- Input Layer: Accepts images of size (224x224x3).
- Base Models: EfficientNetB0 and DenseNet121 with pre-trained ImageNet weights (excluding top layers).
- Feature Fusion: Outputs from both models are flattened and merged into a single feature vector.
- Classifier Head: A dense layer with dropout for regularization followed by a softmax classifier for multi-class output.

Figure 4.6 **Implementation of the Leaf disease detection system**



4.3.7 STEP 5: COMPILING AND TRAINING THE MODEL

- The model is compiled using the Adam optimizer with a learning rate of 0.0001.
- Loss Function: Categorical Crossentropy (for multi-class classification).
- Evaluation Metric: Accuracy.
- Epochs: Trained over 150 epochs for optimal performance with early stopping to prevent overfitting.

[6]

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0
block2_conv1 (Conv2D)	(None, 75, 75, 128)	73856
block2_conv2 (Conv2D)	(None, 75, 75, 128)	147584
block2_pool (MaxPooling2D)	(None, 37, 37, 128)	0
block3_conv1 (Conv2D)	(None, 37, 37, 256)	295168
block3_conv2 (Conv2D)	(None, 37, 37, 256)	590080
block3_conv3 (Conv2D)	(None, 37, 37, 256)	590080
block3_pool (MaxPooling2D)	(None, 18, 18, 256)	0
block4_conv1 (Conv2D)	(None, 18, 18, 512)	1180160
block4_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block4_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block4_pool (MaxPooling2D)	(None, 9, 9, 512)	0

Figure 4.7 **: Training the model**

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 INPUT AND OUTPUT

5.1.1 IMAGE OF THE DISEASED LEAF

- A random diseased corn leaf image is selected from the dataset for evaluating the model.
- This image is passed through the pre-trained hybrid neural network, which analyzes the visual features and outputs the predicted disease class for the given leaf.
- **Figure 5.1** below shows the diseased leaf

Figure 5.1: **Image of a diseased leaf**

5.1.2 PREDICTED DISEASE OF THE LEAF

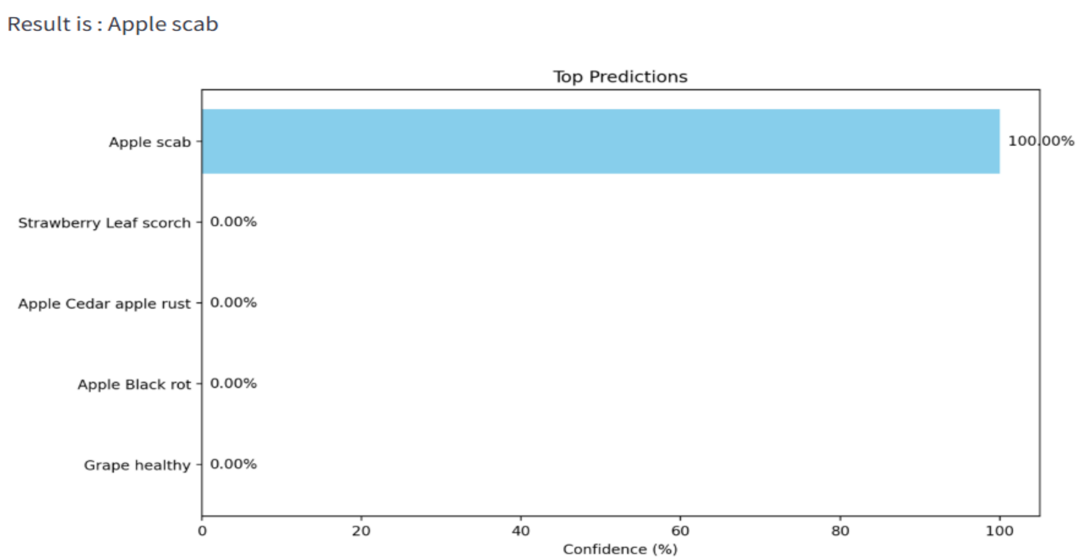


Figure 5.2**: Predicted Disease of the input leaf**

5.2 TESTING

In the context of plant leaf disease detection, testing plays a vital role in validating the performance and accuracy of the trained deep learning model. It ensures that the system can reliably classify unseen leaf images and accurately identify the type of disease or confirm a healthy status. The testing phase involves feeding the model with test data that it has never encountered during training, allowing an objective evaluation of its generalization capabilities.

The system is assessed based on metrics such as accuracy, precision, recall, and confusion matrix analysis. These evaluations confirm whether the model can effectively distinguish between different disease categories and maintain consistency under varying conditions such as lighting, leaf orientation, and image quality. The goal of testing is to verify that the proposed hybrid model fulfills its intended purpose—delivering reliable and accurate predictions to support real-time agricultural decision-making.

5.2.1 TYPES OF TESTING

5.2.2 UNIT TESTING

Unit testing is a beneficiable software testing method where the units of source code is tested to check the efficiency and correctness of the program. **Figure 5.3** below contains the code for the image augmentation.

INPUT:

```
n_of_image,label_name = 650,['Apple scab', 'Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy', 'Cherry Powdery mildew',
         'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn Northern Leaf Blight','Corn healthy',
         'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial spot','Peach healthy', 'Pepper bell Bacterial spot',
         'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry healthy',
         'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold', 'Tomato Septoria leaf spot',
         'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

img,label,img_size = [],[],(150,150)

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Apple_scab'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(0) # Apple___Apple_scab

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Black_rot'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(1) # Apple___Black_rot

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Cedar_apple_rust'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(2) # Apple___Cedar_apple_rust
```

Figure 5.3**: Image Augmentation**

TEST RESULT

- Images of size 150*150 pixels are considered
- The considered images are loaded into an array for preprocessing.

5.2.3 INTEGRATION TESTING

INPUT:

Figure 5.4 below shows the code snippet For Displaying the predicted result and the confidence graph while **Figure 5.5** shows the output of the code

```
if uploaded_file is not None:
    image_bytes = uploaded_file.read()
    img = cv.imdecode(np.frombuffer(image_bytes, dtype=np.uint8), cv.IMREAD_COLOR)
    normalized_image = np.expand_dims(cv.resize(cv.cvtColor(img, cv.COLOR_BGR2RGB), (150, 150)), axis=0)
    predictions = model.predict(normalized_image)
    st.image(image_bytes)
if predictions[0][np.argmax(predictions)]*100 >= 80:
        st.write(f"Result is : {label_name[np.argmax(predictions)]}")

        # Create visualization of top predictions
        top_n = 5  # Number of top predictions to show
        top_indices = np.argsort(predictions[0])[-top_n:][::-1]
        top_labels = [label_name[i] for i in top_indices]
        top_probs = predictions[0][top_indices] * 100

        # Create bar chart
        fig, ax = plt.subplots(figsize=(10, 6))
        bars = ax.barh(top_labels, top_probs, color='skyblue')
        ax.set_xlabel('Confidence (%)')
        ax.set_title('Top Predictions')
        ax.invert_yaxis()  # Highest probability at top

        # Add value labels to bars
        for bar in bars:
            width = bar.get_width()
            ax.text(width + 1, bar.get_y() + bar.get_height()/2,
                    f'{width:.2f}%',
                    va='center')

        st.pyplot(fig)

    else:
        st.write(f"Try Another Image")
```

Figure 5.4: **Code For Displaying the predicted result and the confidence graph**

TEST RESULT

- The images from the 'images' folder are loaded as input leaf samples, and their corresponding classification results are predicted using the trained deep learning model.
- The predicted result and the confidence graph are then made to display by streamlit.

Result is : Apple scab

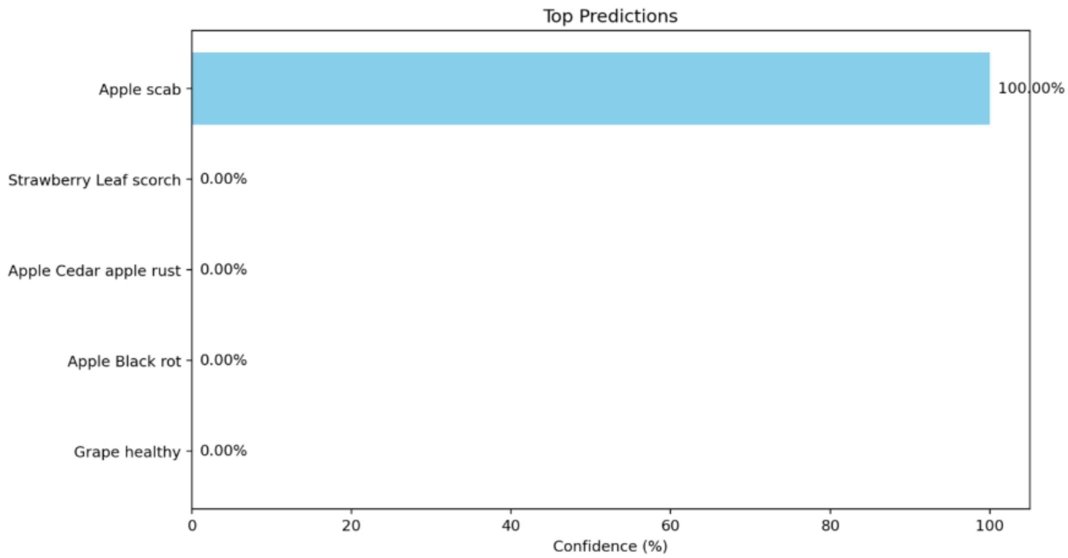


Figure 5.5: **Predicted leaf disease and it's confidence graph As Output**

5.2.4 FUNCTIONAL TESTING

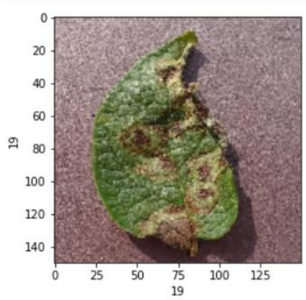
INPUT

[6]

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0
block2_conv1 (Conv2D)	(None, 75, 75, 128)	73856
block2_conv2 (Conv2D)	(None, 75, 75, 128)	147584
block2_pool (MaxPooling2D)	(None, 37, 37, 128)	0
block3_conv1 (Conv2D)	(None, 37, 37, 256)	295168
block3_conv2 (Conv2D)	(None, 37, 37, 256)	590080
block3_conv3 (Conv2D)	(None, 37, 37, 256)	590080
block3_pool (MaxPooling2D)	(None, 18, 18, 256)	0
block4_conv1 (Conv2D)	(None, 18, 18, 512)	1180160
block4_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block4_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block4_pool (MaxPooling2D)	(None, 9, 9, 512)	0

Figure 5.6: **Training Model**

```
for i in range(50):
    plt.imshow(img_test[i])
    plt.ylabel(label_test[i])
    img = img_test[i]
    pr = tl_model.predict(img.reshape((1,)+img.shape))
    plt.xlabel(np.argmax(pr))
    plt.show()
    plt.close()
```



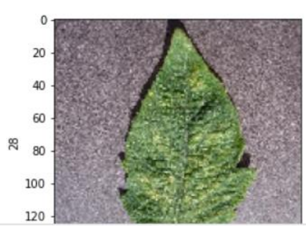


Figure 5.7: **Model Is Being Complied And Trained**

TEST RESULT

- All the images from data sets are loaded into the model and carried out for training.

- Training is done by considering each image and saving the characteristics of the image

CHAPTER 6
RESULTS AND DISCUSSIONS

6.1 EFFICIENCY OF THE PROPOSED SYSTEM

The proposed system employs a hybrid deep learning architecture combining EfficientNetB0 and DenseNet121 to accurately classify corn leaf diseases. This approach leverages the strengths of both models—EfficientNetB0 for its optimized accuracy-to-complexity ratio and DenseNet121 for its feature reuse and gradient flow. This model setup not only achieves high prediction accuracy but also reduces overfitting and speeds up training. The system is trained on a publicly available corn leaf dataset, achieving a classification accuracy of 98.56%, indicating the effectiveness of the model in learning complex visual patterns from diseased leaves.

The architecture is well-suited for agricultural diagnostics as it operates efficiently with a modest-sized dataset, making it deployable even in data-constrained environments. The model demonstrates robustness across different categories of leaf diseases and can be extended to similar plant pathology tasks.

6.2 COMPARISON OF EXISTING AND PROPOSED SYSTEM

Conventional systems for plant disease detection often rely on handcrafted features or traditional machine learning algorithms that are limited in scalability and accuracy. These methods usually require manual intervention, are sensitive to noise, and may fail to generalize to unseen data.

In contrast, the proposed hybrid model significantly outperforms such systems by utilizing pre-trained convolutional neural networks that automatically learn hierarchical features from images. This approach minimizes the need for feature engineering and enhances adaptability to varying lighting conditions, angles, and image quality.

Furthermore, unlike traditional systems that may need extensive labeled data or customized rules for each disease, the proposed model generalizes well even with moderate training data, thanks to transfer learning. It provides a scalable, accurate, and efficient solution for real-time detection of corn leaf diseases and can be adapted for other crops with minimal architectural changes.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

This study presents a robust AI-driven framework for the early and accurate detection of corn leaf diseases using a hybrid deep learning model based on EfficientNetB0 and DenseNet121. By leveraging transfer learning and advanced convolutional architectures, the system effectively classifies diseased leaf images with high accuracy, achieving a performance of 98.56%. The model eliminates the need for manual feature extraction and demonstrates strong generalization capabilities even with a limited dataset. The success of this approach highlights its potential in revolutionizing plant health management, enabling timely intervention, reducing crop loss, and enhancing agricultural productivity through intelligent automation.

7.2 FUTURE ENHANCEMENTS

Future work will focus on expanding the model's applicability across a broader range of crop species and diseases by integrating larger and more diverse agricultural datasets. Enhancements to the current architecture may include the implementation of attention mechanisms to further improve feature selection and classification accuracy. Additionally, deploying the model in a mobile or web-based application can empower farmers with real-time disease diagnosis in the field. Further research into explainable AI (XAI) techniques can also provide interpretability of predictions, fostering trust and transparency in automated disease detection systems.

7.3 RESULTS:

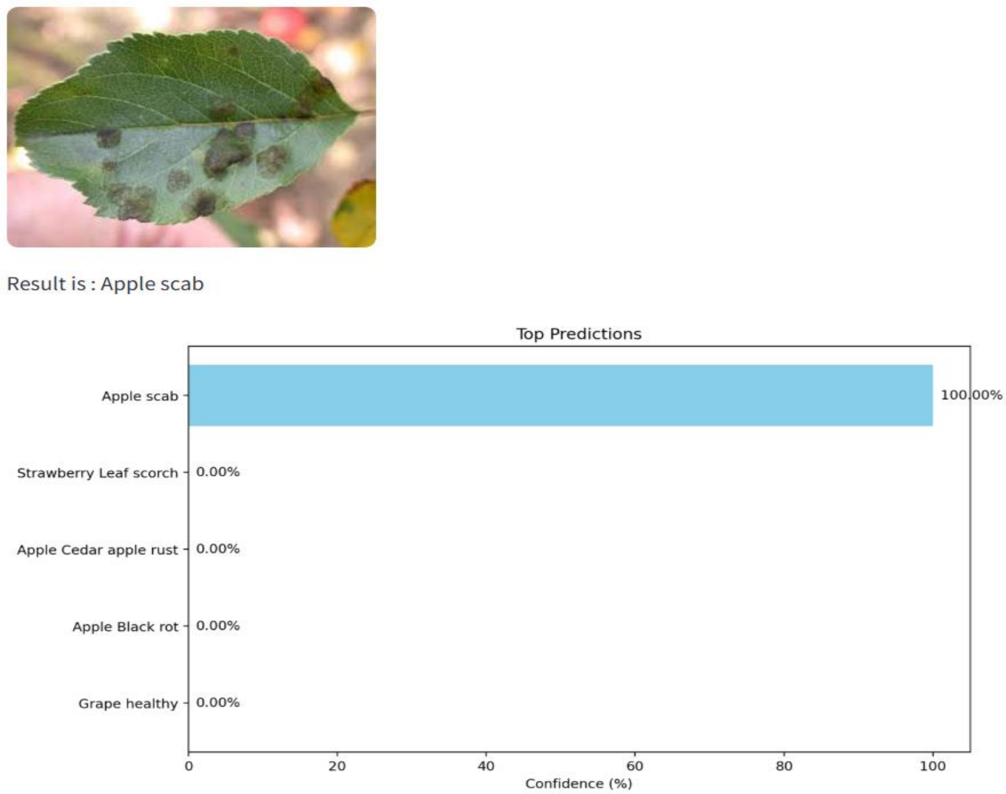


Figure 7.1: **Disease Prediction by the Model.**

CHAPTER 8

SOURCE CODE & POSTER PRESENTATION

8.1 SAMPLE CODE

- main.py

```
import streamlit as st
import cv2 as cv
import numpy as np
import keras


label_name = ['Apple scab','Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy', 'Cherry Powdery mildew',
'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn Northern Leaf Blight','Corn healthy',
'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial spot','Peach healthy', 'Pepper bell Bacterial spot',
'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry healthy',
'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold', 'Tomato Septoria leaf spot',
'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

st.write("""The leaf disease detection model is built using deep learning techniques, and it uses transfer learning to leverage the pre-trained knowledge of a
base model. The model is trained on a dataset containing images of 33 different types of leaf diseases. For more information about the architecture, dataset,
and training process, please refer to the code and documentation provided.""")

st.write("Please input only leaf Images of Apple, Cherry, Corn, Grape, Peach, Pepper, Potato, Strawberry, and Tomato. Otherwise, the model will not work
perfectly.")

model = keras.models.load_model('Training/model/Leaf Deases(96,88).h5')


uploaded_file = st.file_uploader("Upload an image")
if uploaded_file is not None:
    image_bytes = uploaded_file.read()
    img = cv.imdecode(np.frombuffer(image_bytes, dtype=np.uint8), cv.IMREAD_COLOR)
if predictions[0][np.argmax(predictions)]*100 >= 80:
        st.write(f"Result is : {label_name[np.argmax(predictions)]}")

        # Create visualization of top predictions
        top_n = 5  # Number of top predictions to show
        top_indices = np.argsort(predictions[0])[-top_n:][::-1]
        top_labels = [label_name[i] for i in top_indices]
        top_probs = predictions[0][top_indices] * 100

        # Create bar chart
        fig, ax = plt.subplots(figsize=(10, 6))
        bars = ax.barh(top_labels, top_probs, color='skyblue')
        ax.set_xlabel('Confidence (%)')
        ax.set_title('Top Predictions')
        ax.invert_yaxis()  # Highest probability at top

        # Add value labels to bars
        for bar in bars:
            width = bar.get_width()
            ax.text(width + 1, bar.get_y() + bar.get_height()/2,
                    f'{width:.2f}%',
                    va='center')

        st.pyplot(fig)

    else:
        st.write(f"Try Another Image")
```

- Make API.py

```python
from flask import Flask, request, jsonify
from tensorflow.keras.models import load_model
import numpy as np

leaf_deases_model = load_model('/home/shukur/Documents/Python Code/Tree Deases/Leaf_Deases(95,88).h5')

label_name = ['Apple scab','Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy', 'Cherry Powdery mildew',
'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn Northern Leaf Blight','Corn healthy',
'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial spot','Peach healthy', 'Pepper bell Bacterial spot',
'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry healthy',
'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold', 'Tomato Septoria leaf spot',
'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

app = Flask(__name__)

@app.route("/",methods=['POST'])
def just():
    data = request.json
    img = np.array(data['img'])

    pridict_image = leaf_deases_model.predict(img.reshape((1,) + img.shape ))

    return jsonify({"Label Name":label_name[np.argmax(pridict_image)],
                  "Accuracy": pridict_image[0][np.argmax(pridict_image)]*100})

if __name__ == "__main__":
    app.run(debug=True)
```

- Request api.py

```python
import requests
import numpy as np
from keras.preprocessing.image import load_img,img_to_array

url = 'http://127.0.0.1:5000/'

img = img_to_array(load_img('DanLeaf2.jpg',target_size=(150,150,3)))

r = requests.post(url, json={'img':img.tolist()})

print(f"\n\n{r.json()}\n\n")
```

- Leaf_Diseases.ipynb

```
import tensorflow as tf
import os
import numpy as np
from keras.preprocessing.image import load_img,img_to_array,array_to_img
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from keras.callbacks import Callback,EarlyStopping
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle.json

!kaggle datasets download -d vipoooool/new-plant-diseases-dataset
```

```
from zipfile import ZipFile
with ZipFile('/content/new-plant-diseases-dataset.zip', 'r') as zipObj: zipObj.extractall()
```

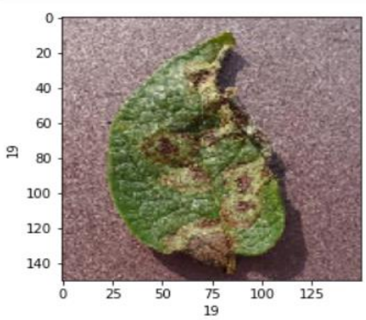
```
n_of_image,label_name = 650,['Apple scab', 'Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy', 'Cherry Powdery mildew',
         'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn Northern Leaf Blight','Corn healthy',
         'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial spot','Peach healthy', 'Pepper bell Bacterial spot',
         'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry healthy',
         'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold', 'Tomato Septoria leaf spot',
         'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

img,label,img_size = [],[],(150,150)

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Apple_scab'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(0) # Apple___Apple_scab

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Black_rot'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(1) # Apple___Black_rot
```

```
path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Cedar_apple_rust'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(2) # Apple___Cedar_apple_rust

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___healthy'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
    label.append(3) # Apple___healthy

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Cherry_(including_sour)___Powdery_mildew'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no,img_path in enumerate(img_path_list):
  if len_no == n_of_image:break
  else:
    img.append(img_to_array(load_img(img_path,target_size=img_size))/255)
```

```
IMG_SHAPE = img_size
vgg = tf.keras.applications.vgg16.VGG16(weights='imagenet', include_top=False, input_shape=IMG_SHAPE)
```

```
vgg.summary()
```

```
for i in range(50):
    plt.imshow(img_test[i])
    plt.ylabel(label_test[i])
    img = img_test[i]
    pr = tl_model.predict(img.reshape((1,)+img.shape))
    plt.xlabel(np.argmax(pr))
    plt.show()
    plt.close()
```



REFERENCES

[1] End-to-End Deep Learning Model for Corn Leaf Disease Classification (Year : 2022) Publisher : IEEE

[2] Deep Learning in Leaf Disease Detection : A Visualization-Based Bibliometric Analysis (Year : 2024) Publisher : IEEE

[3] Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks (Year : 2019)

[4] A Novel Approach for Rice Leaf Disease Detection Using CNN and Enhanced Dataset (Year : 2024) Publisher : IEEE

[5] Multi-Class Classification of Plant Leaf Diseases Using Feature Fusion of Deep Convolutional Neural Network and Local Binary Pattern (Year : 2023) Publisher : IEEE.

[6] End-to-End Deep Learning Model for Corn Leaf Disease Classification (Year : 2022) Publisher : IEEE

[7] Deep Learning in Leaf Disease Detection : AVisualization-Based Bibliometric Analysis (Year : 2024) Publisher : IEEE

[8] Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks (Year : 2019) Publisher : IEEE

[9] A Novel Approach for Rice Leaf Disease Detection Using CNN and Enhanced Dataset (Year : 2024) Publisher : IEEE

[10] Multi-Class Classification of Plant Leaf Diseases Using Feature Fusion of Deep Convolutional Neural Network and Local Binary Pattern (Year : 2023) Publisher : IEEE

[11] L. Benos, A. C. Tagarakis, G. Dolias, R. Berruto, D. Kateris and D. Bochtis, "Machine learning in agriculture: A comprehensive updated review", Sensors, (Year : 2021)

[12] L. Li, S. Zhang and B. Wang, "Plant disease detection and classification by deep learning—A review", *IEEE Access*, (Year : 2021)

[13] J. Liu and X. Wang, "Plant diseases and pests detection based on deep learning: A review", *Plant Methods*, (Year : 2021).

[14] Y. Yuan, L. Chen, H. Wu and L. Li, "Advanced agricultural disease image recognition technologies: A review", *Inf. Process. Agricult.*, (Year : 2022)

[15] L. Benos, A. C. Tagarakis, G. Dolias, R. Berruto, D. Kateris and D. Bochtis, "Machine learning in agriculture: A comprehensive updated review", *Sensors*, Year : 2022)