

# **COMPARITIVE STUDY OF DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF DISEASES IN CROP**

Submitted in partial fulfillment of the requirements for the award of  
Bachelor of Engineering degree in Computer Science and Engineering

by

**T. CHARAN KUMAR REDDY (Register .No - 40111350)**

**V.CHIRANJEEVI ACHARI (Register.No - 40111373)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Category-1 University by UGC**

**Accredited with Grade "A++" by NAAC | 12B Status by UGC | Approved by AICTE**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

**CHENNAI - 600119**

**APRIL - 2024**

# SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

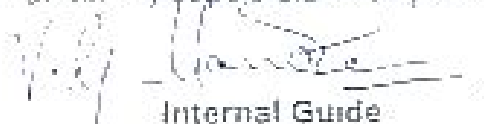
Accredited "A++" by NAAC | Approved by AICTE

[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### BONAFIDE CERTIFICATE

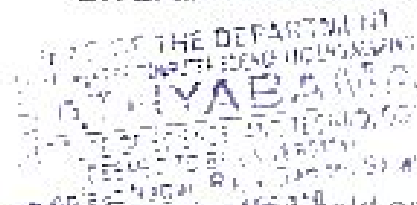
This is to certify that this Project Report is the bonafide work of T.CHARAN KUMAR REDDY(40111350) and V.CHIRANJEEVI ACHARI(40111373) who carried out the Project entitled "COMPARITIVE STUDY OF DEEP TRANSFER LEARNING MODELS FOR CLASSIFYING DIFFERENT TYPES OF DISEASES IN CROP" under my supervision from November 2023 to April 2024

  
Internal Guide

Ms. V. GOWRI MANOHARI, M.E

  
Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.



Submitted for Viva voice Examination held on 12/04/2024

  
Internal Examiner

  
External Examiner

## DECLARATION

I, T.CHARAN KUMAR REDDY(40111350), hereby declare that the Project Report entitled "COMPARITIVE STUDY OF DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF DISEASES IN CROP" done by me under the guidance of Ms.V.GOWRI MANOHARI,M.E is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE : 03/04/2023

PLACE : Chennai



SIGNATURE OF THE CANDIDATE

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms.V.GOWRI MANOHARI,M.E** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

## **ABSTRACT**

The early detection of plant diseases is essential to preventing crop losses in terms of quantity and productivity. Farmers typically identify illnesses using their prior knowledge or by spending a great deal of time, effort, and experience doing so. It is exceedingly challenging to manually monitor plant diseases. In the development of automatic pathogens diagnosis machines, paddy disease detection is crucial. To identify previously known bacterial leaf blight, brown spot, leaf blast, leaf smut, and other narrow illnesses in prior knowledge, we presented a VGG 16 model in this work. Images of various categories diseases are stored in a special data repository, with 80 percent of the data used for training and 20 percent for sample testing. resnet, Inception v2 and Mobile Net were all used in the model-building process to add benefits. The results of the interpretation show that there is high level picture classification accuracy and low mistake rate. As a result, our model excels at identifying paddy diseases and may be used in practical applications in daily life.

**Keywords: Deep learning, Convolution neural network, Image processing, Paddy disease detection, Transfer learning, VGG16, Resnet, Inception v2, Mobile Net, Pathogens**

## TABLE OF CONTENTS

| CHAPTER<br>NO | TITLE                                                             | PAGE<br>NO |
|---------------|-------------------------------------------------------------------|------------|
|               | <b>ABSTRACT</b>                                                   | v          |
|               | <b>LIST OF FIGURES</b>                                            | viii       |
| <b>1</b>      | <b>INTRODUCTION</b>                                               | 1          |
| <b>2</b>      | <b>LITERATURE SURVEY</b>                                          | 3          |
|               | 2.1 INFERENCES FROM LITERATURE SURVEY                             | 3          |
|               | 2.2 EXISTING SYSTEM                                               | 6          |
|               | 2.3 PROPOSED SYSTEM                                               | 7          |
| <b>3</b>      | <b>REQUIRMENTS ANALYSIS</b>                                       | 8          |
|               | 3.1 SCOPE OF THE PRESENT INVESTIGATION                            | 8          |
|               | 3.2 FEASIBILITY STUDY                                             | 10         |
|               | 3.3 SOFTWARE REQUIREMENTS AND<br>SPECIFICATION DOCUMENT           | 13         |
| <b>4</b>      | <b>EXPERIMENTAL OR MATERIALS AND<br/>METHODS, ALGORITHMS USED</b> | 14         |
|               | 4.1 SELECTED METHODOLOGY OR PROCESS<br>MODEL                      | 13         |
|               | 4.2 IMPLEMENTATION                                                | 18         |
|               | 4.3 DESCRIPTION ON PROGRAMMING<br>LANGUAGE AND SOFTWARE           | 19         |

|          |                                                         |           |
|----------|---------------------------------------------------------|-----------|
| 4.4      | ARCHITECTURE                                            | 21        |
| 4.5      | ALGORITHMS                                              | 22        |
| 4.6      | DESIGN PHASE                                            | 29        |
| 4.7      | MODEL DESCRIPTION                                       | 41        |
| <b>5</b> | <b>IMPLEMENTATION DETAILS</b>                           | <b>42</b> |
| 5.1      | TRAINING THE MODEL                                      | 42        |
| 5.2      | TESTING THE MODE                                        | 44        |
| 5.3      | DISEASES DETECTING                                      | 45        |
| <b>6</b> | <b>RESULTS AND DISCUSSION,<br/>PERFORMANCE ANALYSIS</b> | <b>48</b> |
| 6.1      | RESULT                                                  | 48        |
| 6.2      | DISCUSSION                                              | 48        |
| <b>7</b> | <b>SUMMARY AND CONCLUSION</b>                           | <b>50</b> |
| 7.1      | CONCLUSION                                              | 50        |
| 7.2      | FUTURE WORK                                             | 50        |
|          | <b>REFERENCES</b>                                       | <b>53</b> |
|          | <b>APPENDIX</b>                                         | <b>54</b> |
|          | <b>A.SOURCE CODE</b>                                    | <b>56</b> |
|          | <b>B.SCREENSHOTS</b>                                    | <b>59</b> |
|          | <b>C.RESEARCH PAPER</b>                                 | <b>61</b> |
|          | <b>D.CERTIFICATE</b>                                    | <b>68</b> |

## TABLE OF FIGURES

| <b>FIGURES<br/>NO</b> | <b>FIGURES NAME</b>   | <b>PAGE<br/>NO</b> |
|-----------------------|-----------------------|--------------------|
| 4.1                   | SYSTEM ARCHITECTURE   | 20                 |
| 4.2                   | GENERAL ARCHITECURE   | 21                 |
| 4.3                   | DATA FLOW DIAGRAM     | 29                 |
| 4.4                   | USE CASE DIAGRAM      | 30                 |
| 4.5                   | CLASS DIAGRAM         | 32                 |
| 4.6                   | SEQUENCE DIAGRAM      | 35                 |
| 4.7                   | COLLABORATION DIAGRAM | 38                 |
| 4.8                   | ACTIVITY DIAGRAM      | 40                 |

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 INTRODUCTION**

One of the most important agricultural procedures is the classification and identification of paddy plant diseases at an early stage. Genetic disorder infection causes farmers to lose a lot of money every year. Consequently, a prompt, accurate, and early identification of the illness both reduces product loss and raises product quality. As a result, it contributes to the growth of the nation's economy. In conventional medical procedures, the diagnosis of these diseases is based either on the pathogens' visible symptoms or on pathogen identification in a lab .The visual evaluation of the disease area is a subjective matter that may not provide a reliable diagnosis. On the other hand, pathogen growing takes time and can fail to produce results when needed, making pathogen identification in a lab a laborious operation. We looked at pictures of several plant diseases.

#### ***AIM OF THE PROJECT***

This project's primary goal is to develop a deep learning model for the detection and classification of paddy crop diseases using transfer learning and image processing, with all of the work being done in deep learning.

#### ***PROJECT DOMAIN***

The project domain is related to Deep Learning Technology since we are utilizing Deep Learning algorithm to predict plant illnesses. The technique we utilize in this case is the CNN Algorithm are VGG16, Resnet, Inception v2, Mobile Net for crop Disease Prediction

## ***SCOPE OF THE PROJECT***

The project's goal is to help increase yields by preventing crop devastation. Many farmers wish to accept advanced agriculture, but many are unable to do so for a variety of reasons, including lack of knowledge about cutting-edge technology and significant technical costs. It has frequently been noted that paddy diseases are challenging to manage because their prevalence varies depending on the environment.

Many different diseases affect plants, such as leaf spot, dryness, color changes, etc. Some of them might ruin the entire crop if not identified and treated in a timely manner. This could result in a farmer suffering a significant loss and lowering the yield of staple crops. We use image processing techniques to detect the disease of the image based on photos taken of the leaves, which automatically detects the disease of the plant. This destruction of the crops or lower yields result from a lack of knowledge about diseases and inappropriate pesticide usage, among other factors. Moreover, find crop diseases before they spread to the crop's outer layers.

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 INFERENCES FROM LITREATURE SURVEY**

[1] Houda Ochi et al, Increasing agricultural production is a critical requirement to meet demand due to the rapid population expansion. Early detection of agricultural diseases is critical to preventing production loss. Manually monitoring leaf diseases is a laborious activity that involves extensive knowledge of plant pathogens, a lot of work, and long processing times. Researchers have created and analyzed a number of algorithms based on image processing, deep learning, and machine learning for these goals, and they frequently produce meaningful findings.

[2] Ahmed Samit Hate metal, The classification and analysis of images now heavily rely on deep learning. Diseases in maize reduce productivity, which is a significant source of financial losses in the global agriculture industry. In the past, researchers have used hand-crafted traits to categorize photos and spot leaf diseases in maize plants. Researchers can now greatly improve the accuracy of object classification and identification thanks to the advent of deep learning.

[3] Vinay Gautama, The most important and consumable agricultural crop is the paddy crop. Paddy crops' quality and productivity are impacted by leaf disease. Consequently, it is essential to address this problem as soon as possible in order to lessen its effects. As a result, diagnosing and categorizing leaf disease has become increasingly important in recent years thanks to deep learning techniques. To spot disease patterns in crop leaves, deep learning is applied. For instance, it is important to arrange a crop's leaves according to their size, shape, and colour.

[4] Sunu Jatmikaetal, This study examines the use of deep learning in mobile applications to categorize or locate illnesses in rice leaves. By presenting diagnostic results in the form of the disease name along with its taxonomy, disease description, and prescription suggestions for disease treatments, this system will make it simple.

[5] Andrew Jet.al, The agriculture industry is crucial in delivering high-quality food and contributes most to expanding economies and populations. Plant diseases have the potential to significantly reduce food output and wipe out species variety. Early detection of plant diseases using accurate or automatic detection techniques can improve food production quality while minimizing economic losses. Deep learning has significantly increased the identification accuracy of picture categorization and object detection systems in recent years

[6] Gudan Kathiresanet al, The subject of agricultural disease identification has greatly benefited from recent advancements in deep learning techniques. For areas where rice is a staple food, this issue bears a lot of weight due to the well-known detrimental effects of pests and bacterial diseases on rice plants. This research suggests a highly accurate, transfer-learned model that might offer farmers and agricultural organizations a mobile solution to quickly identify illnesses of rice leaves. A generative adversarial network is also used in this study to balance the distribution of illness samples.

[7] SK Mahmudul Hassanet al, Crop diseases must be promptly identified and prevented in order to increase productivity. As deep convolutional neural networks (CNNs) have shown outstanding achievements in the field of machine vision, these models are used in this article to recognize and diagnose plant illnesses from their leaves. Large numbers of parameters and more expensive computations are needed for standard CNN models. The deep CNN model is promising and may significantly influence the effective identification of the diseases, and it may have promise in the detection of diseases in real-time agricultural systems, according to the accuracy findings in the identification of diseases.

[8] Malathiet al, For farmers, recognizing pests in the agricultural sector is a crucial problem that hinders economic growth. Farmers have up until now increased yield production by adhering to traditional procedures. Researchers now use deep learning to practically classify different types of photos. In this study, eleven different types of pests found in the paddy crop are identified using deep convolutional neural networks. The data repository includes photos of pests that harm rice crops, and as deep learning scales effectively with bigger data sets, data augmentation is used.

[9] Siddharth Swarup Rattrayet al, Because it is a necessity for every living thing on this planet, agriculture serves as the backbone of human society. In terms of humanity, paddy agriculture is quite important, particularly in the Asian subcontinent. We must defend the significance and productivity of agriculture since we humans are among the most intellectual creatures on the planet. The productivity of agriculture has increased to some extent with the arrival of the IT sector. It has made significant contributions to agricultural healthcare.

[10] Vimal Shrivastava al, In the crop protection system, early and precise diagnosis of plant diseases is crucial. In conventional procedures, identification is carried out either through visual observation or through laboratory testing. While the laboratory test is time-consuming and might not be able to produce the results in a timely manner, visual observation requires experience and can vary depending on the individual, which could result in an error. In order to address these problems, literature has suggested an image-based machine learning strategy to identify and categories plant diseases. In this work, we have concentrated primarily on the rice plant. From a rice field, pictures of the sick stems and leaves have been taken.

## **2.2 EXISTING SYSTEM**

The major downside in leaf identification is the need to describe leaf pattern. To describe the leaf form, numerous form choices have so far been retrieved. Unfortunately, once the leaf has been photographed and its characteristics have been determined, there is no accurate application to categories it. Plant leaves are classified according to their many morphological characteristics.

The leaf disease prediction at an inopportune moment is explored using the current system paper. To divide the contaminated area into its appropriate classes and segment it, a segmentation model based on colour was developed. For the example photos, experimental analysis has been performed in accordance with or in light of the time complexity and the affected region area.

### ***DISADVANTAGES***

- The training and testing of the data require a lot of time.
- The accuracy is smaller than that of our suggested model.
- The model takes up more space in the system.

## **2.3 PROPOSED SYSTEM**

The major goal of the proposed system is to identify plant leaf diseases using feature extraction techniques that take form, colour, and texture into account. CNN is a method for categorizing plant leaves as healthy or diseased, and if it's a diseased plant leaf, CNN will identify the specific disease. It is made to suggest treatments for certain diseases that will aid in the growth of healthy plants and increase output.

### ***ADVANTAGES***

- The presented approach improves detection accuracy.
- The proposed method is completely automatic, whereas existing methods require user input to select the optimum segmentation of the input image.
- It also gives environmentally friendly illness recovery techniques.

### ***FEASIBILITY STUDY***

In this phase, the project's economic potential is assessed. Because our project is tied to agriculture, some facts must be examined, such as hazards to the natural environment for the pest remedies we provide. We make sure the farmer bears the cost. Understanding the main system requirements is crucial for the feasibility analysis. The feasibility analysis took into account three important factors are:

## **CHAPTER 3**

### **REQUIRMENTS ANALYSIS**

The aim of plant disease detection is to accurately and efficiently identify diseases affecting crops and plants in agricultural settings. This process encompasses various goals and objectives aimed at improving crop health, enhancing yield productivity, reducing economic losses, and promoting sustainable agriculture practices. The scope of plant disease detection is wide-ranging, covering multiple aspects and applications within the agricultural domain:

#### **3.1 SCOPE OF THE PRESENT INVESTIGATION**

##### ***EARLY DETECTION AND DIAGNOSIS:***

One of the primary aims is to detect diseases at an early stage to facilitate prompt intervention and management strategies. Early detection enables farmers and agronomists to take timely actions such as targeted treatments, disease-resistant crop selection, and preventive measures, minimizing the spread and impact of diseases on crop yields.

##### ***DISEASE IDENTIFICATION AND CLASSIFICATION:***

Plant disease detection aims to accurately identify and classify different types of diseases affecting crops, including fungal infections, bacterial diseases, viral infections, nutrient deficiencies, and abiotic stressors. By distinguishing between various disease classes, detection systems can provide tailored recommendations for disease control and management.

##### ***MONITORING AND SURVEILLANCE:***

Plant disease detection systems play a crucial role in continuous monitoring and surveillance of crop health across agricultural fields. These systems utilize sensors, imaging technologies, and remote sensing techniques to collect real-time data on plant conditions, disease outbreaks, pest infestations, and environmental factors. Monitoring capabilities enable early warning systems, disease forecasting, and decision support tools for farmers and stakeholders.

***PRECISION AGRICULTURE:***

The scope of plant disease detection extends to precision agriculture practices, integrating data-driven technologies, analytics, and automation to optimize resource use, minimize inputs, and maximize crop productivity. By incorporating disease detection into precision agriculture frameworks, farmers can adopt targeted strategies for irrigation, fertilization, and pest management, reducing environmental impacts and improving sustainability.

***INTEGRATED PEST MANAGEMENT (IPM):***

Plant disease detection aligns with integrated pest management principles, emphasizing holistic approaches to pest and disease control. IPM strategies integrate biological, cultural, mechanical, and chemical control methods based on scientific knowledge, monitoring data, and risk assessment. Disease detection systems contribute valuable data and insights to support IPM decision-making and sustainable pest management practices.

***TECHNOLOGY INTEGRATION AND INNOVATION:***

The aim of plant disease detection is also to foster technological innovation and integration within the agricultural sector. Advancements in artificial intelligence, machine learning, IoT (Internet of Things), drones, and remote sensing technologies are driving the development of advanced disease detection solutions. These technologies enable scalable, cost-effective, and accessible tools for farmers worldwide, empowering them to make informed decisions and optimize crop health outcomes.

### **3.2 FEASIBILITY STUDY**

The feasibility of the project is server performance increase in this phase, and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Operational feasibility

#### ***ECONOMIC FEASIBILITY***

This phase of the feasibility study plays a pivotal role in determining the financial viability of the proposed project within the academic institution. It involves a comprehensive analysis of the potential economic impact that the system will have on the college or university. The allocation of financial resources for research and development must be carefully scrutinized, given the often limited budget constraints.

Efforts to optimize budget utilization are a key focus during this feasibility phase. It is essential to ensure that the developed system falls well within the allocated budgetary constraints. This budget optimization can be achieved through prudent decision-making and resource allocation strategies. Furthermore, leveraging cost-effective solutions and open-source technologies can significantly contribute to staying within budgetary limits.

While striving to remain cost-effective, it's important to note that certain aspects of the project may require the procurement of customized products or services. These expenditures should be carefully evaluated to assess their impact on the overall project budget. Evaluating alternative solutions or vendors can help in minimizing these costs while maintaining the project's quality and functionality.

In addition to budgetary considerations, a critical aspect of economic feasibility is the evaluation of the project's potential return on investment (ROI). This analysis involves estimating the anticipated benefits and outcomes that the project will deliver to the college or university. Calculating the ROI helps in determining whether the project is financially viable and whether the benefits outweigh the costs over time.

### ***TECHNICAL FEASIBILITY***

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

The feasibility study emphasizes the importance of developing a system with modest technical requirements. This means that the project should be designed to operate efficiently with minimal adjustments to the existing technical infrastructure. Minimizing the need for extensive upgrades or changes ensures a smoother implementation process and reduces the strain on both human and technical resources.

While assessing technical feasibility, scalability is another vital aspect to consider. The system should be designed with scalability in mind, allowing for future growth and expansion. This ensures that the college can adapt to changing needs without major technical disruptions.

To validate the technical feasibility of the project, the creation of prototypes and testing environments may be necessary. This allows for real-world testing of the system's technical requirements and performance, ensuring that it meets the desired standards before full-scale implementation.

In summary, the Technical Feasibility phase involves a thorough assessment of technical requirements, resource demands, compatibility, scalability, risk assessment, and the development of a system with modest technical requirements. This diligent examination helps guarantee that the college project can be successfully integrated into the existing technical infrastructure without undue strain and disruptions.

### ***OPERATIONAL FEASIBILITY:***

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system; instead, they must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make them familiar with it. Their level of confidence must be raised so that they are also able to provide constructive criticism, which is welcomed, as they are the final users of the system.

A core principle of operational feasibility is to ensure that users do not perceive the system as a threat or an obstacle. Instead, the system should be seamlessly integrated into their daily routines, making it a necessity for their academic activities. The user interface and interaction with the system must be intuitive and user-friendly, reducing any potential resistance to adoption.

Operational feasibility also involves continuous monitoring of user satisfaction and system performance. Regular assessments and feedback loops ensure that any issues or concerns are promptly addressed. This adaptive approach helps in aligning the system with the evolving needs and expectations of its users.

### **3.3 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT**

#### ***HARDWARE SPECIFICATION***

- RAM-Minimum of 4GB.
- Storage of at least 15GB will be better.
- processor of min i5 10-gen will be good.
- Internet connectivity.
- 

#### ***SOFTWARE SPECIFICATION***

- Operating System - Windows 10 or 11 will be good.
- TensorFlow - 2.4.1.
- NumPy - 1.24.2

## **CHAPTER 4**

### **EXPERIMENTAL OR MATERIALS AND METHODS AND ALGORITHMS USED**

The major goal of the proposed system is to identify plant leaf diseases using feature extraction techniques that take form, colour, and texture into account. CNN is a method for categorizing plant leaves as healthy or diseased, and if it's a diseased plant leaf, CNN will identify the specific disease. It is made to suggest treatments for certain diseases that will aid in the growth of healthy plants and increase output. The presented approach improves detection accuracy. The proposed method is completely automatic, whereas existing methods require user input to select the optimum segmentation of the input image. It also gives environmentally friendly illness recovery techniques.

#### **4.1 SELECTED METHODOLOGY OR PROCESS MODEL**

##### ***Collection of data***

Collecting data for plant disease detection involves gathering diverse datasets comprising images, sensor data, and metadata related to plant health and environmental conditions. The process begins with capturing high-resolution images of plants using digital cameras, drones, or specialized imaging devices. These images should include samples of both healthy plants and plants exhibiting various disease symptoms.

In addition to images, sensor data such as spectral readings, temperature, humidity, and soil moisture levels may be collected using sensors and IoT devices deployed in agricultural fields. This sensor data provides valuable insights into the biochemical and physiological changes in plants due to diseases, helping to enhance disease detection accuracy.

Metadata such as plant species, disease types, location, weather conditions, and crop management practices are also collected to provide context and facilitate data analysis. Crowd sourcing platforms and collaborations with farmers, researchers, and agricultural experts can help gather large and diverse datasets encompassing different plant species, diseases, and environmental Data collection efforts should prioritize data quality, diversity, and representational to ensure that machine learning models trained on these datasets can generalize well and accurately detect diseases in real-world agricultural settings. Regular updates and maintenance of the datasets are also essential to capture evolving disease patterns, new pathogens, and changes in agricultural practices over time.

### ***Training the data***

Training data in plant disease detection is a crucial phase that involves preparing and feeding labeled datasets into machine learning models to enable them to learn and recognize patterns indicative of different diseases in plants. The process typically begins with the collection of high-quality images of both healthy and diseased plants, along with relevant metadata such as plant species, disease types, and environmental conditions. These images are then annotated or labeled by domain experts to indicate the presence or absence of diseases in each image.

Once the labeled data is ready, it undergoes preprocessing steps to standardize image sizes, normalize pixel values, and augment the dataset to increase its diversity and robustness. Preprocessing techniques may include image re sizing, color normalization, background removal, noise reduction, and data augmentation methods such as rotation, flipping, and zooming. These preprocessing steps ensure that the training data is optimized for machine learning model training and improves the model's ability to generalize to new and unseen data.

The training process involves feeding the prepossessed and labeled data into machine learning algorithms, such as Conventional Neural Networks (CNN's), which are well-suited for image classification tasks. During training, the model learns to extract relevant features and patterns from the input images, adjusting its internal parameters (weights and biases) interactively to minimize prediction errors. This optimization process is typically achieved through optimization algorithms like stochastic gradient descent (SGD) or Adam optimizer, which update the model's parameters based on the computed gradients of the loss function.

### ***Testing the data***

Testing the data in plant disease detection involves evaluating the performance and accuracy of machine learning models trained on labeled datasets. The testing phase is critical for assessing the model's ability to generalize to new and unseen data, ensuring reliable disease detection in real-world scenarios.

The testing process begins by partitioning the labeled dataset into training, validation, and testing subsets. The testing subset, also known as the testing dataset, is kept separate from the training and validation data to simulate real-world conditions where the model encounters new samples for classification.

During testing, the trained model is presented with images or sensor data from the testing data set, and its predictions are compared against the ground truth labels to measure performance metrics such as accuracy, precision, recall, F1 score, and area under the receiver operating characteristic curve (AU C-ORC). These metrics quantify the model's ability to correctly classify plants as healthy or diseased and determine the model's overall effectiveness in disease detection.

Cross-validation techniques, such as k-fold cross-validation, may also be employed to validate the model's performance across different data splits and ensure robustness and generalization. Additionally, testing the model with diverse and representative testing datasets covering various plant species, diseases, and environmental conditions helps validate its reliability and suitability for practical deployment in agricultural settings.

## ***DETECTING THE DISEASES***

Detecting diseases in plant disease detection involves applying machine learning models trained on labeled datasets to identify and classify diseased plants accurately. The detection process is typically carried out using image-based or sensor-based techniques, depending on the data available and the nature of the disease. For image-based detection, the trained machine learning model, often a Convolutional Neural Network (CNN), analyzes input images of plants to identify visual patterns and features indicative of diseases. The model extracts relevant information such as color variations, lesion shapes, texture abnormalities, and leaf deformities to differentiate between healthy and diseased plants. By comparing these features against learned patterns from the training data, the model makes predictions about the presence and type of disease in the plant.

Sensor-based detection leverages data from sensors measuring plant physiological parameters, biochemical changes, or environmental conditions. These sensors can detect anomalies in plant health indicators such as spectral reflectance, chlorophyll content, temperature, humidity, and gas emissions. Machine learning algorithms process sensor data to identify abnormal patterns associated with disease presence and trigger alerts or notifications for further investigation. In both image-based and sensor-based detection methods, the goal is to automate the disease detection process, provide early warnings about potential outbreaks, and assist farmers and agronomists in making informed decisions about disease management strategies, treatment options, and crop protection measures.

## 4.2 IMPLEMENTATION

The implementation of plant disease detection involves several key steps to effectively deploy machine learning models and integrate them into practical applications for agricultural use. Here are the steps involved in implementing plant disease detection:

**DATA COLLECTION:** Gather diverse and representative datasets comprising labeled images of healthy and diseased plants, along with relevant metadata such as plant species, disease types, and environmental conditions.

**DATA PREPROCESSING:** Clean, preprocess, and augment the collected data to enhance its quality, diversity, and suitability for model training. Preprocessing steps may include image re-sizing, normalization, background removal, noise reduction, and data augmentation techniques.

**FEATURE EXTRACTION:** Extract relevant features and patterns from preprocessed data using machine learning algorithms, such as Conventional Neural Networks (CNN's), to capture distinctive characteristics of healthy and diseased plants.

**MODEL TRAINING:** Train the machine learning model on the labeled dataset using optimization algorithms and validation techniques to optimize model performance metrics such as accuracy, precision, recall, and F1 score.

**MODEL EVALUATION:** Evaluate the trained model using independent testing datasets to assess its generalization, robustness, and reliability in accurately detecting plant diseases.

**DEPLOYMENT:** Deploy the trained model into production environments or agricultural systems, such as web-based interfaces, mobile applications, or IoT devices, to enable real-time disease detection and decision support.

**INTEGRATION:** Integrate the disease detection system with data management platforms, IT sensors, agricultural management systems, and other relevant technologies to facilitate seamless data flow, analysis, and communication.

**VALIDATION AND MONITORING:** Continuously validate and monitor the performance of the deployed system, gather user feedback, and interactively improve the model's accuracy, efficiency, and usability based on real-world usage and insights.

**FEEDBACK LOOP:** Establish a feedback loop to incorporate user feedback, domain knowledge, and continuous learning into the system, enabling adaptive and responsive improvements to disease detection capabilities over time.

**SCALABILITY AND MAINTENANCE:** Ensure the system's scalability, scalability, and maintenance by regularly updating datasets, retraining models with new data, addressing model drift, and enhancing system functionality based on emerging technologies and user needs.

#### **4.3 DESCRIPTION OF PROGRAMMING LANGUAGE AND SOFTWARE PYTHON**

Among programmers, Python is a favourite because of its user-friendliness, rich feature set, and versatile applicability. Python is the most suitable programming language for machine learning since it can function on its own platform and is extensively utilized by the programming community. Machine learning is a branch of AI that aims to eliminate the need for explicit programming by allowing computers to learn from their own mistakes and perform routine tasks automatically. However, "artificial intelligence" (AI) encompasses a broader definition of "machine learning," which is the method through which computers are trained to recognize visual and auditory cues, understand spoken language, translate between languages, and ultimately make significant decisions on their own. The desire for intelligent solutions to real-world problems has necessitated the

need to develop AI further in order to automate tasks that are arduous to programme without AI. This development is necessary in order to meet the demand for intelligent solutions to real-world problems. Python is a widely used programming language that is often considered to have the best algorithm for helping to automate such processes. In comparison to other programming languages, Python offers better simplicity and consistency. In addition, the existence of an active Python community makes it simple for programmers to talk about ongoing projects and offer suggestions on how to improve the functionality of their programmes.

### **ANACONDA:**

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Coda environments. Moreover, you can deploy any required project with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them:

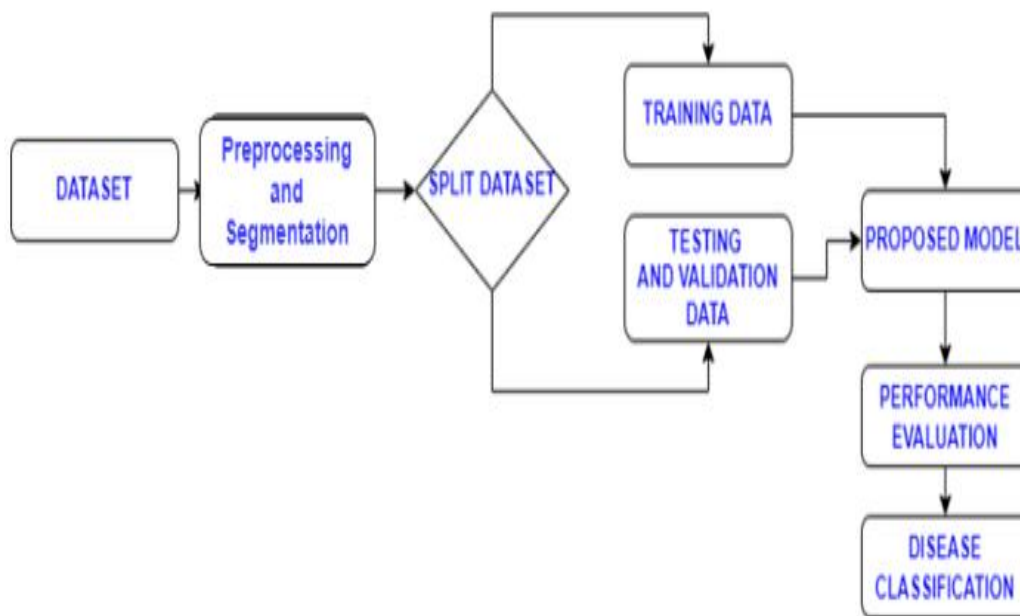
Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues.

It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest.

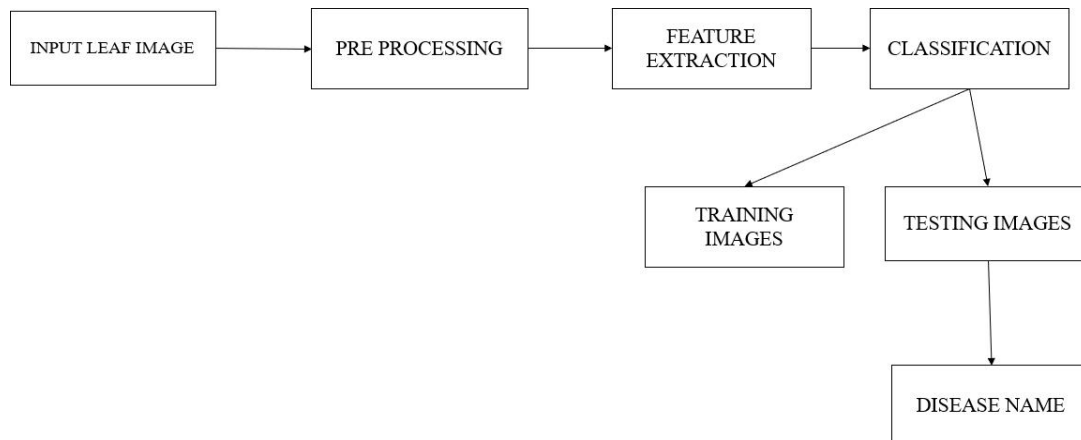
Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Sci kit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

#### 4.4 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM



**Fig 4.4.1: System Architecture**

## GENERAL ARCHITECTURE



**FIG 4.4.2 :GENERAL ARCHITECTURE**

## 4.5 ALGORITHMS

### *Image Segmentation*

The process in which division of an image is processed into multiple parts called Segmentation of an Image, by which relevant information and objects can be identified. The different ways present to perform the process of image segmentation are as follows:

- (i) Threshold method (e.g., the method suggested by Otsu)
- (ii) Segmentation depending on the image colors (e.g., k-means of clustering)
- (iii) Transformation method (e.g., segmentation using watershed technique)
- (iv) Based on the image texture, such as texture filters

Image segmentation plays a crucial role in plant leaf disease detection by enabling the precise delineation and analysis of diseased regions within plant images. This process involves partitioning an image into distinct segments or regions based on pixel intensity, color, texture, or other visual features, allowing for targeted analysis and classification of plant diseases. In the context of plant

leaf disease detection, image segmentation techniques are used to isolate and identify areas of the leaf affected by diseases such as leaf spots, powdery mildew, rust, and blights.

One of the common approaches to image segmentation in plant leaf disease detection is thresholding. Thresholding techniques segment an image by setting intensity or color thresholds to separate foreground (diseased regions) from background (healthy regions) pixels. This method is effective for binary segmentation tasks, where the goal is to differentiate between healthy and diseased areas based on pixel intensity values. However, thresholding may struggle with images containing complex textures, lighting variations, or overlapping regions of different intensities.

Another widely used technique is region-based segmentation, such as the watershed algorithm. This method partitions an image into regions based on gradients or intensity gradients, mimicking the flow of water over the image surface. Watershed segmentation is particularly useful for separating overlapping objects or lesions in plant leaf images, allowing for more accurate disease region delineation. However, it may require preprocessing steps to enhance image contrast and remove noise for optimal segmentation results.

Machine learning-based segmentation approaches, including deep learning models like convolutional neural networks (CNNs), have gained popularity in recent years for their ability to learn and extract complex features from images. Semantic segmentation models, such as U-Net, FCN (Fully Convolutional Networks), and Mask R-CNN, can accurately segment plant leaf images into fine-grained regions corresponding to different disease classes. These models leverage annotated training data to learn the spatial relationships and characteristics of diseased regions, enabling precise disease detection and localization.

Additionally, graph-based segmentation methods, such as graph cuts and minimum spanning trees, offer robustness against noise and can handle irregular-shaped regions in plant leaf images. These methods utilize graph

theory principles to partition an image into homogeneous segments based on pixel similarities and connectivity, facilitating accurate disease region segmentation.

Overall, image segmentation is a critical step in plant leaf disease detection, allowing for the isolation, analysis, and classification of diseased regions within plant images. By employing a combination of thresholding techniques, region-based algorithms, machine learning models, and graph-based methods, researchers and practitioners can achieve accurate and reliable disease detection outcomes, contributing to improved crop management, disease prevention, and agricultural productivity.

### **VGG-16**

The VGG-16 network model, also known as the Very Deep Convolutional Network for Large-Scale Image Recognition, was built by the Visual Geometry Group from Oxford University.

The depth is pushed to 16–19 weight layers and 138 M trainable parameters. The depth of the model is also expanded by reducing the convolution filter size to  $3 \times 3$ . This model requires more training time and occupies more disk space.

**Uniform Architecture:** VGG-16 uses a homogeneous architecture, applying only  $3 \times 3$  convolutional layers with a stride of 1 pixel; the padding is also 1 pixel for convolutional layers to preserve the spatial resolution of the image. Max-pooling is performed over a  $2 \times 2$  pixel window, with stride 2.

**Depth:** The network contains 13 convolutional layers and 3 fully connected layers, leading to a total of 16 weight-bearing layers, hence the name VGG-16. This depth is significant because, at the time of its introduction, it was one of the deepest architectures successfully trained on image recognition tasks.

**ReLU Activation:** The Rectified Linear Unit (ReLU) activation function is used throughout the network, adding non-linearity to the model without affecting the receptive fields of the convolutional layers.

**Fixed Input Size:** The original model was designed to work with a fixed input image size of 224x224 pixels. Images must be resized to this dimension before being input into the network.

**Number of Parameters:** VGG-16 is quite heavy in terms of parameters. It has approximately 138 million parameters, which makes it computationally intensive to train and run, especially on devices with limited resources.

**Applications:** Despite its simplicity and age, VGG-16 is still used in many image processing and computer vision applications. It's often used as a feature extractor for tasks like image classification, object detection, and more.

**Transfer Learning:** Due to its performance and simplicity, VGG-16 is commonly used as a pre-trained model for transfer learning. This approach allows leveraging the VGG model's weights as a starting point for training on a new task with less data or computational resources than training from scratch.

**Convolutional Layers:** The network uses a stack of convolutional layers with small receptive fields in the first layers that capture the fine details of the image. As you move deeper into the network, the receptive fields increase, allowing the network to capture more abstract features.

**Max Pooling Layers:** These are used to reduce the spatial dimensions of the input volume for the next convolutional layer.

**Fully Connected Layers:** Three fully connected layers follow the last convolutional layer. The first two have 4096 channels each, and the third performs 1000-way ILSVRC classification and thus contains 1000 channels

## **K-MEANS CLUSTERING ALGORITHM**

The K-means algorithm is a widely used method for clustering that partitions a dataset into K distinct, non-overlapping subsets (or clusters). The goal is to assign each data point to the cluster whose center (or centroid) is nearest. The result is a partitioning of the data space into Voronoi cells. K-means is popular due to its simplicity and efficiency, particularly for large datasets.

### **ALGORITHM OVERVIEW:**

**Initialization:** Start by selecting K initial centroids. These can be chosen randomly from the dataset or using more complex initialization methods to potentially improve convergence speed and clustering quality.

**Assignment Step:** Assign each data point to the nearest cluster by calculating the distance between each data point and each centroid. The most common distance metric used is Euclidean distance, although other distances can be used depending on the nature of the data.

**Update Step:** After all points have been assigned to clusters, recalculate the centroids of each cluster as the mean of all points assigned to that cluster.

**Repeat:** Repeat the assignment and update steps until the centroids no longer change significantly, indicating that the algorithm has converged, or a predefined number of iterations is reached.

**KEY POINTS:**

**Choosing K:** The choice of K (the number of clusters) is crucial and not defined by the algorithm. Various techniques, such as the Elbow Method or the Silhouette Method, can help in choosing a reasonable K value by evaluating the clustering performance for different K values.

**Convergence:** While K-means is guaranteed to converge, it may converge to a local minimum. This means that the outcome can depend on the initial choice of centroids. To mitigate this, it's common to run the algorithm multiple times with different initializations and choose the best outcome according to some criterion.

**Scalability and Efficiency:** K-means is generally considered efficient for large datasets, but its computational cost increases linearly with the number of data points. Variants such as Mini-Batch K-means can be used to further improve efficiency on very large datasets.

**Applications:** K-means is used across a wide range of applications, including market segmentation, computer vision, geostatistics, and document clustering.

**LIMITATIONS:**

It assumes that the clusters are convex and isotropic, which might not be the case with real-world data.

It is sensitive to outliers, as outliers can significantly shift the position of centroids.

The need to specify K a priori can be a drawback when the number of clusters is not known in advance.

It may not perform well on clusters of varying sizes and density.

Despite these limitations, K-means remains a foundational tool for clustering and data analysis, serving as a benchmark for evaluating more complex clustering algorithms.

## **TRANSFER LEARNING APPROACH**

Transfer learning is a machine learning method where a model developed for a particular task is reused as the starting point for a model on a second task. This approach is widely used in deep learning, where pre-trained models are leveraged to achieve significant improvements in performance, especially when labeled data for the target task is scarce. Transfer learning capitalizes on the idea that instead of starting the learning process from scratch, one can reuse parts of pre-trained models which have already learned some aspects of the task or similar tasks, thereby reducing the computational cost and time required to develop complex models.

### **HOW TRANSFER LEARNING WORKS:**

**Pre-trained Models:** It begins with a model that has been pre-trained on a large dataset. In the context of deep learning, such models often come from tasks with substantial general datasets like ImageNet, which contains millions of images classified into thousands of categories. These models have learned a rich set of features that can be applied to a wide range of tasks beyond those they were originally trained for.

**Feature Extraction:** The simplest form of transfer learning in deep learning contexts involves using the pre-trained model as a feature extractor. Here, you remove the output layer of the model that was specific to the original task and replace it with a new layer (or layers) tailored to the new task. The weights in the pre-trained layers remain fixed, and only the weights in the new layers are trained. This method works well when the new task is similar to the original task that the model was trained on.

**Fine-Tuning:** In scenarios where the new task is significantly different from the original task, or when you have a sufficient amount of labeled data for the new task, fine-tuning can be more effective. This involves not only replacing the output layer but also retraining the entire model (or a substantial part of it) on the new task, starting with the weights of the pre-trained model. Fine-tuning allows the model to adjust its learned features to better suit the new task.

### ***BENEFITS OF TRANSFER LEARNING:***

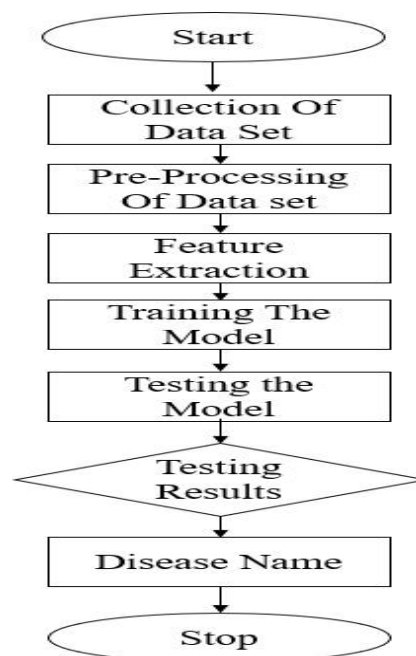
Efficiency training a model from scratch requires substantial computational resources and time, especially for complex tasks. Transfer learning allows for significant reductions in both, making it feasible to train deep learning models on less powerful machines or with limited time.

**Data Requirements:** Deep learning models typically require vast amounts of labeled data. Transfer learning can dramatically reduce the amount of labeled data needed for training because the model has already learned a set of useful features. Improved Performance: Models initialized with pre-trained weights often perform better, especially in tasks where the available dataset is too small to train a deep network from scratch without overfitting.

Image Recognition and Computer Vision: Transfer learning is particularly prevalent in image recognition tasks, where models pre-trained on ImageNet are adapted to tasks like object detection, segmentation, and more.

Natural Language Processing (NLP): Models pre-trained on large text corpora (like BERT, GPT) can be fine-tuned for specific NLP tasks such as sentiment analysis, question-answering, and text classification.

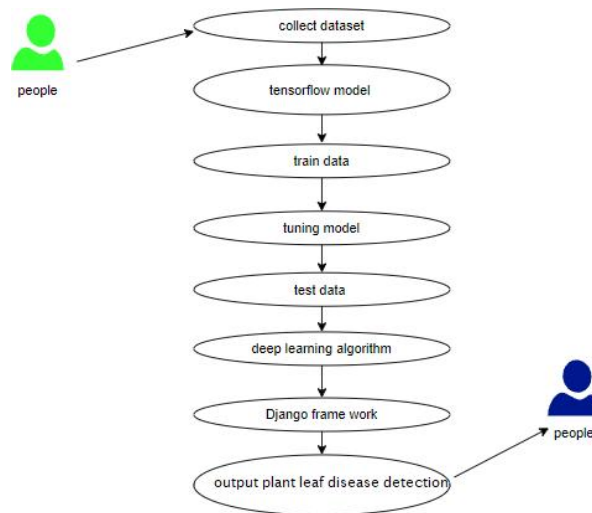
## **4.6 DESIGN PHASE**



**FIG 4.6.1 Data Flow Diagram**

The data flow diagram is a method of illustrating how data flow through a system, typically an information system. In our project, the data flow diagram illustrates the flow of information. We can gather all the information from farmers, web sources, and agriculture groups. Data pre-processing is done after collection to provide a perfect data set. We can test the input data leaf after training the model, and the testing outcome will categories the leaf disease.

### **USE CASE DIAGRAM**



**FIG 4.6.2 USE CASE DIAGRAM**

Use case diagrams are used to show how users interact with systems. It displays several use cases and the system's distinct user categories. Other kinds of diagrams are frequently included with it as well. In the diagram above, the user gathers the dataset, uses the train dataset to train the model, and tests the model as well.

A use case diagram is a visual representation of the interactions between the actors (users or other systems) and the system itself, outlining the functionalities (use cases) provided by the system and how these functionalities interact with the users. In the context of plant disease detection, a use case diagram can

help in identifying and organizing the requirements of the system from the perspective of the end-users, such as farmers, agricultural experts, or researchers.

### ***ACTORS IN A PLANT DISEASE DETECTION SYSTEM:***

***Farmer:*** Primary users who seek to diagnose and understand the diseases affecting their crops.

***Agricultural Expert:*** Professionals who provide insights or validate the disease detection and suggest treatments.

***Researcher:*** Individuals who might use the system to gather data on plant diseases for academic or scientific purposes.

***System Administrator:*** Responsible for managing and maintaining the system, ensuring it runs smoothly and is updated with the latest plant disease databases and detection algorithms.

### ***USE CASES FOR A PLANT DISEASE DETECTION SYSTEM:***

***Upload Image:*** Allows users (farmers, experts, researchers) to upload images of the plant leaves or crops showing symptoms of disease.

***Detect Disease:*** The system processes the uploaded images to identify potential diseases. This involves using machine learning models or algorithms to analyze the images and compare them with known patterns of plant diseases.

***View Diagnosis Results:*** Users can view the results of the disease detection process, which includes the type of disease detected and possibly its severity.

***Receive Treatment Recommendations:*** Based on the diagnosed plant disease, the system provides recommendations for treatment or management practices to address the disease.

***Access Disease Information:*** Users can access detailed information about different plant diseases, their symptoms, causes, and treatments. This is useful for educational purposes and for making informed decisions about crop management.

***Feedback Submission:*** Allows users to provide feedback on the accuracy of the diagnosis and the effectiveness of the recommended treatment, which can be used to improve the system.

***System Update (by System Administrator):*** The system administrator

updates the disease database, improves the detection algorithms based on feedback, and performs routine maintenance.

### **EXAMPLE USE CASE DIAGRAM COMPONENTS:**

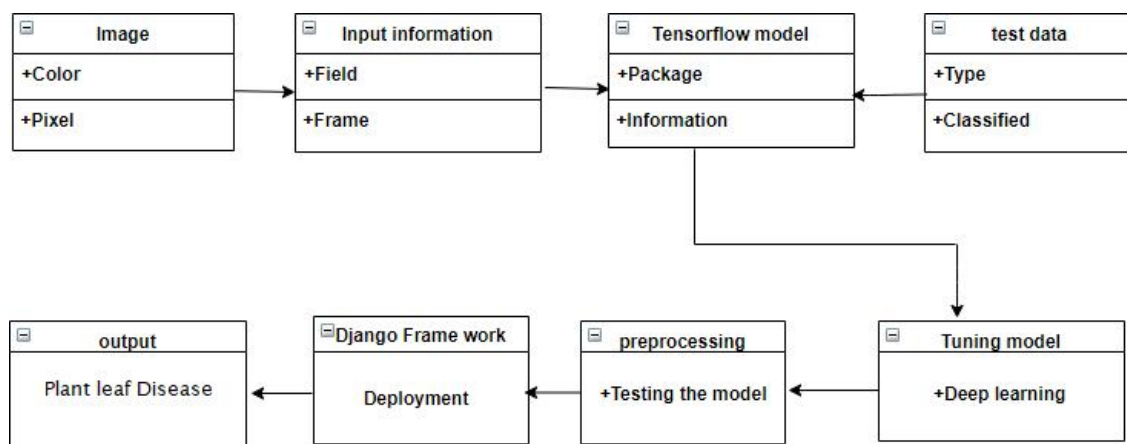
**Actors:** Represented by stick figures or names of the user categories (e.g., Farmer, Agricultural Expert, Researcher, System Administrator).

**Use Cases:** Represented by ovals, showing the actions or functionalities provided by the system (e.g., Upload Image, Detect Disease).

**System:** Typically represented by a box that contains the use cases, labeling the system boundary (e.g., Plant Disease Detection System).

**Relationships:** Lines connecting actors to their use cases, indicating interaction or communication.

### **Class Diagram**



**FIG 4.6.3 CLASS DIAGRAM**

A class diagram is one of the types of diagrams in Unified Modeling Language (UML), which is used for modeling the static structure of an object-oriented system. It describes the types of objects in the system and the static relationships between them, such as associations, inheritances, and aggregations. Class diagrams are essential for visualizing, specifying, and documenting models of systems and are widely used in software development for planning and communicating about system design.

The user's interaction with the model and user interface is described in a class diagram. There are two objects in this class diagram. The user can initially connect to the model. In the model, there are an infinite amount of leaves that may be taught, and those trained leaves can be divided into two categories: healthy and sick. And following the diagnosis of the illness, the illness is treated. Additionally, the interface makes it simple to submit photographs and provide the user with the results.

In the context of a plant disease detection system, a class diagram can be instrumental in laying out the object-oriented structure of the system. This diagram will define how different components of the system interact with each other, their relationships, and their hierarchy. The system might involve classes for managing user interactions, processing images, analyzing data to detect diseases, and providing feedback or recommendations. Below is a conceptual overview of what such a class diagram might include, focusing on key classes and their possible relationships:

#### ***KEY CLASSES IN A PLANT DISEASE DETECTION SYSTEM:***

- ***User:*** Represents the system's users, which could include farmers, agricultural experts, or researchers.
- ***User to Image:*** One-to-many (1..\*), indicating that a single user can upload multiple images but each image is uploaded by only one user.
- ***Image to Disease Detector:*** One-to-one (1..1), as each image is analyzed individually by the disease detector.
- ***Disease Detector to Disease Database:*** Many-to-one (\*..1), since the detector will consult the database to match detected patterns with known diseases.
- ***Disease Database to Recommendation:*** One-to-many (1..\*), because each disease in the database can have multiple associated recommendations.
- ***Diagnosis Result to Image:*** One-to-one (1..1), where each diagnosis result is linked to a specific image.
- ***Diagnosis Result to User:*** Many-to-one (\*..1), as a user can have multiple diagnosis results, but each result is associated with one user.

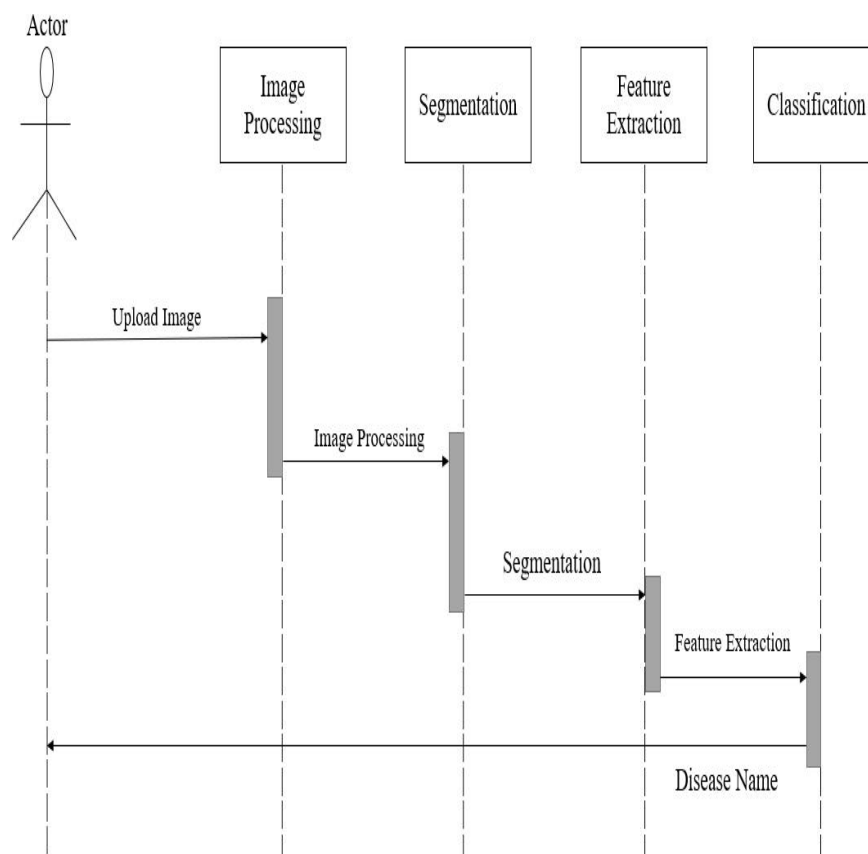
### Simplified Class Diagram Overview:

Classes would be represented as rectangles divided into sections for the class name, attributes, and methods.

Relationships would be illustrated with lines connecting the classes, with symbols at each end to indicate the nature of the relationship (e.g., open diamond for aggregation, filled diamond for composition, arrow for directional).

This conceptual class diagram provides a blueprint for designing and implementing a plant disease detection system. It outlines the system's main entities, their characteristics, and how they interact to facilitate the detection and management of plant diseases. Actual implementation details may vary based on specific requirements, such as the choice of technology stack, the complexity of the disease detection algorithms, and the system's scalability needs.

### **SEQUENCE DIAGRAM**



**FIG 4.6.4 SEQUENCE DIAGRAM**

The process is described in one sequence order in the sequence diagram. The actor can first upload the source image to the image processioning, which is followed by image segmentation before they can extract the leaf's feature data. Following feature extraction, classification can be done using the name of the leaf disease.

A sequence diagram in the context of a Unified Modeling Language (UML) is used to show how objects interact in a given scenario of a use case. This interaction is represented in a time sequence, with the passage of time proceeding down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence.

Below is a description of how a sequence diagram could be structured for a plant disease detection scenario. This scenario involves a user uploading an image of a plant leaf to a system, which then analyzes the image to detect any disease and returns information about the disease and possible treatments.

**Scenario:** User Requests Plant Disease Diagnosis

Participants:

**User:** A farmer or an agricultural expert who uses the system to diagnose plant diseases.

**Web Interface:** The fronted through which the user interacts with the system.

**Controller:** A back-end component that handles web requests and coordinates the disease detection process.

**ImageProcessor:** A service or module responsible for preprocessing images before analysis.

**Disease-detector:** The core component that analyzes the processioned image to detect disease.

**Disease Database:** A database or service that stores information about various plant diseases and recommendations for treatment.

Sequence of Events:

**User uploads an image:** The sequence begins with the User uploading an image of a diseased plant through the Web Interface.

**Web Interface requests image analysis:** The Web Interface sends the image to the Controller to initiate the diagnosis process.

***Disease Detector analyzes the image:*** The Disease Detector uses machine learning models to analyze the image and identify any signs of disease. It then sends the detected disease ID (if any) back to the Controller.

***Controller queries the DiseaseDatabase:*** The Controller uses the disease ID to query the Disease Database for detailed information about the disease and possible treatments.

Disease Database returns disease information: The Disease Database sends detailed information about the disease, including name, symptoms, and treatment recommendations.

Controller sends results to Web Interface: The Controller compiles the diagnosis and treatment information and sends it back to the Web Interface.

User receives diagnosis and recommendations: Finally, the Web Interface displays the disease diagnosis and recommendations to the User.

### ***REPRESENTATION:***

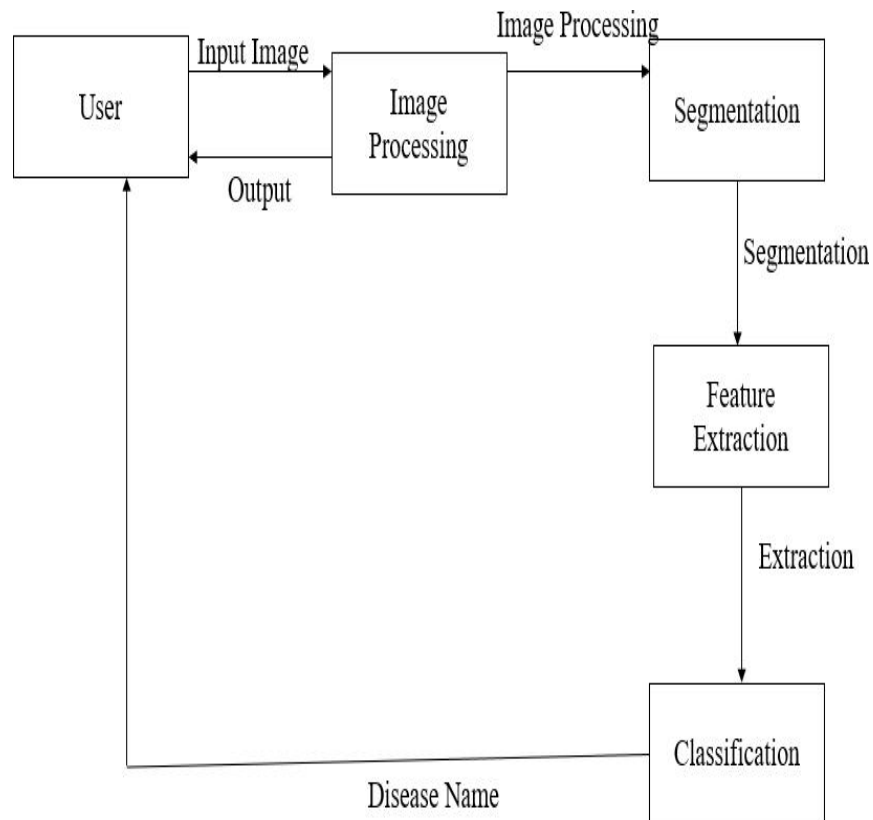
Participants are represented by rectangles at the top, with vertical dashed lines (lifelines) extending downward.

Messages between objects are represented by horizontal arrows, indicating the sequence and direction of communication.

Activation bars (thin rectangles on the lifelines) show when an object is active or performing an operation.

This sequence diagram provides a clear, step-by-step illustration of the interactions between components in a plant disease detection system, from image upload to the delivery of a diagnosis and recommendations. It helps developers and stakeholders understand the system's workflow and the timing of different components' interactions.

## **COLLABORATION DIAGRAM**



**FIG 4.6.5 COLLABORATION DIAGRAM**

A collaboration diagram, also known as a communication diagram in Unified Modeling Language (UML), focuses on showing the interactions and relationships between objects or actors in a system. Unlike sequence diagrams that emphasize the chronological order of messages, collaboration diagrams emphasize the structural organization of objects and how they collaborate to achieve a specific functionality. In the context of a plant disease detection system, a collaboration diagram would illustrate how various system components collaborate to process an image.

The diagram typically includes several key participants or objects, each playing a specific role in the disease detection process. These participants often include the User Interface, Controller, Image Processor, Disease Detector, and Disease Database. The User Interface serves as the entry point for users to upload images and initiate the diagnosis process. The Controller acts as the central

coordinator, managing the flow of requests and interactions between different components.

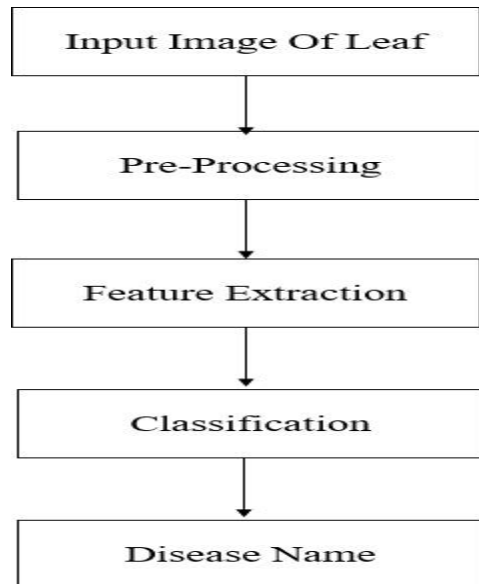
The sequence of interactions depicted in the collaboration diagram begins with the User Interface requesting image analysis from the Controller. The Controller, upon receiving the request, coordinates with the Image Processor to process the uploaded image. Preprocessing tasks may include re-sizing, normalization, or other enhancements to prepare the image for disease analysis.

Once the image is processed, the Controller directs it to the Disease Detector, the core component responsible for analyzing the image using machine learning models to detect any signs of plant diseases. The Disease Detector then interacts with the Disease Database to retrieve detailed information about the detected disease, such as its name, symptoms, causes, and recommended treatments.

After gathering the necessary information from the Disease Database, the Disease Detector communicates the analysis results back to the Controller. The Controller, in turn, compiles the diagnosis information and recommended treatments and communicates this back to the User Interface for display to the user.

In summary, the collaboration diagram provides a comprehensive and structured overview of how the different components of the plant disease detection system collaborate and communicate to analyze images, detect diseases, and provide relevant information and recommendations to users.

## **ACTIVITY DIAGRAM**



**FIG 4.6.6 ACTIVITY DIAGRAM**

Activity diagram can be utilized to represent process activity. It first takes the leaf input image and then sends it for preprocessing. Following preprocessing, we may extract the image's characteristics, and the CNN algorithm can then identify the input image's leaf as either unhealthy or healthy. If an image is infected it can treat that particular medical condition.

An activity diagram in the context of a plant disease detection system provides a visual representation of the workflow or activities involved in detecting and diagnosing plant diseases. It illustrates the sequential and parallel activities, decision points, and flow of control within the system. This diagram is particularly useful for understanding the step-by-step process followed by the system to analyze an uploaded image, detect diseases, and provide recommendations.

The activity diagram for plant disease detection typically starts with an initial activity, such as "Upload Image," representing the user uploading an image of a plant leaf through the User Interface. From there, the diagram branches into various activities that occur concurrently or sequentially. For example, after uploading the image, the system may start preprocessing the image using the Image Processor, which includes tasks like resizing, normalization, and enhancement.

Once the preprocessing is complete, the diagram shows the main activity of "Analyze Image" by the Disease Detector component. This activity involves using machine learning algorithms to analyze the preprocessed image and detect any signs of plant diseases. During this process, the system may also query the Disease Database to gather additional information about detected diseases, such as their symptoms and treatment options.

The activity diagram also includes decision points or branches, represented by diamonds, where the system makes decisions based on certain conditions. For instance, if the Disease Detector detects a disease with a high confidence score, the system may branch to activities related to providing detailed information about the disease and recommending treatments. On the other hand, if no disease is detected or the confidence score is low, the system may branch to activities related to notifying the user about the lack of a conclusive diagnosis or requesting additional images for analysis.

The activity diagram continues to depict the flow of activities, decisions, and control until it reaches the final activities, such as "Display Diagnosis" and "Provide Recommendations," where the system communicates the diagnosis results and treatment recommendations back to the user through the User Interface.

Overall, the activity diagram provides a comprehensive overview of the entire process involved in plant disease detection, making it a valuable tool for system designers, developers, and stakeholders to understand and analyze the workflow and interactions within the system.

## **4.7 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL SYSTEM**

### **4.7.1 MODULE DESCRIPTION**

#### ***Training The Model***

For training our model, we use CNN algorithm models. We must first select the dataset, get it ready for training, then generate training data. Assign the characteristics and labels. The model must now be trained using the tomato dataset. Segmentation, feature extraction, and classification make up the training phase. The necessary modules and packages must be imported. Then, using the sequential method to initialize CNN, we must add elements to the set using the add function. Then, we employ the conv2D function to build output tensors and the Maxpooling function to get the maximum value for patches. Using conv2D and the carpooling method, we need to add another convolutional layer.

Now that CNN has been loaded with an optimizer, loss, and metrics, we must begin to build it. Now use Imaged at a generator function to assign values to the train-data and test-data. The route for training and validating datasets must now be specified using the flow-from-directory function. We utilize the fit-generator function to train the model batch by batch. Finally, we save the model and weights

#### ***Testing the model***

The model's testing phase is crucial since it reveals whether or not our predictions are accurate. We have imported all the modules and packages, same to how we did during the training phase. We must associate the directory of model.h5 with the variable file path.

## **CHAPTER 5**

### **IMPLEMENTATION DETAILS**

#### **5.1 Training the model**

Plant disease detection is a critical aspect of agriculture and plant science, impacting crop yields, food security, and economic stability. Training datasets play a pivotal role in developing effective and accurate disease detection models. These datasets encompass various types of data, each contributing unique insights and challenges to the detection process.

Image datasets are fundamental in plant disease detection. They consist of images captured using different imaging techniques, such as visible light, near-infrared, and hyper spectral imaging. These images showcase plants exhibiting symptoms of diseases, including leaf spots, wilting, discoloration, and deformities. Additionally, healthy plant images are included for comparison and classification purposes. One of the most widely used image datasets is the Plant Village datasets, which features images of multiple plant species affected by various diseases. This datasets enables researchers and developers to train deep learning models to recognize and classify plant diseases accurately.

Sensor data is another valuable type of datasets in plant disease detection. Sensors such as hyper spectral cameras and electronic nose devices capture detailed information about plants' biochemical and physiological changes due to diseases. Hyper spectral imaging, for instance, provides spectral data across a wide range of wavelengths, revealing subtle variations in plant health and disease progression. Electronic nose sensors detect volatile organic compounds emitted by plants under stress, offering early indicators of disease presence. Integrating sensor data with image datasets enhances the overall accuracy and reliability of disease detection models.

Genomic datasets are essential for understanding the genetic factors influencing plant diseases. These datasets contain genetic information about plants, pathogens, and their interactions. By studying gene expression patterns, genetic markers, and disease resistance genes, researchers can identify molecular mechanisms underlying disease susceptibility and develop targeted interventions, such as breeding disease-resistant crop varieties.

Historical data plays a crucial role in predictive modeling and disease forecasting. These datasets include information about past disease outbreaks, weather patterns, soil characteristics, crop management practices, and pest populations. Analyzing historical data enables researchers to identify trends, correlations, and risk factors associated with specific diseases, leading to more informed decision-making and proactive disease management strategies.

Crowdsourced datasets are increasingly valuable in plant disease detection research. Platforms like PlantVillage and Agro-industry collect data from farmers, agronomists, and plant experts worldwide. These datasets encompass diverse geographic regions, crop types, and environmental conditions, enriching the diversity and robustness of disease detection models. Leveraging crowd sourced data facilitates knowledge sharing, collaboration, and collective efforts in combating plant diseases on a global scale.

Augmented datasets combine real data with synthetic data generated through techniques like data augmentation. This approach enhances model generalization and resilience by exposing the model to a broader range of scenarios, variations, and anomalies. Data augmentation techniques include image transformations, noise injection, and simulated environmental conditions, creating a more comprehensive training environment for disease detection models.

In conclusion, training datasets in plant disease detection encompass a wide range of data types, each contributing unique insights and challenges to the development of accurate and reliable detection models. By leveraging diverse datasets, researchers and developers can enhance the effectiveness, efficiency, and scalability of disease detection technologies, ultimately benefiting agricultural productivity, sustainability, and food security globally.

## 5.2 TESTING THE MODE

Testing datasets are crucial components in the development and evaluation of plant disease detection models, serving to validate the performance, accuracy, and robustness of these models under real-world conditions. These datasets complement training datasets by providing independent samples for model evaluation, generalization assessment, and bench marking against established standards. The inclusion of diverse testing datasets enhances the reliability and applicability of disease detection models across different plant species, diseases, and environmental contexts.

One type of testing datasets commonly used in plant disease detection is the validation datasets. This datasets is typically a subset of the overall training datasets, containing images or sensor data that were not used during the model training phase. By evaluating model performance on a validation datasets, researchers can assess how well the model generalizes to unseen data and identify potential over fitting or bias issues. Validation datasets are essential for fine-tuning model hyper parameters, optimizing performance metrics, and ensuring model robustness across various data distributions. External testing datasets are another important category in plant disease detection evaluation. These datasets are collected independently from the training and validation datasets, often sourced from different geographic regions, cultivation practices, and environmental conditions.

External testing datasets provide a more realistic assessment of model performance in diverse agricultural settings, reflecting the variability and complexity of real-world disease scenarios. Evaluating models on external testing datasets helps validate their reliability, scalability, and transfer ability across different agricultural contexts, ensuring practical utility and adoption by farmers and stakeholders.

Cross-validation techniques are commonly employed to further enhance testing datasets diversity and reliability. Cross-validation involves partitioning the datasets into multiple subsets (folds), training the model on a subset of folds, and evaluating its performance on the remaining fold. This process is repeated iteratively, ensuring that each subset serves as both training and testing data. Cross-validation helps assess model stability, variability, and generalization

across different data partitions, reducing the risk of model overfitting and improving overall performance estimation.

Temporal testing datasets focus on evaluating model performance over time, capturing seasonal variations, disease cycles, and evolving pathogen dynamics. These datasets include longitudinal data collected at regular intervals, such as weekly or monthly observations of plant health and disease incidence. By analyzing temporal testing datasets, researchers can assess model robustness against seasonal changes, disease outbreaks, and management interventions, enhancing the model's predictive capabilities and adaptability to dynamic agricultural conditions.

Benchmarking datasets play a crucial role in comparing the performance of different disease detection models, algorithms, and techniques. These datasets are standardized, curated, and widely used within the research community, allowing for fair and consistent performance evaluations. Benchmarking datasets often include ground truth annotations, performance metrics, and reference implementations, facilitating objective model comparisons and advancements in disease detection technology.

In summary, testing datasets in plant disease detection encompass a range of data types, including validation datasets, external testing datasets, cross-validation subsets, temporal datasets, and benchmarking datasets. These datasets are essential for evaluating model performance, generalization, stability, and scalability across diverse agricultural settings, ultimately driving innovation, reliability, and adoption of disease detection solutions in agriculture.

### **5.3 DISEASES DETECTING**

Detecting plant diseases involves several key steps to ensure accurate and reliable identification and management of crop health issues. The process typically begins with data collection, where various types of data such as images, sensor readings, and environmental factors are gathered to assess plant health. Once the data is collected, preprocessing steps are often applied to enhance the quality and usability of the data for disease detection algorithms.

Image preprocessing techniques may include normalization, noise reduction, and image enhancement to improve the clarity and consistency of plant images. For

sensor data, calibration and filtering techniques are applied to remove outliers and irrelevant noise, ensuring the accuracy of the input data for disease detection models.

Following preprocessing, feature extraction plays a crucial role in identifying relevant patterns and characteristics associated with plant diseases. In image-based detection, features such as color histograms, texture descriptors, and shape attributes are extracted to capture unique signatures of diseased plant regions. For sensor data, spectral signatures, chemical compositions, and physiological parameters are analyzed to detect anomalies indicative of disease presence.

Once features are extracted, machine learning and deep learning algorithms are employed for disease classification and detection. Supervised learning algorithms, such as support vector machines (SVM), random forests, and neural networks, are trained on labeled data to learn the relationships between input features and disease classes. These algorithms can then classify new data instances into relevant disease categories based on learned patterns and decision boundaries.

In recent years, deep learning models, particularly convolutional neural networks (CNN's), have shown remarkable performance in plant disease detection tasks. CNN's can automatically learn hierarchical representations of input data, enabling them to capture complex spatial and spectral features associated with different diseases. Models like Alex-net, Vignette, and Resnet have been adapted and fine-tuned for plant disease detection, achieving high accuracy and efficiency in identifying various crop diseases.

After classification, post-processing steps such as result interpretation, validation, and feedback mechanisms are applied to refine the detection outcomes. Interpretability techniques help explain the model's decisions and identify false positives or negatives, enhancing the model's reliability and trustworthiness. Validation methods such as cross-validation, holdout validation, and performance metrics evaluation assess the model's generalization and robustness across different datasets and scenarios.

Furthermore, feedback mechanisms integrate user feedback, domain knowledge, and continuous learning to improve the model's performance over time. By

incorporating feedback loops and adaptive algorithms, plant disease detection systems can evolve and adapt to new disease patterns, environmental changes, and emerging threats, enhancing their effectiveness in supporting precision agriculture and sustainable crop management practices.

## CHAPTER 6

### RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

#### 6.1 RESULT

Plant disease detection involves using various techniques to identify and classify diseases affecting plants. The result of such detection efforts can have significant implications for agriculture, enabling timely interventions to prevent crop losses and ensure food security.

One of the primary methods used in plant disease detection is image processing and machine learning. This approach typically involves capturing images of plant leaves or other parts affected by diseases and then applying algorithms to analyze these images. The result is a classification of whether the plant is healthy or diseased and, if diseased, identification of the specific disease.

#### 6.2 DISCUSSION

The outcome of plant disease detection can be described in several key aspects:

**Accuracy:** The accuracy of disease detection systems is crucial. High accuracy ensures that diseased plants are correctly identified, reducing the risk of misdiagnosis and unnecessary treatments. Advanced machine learning models trained on large datasets can achieve impressive accuracy rates, sometimes surpassing human experts in disease recognition.

**Speed:** Timely detection is essential to prevent the spread of diseases. Modern techniques allow for rapid image processing, enabling real-time or near-real-time detection in agricultural settings. This speed is critical for implementing timely interventions such as targeted pesticide application or disease-resistant crop varieties.

**Precision:** Precision in disease detection refers to the ability to distinguish between different types of diseases accurately. Some plant diseases may have similar visual symptoms, making precise identification challenging. Advanced algorithms can learn subtle differences in disease manifestations, improving precision in classification.

**Scalability:** The scalability of plant disease detection systems determines their practicality for large-scale agricultural applications. Scalable solutions can handle a high volume of image data efficiently, making them suitable for monitoring extensive crop fields or diverse plant species.

**Integration:** Integration with other agricultural technologies enhances the utility of disease detection systems. For example, integrating detection results with automated irrigation systems or robotic farming equipment enables targeted and efficient management practices.

**User-Friendly Interfaces:** The presentation of detection results through user-friendly interfaces is vital for practical adoption. Farmers and agricultural experts should be able to interpret results easily and make informed decisions based on the detected diseases' severity and distribution.

## CHAPTER 7

### SUMMARY AND CONCLUSIONS

#### 7.1 CONCLUSION

In this work, we successfully analyzed the different transfer learning models suitable for the accurate classification of 38 different classes of plant disease. Standardization and evaluation of state-of-the-art convolutional neural networks using transfer learning techniques were undertaken based on the classification accuracy, sensitivity, specificity, and F1 score. From the performance analysis of the various pre-trained architectures, it was found that DenseNet-121 outperformed ResNet-50, VGG-16, and Inception V4. Training the DenseNet-121 model seemed to be easy, as it had a smaller number of trainable parameters with reduced computational complexity. Hence, DenseNet-121 is more suitable for plant disease identification when there is a new plant disease that needs to be included in the model, demonstrating reduced training complexity. The proposed model achieved a classification accuracy of 99.81% and F1 score of 99.8%.

In future work, we will address the problems in real-time data collection and develop a multi-object deep learning model that can even detect plant diseases from a bunch of leaves rather than a single leaf. Furthermore, we are working towards implementing a mobile application with the trained model from this work. It will help farmers and the agricultural sector in real-time leaf disease identification.

#### 7.2 FUTURE WORK

The future of plant disease detection holds several exciting possibilities and areas for further research and development. Here are some potential avenues for future work in plant disease detection:

***Enhanced Accuracy with Deep Learning:*** Deep learning techniques, such as convolutional neural networks (CNN's) and recurrent neural networks (RNNs), can be further optimized for improved accuracy in disease detection. This includes developing novel architectures, leveraging transfer learning from pre-trained models, and exploring advanced feature extraction methods.

**Multi-Spectral Imaging:** Incorporating multi-spectral or hyper spectral imaging can provide additional information about plant health by capturing a broader range of wavelengths beyond visible light. This can help detect subtle changes in plant physiology associated with diseases, enhancing detection capabilities.

**Mobile and IoT Solutions:** Developing mobile applications and IoT (Internet of Things) devices for on-site disease detection can revolutionize agriculture by enabling farmers to monitor crops in real time. These solutions can integrate with drones, sensors, and handheld devices to provide instant feedback on disease presence and severity.

**Robotic Systems:** Integrating plant disease detection capabilities into autonomous robotic systems can enable automated scouting and targeted treatments. Robotic platforms equipped with cameras and AI algorithms can navigate fields, identify diseased plants, and apply interventions such as precision spraying or targeted pruning.

**Data Fusion and Decision Support Systems:** Combining data from multiple sources, such as satellite imagery, weather data, soil sensors, and disease databases, can enhance decision support systems for farmers. By leveraging big data analytics and AI, these systems can provide personalized recommendations for disease management strategies.

**Disease Forecasting Models:** Developing predictive models that forecast disease outbreaks based on environmental conditions, historical data, and disease dynamics can enable proactive management strategies. Early warning systems can alert farmers to potential disease risks, allowing for timely interventions and mitigation efforts.

**Crowd sourced Monitoring:** Engaging farmers and citizen scientists in crowd sourced monitoring programs can create large-scale datasets for training and validating disease detection algorithms. This collaborative approach fosters knowledge sharing and community-driven solutions for agricultural challenges.

Integration of Bio informatics: Incorporating bio informatics tools for genomics, transcriptions, and protoplasmic analysis can deepen our understanding of plant-pathogen interactions. Integrating comics data with image-based disease detection can lead to more comprehensive and accurate diagnostic capabilities.

Resilience to Climate Change: Designing disease detection systems that are resilient to climate change impacts, such as shifting disease patterns and emerging pathogens, is crucial. This includes adaptive algorithms and robust validation frameworks that account for changing environmental conditions.

Accessibility and Affordability: Ensuring that plant disease detection technologies are accessible and affordable for smallholder farmers and agricultural communities worldwide is a key focus. This involves designing low-cost hardware solutions, open-source software platforms, and capacity-building initiatives for technology adoption and skill development.

## REFERENCES

- [1] Honda Ochi et al., "Automation of Crop Disease Detection through Conventional Machine Learning and Deep Transfer Learning Approaches", vol.13(2), pp.352,2023.
- [2] Ahmed Samit Hatem et al., "A Transfer Learning-Based Artificial Intelligence Model for Leaf Disease Assessment", Vol. 29, No. 2, pp. 1030-1038, February 2023.
- [3] Vinay Gautam et al., "Disease detection in rice leaves using transfer learning techniques", vol.14(20), no.13610,2022.
- [4] Sunu Jatmika et al., "Rice Plants Disease Identification Using Deep Learning with Convolutional Neural Network Method", Vol. 7, No 2, July 2022.
- [5] Andrew J et.al "Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications", vol.12, pp.2395,2022.
- [6] Andrew J et.al., "Deep Learning-Based Leaf Disease Detection in Crops Images for Agricultural", vol.12, pp.2395,2022.
- [6] V. Malathi et al., "Classification of pest detection in paddy crop based on transfer learning approach, Acta Agriculture Scandinavica", vol.71:7, pp.552-559,2021.
- [8] Siddharth Swarup Rattray et al., "International Journal of Recent Technology and Engineering (IJRTE)", Volume-8 Issue-6, ISSN: 2277-3878, March 2020

## APPENDIX

### A. SOURCE CODE

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from matplotlib.image import imread

import cv2

import random

import os

from os import listdir

from PIL import Image

import tensorflow as tf

from keras.preprocessing import image

from tensorflow.keras.utils import img_to_array, array_to_img

from keras.optimizers import Adam

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D

from keras.layers import Activation, Flatten, Dropout, Dense

from sklearn.model_selection import train_test_split

from keras.models import model_from_json

from keras.utils import to_categorical

plt.figure(figsize=(12,12))

path="/content/drive/MyDrive/Plant-Leaf-Diseases

Prediction/Dataset/Potato_Early_blight"

for i in range(1,17):
```

```

plt.subplot(4,4,i)

plt.tight_layout()

rand_img = imread(path +'/' + random.choice(sorted(os.listdir(path))))

plt.imshow(rand_img)

plt.xlabel(rand_img.shape[1], fontsize = 10)#width of image

plt.ylabel(rand_img.shape[0], fontsize = 10)#height of image

def convert_image_to_array(image_dir):

    try:

        image = cv2.imread(image_dir)

        if image is not None :

            image = cv2.resize(image, (256,256))

            return img_to_array(image)

        else :

            return np.array([])

    except Exception as e:

        print(f"Error : {e}")

        return None

dir = "/content/drive/My Drive/Plant-Leaf-Disease-Prediction/Dataset"

image_list, label_list = [], []

all_labels = ['Tomato-Bacterial_spot', 'Potato-Early_blight', 'Corn-Common_rust']

binary_labels = [0,1,2]

temp = -1

for    directory    in    ['Tomato_Bacterial_spot',    'Potato_Early_blight',

'Corn_(maize)_Common_rust']:

    plant_image_list = listdir(f"{dir}/{directory}")

```

```

temp += 1

for files in plant_image_list:

    image_path = f"{dir}/{directory}/{files}"

    image_list.append(convert_image_to_array(image_path))

    label_list.append(binary_labels[temp])

label_counts = pd.DataFrame(label_list).value_counts()

label_counts.head()

image_list[0].shape

label_list = np.array(label_list)

label_list.shape

x_train, x_test, y_train, y_test = train_test_split(image_list, label_list,
test_size=0.2, random_state = 10)

x_train = np.array(x_train, dtype=np.float16) / 255.0

x_test = np.array(x_test, dtype=np.float16) / 255.0

x_train = x_train.reshape(-1, 256,256,3)

x_test = x_test.reshape(-1, 256,256,3)

y_train = to_categorical(y_train)

y_test = to_categorical(y_test)

model = Sequential()

model.add(Conv2D(32, (3, 3), padding="same",input_shape=(256,256,3),
activation="relu"))

model.add(MaxPooling2D(pool_size=(3, 3)))

model.add(Conv2D(16, (3, 3), padding="same", activation="relu"))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

```

```

model.add(Dense(8, activation="relu"))
model.add(Dense(3, activation="softmax"))
model.summary()
model.compile(loss = 'categorical_crossentropy', optimizer =
Adam(0.0001), metrics=['accuracy'])
x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size = 0.2,
random_state = 10)
epochs = 50
batch_size = 128
history = model.fit(x_train, y_train, batch_size = batch_size, epochs = epochs,
validation_data = (x_val, y_val))
plt.figure(figsize=(12, 5))
plt.plot(history.history['accuracy'], color='r')
plt.plot(history.history['val_accuracy'], color='b')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['train', 'val'])

plt.show()
print("Calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")
y_pred = model.predict(x_test)
img = array_to_img(x_test[11])

```

```
img

print("Original Label: ",all_labels[np.argmax(y_test[11])])

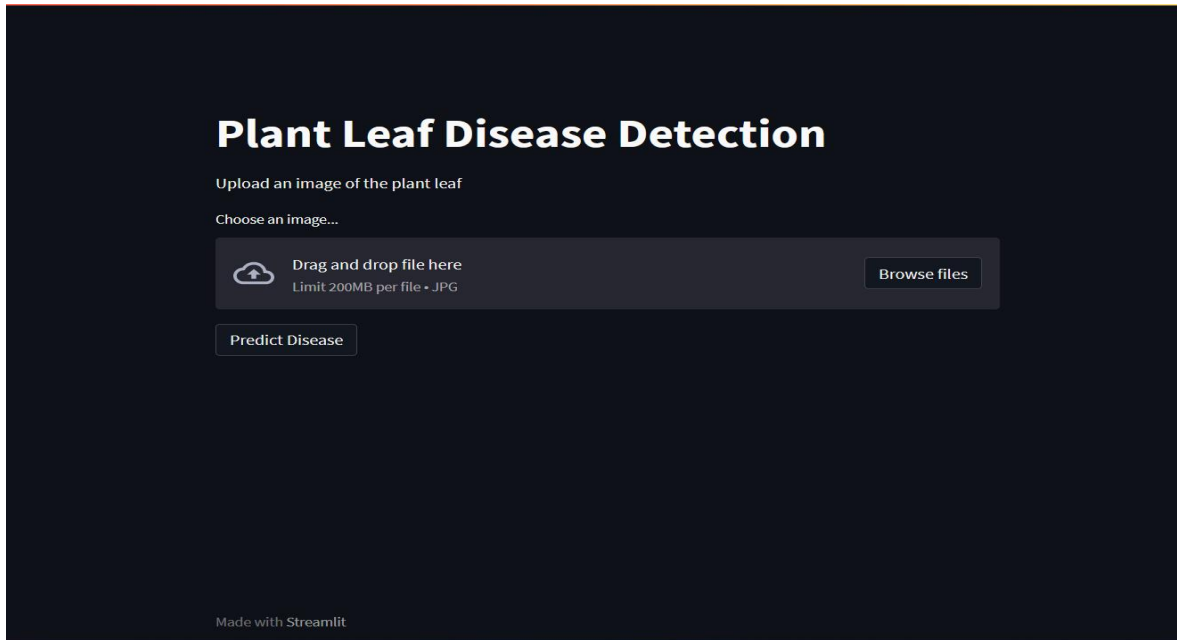
print("Predicted Label: ",all_labels[np.argmax(y_pred[4])])

print (y_pred[2])

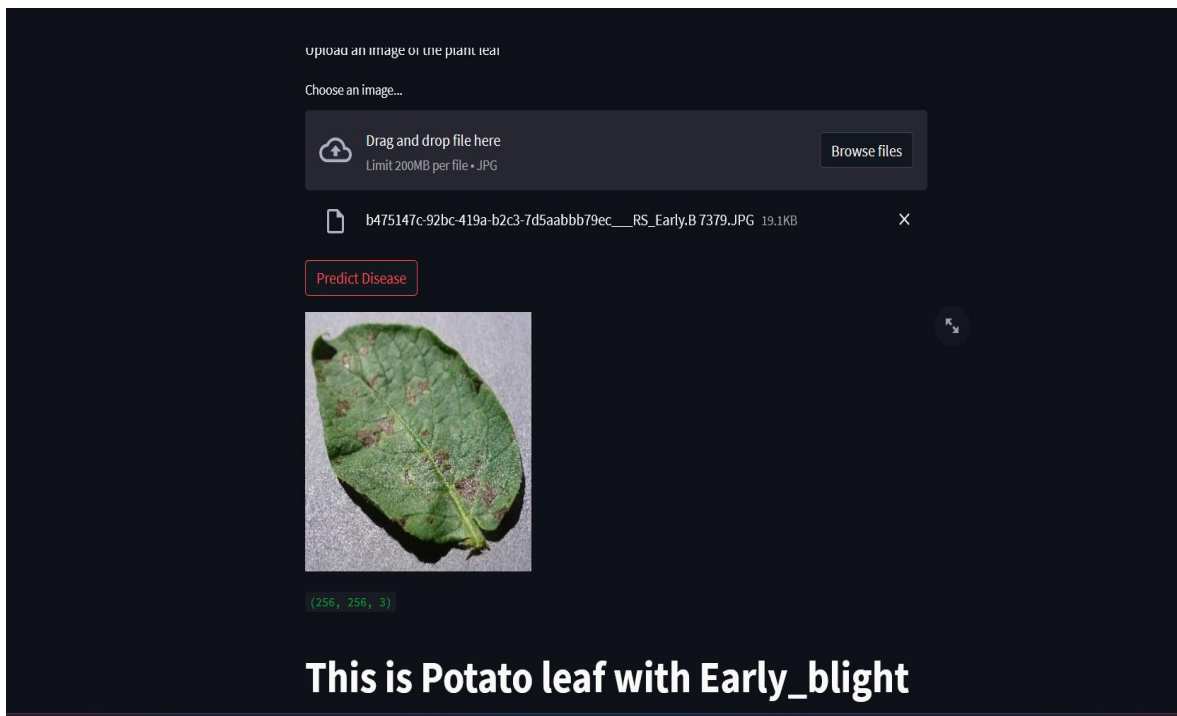
for i in range(50):

    print (all_labels[np.argmax(y_test[i])], "-", all_labels[np.argmax(y_pred[i])])
```

## B.SCREENSHOTS



### *LOGIN PAGE*



### *POTATO LEAF WITH EARLY BLIGHT*

Upload an image of the plant leaf

Choose an image...

Drag and drop file here  
Limit 200MB per file • JPG

Browse files

RS\_Rust\_2494.JPG 13.0KB

Predict Disease



(256, 256, 3)

**This is Corn leaf with Common\_rust**

***CORN LEAF WITH COMMON\_RUST***

Upload an image of the plant leaf

Choose an image...

Drag and drop file here  
Limit 200MB per file • JPG

Browse files

dfce64d1-9954-4922-8beb-2681e2e6b896\_\_GCREC\_Bact.Sp 2961.JPG 13.5KB

Predict Disease



(256, 256, 3)

**This is Tomato leaf with Bacterial\_spot**

***TOMATO LEAF WITH BACTERIAL\_SPOT***

## C.RESEARCH PAPER

# COMPARITIVE STUDY OF DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF DISEASES IN CROP

Tippasani Charan kumar  
reddy

*Department of computer  
science and Engineering*

*Sathyabama Institute of  
Science and Technology*

Chennai, India

[tippasanicharan@gmail.com](mailto:tippasanicharan@gmail.com)

Vadla Chiranjeevi Achari

*Department of computer  
science and Engineering*

*Sathyabama Institute of  
Science and Technology*

Chennai, India

[vadlachiru221@gmail.com](mailto:vadlachiru221@gmail.com)

Ms.Gowri Manohari

*Assistant Professor, Department of  
computer science and Engineering*

*Sathyabama Institute of Science and  
Technology*

Chennai, India

[gowrimanohari.cse@sathyabama.ac.in](mailto:gowrimanohari.cse@sathyabama.ac.in)

**Abstract**— Plant diseases have the potential to seriously harm crops, lowering yields and costing farmers money. Effective disease management requires early diagnosis of plant diseases. In this research, we use deep learning and image processing to create an automated system for identifying plant illnesses. To precisely detect the existence of disease, the system will examine photos of sick plant leaves and contrast them with healthy plant leaves. Plant disease identification may become much more accurate and efficient with the help of the suggested approach, which would eventually boost agricultural yields and enhance food security. With plant diseases responsible for an estimated 20–40% of annual crop production losses worldwide, they represent a serious danger to global food security. For plant diseases to be effectively managed, early detection is essential since it allows farmers to respond quickly to stop the disease's spread. On the other hand, labour- and time-intensive classical disease detection techniques like visual inspection can be used. Consequently, the demand for automated plant disease detection systems is rising. By helping farmers and other agricultural experts to rapidly and effectively identify damaged plants, the suggested method has the potential to completely transform the field of plant disease identification. In the end, this will boost crop production and enhance food security by halting the spread of illness. Additionally, the system will be expandable, making it simple to adjust to newly discovered plant species and illnesses. All things considered, this initiative marks a significant advancement in the creation of automated systems for the identification of plant diseases, whether or not the app is authentic. The machine is able to learn and understand the sentiments and emotions of reviews and other writings by using sentimental analysis and data mining. One of the most important components of app ranking fraud is the manipulation of reviews. To anticipate the outcomes, we employed the LSTM model.

**Keywords**— convolutional neural networks (CNNs), ANN Segmentation, SVM

## 1.INTRODUCTION

Inexperienced pesticide use can lead to the development of long-term pathogen resistance. There are various methods for identifying plant diseases. The issue of effective plant disease protection is intimately linked to the issue of sustainable agriculture. A farmer may not be able to diagnose a disease until it affects crop productivity since certain diseases have no outward signs or their consequences manifest too late for action. In such situations, an automated system that can detect plant diseases based on the appearance of the plant and its visible symptoms might be very beneficial to both inexperienced farmers and qualified experts as a means of confirming the diagnosis of diseases. Due to repeated application of the same pest, any bacterium or fungus can become accustomed to the pesticide. Conventional illness detection techniques rely on the laborious and sometimes error-prone visual assessment of human experts. With the use of photographs of leaves, plant diseases can now be automatically detected thanks to developments in image processing and machine learning algorithms.

These algorithms can learn to effectively detect and diagnose plant diseases in a fraction of the time it would take a human expert by using enormous datasets of photos of both healthy and diseased plants. Relevant features are extracted from photos of plants or plant sections that show signs of disease using image processing techniques. Among other things, these characteristics can have colour, texture, shape, and size. The CNNs can also be trained to identify specific diseases based on the symptoms present in the images. There are several challenges in developing an effective automatic detection

system for plant diseases. One of the main challenges is the availability of large datasets of high-quality images of diseased and healthy plants. Another challenge is the development of robust algorithms that can handle variations in lighting, background, and plant growth stage. Finally, there is a need for user-friendly interfaces that can be easily adopted by farmers and other stakeholders.

The use of automated systems for disease detection has several advantages over traditional methods, including faster detection, higher accuracy, and lower cost. It can also facilitate the early detection of diseases, allowing for prompt and targeted intervention to prevent further spread and damage. Image processing and deep learning techniques offer a promising solution for detecting plant diseases accurately and efficiently. These techniques enable the automated analysis of large volumes of plant images, identifying the presence and severity of diseases in Realtime. Applications for this technique range from small-scale gardens to large-scale agricultural regions. Large datasets of annotated photos are used to train these algorithms to identify patterns and characteristics linked to various diseases. The algorithm can be taught and then used to identify and categorize the disease in new photos. Deep learning and image processing have the ability to detect plant illnesses automatically. Some advantages include early disease identification, less pesticide and herbicide use, higher crop production and quality, and eventually, enhanced food security. However, there are still challenges to overcome, such as the need for large datasets of high-quality images, the variability of plant diseases, and the need for reliable and accessible technology for farmers.

### 1.1 Existing System

**Limited Scope:** The existing system is designed to identify diseases in paddy crops only. This limited scope hinders its applicability to a broader range of agricultural crops, which is a significant drawback in the context of diverse farming needs. **Inadequate Training Data:** The system was trained with a minimal dataset of 75 images, which is insufficient for accurate and robust disease prediction. A small dataset can lead to inadequate model performance and a lack of generalizability. **Inaccurate Image Segmentation:** The use of K-means clustering and Otsu's classifier for image segmentation is noted to lack accuracy. Inaccurate segmentation can result in imprecise identification of disease affected areas on leaves. **Limited Disease Identification:** The system provides information about the extent of infected regions but does not identify the specific disease affecting the plant. This limitation is a considerable drawback, as identifying the exact disease is crucial for appropriate treatment. **No Real-time Monitoring:** The existing system does not support real-time or near-real-time disease monitoring. Real-time monitoring is vital for timely intervention and disease management in agriculture, making this a significant limitation. These disadvantages underscore the need for improvements and advancements in the system to enhance its accuracy, versatility, and practicality for disease prediction and monitoring in agriculture.

### 1.2 Proposed System

There are many existing systems present using different algorithms for particular crop but our software has the feasibility of any identifying any crop pest that we can train diseases for paddy or teak plant etc. so our system is beneficial in that way for any kind of farmer as technology is developing pests are also developing there immunity to the pesticides this term is called adaption so we will keep updated remedies for the immunised pests our remedies are beneficial to the farmers with minimum cost as well as Eco friendly to the environment. Farmers do not have lot of task to be involved they are simple so that they can bring the images to the organisation there are sample of thousands of images are taken to get maximum efficiency from the algorithm.

## 2.LITERATURE SURVEY

Saradhamahal et al. demonstrated in their research based on how they acquired the images of the healthy and diseased crop leaves from the agricultural field. Initially the obtained image contains the dust so to remove that image pre-processing is used. After that this RGB image is converted to  $L^*a^*b$  color space and then is done with the help of Knn algorithm (Euclidean distance). And then otus classifier is used to generate the image from the cluster formed by then KNN. In this way they classified and detected the diseased region of the leaf. if the new image is given to system, it can identify the disease of the plant [1].

Sachin D. Khirade et al. proved in their study about disease identification has significance in preventing agricultural losses. By applying the normal methods for identification of the disease it takes some period of time to get the result whether disease is present or not. So by using image processing we can easily find the disease of the crop. To diagnose crop disease, a few stages are involved: acquiring the image, preprocessing it, segmenting it, extracting features, and classifying it. Here, they discussed a number of techniques for breaking down the illness portion. In order to extract the characteristics of the diseased plant leaf and classify different plant illnesses, they also talked about certain feature extraction and classification strategies. They stated that ANN techniques, such as self-organizing feature maps, back propagation algorithms, SVMs, etc., can be effectively applied for the classification of sick crop leaves. These techniques use image processing to measure plant diseases effectively and precisely [2].

Dhiman Mondal et al. exhibited how they used classification and detection methods for classification of the plant diseases. Initially preprocessing is done after that feature extraction is done. To extract the vein image from each plant leaf, RGB photos are first transformed to white, and then to greyscale. Basic morphological functions are applied to the image after the conversion. After that, picture 4 is changed to a binary image. We should transform any 0 pixel values to RGB values if they exist. In their study, R. Meena Prakash et al.

demonstrated how they assessed the use of digital image processing for leaf disease identification and categorization[3]. They used a method for detection and classification. Kmeans segmentation is used for segmentation of diseased part of the leaf. GLCM features are extracted and classification is done with help of SVM algorithm. This method is used for detection of disease in citrus leaves. In future implementations we can add different diseases of plant to improve classification accuracy [4].

Gurleen Kaur Sandhu et al. exemplified their study about different Plant Disease Detection Techniques. The analysis surveys and sums up different methods of plant sickness location utilizing picture handling that have been utilized by various scientists in the previous barely any years. The methods that are utilized are BPNN, SVM, K-implies bunching,Otsu's calculation, CCM and SGDM. These procedures are utilized to recognize if the leaves are solid or ailing. Different difficulties emerge in this procedure including the mechanization of the identification framework utilizing complex pictures caught in outside lightning and serious natural conditions [5].Mohanty et al. evaluated in their study how they used deep L learning and Convolutional Neural Network (CNN) to classify diseased and healthy plants. They achieved an accuracy of 99.35 percent using the Leafsnap dataset, which consists of over 50,000 images of plant leaves [6].

The study highlighted the potential of deep learningbased approaches for early detection and management of plant diseases and identified the current challenges and future directions for research in this field [7].

### 3.METHODOLOGY

#### 3.1 System Architecture

The architecture and overall design of a plant disease prediction system utilizing Convolutional Neural Networks (CNNs) is a multifaceted process that begins with the collection of a diverse dataset of plant images encompassing both healthy and diseased instances. This data serves as the foundation for training and validating the CNN-based model. Subsequently, the collected images undergo rigorous preprocessing, involving tasks like resizing, normalization, and enhancement to ensure uniformity and quality. Once the data is primed, the system employs image segmentation techniques to pinpoint regions within the images that are affected by disease. This segmentation can be achieved through sophisticated methods, including semantic segmentation using a dedicated CNN trained for pixel-level disease mask prediction.Feature extraction is a pivotal step where the system captures pertinent information from the segmented regions. This may encompass extracting deep learning feature vectors from fine-tuned CNN layers, analyzing color histograms to capture color intensity distributions, and scrutinizing texture features to detect textural patterns. The selection of an appropriate CNN architecture is

central to the design, with choices ranging from well-established models like VGG, ResNet, or Inception, tailored to the specific task through the modification of the output layer to accommodate the number of disease classes. Model training entails training the CNN model using the preprocessed data, including data augmentation to diversify the dataset, hyperparameter tuning to optimize learning rates and other parameters, and cross-validation to rigorously assess model performance. Evaluation is carried out using a dedicated testing dataset, with performance metrics such as accuracy, precision, recall, F1-score, and confusion matrices serving as key indicators.rates and other parameters, and cross-validation to rigorously assess model performance. Evaluation is carried out using a dedicated testing dataset, with performance metrics such as accuracy, precision, recall, F1-score, and confusion matrices serving as key indicators.



Fig 3.1 System Architecture

The Figure 3.1, shows the system architecture which represents the conceptual model that defines the structure, behavior and more view of the system. Process in the system is running step by step . First we have to take the input leaf and pre-process the whole data of input images and extract the features of the leaf and classify the leaves according to the features and collect the data of the training images from the database and compare the input leaf with the training data and classify the leaf whether leaf is healthy or unhealthy.

#### 3.2 Model Description

##### 3.2.1 Data Collection

Collection of dataset is done through online resources, agriculture resources and classified them accordingly for training and testing process. Collecting data for a plant disease prediction project is a critical step to ensure the effectiveness of

the system. Here are steps for data collection: 1. Identify Target Crops: Determine the specific crops for which you want to build a disease prediction system. Different crops may have unique diseases, so it's important to focus on one or more specific types. 2. Collaborate with Agricultural Experts: Collaborate with agricultural experts, local farmers, or agricultural extension services to gain insights into common diseases affecting the chosen crops in your region. They can provide valuable information on disease prevalence and symptoms.

3. Data Sources: a. Field Surveys: Conduct field surveys to visually inspect and collect data on infected and healthy plants. Document the symptoms and capture high-quality images of the leaves, stems, and fruit. b. Plant Pathology Labs: Collaborate with plant pathology labs or agricultural research institutions to obtain data on known diseases, including microscopic images of pathogens and infected plant tissue. c. Online Databases: Utilize existing online databases or repositories that provide open-access plant disease images and information. Some examples include the PlantVillage and UCI Machine Learning Repository. 4. Image Collection: a. Use high-resolution cameras or smartphones to capture images of both healthy and diseased plants. Ensure consistent lighting and angle for image quality. b. Annotate the images to specify the type of disease, its severity, and any relevant metadata such as crop type, location, and date. 5. Data Augmentation: These may include flipping, rotating, and altering image brightness and contrast. 6. Validation Data: Collect a separate set of images for validation and testing.

7. Metadata: Record metadata associated with each image, including location, climate conditions, and disease stage, to enable further analysis. 8. Ethical Considerations: Ensure you have the appropriate permissions and follow ethical guidelines when collecting data from farmers' fields or collaborating with research institutions. Respect privacy and intellectual property rights. 9. Data Quality Control: Review collected data for accuracy and consistency. Eliminate duplicate or low-quality images and ensure proper labeling and metadata. 10. Scale and Balance: Aim for a balanced dataset that represents different disease types and severities. Scaling the dataset appropriately will prevent biases in the machine learning model. 11. Data Storage and Management: Organize your data in a structured manner, store it securely, and create backups to prevent data loss. 12. Data Anonymization: If data includes sensitive information about the farmers or locations, consider anonymizing the data to protect privacy. 13. Data Documentation: Maintain thorough documentation of your dataset, including data sources, collection dates, and any preprocessing steps.

Data collection is an ongoing process that may require continuous updates to reflect changes in disease prevalence or the introduction of new crops. The quality and diversity of your dataset are crucial for the success of your plant disease prediction system.

### 3.3 Dataset:

The dataset consists of 12495 individual data. There are 5 columns in the dataset, which are described below

Index : unique

Disease : disease name

Description : disease description

Possible steps : remedies

Image : image url

#### 3.4 Importing the necessary libraries:

Python will be the language we use for this. To begin, we will load the required libraries, including pandas, numpy, matplotlib, and tensorflow, as well as keras for creating the primary model, sklearn for dividing the training and test data, and PIL for transforming the images into an array of integers..

#### 3.5 Data Preprocessing:

When employing Convolutional Neural Networks (CNNs) to identify plant diseases, data preprocessing is an essential first step. To make sure the input photos are appropriate for training and analysis, it entails preparing them.

**3.5.1 Resizing:** To standardize their dimensions, all of the photos in the collection were shrunk to a common resolution (e.g., 224x224 pixels). To guarantee consistency during training and analysis, this step is crucial.

**3.5.2 Normalization:** The image pixel values were adjusted to a standard scale, usually ranging from 0 to 1. By reducing brightness and contrast differences, this normalization makes it possible for the CNN to extract useful patterns from the pictures.

**3.5.3 Image Enhancement:** To raise the caliber of the images, image improvement methods such noise removal and contrast correction were used. These improvements aid in highlighting significant characteristics and lowering noise levels in the dataset.

#### 3.6 Image Segmentation:

Image segmentation is a critical step to isolate the regions of interest, specifically the diseased parts of plant leaves. In this project, the following image segmentation techniques were used:

**3.6.1 Semantic Segmentation with CNNs:** Instead of using traditional segmentation methods, a CNN-based approach for semantic segmentation was employed. A fully convolutional network was trained to predict pixel-level disease masks. This approach allows for accurate delineation of disease-affected regions. **3.7 Feature Extraction:**

Feature extraction is the process of capturing relevant information from the preprocessed images, which is crucial for CNN-based disease detection.

**3.7.1 Deep Learning Feature Vectors:** Deep learning models, specifically CNNs, were used to extract feature vectors. These feature vectors represent high-level features learned by the network during training. These features capture the unique characteristics of diseased and healthy plant regions. **3.8 Classification:**

Classification is the final stage in the plant disease detection process, where the extracted features are used to

categorize images into disease classes. In this project, CNNs were used for image classification:

**3.8.1 Convolutional Neural Networks (CNNs):** CNNs were employed as both the feature extractor and classifier. A CNN model was trained to learn and differentiate between various disease classes. This deep learning approach has demonstrated success in identifying plant diseases with high accuracy. The use of CNNs for both feature extraction and classification offers a powerful end-to-end solution for plant disease detection. The CNNs can automatically learn and recognize complex patterns in the images, making them well-suited for this task. The accuracy of the CNN-based model may vary depending on factors such as the dataset size, model architecture, and training process. However, CNNs have proven to be effective in achieving high accuracy in plant disease detection tasks.

### 3.9 Finding the Data :

Finding a suitable dataset for your plant disease detection project is a crucial step. You can explore various sources to obtain relevant data for training and testing your CNN-based model. Here are some methods to find datasets for plant disease detection:

#### 3.9.1 Online Repositories:

Many online repositories provide free access to plant disease datasets.

Some popular repositories include:

[PlantVillage](<https://plantvillage.psu.edu/>)

[Kaggle Datasets](<https://www.kaggle.com/datasets>)

-[UCIMachineLearning

Repository](<https://archive.ics.uci.edu/ml/datasets/>)

#### 3.9.2 Agricultural Research Institutions:

Contact agricultural research institutions, universities, or government agencies that may have datasets related to plant diseases. They often conduct research on crop health and may have collected relevant data.

#### 3.9.3 Collaboration with Farmers:

Collaborate with local farmers or agricultural communities. They may provide you with access to their field data and images of diseased plants. Ensure you have their consent and follow ethical considerations.

#### 3.9.4 Data Crowdsourcing:

Create a crowdsourcing campaign to collect images from a wider audience. Online platforms like Zooniverse or iNaturalist can be used to engage volunteers in collecting data.

#### 3.9.5 Research Papers and Citations:

Explore research papers related to plant disease detection. Many papers include references to the datasets they used. You can follow these citations to locate relevant datasets.

#### 3.9.6 Open Data Initiatives:

Check if there are open data initiatives related to agriculture in your region. Government departments and organizations may share agricultural datasets.

#### 3.9.7 Sensor Data:

Consider integrating sensor data from IoT devices or drones for real-time monitoring of plant health. Sensor data can provide valuable information for your project.

### 3.9.8 Data Augmentation:

If you have limited data, consider augmenting your dataset by applying transformations like cropping, rotation, and changing lighting conditions to your existing images. This can increase the diversity of your dataset. When selecting a dataset, ensure that it contains images of both healthy and diseased plants. The dataset should be well-annotated, indicating the type of disease and, if possible, its severity. It's important to respect data licensing agreements and ethical considerations when using datasets from various sources. Remember that the quality and diversity of your dataset will significantly impact the performance of your CNN-based model. Ideally, you should aim for a balanced and representative dataset with sufficient examples of each disease class.

### 3.10 Testing the Data :

Testing phase is very important because it shows whether our prediction is correct or not. Similar to that training phase we have to import all the modules and packages. We need to assign the variable path with the directory of model. We load the model now and store the variable. Now the model has been loaded and ready for testing., the testing dataset is meticulously prepared through the same preprocessing steps applied during training, including image resizing, normalization, and enhancements to ensure consistency. Subsequently, the model's performance is assessed using various evaluation metrics, such as accuracy, precision, recall, F1-score, and the construction of a confusion matrix. These metrics help gauge the model's ability to correctly identify and classify images of healthy and diseased plants.

In addition to numerical assessments, a visual inspection of the model's predictions is conducted using a sample of images from the testing dataset. This qualitative evaluation offers insights into the model's strengths and weaknesses, revealing where it excels and where it might make errors. If the model's performance on the testing dataset falls short of expectations, further fine-tuning may be necessary. This could involve adjusting hyperparameters, incorporating data augmentation techniques, or revisiting the training process with a larger and more diverse dataset. Furthermore, for practical applications, real-world testing is considered, which entails deploying the model in agricultural fields to capture images of plants and predict diseases on-site. Ultimately, the testing phase is crucial for validating the model's readiness for real-world deployment, and the results should be carefully documented and reported to stakeholders. Continuous monitoring and refinement are also vital postdeployment to ensure sustained performance in dynamic agricultural environments.

### 3.11 Detecting the disease :

This module is for checking the image identified is diseased or not. Now we have the prepared model which is ready to predict the diseases but the only problem is it is not interactive because the common people or farmers cannot use it. So to make it interactive we need to have a front-end design for our model. The frontend have to be simple and interactive so we use basic html and css. Similar to that of training and testing we have to import the modules. Store the directory of

module.h5 in the filepath variable. Now load the model, convert it into the np-array and change its dimension using img-to-array and np.dimsen() functions respectively. Get the predicted value. Create the flask instance, render the html file using rendertemplate() function. Get the local host link using the app.run() function. Open that link in any browser there we will get a box to upload a picture, upload a picture then it shows the disease description and treatment method if the leaf is diseased, if not it shows the output as heal.

### 3.12 Identifying the remedies :

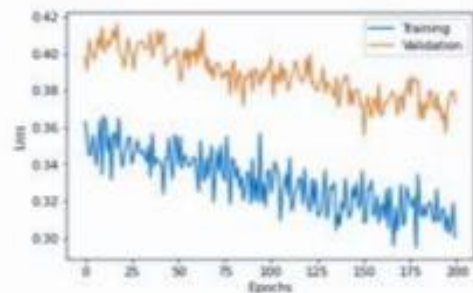
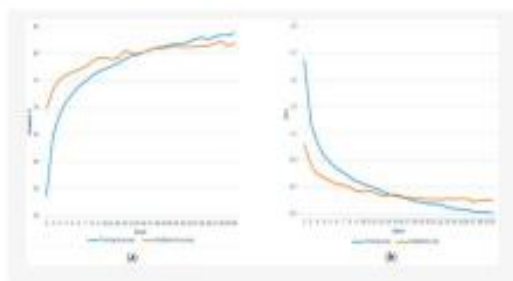
Finally if the disease is identified it will give the remedies for the disease, If the leaf is healthy it will show status as healthy.

## 4.RESULTS AND DISCUSSION

Part of this research uses state-of-the-art deep learning models to detect plant diseases using adaptive learning. PlantVillage is a publicly available dataset for further training of pre-trained deep CNN networks previously trained using the ImageNet dataset. In our experiment, each model is normalized with a learning of 0.01, an output of 0.5, and 38 output groups. The dataset is divided into training samples, test samples, and validation samples. A total of 80% of the PlantVillage samples were used for pretraining the Inception V4, VGG-16, ResNet, and DenseNet-121 models. Each model was run 30 times and it was observed that our model started to converge with high accuracy after 10 times.

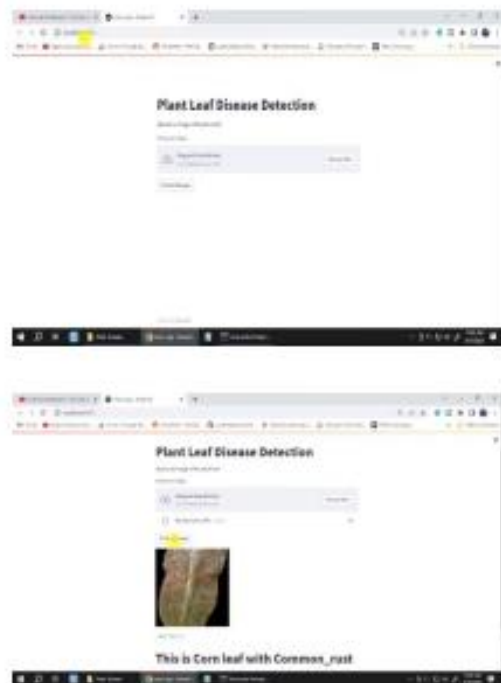
The second experiment evaluates the VGG-16 model using the same data set. After hyperparameter normalization, the model is trained using 80% of the same data; 10% is used for testing and 10% is used for testing and verification. It can be seen that the pattern recognition accuracy reaches 78% in the first 10 times, and then continuously improves, reaching the highest recognition rate of 84.27%, which is lower than the standard Inception V4. Learning loss and validity were 0.52% and 0.64%, respectively.

Model Accuracy



Prediction Page

On the prediction page the user can give input to create a text type and select a dog image. After selecting the dog picture, it will be displayed by clicking on the guess button.



## II. CONCLUSION

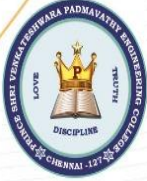
In this study, we evaluated the different mutation patterns required for the classification of 38 different strains. State-of-the-art convolutional neural networks are modeled and evaluated using classification accuracy, sensitivity, specificity, and learning variables based on F1 scores. By evaluating the performance of various pre-trained methods, DenseNet-121 was found to outperform ResNet-50, VGG-16, and Inception V4. The DenseNet-121 model appears easy to train due to the smaller number of training parameters and lower operational complexity. Therefore, when there are new diseases that need to be included in the sample, DenseNet-121 is suitable for recognizing plant diseases, thus reducing the learning difficulty. The proposed model achieved 99.81% classification rate and 99.8% F1 score.

In future work, we will solve the problem of real-time data collection and develop a deep learning model that can detect viruses even in a group of more than one leaf. We are also working to ensure that the education model in this project can also use mobile applications. It will help farmers and ranches quickly detect leaf diseases.

## III. References

- [1]Hosada Orchi,"Automation of Crop Disease Detection through Conventional Machine Learning and Deep Transfer Learning Approaches ",vol.13(2), pp.352,2023.
- [2]Ahmed Samir Hatem," A Transfer Learning-Based Artificial Intelligence Model for Leaf Disease Assessment", Vol. 29, No. 2, pp. 1030-1038, February 2023.
- [3]Vinay Gautam,"Disease detection in rice leaves using transfer learning techniques ", vol.14(20), no.13610,2022.
- [4]Sunu Jatmika," Rice Plants Disease Identification Using Deep Learning with Convolutional Neural Network Method",Vol. 7, No 2, July 2022.
- [5]Andrew J, "Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications ", vol.12, pp.2395,2022.
- [6]Sunu Jatmika, "Synchronous: Journal and Research in Informatics Engineering ", Vol. 7, No. 2, July 2022.
- [7]Andrew J," Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural ", vol.12, pp.2395,2022.
- [8]V. Malathi," Classification of pest detection in paddy crop based on transfer learning approach, Acta Agriculturae Scandinavica", vol.71:7, pp.552-559,2021.
- [9]Siddharth Swarup Rautaray," International Journal of Recent Technology and Engineering (IJRTE) ", Volume-8 Issue-6, ISSN: 2277-3878, March 2020.
- [10]Vimal K. Shrivastava," The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences", Volume XLII-3/W6, 2019.

## D.CERTIFICATE



**12th INTERNATIONAL CONFERENCE ON CONTEMPORARY ENGINEERING  
AND TECHNOLOGY 2024**

ORGANIZED BY  
ORGANIZATION OF SCIENCE & INNOVATIVE ENGINEERING AND TECHNOLOGY (OSIET), CHENNAI, INDIA.

IN COLLABORATION WITH  
**SAMARKAND STATE UNIVERSITY, UZBEKISTAN**

IN ASSOCIATION WITH  
**PRINCE SHRI VENKATESHWARA PADMAVATHY ENGINEERING COLLEGE**  
**PRINCE DR. K. VASUDEVAN COLLEGE OF ENGINEERING & TECHNOLOGY**

(Approved By All India Council For Technical Education, Affiliated To Anna University)  
Medavakkam - Mambakkam Main Road, Ponmar, Chennai - 600 127.

*Certificate of Presentation*

This is to certify that Mr/Mrs/Dr..... **Vadla Chiranjeevi Achari** ..... from  
**Sathyabama Institute of Science and Technology** ..... has presented a paper titled  
**COMPARITIVE STUDY OF DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING**  
**DIFFERENT TYPES OF DISEASES IN CROP** ..... in the "12th International  
Conference on Contemporary Engineering and Technology 2024" held on 23<sup>rd</sup> & 24<sup>th</sup> March 2024.

  
**Dr. Akhatov Akmal Rustamovich**  
Vice Rector of International Affairs,  
Samarkand State University, Uzbekistan

  
**Dr. Christo Ananth**  
Professor,  
Samarkand State University, Uzbekistan

  
**Dr. P. Aravinthan, M.E., Ph.D.,**  
Technical Lead, OSIET  
Secretary

  
**Prof. A. Krishnamoorthy, M.E., (Ph.D.)**  
Technical Lead, OSIET  
Administrator

  
**K. Janani, M.Tech.,**  
CEO, OSIET  
Director