LEAF DISEASE DETECTION USING DEEP LEARNING

A DESIGN PROJECT REPORT

Submitted by

ABINAYA.P

CHARUMATHI.P

JAI PRADHA.S

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM -621112

NOVEMBER 2024

LEAF DISEASE DETECTION USING DEEP LEARNING

A DESIGN PROJECT REPORT

Submitted by

ABINAYA P (811722104003)

CHARUMATHI P (811722104024)

JAI PRADHA S (811722104058)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)
SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report titled **"LEAF DISEASE DETECTION USING DEEP LEARNING"** is the Bonafide work of the students ABINAYA P (811722104003), CHARUMATHI P (811722104024), JAI PRADHA S(811722104058) who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on thebasis of which a degree or award was conferred on an earlier occasion on this orany other candidate.

SIGNATURE

Dr.Delphin Carolina Rani, M.E, Ph.D**.,**

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K. Ramakrishnan College of Technology(Autonomous)

Samayapuram – 621 112

SIGNATURE

Mrs.ValliPriyaDharshini,M.E,(Ph.D.,)

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K. Ramakrishnan College of Technology (Autonomous)

Samayapuram – 621 112

Submitted for the viva-voce examination held on …….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We jointly declare that the project report on **"LEAF DISEASE DETECTION USING DEEP LEARNING"** is the result of original workdone by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the award of Degree of **BACHELOR OF ENGINEERING**.

Signature

ABINAYA P

CHARUMATHI P

JAI PRADHA S

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and in-debt to our institution "**K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)**", for providing us with the opportunity to do this project.

We are glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

We would like to thank **Dr. N. VASUDEVAN, M.Tech, Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

We heartily thanks to **Dr. DELPHIN CAROLINA RANI, M.E, Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express my deep and sincere gratitude to our beloved project guide **Mrs.ValliPriyaDharshini,M.E,(Ph.D.,)** Department of **COMPUTER SCIENCE AND ENGINEERING**, for her incalculable suggestions, creativity, assistance and patience which motivated me to carry out this project.

I render my sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express my special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

Detecting diseases in crop leaves is important for keeping plants healthy and ensuring good agricultural yields. This project uses deep learning techniques in Python 3.7 to build a reliable system that identifies leaf diseases and offers remedies to help farmers take effective actions. Several Python libraries are used to make the system powerful and efficient. TensorFlow is used to build and train the deep learning model, OpenCV is used for image preprocessing (like resizing and augmenting images), NumPy helps with handling large arrays for faster computations, Pandas is used for managing the data, and Matplotlib and Seaborn are used to create visual representations of data and results.

The process begins by preparing the images through resizing them to a standard size for consistency and normalizing pixel values for better performance. To ensure the model learns effectively, techniques like rotating and flipping images are applied to make the dataset more diverse. The main model is a Convolutional Neural Network (CNN), which includes several layers. The input layer takes in the images, convolutional layers extract important features, pooling layers reduce the size of the data, dense layers process the information, and the output layer provides the final classification. The model uses activation functions like ReLU for extracting features and Softmax for classifying the diseases. The dataset is split into 80% for training and 20% for testing to ensure the model performs well on unseen data. The final system is not only good at detecting diseases but also provides suggestions on how to treat the detected issues and maintain healthy crops. This feature makes the tool highly beneficial for farmers, helping them take quick and effective steps to protect their crops.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	**v**
	LIST OF TABLES	**vi**
	LIST OF FIGURES	**x**
	LIST OF ABBREVIATIONS	**xi**
1	**INTRODUCTION**	**1**
	1.1 PROJECT OVERVIEW	**2**
	1.2 PROBLEM STATEMENT	**3**
	1.2.1 GOAL	**3**
	1.3 OBJECTIVES	**4**
	1.4 SCOPE OF THE PROJECT	**5**
2	**LITERATURE REVIEW**	**6**
	2.1 PLANT LEAF DISEASE DETECTION USING CONVOLUTION NEURAL NETWOK	**6**
	2.2 A COMPRENSHIVE REVIEW ON LEAF DISEASE DEEP LEARNING	**7**

	2.3 LEAKY ReLU-ReSNET FOR PLANT LEAF DISAESE DETECTION BY DEEP LEARNING APPROACH	**8**
	2.4 DEEP LEARNING IN IMAGE CLASSIFICAATION	**9**
	2.5 A COMPRENSHIVE REVIEW ON DETECTION BY DEEP LEARNING AND MACHINE LEARNING	**10**
3	**EXISTING SYSTEM**	**11**
4	**PROPOSED SYSTEM**	**14**
5	**SYSTEM DESIGN**	**20**
	5.1 SYSTEM ARCHITECTURE	**21**
	5.2 DATA FLOW DIAGRAM	**24**
	5.3 UML DIAGRAM	**24**
	5.3.1 USE-CASE DIAGRAM	**25**
	5.3.2 ACTIVITY DIAGRAM	**26**
	5.3.3 SEQUENCE DIAGRAM	**27**
	5.4 SYSTEM STUDY	**28**
	5.4.1 FEASIBILITY STUDY	**28**
	5.4.1.1 TECHNICAL FEASIBILITY	**28**
	5.4.1.2 ECONOMICAL FEASIBILITY	**28**
6	**SYSTEM REQUIREMENT**	**25**

6.1 HARDWARE REQUIREMENTS 25

6.2 SOFTWARE REQUIREMENTS 25

6.3 HARDWARE DESCRIPTION 26

6.4 SOFTWARE DESCRIPTION 26

6.4.1 INTEGRATED DEVELOPMENT AND LEARNING ENVIRONMENT(IDLE) 26

6.4.2 ENVIRONMENT SETUP 26

7 SYSTEM TESTING 28

7.1 TESTING STEPS 28

7.1.1 UNIT TESTING 28

7.1.2 INTEGRATION TESTING 29

7.1.3 FUNCTIONAL TESTING 29

7.1.4 WHITE BOX TESTING 29

7.1.5 BLACK BOX TESTING 30

7.1.6 ACCEPTANCE TESTING 3

8 CONCLUSION AND FUTURE WORKS 31

APPENDIX A 34

APPENDIX B 38

REFERENCES 45

LIST OF FIGURES

FIGURE	FIGURE NAME	PAGE NO
3.1	EXI`1STING SYSTEM	13
4.1	PROPOSED SYSTEM	16
5.1	SYSTEM ARCHITECTURE	18
5.2	FLOW CHART	20
5.3	USE CASE DIAGRAM	21
5.4	ACTIVITY DIAGRAM	22
5.5	SEQUENCE DIAGRAM	23
7.1	BLACK BOX TESTING	30

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM
CNN	Convolutional Neural Networks
ANN	Artificial Neural Network
RNN	Recurrent Neural Networks (RNNs)
GNN	Graph Neural Networks
YOLO	You Only Look Once
ViTs	Vision Transformers
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
VGG	Visual Geometry Group Network.
RGB	Red, Green, Blue
HSV	Hue, Saturation, Value
HSI	Hue, Saturation, Intensity
SURF	Speeded-Up Robust Features
SIFT	Scale-Invariant Feature Transform

HOG	Histogram of Oriented Gradients
SVM	Support Vector Machine
BPNN	Backpropagation Neural Network
DFD	Data Flow Diagram
UML	Unified Modelling Language
IDLE	Integrated Development and Learning Environment.
IoT	Internet Of Things
ReLU	Rectified Linear Unit.
API	Application Programming Interface.

CHAPTER 1

INTRODUCTION

Agriculture is the backbone of human survival and a critical sector for global food security. However, the increasing prevalence of crop diseases poses a significant threat to agricultural productivity, leading to severe economic losses for farmers and endangering food supplies. Leaf diseases, in particular, are among the most common and damaging issues faced by crops. Detecting and managing these diseases in their early stages is vital to minimizing their impact.

Traditionally, farmers rely on manual inspections or consultations with agricultural experts to identify diseases, but these methods are often time-consuming, expensive, and inaccessible, especially in rural areas. With advancements in artificial intelligence, particularly deep learning, there is now an opportunity to revolutionize the way crop diseases are managed. This project aims to develop a deep learning-based system to detect leaf diseases, provide targeted remedies, and recommend maintenance strategies to ensure sustainable crop health.

By leveraging technology, the project seeks to empower farmers with real-time tools to enhance productivity, reduce losses, and promote eco-friendly farming practices. The integration of a user-friendly interface further ensures accessibility for farmers, agricultural advisors, and researchers, making it a comprehensive solution to one of agriculture's most pressing challenges. This initiative not only addresses the immediate need for effective disease management but also contributes to the broader goal of sustainable agriculture and food security.

1.1 PROJECT OVERVIEW

Leaf disease detection in crops involves using technology to identify the presence of diseases affecting plant leaves, enabling early diagnosis and management. The process begins with capturing high-quality images of the affected leaves, which are then analyzed using advanced techniques like Convolutional Neural Networks (CNNs). These models can accurately classify diseases such as Bacterial Blight, Powdery Mildew, and Soybean Mosaic Virus, among others, by recognizing specific patterns and symptoms on the leaves. Once a disease is identified, targeted remedies can be suggested, such as applying appropriate fungicides, adjusting watering schedules, or using organic treatments to prevent further spread. Early detection helps farmers intervene promptly, minimizing crop damage and preserving yield. Following disease management, proper crop maintenance practices, including regular monitoring, optimizing soil health, and ensuring adequate nutrient supply, are essential to promote plant resilience and prevent future outbreaks. This holistic approach not only improves crop productivity but also supports sustainable agricultural practices, ensuring long-term food security.

The rapid advancements in artificial intelligence, particularly deep learning, provide a promising solution to these challenges. This project aims to develop an automated deep learning-based system that detects leaf diseases with high accuracy, recommends targeted remedies, and provides crop maintenance strategies. By leveraging technology, the system offers real-time, cost-effective, and user-friendly tools to empower farmers and promote sustainable agriculture. It focuses on minimizing crop losses, optimizing resource usage, and fostering environmentally friendly practices, ensuring a comprehensive approach to addressing one of agriculture’s most pressing issues.

1.2 PROBLEM STATEMENT

Plant diseases significantly impact agricultural productivity, causing economic losses and threatening food security. Early and accurate detection of leaf diseases is essential to minimize these effects. Traditional methods of disease detection are manual and labor-intensive, limiting their scalability. The goal of this project is to develop a deep learning-based automated system to classify and identify diseases in plant and remedy to the leaves with high accuracy and reliability.

The Risk Identified:

1. **Early Disease Detection Challenges:** Difficulty in identifying leaf diseases in their early stages without expert knowledge.
2. **Mismanagement of Resources:** Overuse or misuse of pesticides due to incorrect or delayed diagnosis.
3. **Lack of Scalable Solutions:** Existing methods are not easily scalable for diverse crops and regions.
4. **Environmental Concerns:** Over-reliance on chemical treatments negatively impacts soil health and ecosystems.
5. **Time and Cost Constraints:** Manual inspections are time-consuming and expensive for farmers.

1.2.1 GOALS

The goals of this project are as follows:

- **Detect Leaf Diseases Accurately:** Use deep learning to identify and classify diseases in crop leaves with high precision.
- **Recommend Effective Remedies:** Provide targeted solutions like suitable pesticides, fertilizers, and eco-friendly treatments for each disease.

- **Guide Crop Maintenance:** Offer practical advice on irrigation, fertilization, and preventive care to ensure healthy crops.
- **Enhance Agricultural Sustainability:** Help farmers reduce crop losses, minimize pesticide use, and adopt sustainable farming practices.

1.3 OBJECTIVE OF THE PROJECT

To create a deep learning-based solution for the accurate detection of diseases in plant leaves. By utilizing advanced image processing techniques, the system will automate the process of identifying and classifying various plant diseases. This will reduce reliance on manual inspections, which can be time-consuming and prone to errors, especially in large-scale agricultural settings.

In addition to identifying diseases, the system will provide tailored remedies to address the specific issues detected. These remedies will include both organic and chemical treatment options, ensuring farmers have a range of solutions to choose from based on their preferences and the severity of the disease. By offering targeted advice, the system aims to mitigate the impact of diseases effectively and promptly. Furthermore, the project will focus on promoting better crop maintenance practices by providing preventive measures to minimize future outbreaks. This includes guidelines on optimal farming practices, environmental controls, and regular monitoring schedules to maintain healthy crops. By empowering farmers with actionable insights, the system will help enhance crop resilience and overall productivity.

1.4 SCOPE OF THE PROJECT

The Leaf Disease Detection Using Deep Learning with Remedies and Crop

Maintenance project operates within a defined scope to ensure efficient and targeted outcomes. The key components of the project are as follows:

1.4.1. Data Collection

- **Source:** The project focuses on collecting images of plant leaves from various open-source agricultural datasets and user-provided inputs. These datasets will include high-resolution images capturing healthy leaves and leaves affected by diseases.
- **Preprocessing:** The images will be preprocessed for clarity, including resizing, noise reduction, and augmentation techniques to improve model performance.

1.4.2. Disease Detection and Classification

- **Methodology**: The project will use a convolutional neural network (CNN) as the core deep learning model to detect and classify diseases. Advanced techniques like transfer learning may be employed to improve accuracy and reduce training time.
- **Output:** The system will classify each leaf image into categories such as "Healthy," "Disease A," "Disease B," etc., with confidence scores for each prediction.

1.4.3. Remedy Recommendation

Personalized Remedies: For each detected disease, the system will suggest tailored remedies. Recommendations will include organic solutions, chemical treatments, and preventive measures based on the severity of the disease.

CHAPTER 2

LITERATURE SURVEY

2.1 TITLE: PLANT LEAF DISEASE DETECTION USING CONVOLUTION NEURAL NETWORK

AUTHOR: M. VINITHA, MALLIKARJUNA NANDHI
YEAR: 2024

ABSTRACT

This research focuses on using deep learning techniques, specifically Convolutional Neural Networks (CNNs) implemented in the PyTorch framework, to automate the detection and classification of plant diseases. The model is trained on the Plant Village dataset, which includes images of 39 different plant disease categories. By utilizing CNNs, the system can accurately classify leaf images, enabling early and efficient diagnosis of diseases.

MERITS

- **High Classification Accuracy:** The CNN model can accurately classify plant diseases across 39 categories, providing reliable diagnosis and supporting timely interventions.

DEMERITS

- **Computational Complexity:** Training deep learning models like CNNs requires significant computational resources, which may not be feasible for all users, especially small-scale farmers.

2.2 TITLE: A COMPREHENSIVE REVIEW ON LEAF DISEASE DETECTION USING DEEP ANALYSIS

AUTHORS: SUMAYA MUSTOFA, MD MEHEDI HASAN MUNNA, YOUSUF RAYHAN EMON, GOLAM RABBANY

YEAR: 2023

ABSTRACT

Leaf diseases are a major threat to global agriculture, significantly affecting crop yield and quality. Early detection and accurate diagnosis are essential for effective management. This paper provides a comprehensive review of the use of deep learning for plant leaf disease detection. It discusses various models and methods applied to diagnose leaf diseases, including Convolutional Neural Networks (CNNs), Vision Transformers (ViTs), and the YOLO (You Only Look Once) architecture. The study highlights the advantages of using deep learning models, such as their ability to handle large datasets and automatically extract relevant features from leaf images, reducing human intervention in feature engineering.

MERITS:

- **High Accuracy**: Deep learning models, particularly CNNs and ViTs, achieve high accuracy in detecting and classifying leaf diseases, surpassing traditional machine learning techniques.

DEMERITS

- **Data Dependency:** Deep learning models require large amounts of high-quality labeled data for training, which can be expensive and time-consuming to collect.

2.3 TITLE: TITLE: LEAKY ReLU-ResNET FOR PLANT LEAF DISEASE DETECTION: A DEEP LEARNING APPROACH

AUTHOR: ZHONG ET AL.,

YEAR: 2023

ABSTRACT

He provides a comprehensive overview of the advancements in image classification using deep learning techniques. It explores various architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and graph neural networks (GNNs), highlighting their respective roles in processing spatial and sequential data. The paper emphasizes the evolution of CNN models like AlexNet, VGGNet, ResNet, and DenseNet, which have significantly improved image classification performance.

MERITS

- **High Accuracy:** The Leaky ReLU-ResNet model detects leaf diseases with great precision, reducing errors in diagnosis.
- **Early Detection:** Identifies diseases early, enabling timely action to save crops.

DEMERITS

- **Needs High Resources:** Requires powerful computers for training and running the model.
- **Data Dependence:** Relies on well-labeled datasets for effective training.

2.4 TITLE: DEEP LEARNING IN IMAGE CLASSIFICATION

AUTHOR: ZHENGYU HE

YEAR: 2022

ABSTRACT

Deep Learning in Image classification provides a comprehensive overview of the advancements in image classification using deep learning techniques. It explores various architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and graph neural networks (GNNs), highlighting their respective roles in processing spatial and sequential data. The paper emphasizes the evolution of CNN models like AlexNet, VGGNet, ResNet, and DenseNet, which have significantly improved image classification performance. Additionally, it discusses emerging trends such as GNNs, which extend deep learning capabilities to non-Euclidean data domains, offering new approaches for graph-structured data tasks.

MERITS

- **High Accuracy:** Deep learning models, especially CNNs, achieve high accuracy in image classification tasks due to their ability to learn hierarchical features.
- **Automated Feature Extraction:** Unlike traditional methods, deep learning automatically extracts features from raw data without human intervention.

DEMERITS

- **Data Dependency:** Requires large volumes of labeled data for training, which can be expensive and time-consuming to obtain.

2.5 TITLE: A COMPREHENSIVE REVIEW ON DETECTION OF PLANT DISEASE USING MACHINE LEARNING AND DEEP LEARNING APPROACHES

AUTHOR: C. JACKULIN, S. MURUGAVALLI
YEAR: 2022

ABSTRACT

The paper "**A Comprehensive Review on Detection of Plant Disease Using Machine Learning and Deep Learning Approaches**" explores advancements in plant disease detection through machine learning (ML) and deep learning (DL). The study highlights the significance of AI technologies in agriculture for improving crop yields by addressing plant diseases caused by bacteria, fungi, and viruses. It reviews various ML and DL techniques, emphasizing the superior performance of DL in detecting diseases via computer vision. A comparative analysis of these techniques reveals their applications, strengths, and limitations in real-world scenarios.

MERITS

- **Enhanced Accuracy**: DL models provide higher accuracy in disease detection compared to traditional ML methods.
- **Automated Analysis:** Reduces manual labor by enabling automated disease identification from images.

DEMERITS

- **Resource-Intensive:** Requires high computational power and large datasets for training DL models.

CHAPTER 3

EXISTING SYSTEM

1. Database

- The process starts with a database containing images of leaves.
- The images are typically captured in various formats, such as:
    - **HSV, HIS, RGB:** Common color spaces for image representation.
    - **Hyperspectral:** Captures a wide spectrum of wavelengths.
    - **Thermal:** Captures heat patterns of the leaf surface.

2. Image Acquisition

This step involves capturing images of leaves, which can be done using various devices such as cameras, hyperspectral sensors, or thermal scanners. The captured images are fed into the system. Key points:

- Ensures that the input data is diverse and representative of real-world scenarios.
- Multiple formats (HSV, HIS, etc.) are used to increase detection accuracy.
- The acquired images serve as input for pre-processing.

3. Image Pre-processing

The raw images undergo pre-processing to enhance quality and remove noise. This includes:

- **Histogram Equalization**: Adjusts image contrast for better visibility.
- **Filters (General and Customized)**: Used to smooth or sharpen

images.

- **Color Space Conversion**: Transforms the image into a suitable color format for analysis.

4. Image Segmentation

The pre-processed images are segmented to isolate the diseased areas of the leaf. Methods include:

- **Clustering Methods**: Groups pixels with similar characteristics.
- **Region Growing**: Starts with seed points and expands regions based on similarity.
- **Thresholding**: Differentiates healthy and diseased areas based on pixel intensity.
- **Watershed Methods**: Splits the image into regions based on topographical features.

5. Feature Extraction

After segmentation, features are extracted for classification:

- **Shape**: Includes methods like Hu moments, SURF, SIFT, and HOG.
- **Texture**: Uses techniques like Haralick features to analyze patterns.
- **Color**: Analyzes color distribution using color histograms.

6. Image Classification

- The extracted features are used to classify the leaf as either **healthy** or **diseased**.
- Classification algorithms include:
  - **SVM (RBF Kernel)**: Support Vector Machine with Radial Basis Function kernel.
  - **K-Nearest Neighbor (KNN)**: Classifies based on nearest data points.

- **ANN/BPNN**: Artificial Neural Networks and Backpropagation Neural Networks.
- **Random Forest, Naive Bayes, Decision Tree**: Other machine learning classifiers.

7. Output

The system generates a final output indicating the health status of the leaf:

- **Healthy**: No disease is detected.
- **Diseased**: The system identifies the presence of a disease, and further details (e.g., severity, type) can be provided.

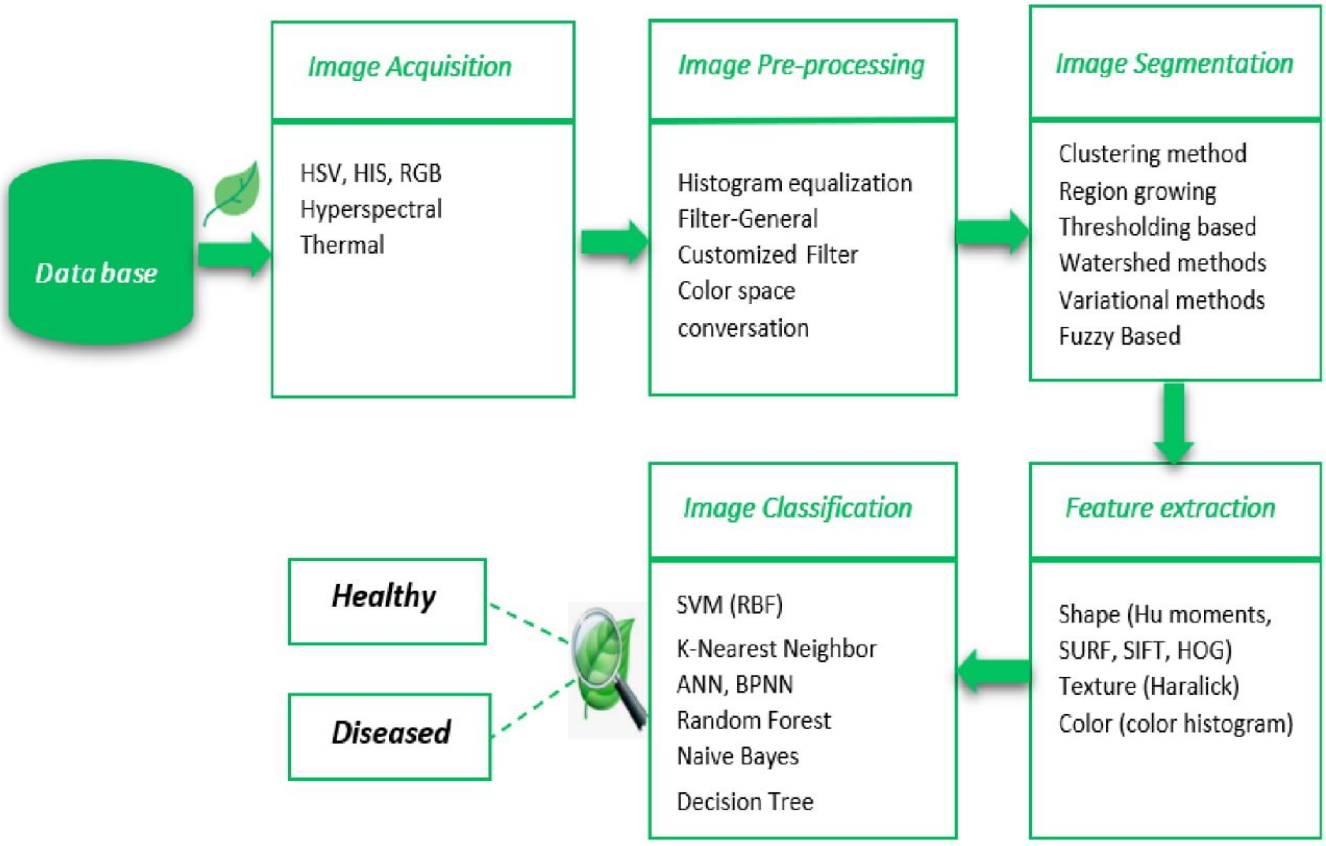


Fig.3.1 Existing System

CHAPTER 4

PROPOSED SYSTEM

The proposed system for Leaf Disease Detection is an advanced and automated solution designed to address the challenges faced by farmers and agricultural practitioners in identifying and managing plant diseases. With the growing demand for sustainable agriculture and increased crop yield, timely and accurate detection of leaf diseases has become crucial. This system leverages modern technology, such as image processing and deep learning, to provide an efficient, user-friendly, and cost-effective alternative to traditional methods of plant disease diagnosis.

Unlike conventional approaches that rely heavily on manual observation or expert consultations, the proposed system automates the entire process. Starting from image capture to preprocessing, disease classification, and actionable outputs, the system streamlines operations to ensure quick and reliable results. By incorporating cutting-edge deep learning models, it enhances accuracy in detecting and classifying diseases, significantly reducing human error.

The system focuses not only on identifying diseases but also on providing actionable outcomes. Once a leaf image is analyzed, it classifies the plant into "Healthy" or "Diseased" categories. For diseased plants, it suggests specific remedies, while for healthy plants, it provides maintenance tips to prevent future infections. This dual-purpose functionality ensures both immediate problem resolution and long-term crop health management.

Designed with scalability and ease of use in mind, the system is accessible to a

wide range of users, from small-scale farmers to agricultural researchers. It operates seamlessly with commonly available devices like smartphones or cameras, making it adaptable to diverse agricultural environments, including remote areas. Moreover, its cost-effective nature ensures that even resource-limited users can benefit from its capabilities.

Key Features of the Proposed System:

1. **Automated Disease Detection**
   - The system automates the process of detecting leaf diseases, reducing reliance on manual inspection and expert involvement.
2. **Deep Learning-Based Classification**
   - Utilizes a pre-trained deep learning model for accurate classification of leaves into "Healthy" or "Diseased" categories.
3. **Real-Time Analysis**
   - Offers quick and efficient processing, enabling immediate diagnosis and actionable recommendations.
4. **Actionable Remedies and Maintenance Tips**
   - Provides disease-specific remedies for diseased plants and preventive maintenance tips for healthy crops.
5. **User-Friendly Design**
   - Simplified workflow ensures ease of use for non-technical users like farmers and agricultural workers.
6. **Cost-Effective Solution**
   - Reduces costs associated with manual disease detection or hiring experts by providing a self-contained system.

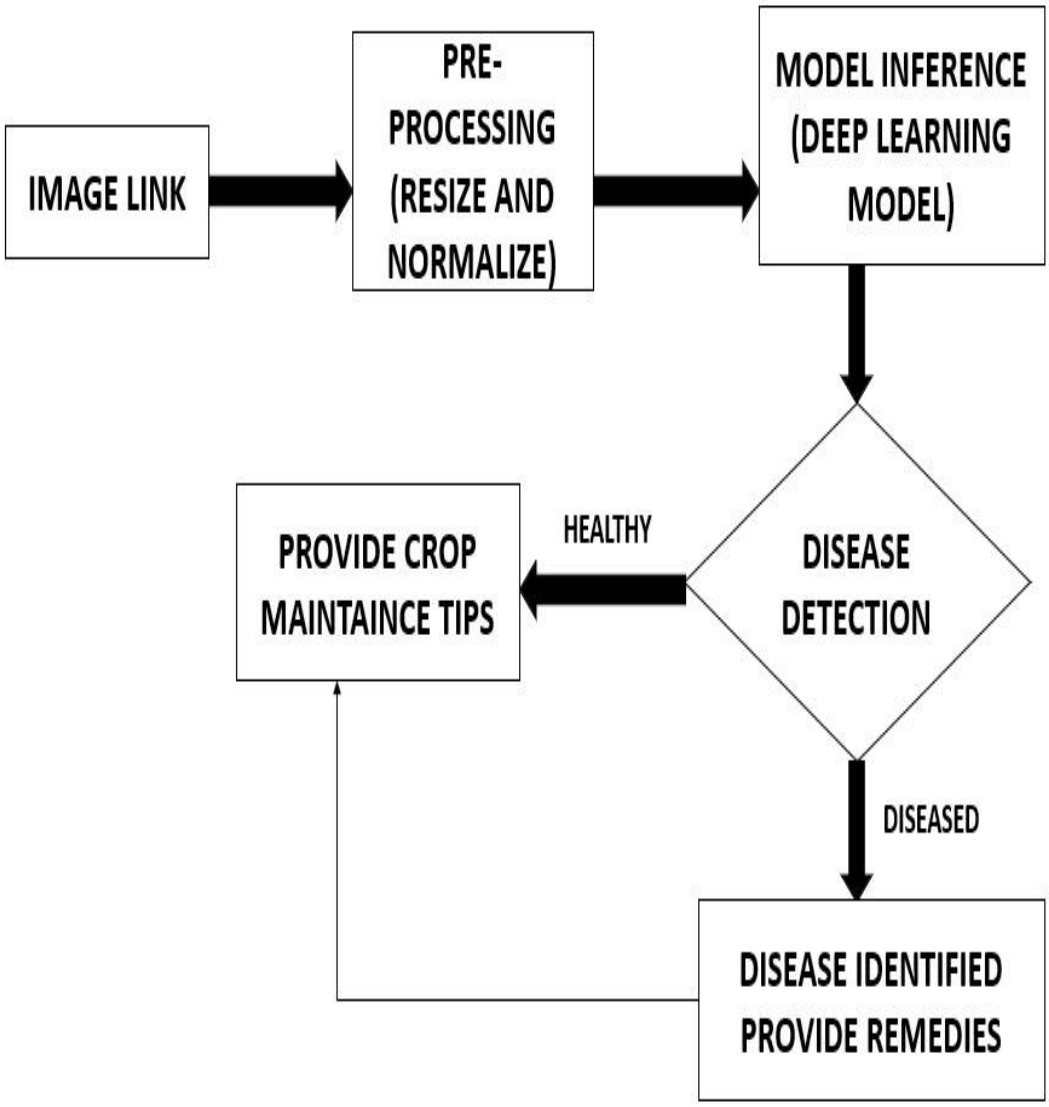


Fig. 4.1 Proposed System

CHAPTER 5

SYSTEM DESIGN

SYSTEM ARCHITECTURE

This system architecture outlines the workflow of a leaf disease detection system. The process begins with the user, who uploads an image of a leaf through the system's interface. This image is handled by the image upload module, which ensures the image is ready for processing and sends it to the next component. The deep learning model then analyzes the uploaded image, using advanced algorithms to detect any diseases affecting the leaf. Once the disease is identified, the system retrieves detailed information about it from the disease database.

The disease information is passed to the **detection service**, which acts as a central component for generating recommendations. It fetches appropriate remedies from the **remedies database** and preventive tips from the **crop maintenance tips database**. These remedies and tips are combined to create actionable recommendations. Finally, the system provides these recommendations as output to the user, ensuring that they receive precise solutions for managing and preventing crop diseases. This architecture seamlessly integrates user interaction, image analysis, and database retrieval to deliver a comprehensive disease management solution.

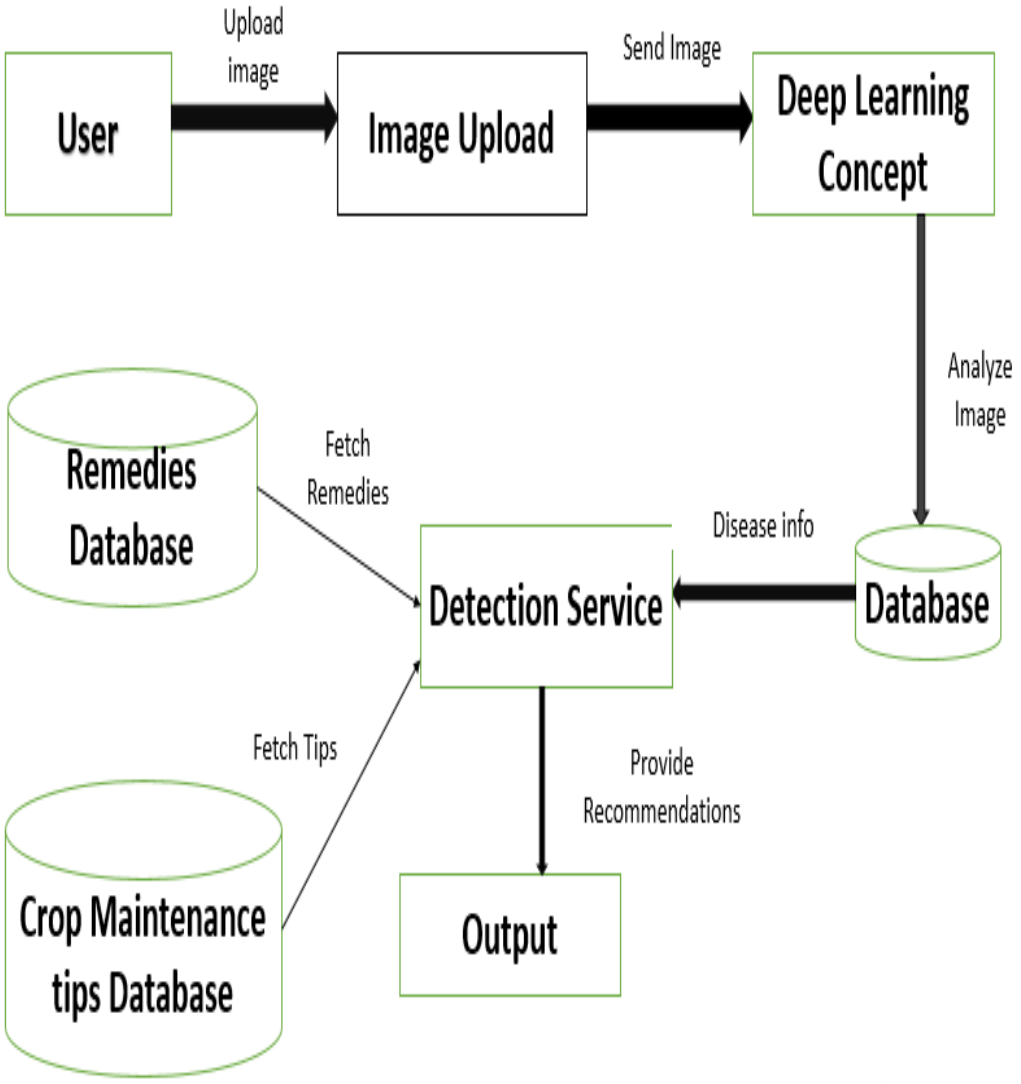


Fig .5.1 System Architecture

5.1 DATA FLOW DIAGRAM

The Data Flow Diagram (DFD) for a leaf disease detection system that provides remedies and crop maintenance involves a series of steps that ensure the user receives accurate and actionable information. First, the user uploads an image of a leaf, which is the starting point for the system. The uploaded image is processed by the system to ensure it is in the correct format and ready for analysis. Once validated, the image is sent to the deep learning model, which analyzes the leaf's features, such as discoloration or spots, to detect any signs of disease.

After identifying the disease, the system queries two key databases: the disease database to get detailed information about the detected disease, and the remedies database for treatment options. In parallel, the system also accesses the crop maintenance database to retrieve guidelines for maintaining healthy plants and preventing future diseases. The system then combines the information from these databases to generate a set of recommendations, including treatments for the disease and preventative care tips for the user.

Finally, the system presents these recommendations to the user in a clear and easy-to-understand format. The output includes the name of the disease, suggested remedies (such as pesticides or fertilizers), and crop maintenance tips for overall plant health. This DFD ensures a streamlined process that enables the user to effectively manage plant health through accurate disease detection and practical advice.

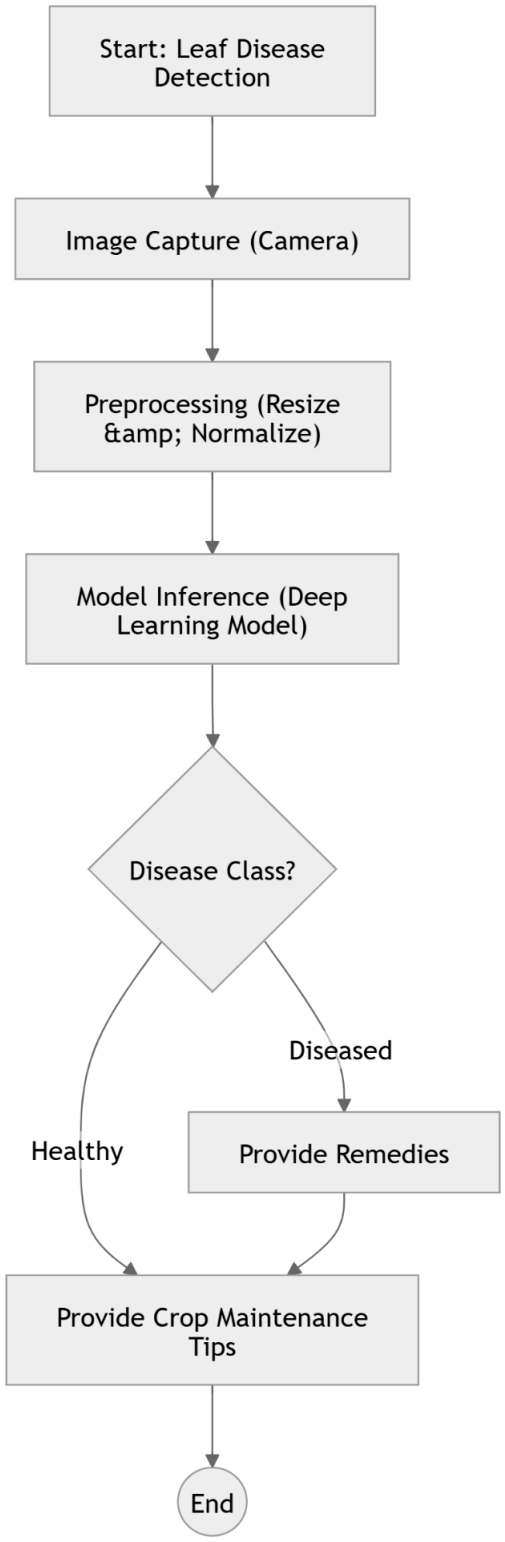


Fig 5.2 Flow chart

5.2 UML DIAGRAM

5.2.1 USE CASE DIAGRAM

A use case diagram for a leaf disease detection system involves several key components. The primary actor is the **Farmer**, who uploads images of leaves to the **System** for analysis. The system detects any diseases and provides a **Diagnosis**, along with suggested **Remedies**. Additionally, the farmer can seek advice from an **Agricultural Expert**. The system also stores user data and allows the farmer to access historical information about previous diagnoses. This diagram illustrates the interactions between the farmer, the system, and the expert, highlighting the essential functionalities for effective leaf disease management.

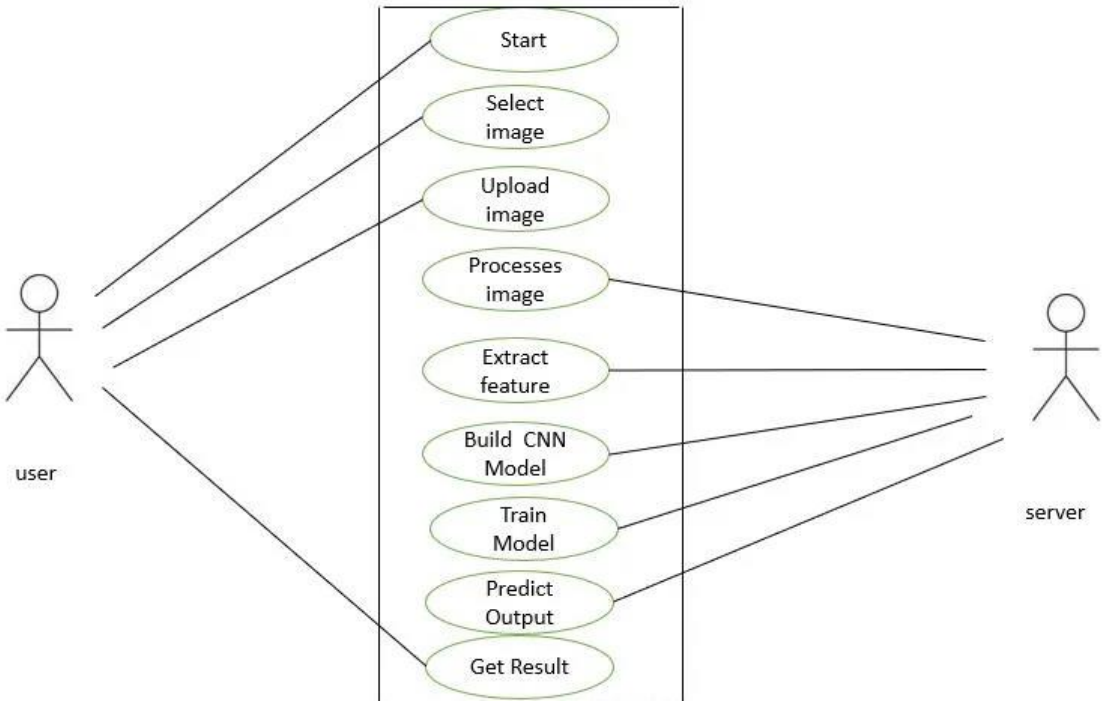


Fig 5.3 Use Case Diagram

5.2.2 ACTIVITY DIAGRAM

The leaf disease detection process begins when a user uploads a photo of a leaf. The system then preprocesses the image to enhance its features before using algorithms to detect any diseases present. Upon identifying a disease, it classifies it and presents the user with suitable remedies or treatment options. The user can review these suggestions and choose how to proceed, whether they want to apply the remedies or seek additional information.

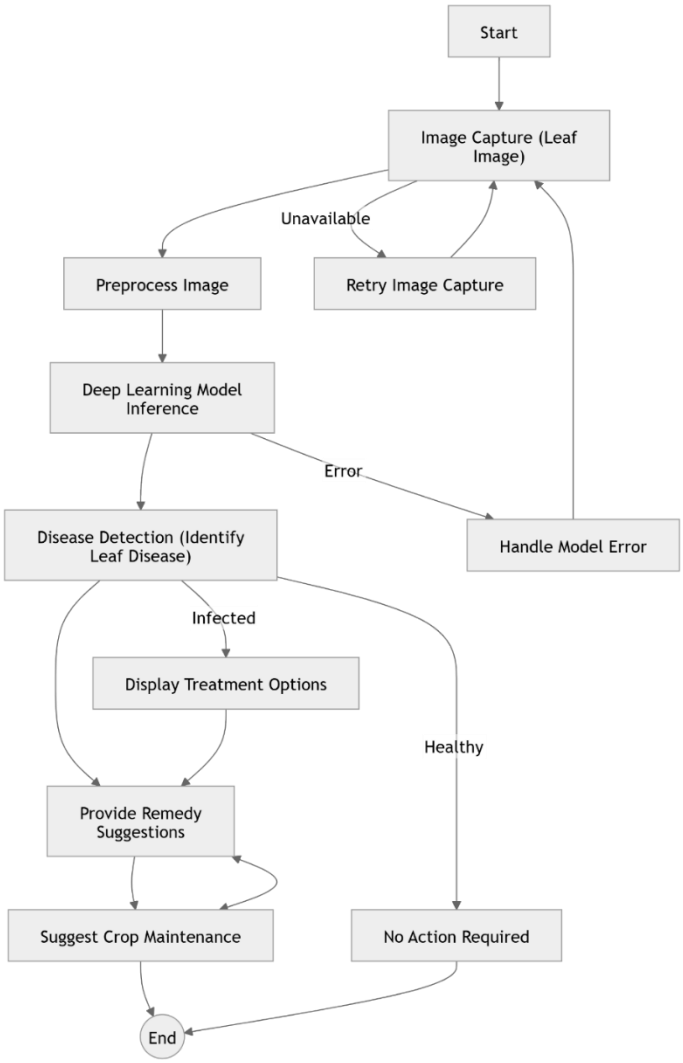


Fig 5.4 Activity Diagram

5.2.3 SEQUENCE DIAGRAM

In the leaf disease detection system, the farmer uploads a leaf image, which the system preprocesses and analyzes using a deep learning model (CNN). The model classifies the leaf as healthy or diseased. If a disease is detected, the system provides information about the disease and recommends remedies, such as applying specific pesticides, adjusting watering, or removing affected leaves. The system also suggests crop maintenance practices like soil treatment, crop rotation, or improving plant care to prevent future diseases.

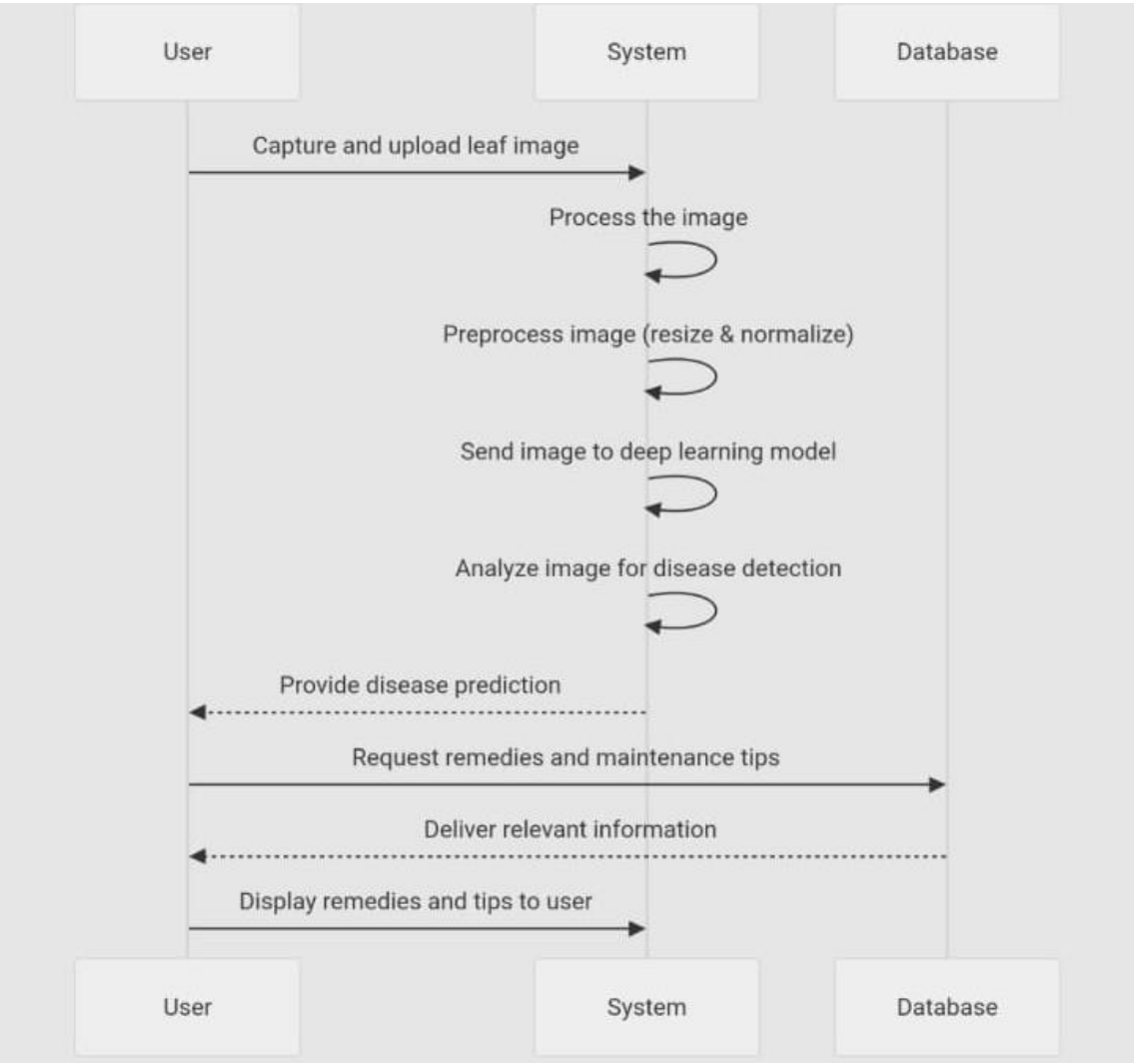


Fig. 5.5 Sequence Diagram

5.3 SYSTEM STUDY

5.3.1 FEASIBILITY STUDY

The leaf disease detection system using deep learning is technically feasible with CNNs for accurate image classification, operationally viable through easy integration with mobile apps and cloud deployment, and economically beneficial by reducing crop loss, optimizing pesticide use, and increasing yields.

5.3.1.1 TECHNICAL FEASIBILITY

The technical feasibility of the leaf disease detection system is high, as it leverages advanced deep learning models like CNNs to accurately classify leaf images and detect diseases. With cloud-based infrastructure for scalable real-time analysis and preprocessing techniques to handle environmental variations, the system is both efficient and robust. Economically, the system offers significant benefits by reducing crop loss, optimizing pesticide use, and improving overall yield, leading to cost savings and increased profits for farmers. The initial development costs are outweighed by the long-term savings and increased productivity, making it a financially viable solution for modern agriculture.

5.3.1.2 ECONOMICAL FEASIBILITY

The economic feasibility of the leaf disease detection system is strong, as it offers significant cost savings by preventing crop loss through early disease detection, optimizing pesticide use, and improving overall crop health. The initial costs, including software development, data collection, and cloud infrastructure, are outweighed by the long-term benefits of increased yields, reduced pesticide expenditure, and more efficient resource management.

CHAPTER 6

SYSTEM REQUIREMENTS

6.1 SOFTWARE REQUIREMENTS

Operating System: Windows 10 or Higher

Coding Language: Python 3.7

Environment Setup : leaf_disease_env\Scripts\activate, python -m venv leaf_disease_env

Platform : Idle

6.2 HARDWARE REQUIREMENTS

Processor: Intel Core i7

Hard Disk: Not Applicable

Monitor: Responsive to all Screen Sizes (Responsive Design)

RAM: Minimum 2GB

6.3 HARDWARE DESCRIPTION

The Leaf Disease Detection System is designed to identify and classify plant leaf diseases efficiently, leveraging **Windows 10 or higher** as the operating platform for stability and performance. Developed using **Python 3.7**, the project utilizes the **IDLE environment** for its simplicity and seamless integration with Python libraries. By applying advanced image processing techniques with libraries like OpenCV and machine learning frameworks such as TensorFlow, the system analyzes features like

color, texture, and shape to detect diseases accurately. This solution aims to assist farmers and researchers in diagnosing plant health issues quickly, enabling timely intervention and promoting sustainable agricultural practices.

6.4 SOFTWARE DESCRIPTION

6.4.1 Integrated Development and Learning Environment(IDLE)

IDLE (Integrated Development and Learning Environment) is the default editor and IDE for Python, designed for simplicity and ease of use. It is particularly suited for projects like leaf disease detection for the following reasons:

- **Ease of Setup:** IDLE is pre-installed with Python, eliminating additional installation steps.
- **User-Friendly Interface:** Its straightforward interface is ideal for coding, debugging, and running Python scripts in a single window.
- **Lightweight:** IDLE is lightweight and efficient, making it suitable for smaller to medium-scale projects like disease detection.
- **Visualization:** IDLE's shell allows immediate visualization of data outputs, making debugging and data analysis faster.

6.4.3 Environment Setup

Both commands are related to software configuration and are part of Python's environment management tools.

- **python -m venv leaf_disease_env:**
    - This command creates a virtual environment called leaf_disease_env using Python.
    - **Software:** It sets up an isolated environment for Python projects, allowing you to install and manage dependencies specific to the project without affecting other Python projects on your system.

- **leaf_disease_env\Scripts\activate:**
  - This command activates the virtual environment on Windows.
  - **Software:** It activates the virtual environment so that you can work within it, ensuring that the Python interpreter and libraries used are those associated with the leaf_disease_env environment.

6.5 FUTURE WORK

- **Real-Time Monitoring**: Use IoT devices and drones for automatic crop monitoring.
- **Better AI Models**: Use advanced machine learning and deep learning for more accurate results.
- **Larger Databases**: Create a global collection of leaf images to improve disease detection.
- **Explainable AI**: Show farmers which parts of the leaf are affected and why.
- **Multilingual Support**: Add language options to help farmers from different regions.
- **Sustainability**: Include features to detect environmental issues and promote eco-friendly farming.

CHAPTER 7

SYSTEM TESTING

System testing for a leaf disease detection system involves thoroughly evaluating the entire system to ensure it functions as intended, meets user requirements, and delivers accurate results. This process tests every component, from image upload and disease detection to the retrieval of remedies and maintenance tips, ensuring seamless integration and reliable performance. The goal is to verify that the system can handle different scenarios, including various image inputs, large datasets, and simultaneous user requests, without errors or delays. Additionally, system testing ensures the accuracy of disease detection, the relevance of provided recommendations, and the security of user data. This comprehensive testing approach ensures the system is ready for real-world use, providing farmers and agricultural experts with a dependable tool for managing plant health.

7.1 TESTING STEPS

7.1.1 UNIT TESTING

- Ensures the accuracy of each component, such as the deep learning model's ability to identify diseases or the database's ability to retrieve correct disease information.
- Helps detect bugs in isolated modules, like image upload handling or output formatting.
- Verifies that each feature, such as fetching remedies or tips, works as intended before integration.

7.1.2 INTEGRATION TESTING

- Confirms smooth data flow between modules, such as passing the image from upload to analysis and retrieving relevant disease information.
- Ensures that combined operations, like generating recommendations from database queries, work seamlessly.
- Validates the complete process, ensuring the system delivers accurate results and recommendations to users.

7.1.3 FUNCTIONAL TESTING

- **Purpose:** Verifies that all system features and functionalities work as expected based on requirements.
- **Focus:** Testing individual functions, data flow, and output accuracy.
- **Example in Leaf Disease Detection:**
  - Check if the image upload accepts valid images and rejects invalid formats.
  - Ensure the deep learning model correctly detects diseases for different types of leaf images.
  - Test if the system fetches accurate remedies and tips from the database.

7.1.4 WHITE BOX TESTING

- **Focus:** Examines the internal workings, logic, and code structure of the system.
- **Purpose:** To ensure the code is efficient, error-free, and meets the design specifications.
- **Example in Leaf Detection:**
  - Check the deep learning model's implementation to ensure it processes features correctly.

- Verify database query execution for retrieving disease information and remedies.

7.1.5 BLACK BOX TESTING

- **Focus**: Tests the system's functionality without looking at the internal code or structure.
- **Purpose**: To ensure the system behaves as expected based on input and output.
- **Example in Leaf Detection**:
  - Test invalid inputs like corrupted files or non-leaf images to ensure proper error handling.

Fig.7.1 Black Box Testing

7.1.6 ACCEPTANCE TESTING

- **Purpose:** Validates the system's readiness for real-world use by ensuring it meets user expectations and requirements.
- **Focus:** User satisfaction and usability of the system.
- **Example in Leaf Disease Detection:**
    - Conduct tests with actual users (e.g., farmers) to ensure the recommendations are clear and actionable.
    - Verify that the system provides a seamless workflow, from image upload to displaying disease remedies.
    - Gather user feedback on the relevance and ease of using the system.

CHAPTER 8
CONCLUSION AND FUTURE WORK

In this project, we developed an effective leaf disease detection system using Deep Learning and Python 3.7. By utilizing core libraries such as TensorFlow 2.11.0 for neural network implementation, OpenCV for image pre-processing and augmentation, NumPy for array handling, and Pandas for data management, we created a robust system for identifying plant diseases. We utilized Matplotlib and Seaborn for visualizing data trends and results, ensuring an insightful analysis of the model's performance.

The system employs **Convolutional Neural Networks (CNNs)**, which consist of multiple layers, including **input**, **convolutional**, **pooling**, **dense**, and **output layers**. The **ReLU** activation function is used for hidden layers to introduce non-linearity, while **Softmax** is applied in the output layer for multi-class classification. The **Adam optimizer** ensures efficient training and model convergence.

To enhance model performance, the dataset was pre-processed with techniques such as **image resizing** to standard dimensions, **normalization** for optimization, and **augmentation** (rotation, flipping) to diversify the training data. The dataset was split into **80% training** and **20% testing**, ensuring that the model was well-trained and capable of generalizing to unseen data.

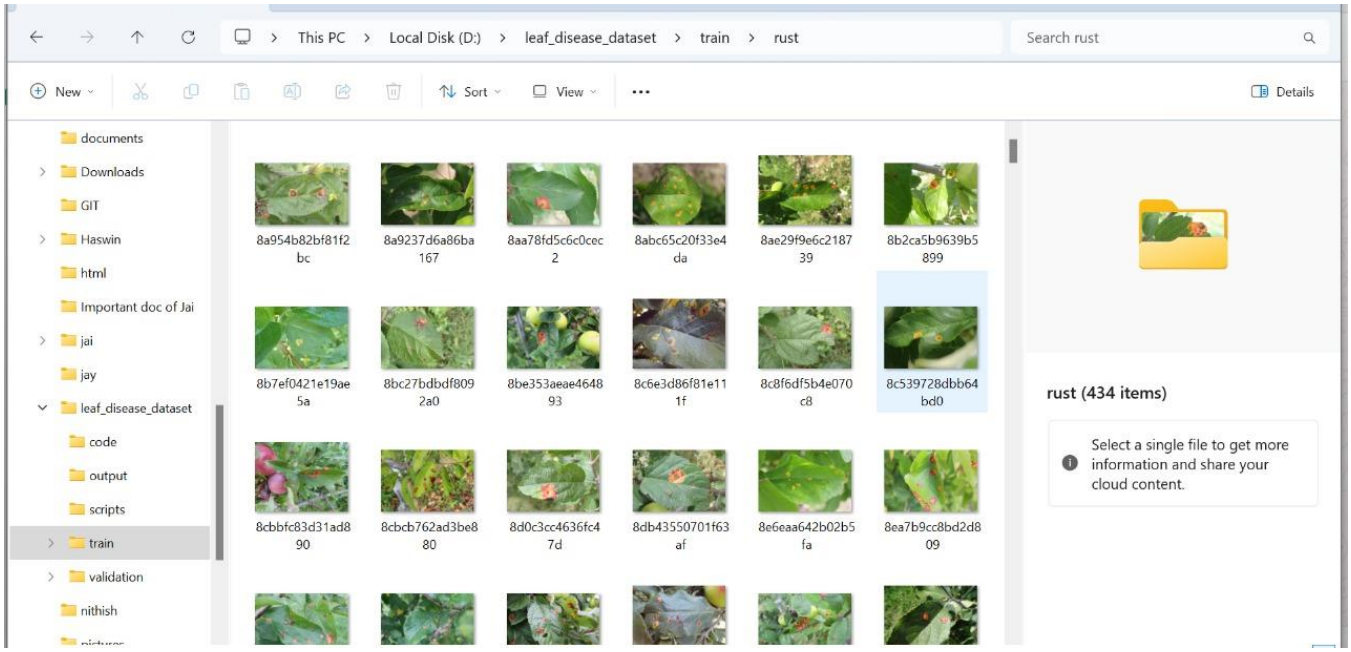
The result is a highly accurate model capable of detecting leaf diseases, with output that includes not only the diagnosis but also **remedies and crop maintenance recommendations**. This system aids farmers in taking timely actions, helping to improve crop health and yield while promoting sustainable farming practices.

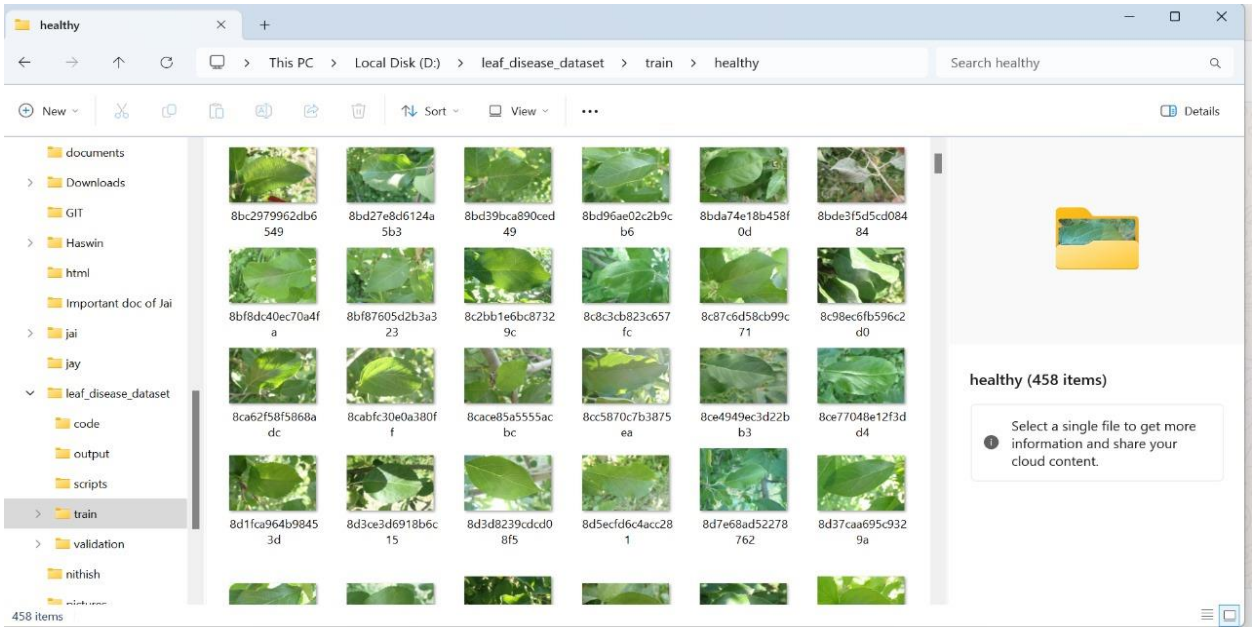
Future Work:

The future work for the project includes several exciting directions aimed at enhancing its functionality and reach. One key development is mobile application development, which will enable the platform to be accessible across a wider audience, allowing users to access resources and tools directly from their smartphones. Additionally, AI-driven insights will be incorporated, using machine learning algorithms to provide personalized mental health resources tailored to individual needs, ensuring more relevant and impactful support. To further enhance user engagement, gamification features such as rewards, challenges, and leaderboards will be introduced, fostering a more interactive and motivating experience. Finally, real-time data integration will be explored by incorporating live APIs that provide dynamic updates on sentiment trends, enabling users to stay informed with up-to-date insights and improving the platform's responsiveness to changes in user behavior and needs. These advancements will create a more comprehensive and accessible mental health resource, driving further innovation in user experience and effectiveness.

APPENDIX A

OUTPUT SCREENSHOTS



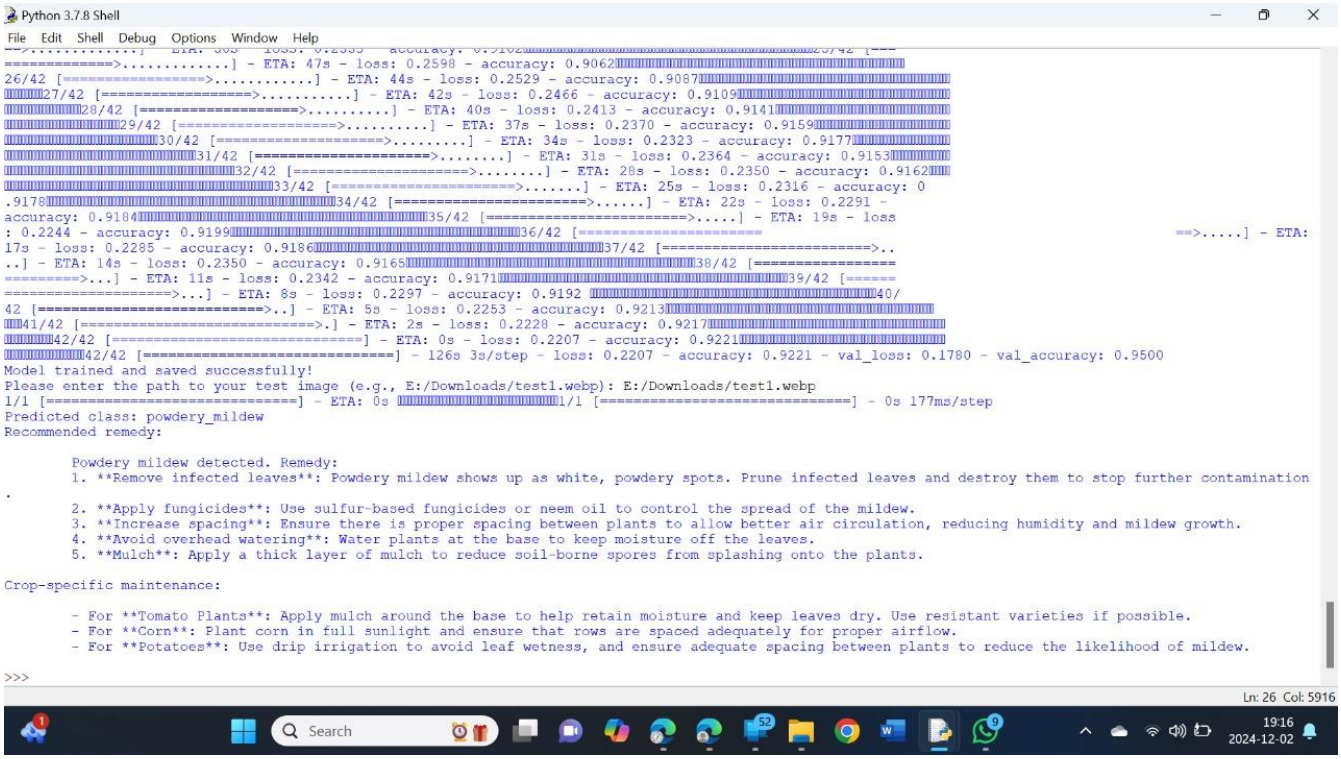


DataSet

Classification Leaf


Python 3.7.8 Shell
File Edit Shell Debug Options Window Help
42/42 [==========] - ETA: 0s - loss: 0.4776 - accuracy: 0.8230
42/42 [==========] - 188s 4s/step - loss: 0.4776 - accuracy: 0.8230 - val_loss: 0.3489 - val_a
ccuracy: 0.9000
Epoch 5/10
1/42 [..........] - ETA: 3:39 - loss: 0.2743 - accuracy: 0.9688
2/42 [>.........] - ETA: 3:03 - loss: 0.3554 - accuracy: 0.9062
3/42 [=>........] - ETA: 3:04 - loss: 0.3909 - accuracy: 0.8958
4/42 [=>........] - ETA: 2:54 - loss: 0.4482 - accuracy: 0.8516
5/42 [==>.......] - ETA: 2:54 - loss: 0.4508 - accuracy: 0.8500
6/42 [===>......] - ETA: 2:46 - loss: 0.4493 - accuracy: 0.8385
7/42 [====>.....] - ETA: 2:37 - loss: 0.4194 - accuracy:
0.8482
8/42 [====>.....] - ETA: 2:31 - loss: 0.3979
- accuracy: 0.8555
9/42 [=====>....] - ETA: 2:12 -
loss: 0.3945 - accuracy: 0.8534
10/42 [======>...] -
ETA: 2:10 - loss: 0.3798 - accuracy: 0.8624
11/42 [======>.....
.....] - ETA: 2:06 - loss: 0.3651 - accuracy: 0.8606
12/42 [=======>
.....] - ETA: 2:02 - loss: 0.3667 - accuracy: 0.8619
13
/42 [========>....] - ETA: 1:57 - loss: 0.3670 - accuracy: 0.8629
14/42 [=========>...] - ETA: 1:53 - loss: 0.3619 - accuracy: 0.8662
15/42 [=========>...] - ETA: 1:50 - loss: 0.3814 - accuracy: 0.8603
16/42 [==========>..] - ETA: 1:46 - loss: 0.3681 - accuracy: 0.8673
17/42 [==========>..] - ETA: 1:42 - loss: 0.3545 - accuracy: 0.8716
18/42 [==========>..] - ETA: 1:38 - loss: 0.4071 - accuracy: 0.8664
19/42 [===========>.] - ETA: 1:33 - loss: 0.3909 - accuracy: 0.
8720
20/42 [===========>.] - ETA: 1:29 - loss: 0.3918 -
accuracy: 0.8706
21/42 [==========>...] - ETA: 1:25 - lo
ss: 0.3868 - accuracy: 0.8708
22/42 [==========>...] - E
TA: 1:21 - loss: 0.3812 - accuracy: 0.8724
23/42 [==========>....
.....] - ETA: 1:17 - loss: 0.3821 - accuracy: 0.8711
24/42 [=========
=======>.....] - ETA: 1:13 - loss: 0.3820 - accuracy: 0.8700
25/
42 [==========>....] - ETA: 1:09 - loss: 0.3896 - accuracy: 0.8650
26/42 [==========>....] - ETA: 1:06 - loss: 0.3862 - accuracy: 0.8654
27/42 [==========>....] - ETA: 1:02 - loss: 0.3839 - accuracy: 0.8658
28/42 [==========>....] - ETA: 58s - loss: 0.3854 - accuracy: 0.8673
29/42 [==========>....] - ETA: 54s - loss: 0.3902 - accuracy: 0.8653
30/42 [==========>....] - ETA: 51s - loss: 0.3898 - accuracy: 0.8657
31/42 [==========>....] - ETA: 47s - loss: 0.3878 - accuracy: 0.8670
32/42 [==========>....] - ETA: 42s - loss: 0.3808 - accuracy
: 0.8703
33/42 [==========>...] - ETA: 38s - loss: 0.3771
- accuracy: 0.8694
34/42 [==========>...] - ETA: 34s - lo
ss: 0.3744 - accuracy: 0.8696

Processing


Python 3.7.8 Shell
File Edit Shell Debug Options Window Help
Model trained and saved successfully!
Please enter the path to your test image (e.g., E:/Downloads/test1.webp): E:/Downloads/test1.webp
Predicted class: powdery_mildew
Recommended remedy:
Powdery mildew detected. Remedy:
1. **Remove infected leaves**: Powdery mildew shows up as white, powdery spots. Prune infected leaves and destroy them to stop further contamination
.
2. **Apply fungicides**: Use sulfur-based fungicides or neem oil to control the spread of the mildew.
3. **Increase spacing**: Ensure there is proper spacing between plants to allow better air circulation, reducing humidity and mildew growth.
4. **Avoid overhead watering**: Water plants at the base to keep moisture off the leaves.
5. **Mulch**: Apply a thick layer of mulch to reduce soil-borne spores from splashing onto the plants.
Crop-specific maintenance:
- For **Tomato Plants**: Apply mulch around the base to help retain moisture and keep leaves dry. Use resistant varieties if possible.
- For **Corn**: Plant corn in full sunlight and ensure that rows are spaced adequately for proper airflow.
- For **Potatoes**: Use drip irrigation to avoid leaf wetness, and ensure adequate spacing between plants to reduce the likelihood of mildew.
>>>
Ln: 26 Col: 5916
Search
19:16
2024-12-02

Output

APPENDIX B

SOURCE CODE:

leaf.py

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras import layers, models

# Define directories
train_dir = 'D:/leaf_disease_dataset/train'
validation_dir = 'D:/leaf_disease_dataset/validation'

# Image dimensions and batch size

image_size = (128, 128)  # Resize to 128x128 or (224, 224) if needed
batch_size = 32

# Data Augmentation for Training Set

train_datagen = ImageDataGenerator(
    rescale=1.0/255,           # Normalize pixel values between 0 and 1
    rotation_range=40,         # Random rotations
    width_shift_range=0.2,     # Random width shifts
    height_shift_range=0.2,    # Random height shifts
    shear_range=0.2,           # Random shear
    zoom_range=0.2,            # Random zoom
    horizontal_flip=True,      # Random horizontal flip
    fill_mode='nearest'        # How to fill in newly created pixels
)

# Data Preprocessing for Validation Set

validation_datagen = ImageDataGenerator(rescale=1.0/255)

# Create Data Generators
```

```
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=image_size,      # Resize images to 128x128
    batch_size=batch_size,
    class_mode='categorical'     # For multi-class classification
)

validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    target_size=image_size,      # Resize images to 128x128
    batch_size=batch_size,
    class_mode='categorical'     # For multi-class classification
)

# Print class indices

print(f"Classes found: {train_generator.class_indices}")
print(f"Total training images: {train_generator.samples}")
print(f"Total validation images: {validation_generator.samples}")

# Build the CNN Model

model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),  # First
        Convolutional Layer
    layers.MaxPooling2D(2, 2),  # Max Pooling Layer
    layers.Conv2D(64, (3, 3), activation='relu'),  # Second Convolutional Layer
    layers.MaxPooling2D(2, 2),  # Max Pooling Layer
    layers.Conv2D(128, (3, 3), activation='relu'),  # Third Convolutional Layer
    layers.MaxPooling2D(2, 2),  # Max Pooling Layer
    layers.Flatten(),  # Flatten layer to prepare for Dense layer
    layers.Dense(128, activation='relu'),  # Fully Connected layer
    layers.Dense(3, activation='softmax')  # Output layer (3 classes)
])

# Compile the Model

model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',  # For multi-class classification
    metrics=['accuracy']
```

```
)

# Train the Model
history = model.fit(
    train_generator,
    epochs=10,  # Change epochs as needed
    validation_data=validation_generator
)
model.save('leaf_disease_detection_model.h5')
```

leaf1.py

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras import layers, models
import os
import numpy as np
from tensorflow.keras.preprocessing import image

train_dir = 'D:/leaf_disease_dataset/train'  # Modify this path if needed
validation_dir = 'D:/leaf_disease_dataset/validation'  # Modify this path if needed

image_size = (128, 128)  # Resize images to 128x128 pixels

# Set up the ImageDataGenerators for training and validation data

train_datagen = ImageDataGenerator(
    rescale=1./255,  # Normalize pixel values to [0, 1]
    rotation_range=40,  # Random rotations
    width_shift_range=0.2,  # Random horizontal shift
    height_shift_range=0.2,  # Random vertical shift
    shear_range=0.2,  # Random shear transformation
    zoom_range=0.2,  # Random zoom
    horizontal_flip=True,  # Random horizontal flip
    fill_mode='nearest'  # Fill pixels that are left blank by transformations
)

validation_datagen = ImageDataGenerator(rescale=1./255)

# Load and preprocess the training and validation data
```

```
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=image_size,
    batch_size=32,
    class_mode='categorical'  # For multi-class classification
)

validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    target_size=image_size,
    batch_size=32,
    class_mode='categorical'
)

# Build the model using Convolutional Neural Networks (CNN)

model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
    layers.Flatten(),
    layers.Dense(512, activation='relu'),
    layers.Dense(3, activation='softmax')  # 3 classes (healthy, rust, powdery_mildew)
])

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(
    train_generator,
    epochs=10,  # You can increase this depending on your dataset and hardware
    validation_data=validation_generator
)

# Save the trained model

model.save('leaf_disease_detection_model.h5')
```

```
print("Model trained and saved successfully!")

# Dictionary of remedies and crop maintenance tips for each disease class

remedies = {
    'healthy': {
        'remedy': """
        No disease detected. Your crops are healthy and thriving.
        General maintenance tips:
        1. Ensure consistent watering, avoiding overwatering or underwatering.
        2. Provide adequate sunlight for at least 6-8 hours a day.
        3. Perform soil testing for pH and nutrient levels regularly.
        4. Use organic fertilizers during the growing season to encourage healthy growth.
        5. Inspect your crops periodically for pests, weeds, or diseases, even when they
           seem healthy.
        """,
        'crop_maintenance': """
        - For Tomato Plants: Prune lower leaves to avoid soil contact and improve air
          circulation. Mulch around the base to retain moisture.
        - For Corn: Ensure a minimum of 30 cm spacing between rows for proper airflow.
          Fertilize with a nitrogen-rich fertilizer during early growth stages.
        - For Potatoes: Use raised beds to avoid waterlogging and ensure proper drainage.
          Apply compost to enrich the soil with nutrients.
    },
    'rust': {
        'remedy':
        Rust disease detected. Remedy:
        1. Remove infected leaves: Rust fungi appear as orange or red spots. Remove and
           destroy infected leaves to prevent further spread.
        2. Apply fungicides: Use copper-based or sulfur-based fungicides to treat the
           infection.
        3. Increase plant spacing: Improve air circulation around plants by increasing
           spacing to reduce humidity.
        4. Rotate crops: Avoid planting the same crop in the same soil each year to reduce
           rust build-up.
        5. Water carefully: Water plants at the base to avoid wetting leaves and promoting
           rust growth.
        """,
        'crop_maintenance': """
        - For Tomato Plants: Improve airflow by spacing plants apart and pruning any
          dense foliage. Water plants at the soil level.
```

```
        - For Corn: Rotate crops annually to avoid rust spores overwintering in the soil.
          Apply a fungicide early in the season as a preventive measure.
        - For Potatoes: After harvesting, rotate the crop with non-solanaceous plants to
          reduce the risk of rust reinfection.
        """
    },
    'powdery_mildew': {
        'remedy':
        Powdery mildew detected. Remedy:
        1. Remove infected leaves: Powdery mildew shows up as white, powdery spots.
           Prune infected leaves and destroy them to stop further contamination.
        2. Apply fungicides: Use sulfur-based fungicides or neem oil to control the spread
           of the mildew.
        3. Increase spacing: Ensure there is proper spacing between plants to allow better
           air circulation, reducing humidity and mildew growth.
        4. Avoid overhead watering: Water plants at the base to keep moisture off the
           leaves.
        5. Mulch: Apply a thick layer of mulch to reduce soil-borne spores from splashing
           onto the plants.

        'crop_maintenance':
        - For Tomato Plants: Apply mulch around the base to help retain moisture and keep
          leaves dry. Use resistant varieties if possible.
        - For Corn: Plant corn in full sunlight and ensure that rows are spaced adequately
          for proper airflow.
        - For Potatoes: Use drip irrigation to avoid leaf wetness, and ensure adequate
          spacing between plants to reduce the likelihood of mildew.

    }
}

# Request the user to input the path for the test image

img_path = input("Please enter the path to your test image (e.g.,
          E:/Downloads/test1.webp): ")

# Check if the file exists

if not os.path.exists(img_path):
    print("Error: The image file does not exist. Please check the path and try again.")
else:
```

```
test_img = image.load_img(img_path, target_size=image_size)
test_img_array = image.img_to_array(test_img)
test_img_array = np.expand_dims(test_img_array, axis=0)  # Add batch dimension
test_img_array = test_img_array / 255.0  # Normalize the image

predictions = model.predict(test_img_array)
class_names = list(train_generator.class_indices.keys())  # Get class names from the
      training set

predicted_class = class_names[np.argmax(predictions)]
print(f"Predicted class: {predicted_class}")

# Output the remedy for the predicted disease along with crop-specific maintenance

remedy = remedies.get(predicted_class, {"remedy": "No remedy available for this
      disease.", "crop_maintenance": "No maintenance tips available."})
print(f"Recommended remedy:\n{remedy['remedy']}")
print(f"Crop-specific maintenance:\n{remedy['crop_maintenance']}")
```

REFERENCES

1. M. Vinitha, Mallikarjuana Nandhi. Plant Leaf Disease Detection Using Convolution Neural Network

2. Sumaya Mustofa, MD Mehedi Hasan Munna, Yousuf. A Comprehensive Review on Leaf Disease Detection Using Deep Analysis.

3. Chohan, Murk & Khan, Adil & Chohan, Rozina & Hassan, Muhammad. Plant Disease Detection using Deep Learning.

4. S. Arivazhagan, R. Newlin Shebiah, S. Ananthi, S. Vishnu Varthini.Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features

5. Mohanty SP, Hughes DP. Using Deep Learning for Image-Based Plant Disease Detection.

6. Vijeta Shrivastava, Pushpanjali, Samreen Fatima, Indrajit Das. Plant leaf diseases detection and classification using machine learning.

7. Md. Z. Hasan, Md. S. Ahamed, A. Rakshit and K. M. Z. Hasan. Recognition of jute diseases by leaf image classification using convolutional neural network.

8. S. Barburiceanu, S. Meza, B. Orza, R. Malutan and R. Terebes. sConvolutional neural networks for texture feature extraction. Applications to leaf disease classification in precision agriculture

9. S. P. Mohanty, D. P. Hughes and M. Salathé. Using deep learning for image-based plant disease detection.

10. S. M. Hassan, A. K. Maji, M. Jasiński, Z. Leonowicz and E. Jasińska. Identification

of plant-leaf diseases using CNN and transfer-learning approach.

11. Anand H. Kulkarni, Ashwin Patil R. K. .Applying image processing technique to detect plant diseases.

12. F. Argenti,L. Alparone,G. Benelli .Fast algorithms for texture analysis

13. P. Revathi, M. Hemalatha. Classification of Cotton Leaf Spot Diseases Using Image Processing Edge Detection Techniques.

14. Tushar H. Jaware, Ravindra D. Badgujar and Prashant G. Patil, Crop disease detection using image segmentation.

15. Prof.Sanjay B. Dhaygude, Mr.Nitin P. Kumbhar, Agricultural plant Leaf Disease Detection Using Image Processing.