

PLANT DISEASE DETECTION AND PREVENTION USING CNN

*Major project report submitted
in partial fulfillment of the requirement for award of the degree of*

**Bachelor of Technology
in
Computer Science & Engineering**

By

GUTHA SAINATH REDDY	(20UECS0368)	(17707)
AMUDALA BHANU TEJA	(20UECS0046)	(18300)
AMUDALA POOJITHA	(20UECS0047)	(18301)

*Under the guidance of
Mr. IGNATIUS K PIOUS, ME.,
ASSISTANT PROFESSOR*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING**

**VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF
SCIENCE & TECHNOLOGY**

(Deemed to be University Estd u/s 3 of UGC Act, 1956)

**Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA**

May, 2024

PLANT DISEASE DETECTION AND PREVENTION USING CNN

*Major project report submitted
in partial fulfillment of the requirement for award of the degree of*

**Bachelor of Technology
in
Computer Science & Engineering**

By

GUTHA SAINATH REDDY	(20UECS0368)	(17707)
AMUDALA BHANU TEJA	(20UECS0046)	(18300)
AMUDALA POOJITHA	(20UECS0047)	(18301)

*Under the guidance of
Mr. IGNATIUS K PIOUS, ME.,
ASSISTANT PROFESSOR*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING**

**VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF
SCIENCE & TECHNOLOGY**

(Deemed to be University Estd u/s 3 of UGC Act, 1956)

**Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA**

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "PLANT DISEASE DETECTION AND PREVENTION USING CNN" by "GUTHA SAINATH REDDY (20UECS0368), AMUDALA BHANU TEJA (20UECS0046), AMUDALA POOJITHA (20UECS0047)" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor

Computer Science & Engineering

School of Computing

Vel Tech Rangarajan Dr. Sagunthala R&D

Institute of Science & Technology

May, 2024

Signature of Professor In-charge

Computer Science & Engineering

School of Computing

Vel Tech Rangarajan Dr. Sagunthala R&D

Institute of Science & Technology

May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(GUTHA SAINATH REDDY)

Date: / /

(AMUDALA BHANU TEJA)

Date: / /

(AMUDALA POOJITHA)

Date: / /

APPROVAL SHEET

This project report entitled (PLANT DISEASE DETECTION AND PREVENTION USING CNN) by (GUTHA SAINATH REDDY (20UECS0368), (AMUDALA BHANU TEJA (20UECS0046), (AMUDALA POOJITHA (20UECS0047) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners

Supervisor

Mr. Ignatious K Pious, ME(Networking)

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN**, for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.**, for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.**, for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Mr. IGNATIOUS K PIOUS, M.E.**, for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E.**, for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

GUTHA SAINATH REDDY	(20UECS0368)
AMUDALA BHANU TEJA	(20UECS0046)
AMUDALA POOJITHA	(20UECS0047)

ABSTRACT

Plant diseases have a disastrous effect on the safety of food production; they lower the value and quantity of agricultural products. Plant diseases can result in extremely high losses or, in the worst circumstances, no harvest at all. Crop yields and plant growth at different stages are impacted by a variety of pests and diseases. The focus of this study was on plant disease identification and prevention using Leaf Spot. Many techniques have been put out for the diagnosis of plant diseases, however deep learning has emerged as the method of choice due to its extraordinary success. In this study, multiscale features were chosen using Convolutional Neural Network in order to exclude the undesirable background from an input image. This work suggests use the EfficientNetV2 model to detect plant diseases. To determine the effectiveness of the suggested strategy and evaluate it against alternative models like the Convolutional Neural Network and Efficient Net, a thorough series of tests was conducted. According to the experimental findings, the suggested method has a 93.26 percent detection accuracy. This high degree of accuracy demonstrates how well deep learning methods work for identifying plant diseases. By accurately identifying Leaf Spot, farmers may take prompt and targeted action to stop the disease's spread, reducing crop losses and guaranteeing food security. The results have applications outside the realm of academia, providing doable ways to improve food safety and agricultural output. Farmers are better able to control pests, allocate resources, and develop crop protection plans when deep learning-based disease detection technologies are included into agricultural processes. Moreover, the approach's versatility and scalability allow for its application in a variety of agricultural contexts across the globe. There are a number of directions that future study and development could go. It could be possible to improve detection accuracy even further by tweaking hyper parameters and fine tuning the model architecture.

Keywords: Computer Vision, Convolutional Neural Networks, Deep Learning, Disease Classification, Image Processing

LIST OF FIGURES

4.1	Convolutional Neural Networks Diagram	11
4.2	Data Flow Diagram	12
4.3	Use Case Diagram	13
4.4	Class Diagram	14
4.5	Sequence Diagram	15
4.6	Collaboration Diagram	16
4.7	Activity Diagram	17
4.8	Mobile Inverted Bottleneck Convolution Layers	19
5.1	Input Design	23
5.2	Output Design	24
5.3	Unit Testing	26
5.4	Integration Testing	28
5.5	System Testing	30
6.1	Sample Code Output	34
6.2	Sample Code Accuracy	35
8.1	Plagiarism Report	38
9.1	Poster Presentation	41

LIST OF TABLES

6.1	Comparison between Proposed System and Existing System	32
-----	--	----

LIST OF ACRONYMS AND ABBREVIATIONS

ADAM	Adaptive Moment Estimation
API	Application Programming Interface
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
CV	Computer Vision
DL	Deep Learning
F1	F1 Score (Harmonic Mean of Precision and Recall)
FN	False Negative
FP	False Positive
GPU	Graphics Processing Unit
GUI	Graphical User Interface
RMSProp	Root Mean Square Propagation
ROC	Receiver Operating Characteristic
ROI	Region of Interest
SGD	Stochastic Gradient Descent
TN	True Negative
TP	True Positive
VGG	Visual Geometry Group

TABLE OF CONTENTS

	Page.No
ABSTRACT	v
LIST OF FIGURES	vi
LIST OF TABLES	vii
LIST OF ACRONYMS AND ABBREVIATIONS	viii
1 INTRODUCTION	1
1.1 Introduction	1
1.2 Aim of the Project	2
1.3 Project Domain	2
1.4 Scope of the Project	2
2 LITERATURE REVIEW	3
3 PROJECT DESCRIPTION	6
3.1 Existing System	6
3.2 Proposed System	7
3.3 Feasibility Study	7
3.3.1 Economic Feasibility	8
3.3.2 Technical Feasibility	8
3.3.3 Social Feasibility	9
3.4 System Specification	9
3.4.1 Hardware Specification	9
3.4.2 Software Specification	10
3.4.3 Standards and Policies	10
4 METHODOLOGY	11
4.1 General Architecture	11
4.2 Design Phase	12
4.2.1 Data Flow Diagram	12

4.2.2	Use Case Diagram	13
4.2.3	Class Diagram	14
4.2.4	Sequence Diagram	15
4.2.5	Collaboration Diagram	16
4.2.6	Activity Diagram	17
4.3	Algorithm & Pseudo Code	18
4.3.1	Algorithm	18
4.3.2	Pseudo Code	18
4.3.3	Mobile Inverted Bottleneck Convolution Layers	19
4.4	Module Description	20
4.4.1	Dataset Collection:	20
4.4.2	Data Preprocessing:	20
4.4.3	Feature Extraction:	21
4.4.4	Model Selection and Metrics:	21
4.4.5	Configuration of the Classification Model:	21
4.4.6	Detection of Leaf Disease:	21
4.5	Steps to execute/run/implement the project	22
4.5.1	Installing Packages	22
4.5.2	Creating Virtual Environments	22
4.5.3	Evaluate the Model	22

5 IMPLEMENTATION AND TESTING 23

5.1	Input and Output	23
5.1.1	Input Design	23
5.1.2	Output Design	24
5.2	Testing	25
5.3	Types of Testing	25
5.3.1	Unit Testing	25
5.3.2	Integration Testing	27
5.3.3	System Testing	29
5.3.4	Test Result	30

6 RESULTS AND DISCUSSIONS 31

6.1	Efficiency of the Proposed System	31
6.2	Comparison of Existing and Proposed System	32
6.3	Sample Code	33

7	CONCLUSION AND FUTURE ENHANCEMENTS	36
7.1	Conclusion	36
7.2	Future Enhancements	37
8	PLAGIARISM REPORT	38
9	SOURCE CODE & POSTER PRESENTATION	39
9.1	Source Code	39
9.2	Poster Presentation	41
	References	41

Chapter 1

INTRODUCTION

1.1 Introduction

Plant diseases are any number of anomalous circumstances, illnesses, or pathogens that negatively impact a plant's health and vitality. There are several ways in which these illnesses might show up, including as physical harm, stunted development, lower yields, or even plant mortality. They present serious obstacles to global forestry, horticulture, and agricultural activities. Microorganisms that infiltrate plant tissues and interfere with regular cellular processes are known as pathogens. These include bacteria, fungus, viruses, and nematodes. They can proliferate by vectors, water, air, or soil. Extreme weather, drought, flooding, or pollution are examples of abiotic variables that can weaken plants and increase their susceptibility to disease. Certain diseases are genetically prone to affect certain plants. The goal of breeding initiatives is to create disease-resistant cultivars. Inadequate cultural practices can stress plants and foster an environment that is conducive to illness, such as overcrowding, poor watering, or inadequate nourishment. Crops and ecosystems must remain healthy for plant diseases to be prevented. To disrupt the life cycle of viruses, avoid planting the same crop in the same spot every year. Select plant kinds that are bred to be resistant to the illnesses that are common in your area. Remove any sick plant debris from planting areas, including any fallen leaves or fruits, to keep the area tidy. Water plants from the base up to reduce leaf moisture, but avoid overwatering as this might foster an environment that is favorable to fungal diseases. Plant with enough distance between each other to encourage ventilation and lower surrounding humidity. Addition of organic matter and appropriate pH adjustments will help maintain healthy soil. A variety of techniques suited to particular plant species, illnesses, and environmental factors, as well as attentiveness, are needed to prevent and manage plant diseases. By using convolutional neural network we get 93.26 percent detection accuracy. Protecting plant health and ensuring strong agriculture and horticultural practices require a proactive approach.

1.2 Aim of the Project

Plant leaf disease is often detected using deep learning-based methods and modern image processing. Convolutional neural networks and pre-trained models are used in several diagnostic techniques to identify and categorize healthy and ill plants.

1.3 Project Domain

Machine learning includes the use of Convolutional Neural Network for plant disease identification and prevention. Among artificial neural networks, Convolutional Neural Networks are particularly good at interpreting visual data, such as photographs. Convolutional Neural Networks are trained on labeled datasets in this application, which contain pictures of both healthy and sick plants. By automatically extracting features from the photos during training, the Convolutional Neural Network becomes capable of distinguishing between healthy and diseased plants based on visible symptoms. Once trained, unseen plant photos can be reliably classified by Convolutional Neural Network, allowing for early disease diagnosis. Farmers can minimize crop losses and lessen their need on chemical treatments by promptly identifying and addressing plant diseases through the integration of Convolutional Neural Network-based systems into agricultural workflows. In general, the use of Convolutional Neural Networks for plant disease identification and prevention is an excellent example of how machine learning may transform agricultural practices and improve sustainability and food security.

1.4 Scope of the Project

Plant leaf disease is often detected using deep learning-based methods and modern image processing. Many diagnostic techniques employ segmentation to exclude background data, a trained neural network for classification, and a Convolutional Neural Network using a pre-trained model to identify and categorize healthy and diseased plants. Zhang et al. suggested a method to diagnosis diseases of cucumber plants by integrating K-means, the condition and color of infected leaf lesions, and the sparse resentment approach to separate images containing diseased patches.

Chapter 2

LITERATURE REVIEW

[1] Nagaraju, et al., (2020), Plant leaf endemics are part of human life. Any changes in crops yield leads to shortage of food. Apple and Grape fruits are most profitable and also prone to many diseases. Apple trees often suffers from Scab, Black rot, and Cedar rust diseases and Grape plants suffers from Black Measles, Black rot, and Leaf Blight diseases. The productivity of Apple and Grape depends on early detection and diagnosis of diseases. The various parts of plants such as leaf, trunk, seed, blossom and fruit growth gets affected. Identification and classification of these endemics requires presence of farmer or plant pathologists. Manual diagnosis of disease might lead to misidentification and inappropriate use of pesticides and also consumes lot of time. Plants suffer from incorrect pesticide usage to diagnose endemics. There is a need for artificial ways in classifying diseases.

[2] Tasnim Ahmed, et al., (2020), Ensure global food security and the overall profit of stakeholders, the importance of correctly detecting and classifying plant diseases is paramount. In this connection, the emergence of deep learning-based image classification has introduced a substantial number of solutions. However, the applicability of these solutions in low-end devices requires fast, accurate, and computationally inexpensive systems. This work proposes a lightweight transfer learning-based approach for detecting diseases from tomato leaves. It utilizes an effective preprocessing method to enhance the leaf images with illumination correction for improved classification. Our system extracts features using a combined model consisting of a pretrained MobileNetV2 architecture and a classifier network for effective prediction. Traditional augmentation approaches are replaced by runtime augmentation to avoid data leakage and address the class imbalance issue.

[3] Elhoucine Elfatimi, et al., (2021), Angular leaf spot disease and bean rust disease are two of the many diseases associated with beans that prevent them from being produced. In recent years, plant leaf diseases have become a common problem for which accurate research and quick application of deep learning in plant disease classification is required. Beans are also one of the most important plants and seeds that are used worldwide for cooking in either dried or fresh form; they are a great source

of protein and offer many health benefits. Therefore, in order to address the issue at its earliest stages, a precise classification of bean leaf diseases is required. Using an open-source TensorFlow library, a MobileNet model, and a public collection of leaf images, a deep learning method is suggested to detect and categorize bean leaf disease. In this paper, we suggested a strategy for identifying and characterizing the effective network architecture (hyperparameters and optimization techniques) as well as for classifying bean leaf disease. In addition, we compared the outcomes of applying each architecture independently in order to determine which architecture configuration produced the best results for bean leaf disease classification.

[4] Lili Li, et al., (2021), One area of artificial intelligence is deep learning. Because of the benefits of autonomous learning and feature extraction, academic and industry circles have become increasingly concerned with it in recent years. Natural language processing, speech processing, and picture and video processing have all made extensive use of it. In addition, it has developed into a hub for research on agricultural plant protection, including the identification of plant diseases and the evaluation of pest ranges. Deep learning can be used to recognize plant diseases without the drawbacks of artificially selecting disease spot features. It can also increase the objectivity of plant disease feature extraction and accelerate the pace of technological advancement and research.

[5] Mobeen Ahmad, et al., (2021), Convolutional neural networks have proven to be the most effective in a variety of computer vision applications, including picture categorization. An important use of deep learning is the identification of plant diseases, which has been tackled by numerous contemporary techniques. Nonetheless, it is imperative to adapt these solutions for portable devices with limited resources, such smartphones. Because deep learning models require a lot of resources, this is a difficult task. In this paper, a systematic approach to classify symptoms of plant diseases using convolutional neural networks is proposed. These networks are memory-efficient, and by shortening training times, they facilitate the quick creation of industrial applications when combined with the suggested training configuration.

[6] Malusi Sibiyi, et al., (2021), Plant pathology had been enabled by the evolution of artificial intelligence. For instance, many researchers had used pre-trained convolutional neural networks such as the Visual Geometry Group-16, Inception, and Google Net to mention a few, for the classifications of plant diseases. The trend of using artificial intelligence for plant disease classification has grown to such an extent that some researchers were able to use artificial intelligence to also detect their

severities. The purpose of this study is to introduce a novel approach that is reliable in predicting severities of the maize common rust disease by convolutional neural networks deep learning models.

[7] Florent Retraint, et al., (2023), The Food and Agriculture Organization of the United Nations suggests increasing the food supply by 70 percent to feed the world population by 2050, although approximately one third of all food is wasted because of plant diseases or disorders. To achieve this goal, researchers have proposed many deep learning models to help farmers detect diseases in their crops as efficiently as possible to avoid yield declines. These models are usually trained on personal or public plant disease datasets such as PlantVillage or PlantDoc. PlantVillage is composed of laboratory images captured under laboratory conditions, with one leaf each and a uniform background.

[8] Khalid M. Hosny, et al., (2023), Plant diseases are one of the primary causes of decreased agricultural production quality and quantity. With ongoing changes in plant structure and cultivation techniques, new diseases are constantly arising on plant leaves. Thus, accurate classification and detection of plant leaf diseases in their early stages will limit the spread of the infection and support the healthy development of plant production. This work proposes a novel lightweight deep convolutional neural network (CNN) model for obtaining high-level hidden feature representations.

[9] Vasileios Balafas, et al., (2023), Precision agriculture is a rapidly developing field aimed at addressing current concerns about agricultural sustainability. Machine learning is the cutting edge technology underpinning precision agriculture, enabling the development of advanced disease detection and classification methods. This paper presents a review of the application of machine learning and deep learning techniques in precision agriculture, specifically for detecting and classifying plant diseases. We propose a novel classification scheme that categorizes all relevant works in the associated classes.

[10] Wasswa Shafik, et al., (2023), Plant pests and diseases are a significant threat to almost all major types of plants and global food security. Traditional inspection across different plant fields is time-consuming and impractical for a wider plantation size, thus reducing crop production. Therefore, many smart agricultural practices are deployed to control plant diseases and pests. Most of these approaches, for example, use vision-based artificial intelligence, machine learning, or deep learning methods and models to provide disease detection solutions.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The implementation of proper techniques to identify healthy and diseased leaves helps in controlling crop loss and increasing productivity. This section comprises different existing machine-learning techniques for the identification of plant diseases. The decrease in time per epoch was because the number of parameters in these models was quite smaller than that of other existing models. The decrease in time per epoch was because the number of parameters in these models was quite smaller than that of other existing models. To enhance the training and efficacy, it uses Fused mobile inverted bottleneck convolution for the first three stages and mobile inverted bottleneck convolution for the subsequent stages and this is slow than existing models, which are up to 6.8x smaller. In a computer vision and machine learning based techniques were developed for plant leaf disease detection. Real-time plant disease detection has some significant challenges, such as complex background and severity of the disease due to the images being captured in real-time scenarios from the farm field. These models undergo rigorous training processes where they learn to automatically extract intricate features from the images, enabling accurate classification of plant diseases. Through meticulous data preprocessing and model optimization, the Convolutional Neural Networks achieve high levels of accuracy in disease detection. Once trained and validated, the Convolutional Neural Network models are deployed in real-world scenarios, integrated into agricultural technologies such as drones or mobile applications.

Disadvantages

- The traditional Convolutional Neural Network is a simple supervised-machine-learning algorithm used in classification problems.
- Hyper parameters that deal with training include batch size, learning rate, and dropout the loss function is expressed.

3.2 Proposed System

A plant leaf disease detection method in this study that uses a background removal technique with U2-Net to eliminate the image's complex background. For classification, the EfficientNetV2 deep learning model is employed. To determine the effectiveness of the suggested strategy and evaluate it against alternative models like Efficient-Net and Convolutional Neural Network, an extensive series of tests was conducted. Using EfficientNetV2, a plant leaf disease detection method was suggested. Computer vision methods, such as grab cut approaches and image thresholding in Open Source Computer Vision Library, typically remove background from images. U2-Net was the background removal method employed in this investigation. A series of tests was conducted to ascertain the detection efficiency of the proposed approach.

Advantages of Proposed system

- Transfer learning has several advantages; for example, it does not need a large amount of data to train the network.
- This is hybrid deep learning model has the advantages of a residual network and retains unique properties.
- Their technical status, performance and stability are directly related to the train operation safety.

3.3 Feasibility Study

A feasibility study of plant disease detection and prevention using convolutional neural networks involves assessing technical, economic, and practical aspects. Firstly, the availability of labeled datasets and computational resources for training and deploying convolutional neural network models must be evaluated. Accuracy and reliability are critical, comparing convolutional neural network performance to traditional methods in detecting and classifying diseases. Cost analysis considers expenses for data collection, infrastructure, personnel, and maintenance. Scalability is crucial for adapting the system to diverse agricultural settings. Integration with existing systems, regulatory compliance, and ethical considerations are also examined.

3.3.1 Economic Feasibility

The economic feasibility of plant disease detection and prevention using convolutional neural networks involves evaluating the costs and benefits associated with implementing such a system. Initial costs include expenses for data collection, hardware and software infrastructure, model development, and personnel training. Ongoing costs may include maintenance, updates, and data storage. However, the long-term benefits of convolutional neural network-based disease detection can outweigh these costs by reducing crop losses, increasing yields, and optimizing resource utilization. By enabling early detection and targeted intervention, convolutional neural networks can minimize the need for costly chemical treatments and labor-intensive manual inspections. Additionally, the scalability of convolutional neural network-based systems allows for broader adoption across diverse agricultural settings, maximizing their economic impact. Overall, while initial investments may be significant, the potential economic returns and improvements in agricultural productivity make plant disease detection and prevention using convolutional neural networks economically feasible and advantageous for farmers and agricultural stakeholders.

3.3.2 Technical Feasibility

The technical feasibility of plant disease detection and prevention using convolutional neural networks hinges on several factors. Firstly, the availability of high-quality labeled datasets containing images of diseased and healthy plants is crucial for training accurate models. Additionally, the computational resources required for training and deploying convolutional neural networks, such as Graphics Processing Unit and deep learning frameworks, must be accessible. Integration with existing agricultural technologies and practices is essential to ensure seamless deployment and compatibility. Furthermore, the scalability of convolutional neural network-based systems enables adaptation to various crops, regions, and environmental conditions. Overall, while technical challenges such as data quality, computational requirements, and integration complexities may exist, advancements in machine learning and computer vision make plant disease detection and prevention using convolutional neural networks technically feasible and increasingly accessible for agricultural applications.

3.3.3 Social Feasibility

The social feasibility of employing convolutional neural networks for plant disease detection and prevention involves assessing the acceptance, impact, and implications of such technology within society. Stakeholder engagement and awareness campaigns are crucial for garnering support and addressing concerns among farmers, agricultural communities, and relevant authorities. Ensuring accessibility and affordability of convolutional neural network-based solutions is essential to promote equitable access to agricultural innovations across diverse socio-economic backgrounds. Moreover, collaboration with local communities and extension services can facilitate knowledge sharing and capacity building, empowering farmers to effectively utilize convolutional neural network based tools for disease management. By fostering inclusivity, transparency, and responsible innovation, plant disease detection and prevention using convolutional neural networks can contribute to societal well-being, sustainable agriculture, and food security while addressing social needs and priorities.

3.4 System Specification

- Large-scale labeled datasets comprising diverse images of diseased and healthy plants, annotated with relevant metadata.
- High-resolution images (e.g., 224x224 pixels or higher) for accurate feature extraction and classification by the convolutional neural network.
- Annotated metadata containing information about plant species, disease types, and environmental conditions for each image.

3.4.1 Hardware Specification

- High-performance Graphics Processing Unit with Compute Unified Device Architecture support for accelerated deep learning computations.
- Minimum of 16 GB RAM for efficient model training and inference processes.
- Multi-core Central Processing Unit for handling data preprocessing tasks and model deployment.

3.4.2 Software Specification

- Deep learning frameworks like TensorFlow or PyTorch for building and training convolutional neural network models.
- Python programming language for scripting and implementing machine learning pipelines.
- Image processing libraries such as OpenCV for data preprocessing, augmentation, and feature extraction.
- Development environments like Jupyter Notebook or Integrated Development Environments for coding and experimentation.

3.4.3 Standards and Policies

Standards and policies governing plant disease detection and prevention using convolutional neural networks are critical for ensuring ethical, secure, and effective practices. These guidelines encompass various aspects, including data privacy and security, ethical data usage, model transparency, regulatory compliance, collaboration, security measures, and environmental and social impact.

Tensorflow model optimizer:

- Usage: This command is used to optimize a TensorFlow model for inference on a specific hardware platform.
- Example: It takes the input model file model.pb, specifies the input shape, output format, and other parameters to optimize the model for efficient inference.

Torch-model-archiver:

- Usage: This command is used to archive a PyTorch model for deployment.
- Example: It archives the model plant disease model.pt into a serialized file model.pth along with a handler script plant disease handler.py for inference.

Chapter 4

METHODOLOGY

4.1 General Architecture

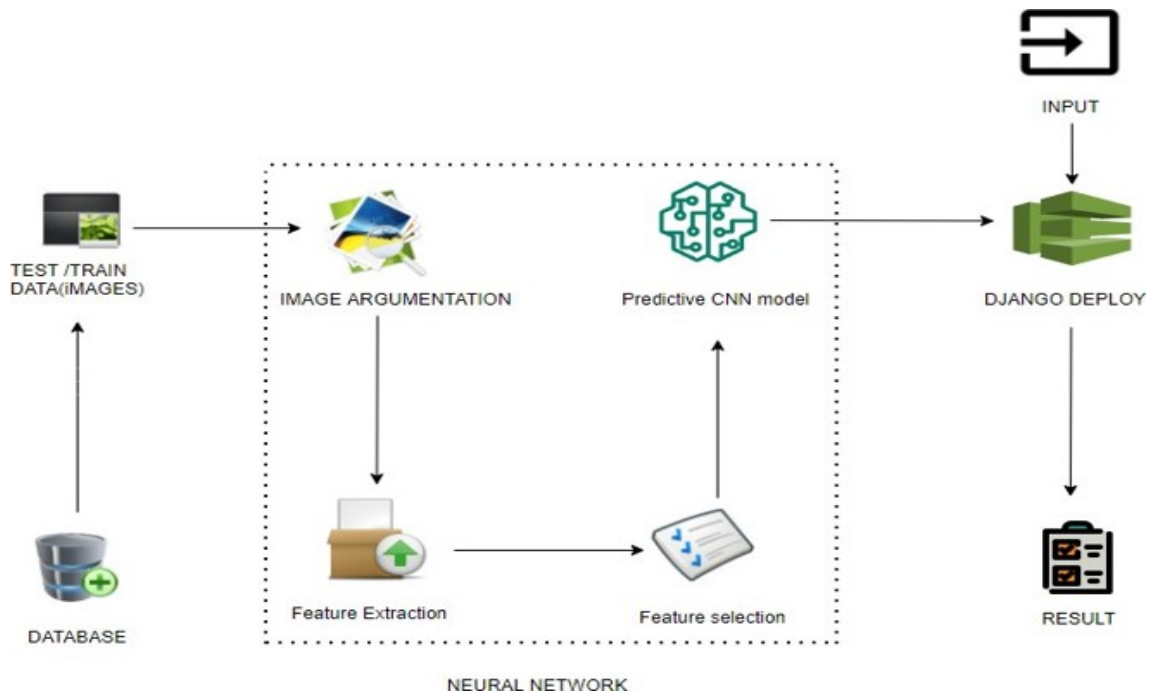


Figure 4.1: Convolutional Neural Networks Diagram

Figure 4.1 describes the general architecture of plant disease detection and prevention using Convolutional Neural Networks follows a systematic process to automate and enhance the identification and management of plant diseases. It begins with the acquisition of high-quality images of plants, including both diseased and healthy samples, from diverse sources. These images undergo preprocessing to standardize their format, enhance quality, and augment the dataset. Next, convolutional neural network models are trained on labeled image data, learning to automatically extract relevant features and classify plants into different disease categories.

4.2 Design Phase

4.2.1 Data Flow Diagram

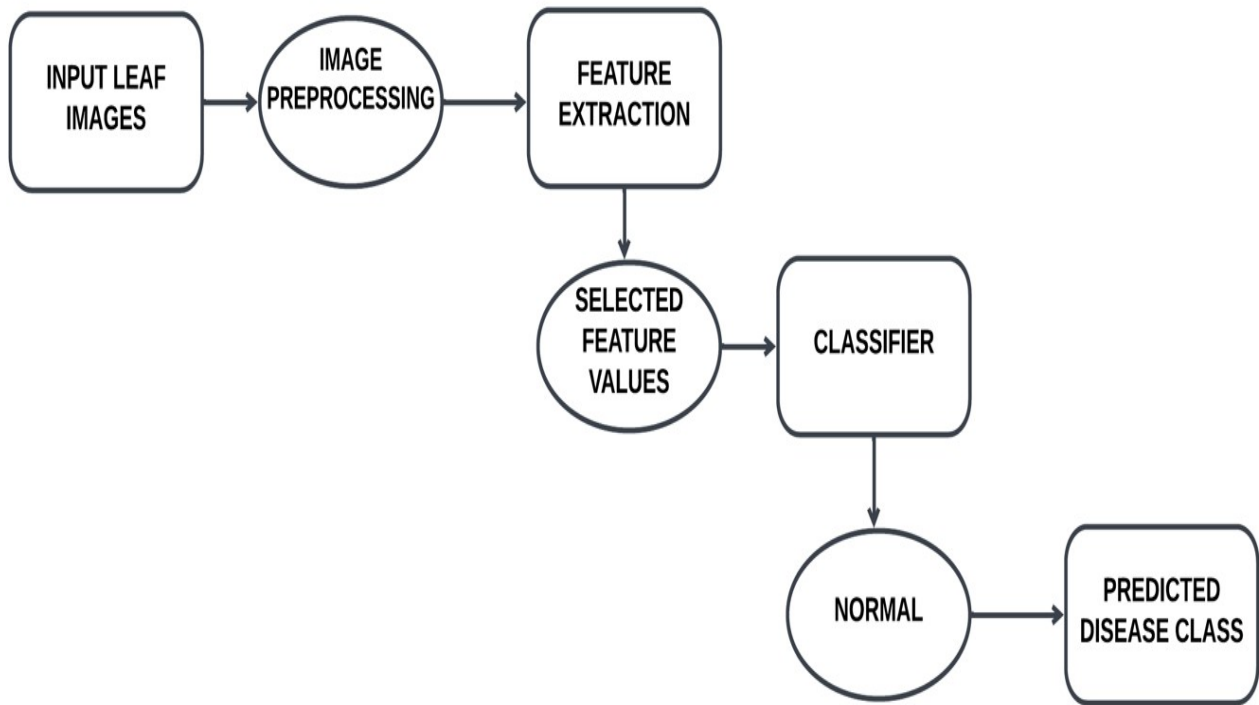


Figure 4.2: Data Flow Diagram

Figure 4.2 represents the plant disease diagnosis and prevention using convolutional neural networks is a methodical procedure that starts with the collection of high-resolution plant images from several sources, including digital cameras or field surveys. Preprocessing is applied to these photos in order to improve quality, standardize format, and expand the dataset. The convolutional neural network model is then trained using the preprocessed images, allowing it to automatically extract pertinent features and classify them into various illness categories using optimization techniques like gradient descent and backpropagation. The convolutional neural network model is assessed for performance measures using an independent dataset once it has been trained. After a successful evaluation, the trained model is used to real-world inference tasks, such as predicting diseases based on fresh plant photos. Model modification is informed by ongoing user and domain expert feedback, guaranteeing adaptability and upholding excellent illness detection accuracy.

4.2.2 Use Case Diagram

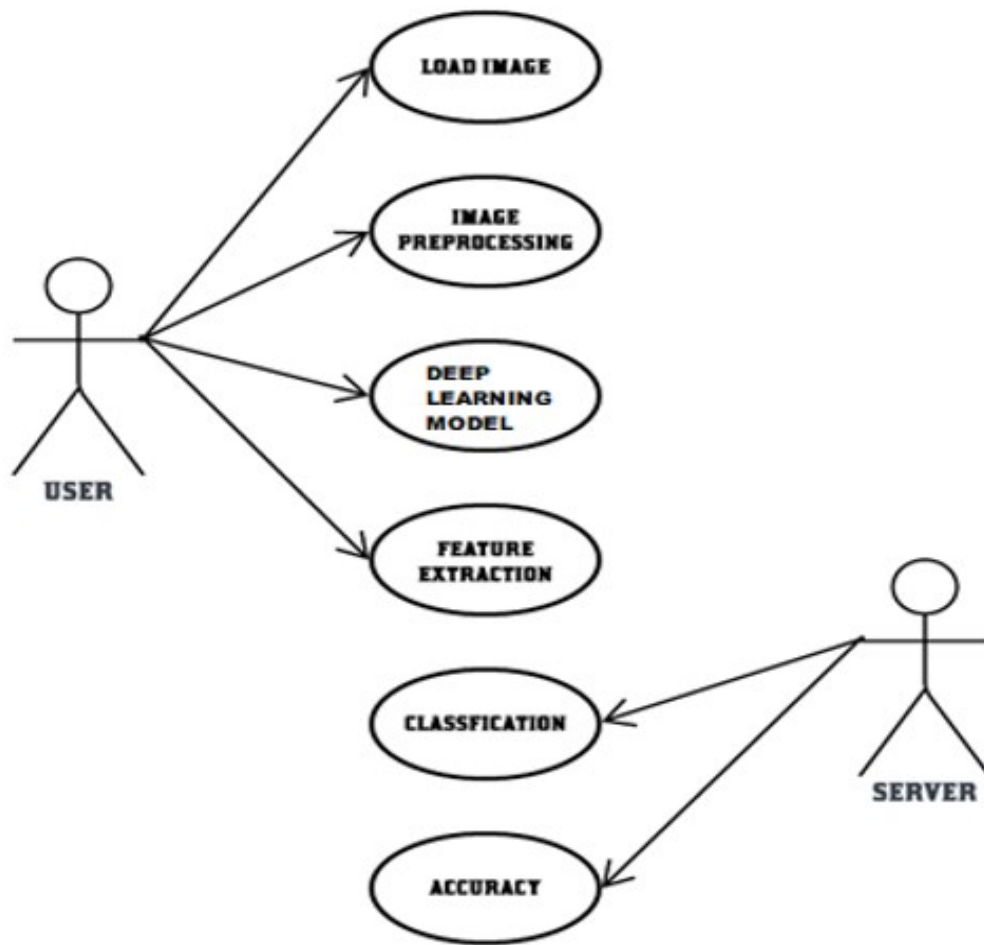


Figure 4.3: Use Case Diagram

Figure 4.3 represents the real-world use case, farmers who are confronted with possible outbreaks of crop diseases employ convolutional neural networks-based plant disease diagnosis and prevention technologies to accurately identify and address the problem. Farmers use easily accessible gadgets like cellphones or digital cameras to take pictures of the crop leaves that are impacted by this process. Preprocessing is applied to these images in order to improve their quality and standardize their format so that the convolutional neural networks model can work with them.

4.2.3 Class Diagram

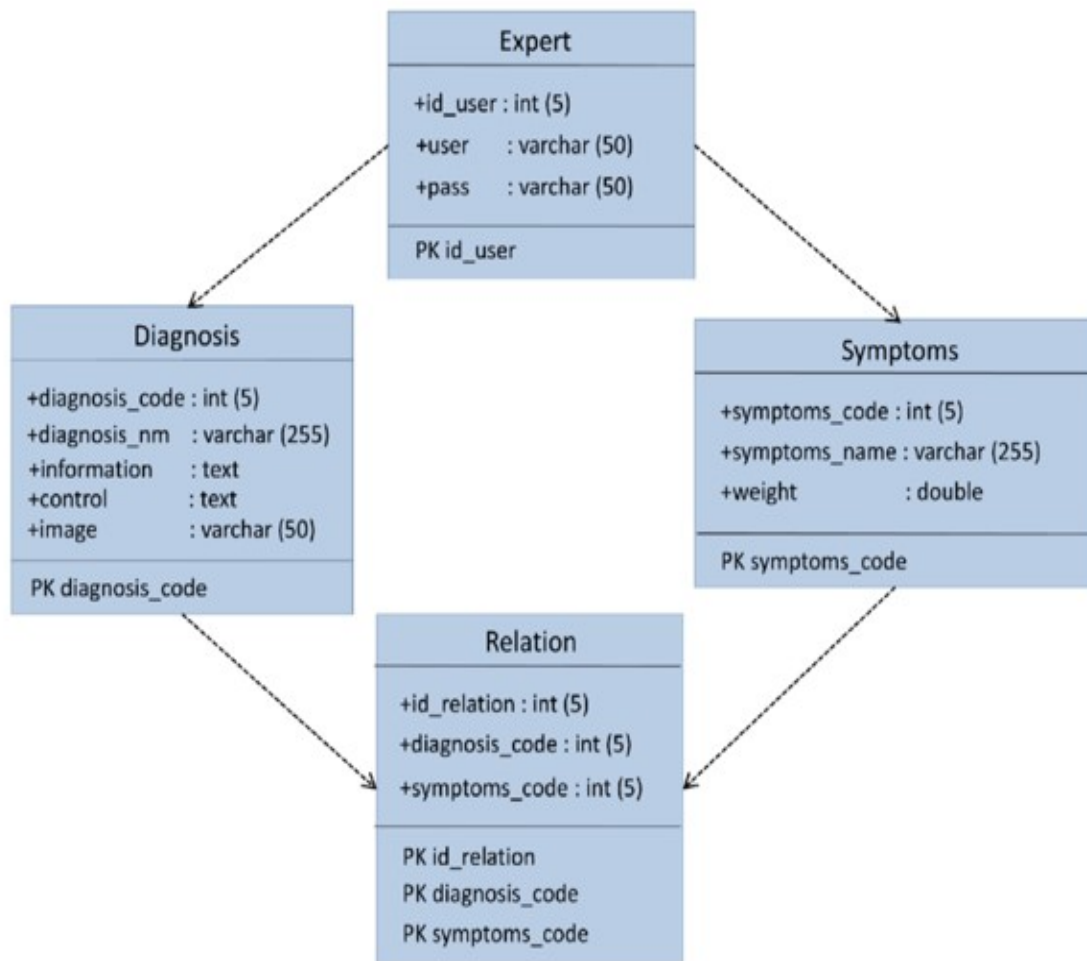


Figure 4.4: **Class Diagram**

Figure 4.4 describes the key classes and their relationships are shown in a class diagram to show the structure and interactions of a plant disease diagnosis and prevention system utilizing convolutional neural networks. The Convolutional Neural Network model, which is in charge of disease identification, is represented by the "convolutional neural network Model" class at the center of the diagram. This class contains the methods for predicting new input photos and training the model with image datasets. "Image Data Processor," another crucial class, is in charge of pre-processing raw picture data before supplying it to the convolutional neural network Model. This class prepares the images for training or inference by handling operations like augmentation, normalization, and resizing. Furthermore, during training, the loading and batching of image datasets is handled by the "Data Loader" class.

4.2.4 Sequence Diagram

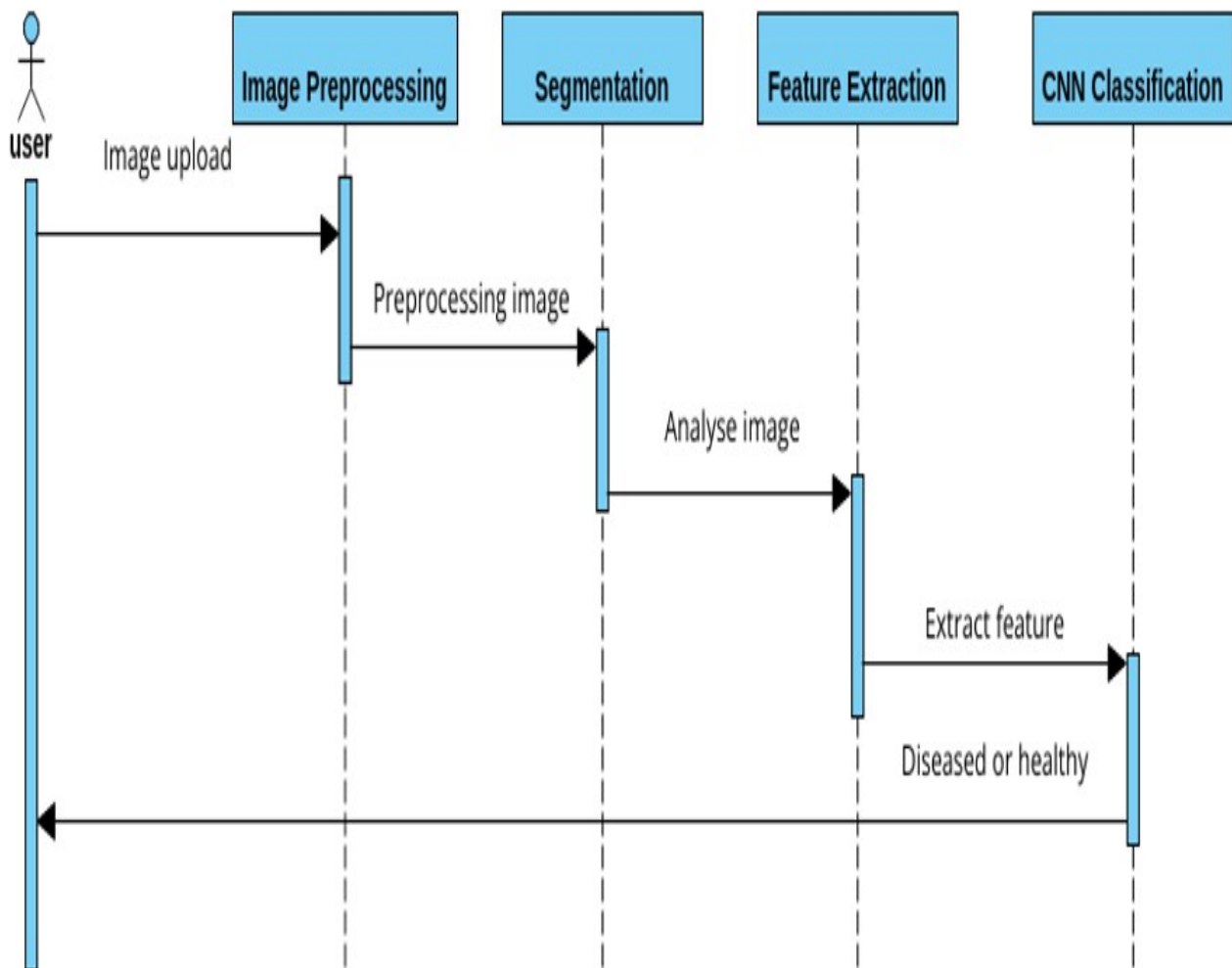


Figure 4.5: Sequence Diagram

Figure 4.5 represents the chronological flow of interactions between system components during the detection and prevention process. The process is depicted in a Sequence Diagram for plant disease detection and prevention using convolutional neural networks. The first part of the process is shown in the illustration, where a farmer uses a smartphone or digital camera to take pictures of crop leaves that seem sick. After that, the preprocessing module receives these images and uses augmentation, normalization, and scaling to get them ready for analysis. The preprocessed photos are then sent to the convolutional neural network model so that diseases can be classified. After undergoing training on labeled image datasets, the convolutional neural network model analyzes the input images and makes predictions on the presence or absence of disease. The likelihood and severity of the disease are then predicted, and the farmer receives the data through an interface.

4.2.5 Collaboration Diagram

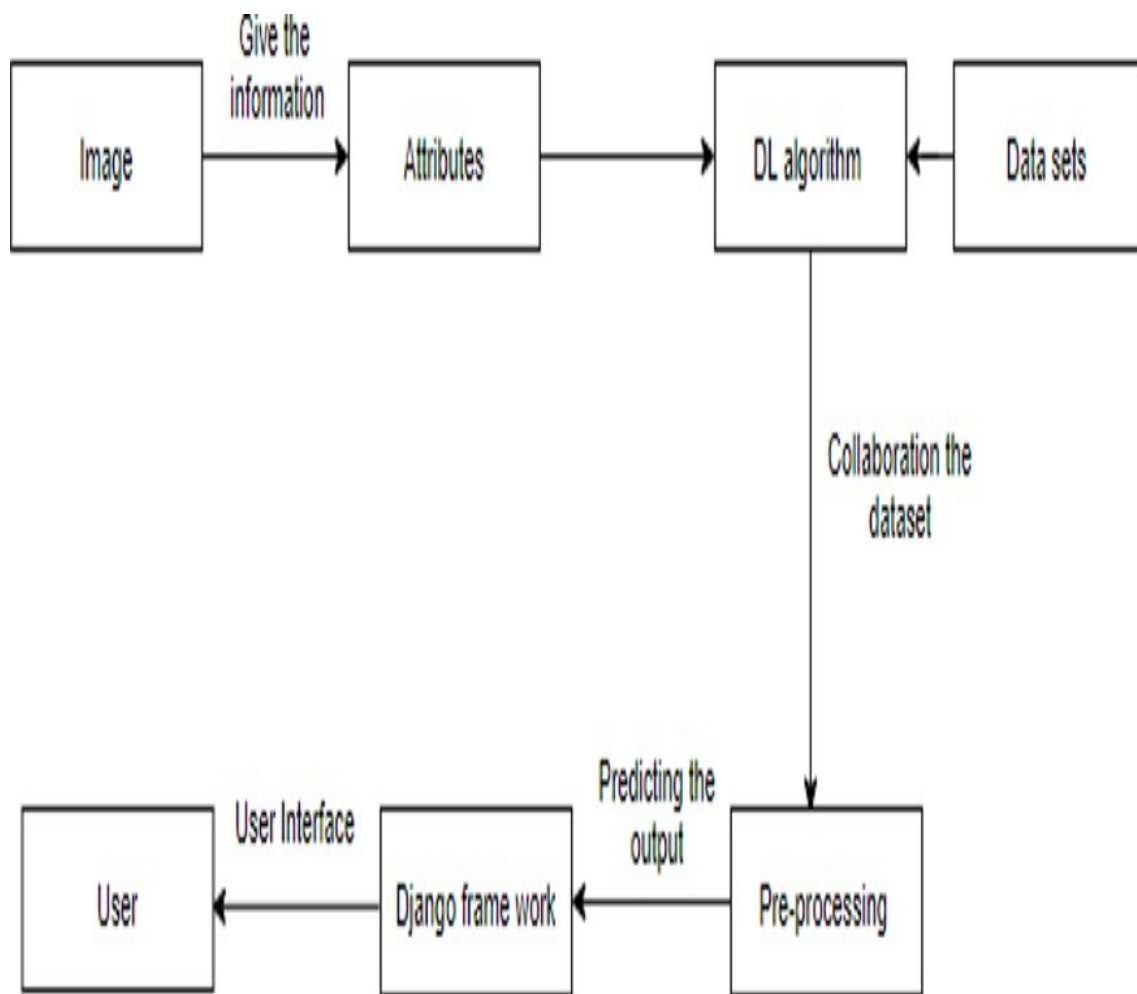


Figure 4.6: Collaboration Diagram

Figure 4.6 represents the cooperation and interactions between different system components to achieve the goal of disease detection and prevention are visualized in a collaboration diagram for plant disease detection and prevention using convolutional neural networks. Fundamentally, the diagram shows how important components like the convolutional neural network model, farmer, preprocessing module, and user interface work together. The farmer starts the procedure by taking pictures of crop leaves that appear to be diseased. The preprocessing module then uses these photos to prepare the data. Preprocessing extracts pertinent information from the input photos and classifies them into illness categories in conjunction with the convolutional neural network model. Deploying a django-based web application for plant disease detection and prevention utilizing convolutional neural networks involves several key steps.

4.2.6 Activity Diagram

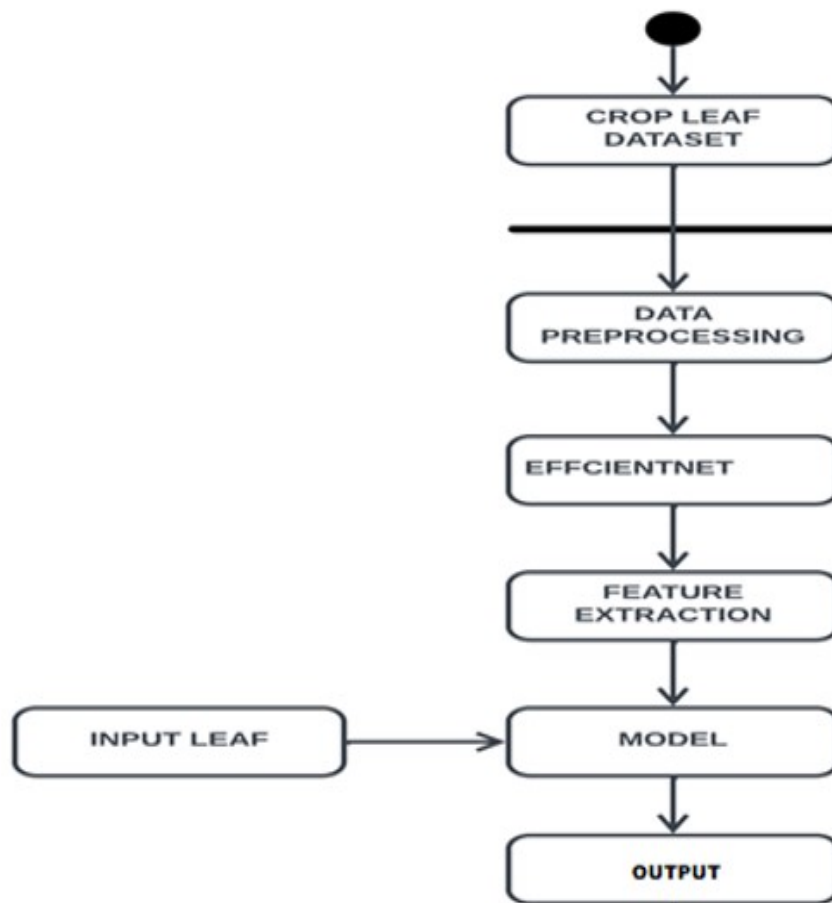


Figure 4.7: Activity Diagram

Figure 4.7 represents the flow of decisions and actions involved in the detection and prevention process. The process is shown in an activity diagram for plant disease detection and prevention using convolutional neural networks. First comes the "Image Capture" task, in which farmers use digital cameras or smartphones to take pictures of crop leaves that they think may be diseased. After that, these photos are sent to the "Preprocessing" task, where they are resized, normalized, and enhanced in order to get ready for analysis. The preprocessed images are then used as input for the "convolutional neural network Model Training" activity, which trains the convolutional neural network model to identify patterns and classifications of disease using labeled image datasets.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Step 1: Start

Step 2: Choose a Dataset

Step 3: Prepare Dataset for Training

Step 4: Create Training Data

Step 5: Shuffle the Dataset

Step 6: Assigning Labels and Features

Step 7: Normalising X and converting labels to categorical data

Step 8: Split X and Y for use in CNN

Step 9: Define, compile and train the CNN Model

Step 10: Accuracy and Score of model

Step 11: Stop

4.3.2 Pseudo Code

```
1 # Step 1: Data Acquisition
2 images = acquire_images() # Acquire images of plants from various sources
3
4 # Step 2: Data Preprocessing
5 preprocessed_images = preprocess_images(images) # Preprocess images (e.g., resize, normalize)
6
7 # Step 3: Model Training
8 trained_model = train_cnn_model(preprocessed_images) # Train CNN model on preprocessed images
9
10 # Step 4: Model Evaluation (Optional)
11 evaluation_metrics = evaluate_model(trained_model, validation_data)
12
13 # Step 5: Inference
14 new_image = capture_new_image() # Capture new image of a plant leaf
15 preprocessed_image = preprocess_image(new_image) # Preprocess new image
16 prediction = infer_disease(trained_model, preprocessed_image)
17
18 # Step 6: Prevention Measures
19 if prediction indicates disease:
20     implement_prevention_measures() # Implement preventive measures (e.g., pesticide application)
21
22 # Step 7: Continuous Improvement
23 if feedback_received:
24     update_model() # Update model based on feedback
25
26 # Step 8: Repeat Steps 5-7 as necessary
```

4.3.3 Mobile Inverted Bottleneck Convolution Layers

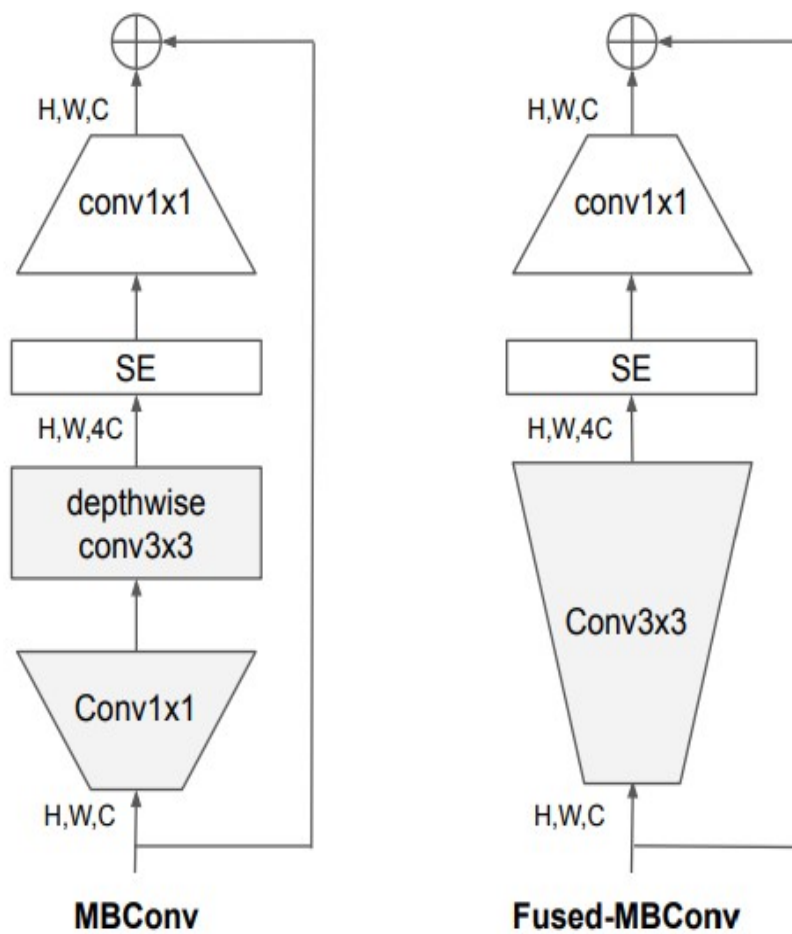


Figure 4.8: Mobile Inverted Bottleneck Convolution Layers

Figure 4.8 represents the Mobile Inverted Bottleneck Convolution layer is a key component of efficient convolutional neural network, particularly in models like MobileNet and EfficientNet. In the context of plant disease detection and prevention using convolutional neural network, integrating Mobile Inverted Bottleneck Convolution layer can significantly enhance the efficiency of the model, making it suitable for deployment on mobile devices or resource constrained environments. Pre-trained Mobile Inverted Bottleneck Convolution based models, such as those trained on large-scale image datasets like ImageNet, can be fine tuned on plant disease datasets.

4.4 Module Description

LIST OF MODULES:

- Dataset Collection
- Data Preprocessing
- Feature Extraction
- Model Selection and Metrics
- Configuration of the Classification Model
- Detection of Leaf disease

4.4.1 Dataset Collection:

An essential component of every machine learning system is data. The disease leaf was predicted using datasets from Kaggle and other government websites. 22 plants that are cultivated all throughout India make up the dataset for the plant disease system. photos of 14 plants' leaves healthy leaves excluded as well as 26 other types that depict a specific plant disease make up the dataset for classifying plant leaf diseases.

4.4.2 Data Preprocessing:

Next, each plant image's labels are mapped to a distinct Any machine learning model's pre-processing phase is crucial and should ideally shape the models' performance and outcomes. The following actions were taken in this report to ensure that the models yielded the best possible results. Following the reading and resizing of the images, we use `np.array()` to transform the images into an array format.

1. The e value using `LabelBinarizer()`
2. Finally, the plant village dataset is split into two different sets, namely, train and test set with a 75:25 ratio respectively.

4.4.3 Feature Extraction:

In machine learning, feature extraction is the process of extracting pertinent information from unprocessed data to provide a more simple and comprehensible representation that a learning algorithm can process more quickly. Feature extraction for the plant leaf disease picture dataset entails obtaining details about the color, texture, and form of the leaves. Convolutional neural networks, color histograms, edge detection, and other image processing techniques can be used to recognize and extract these properties.

4.4.4 Model Selection and Metrics:

In order to obtain optimal performance, it entails choosing the right architecture and algorithm for the model and fine tuning its hyperparameters. Convolutional neural networks, one of the deep learning architectures, are frequently utilized for plant leaf disease diagnosis utilizing picture datasets. Since convolutional neural networks can automatically extract pertinent features from images without the need for manual feature engineering, they are especially well adapted to handle image data.

4.4.5 Configuration of the Classification Model:

The following is the architecture of the convolutional neural network used in this project to detect plant diseases: the first block has a convolutional layer with 32 filters of 3 x 3 in size, and the activation function utilized is the rectified linear unit activation function. After that, we carry out batch normalization, select the Max Pooling layer with a pool size of, and add a dropout layer with a drop-out rate of 25 percent.

4.4.6 Detection of Leaf Disease:

By identifying the type of disease present and extracting pertinent information from the photos, pre trained convolutional neural network models can be utilized

to detect plant leaf diseases. This method can decrease the requirement for human feature engineering while increasing prediction accuracy. Nonetheless, a substantial volume of data is needed to adjust the pre trained model to the particular task. To identify the sort of disease present in new plant leaf photos, use the pre-trained convolutional neural network model to make predictions.

4.5 Steps to execute/run/implement the project

4.5.1 Installing Packages

- Requirements for Installing Packages
- Ensure you can run Python from the command line
- Traceback most recent call last
- Ensure you can run pip from the command line
- Ensure pip, setuptools, and wheel are up to date
- Optionally, create a virtual environment

4.5.2 Creating Virtual Environments

- Use pip for Installing
- Source Distributions vs Wheels
- Upgrading packages
- Requirements files

4.5.3 Evaluate the Model

- Installing from other Indexes
- Installing from a local src tree
- Installing from local archives
- Installing from other sources
- Installing Prereleases
- Installing Setuptools “Extras”

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

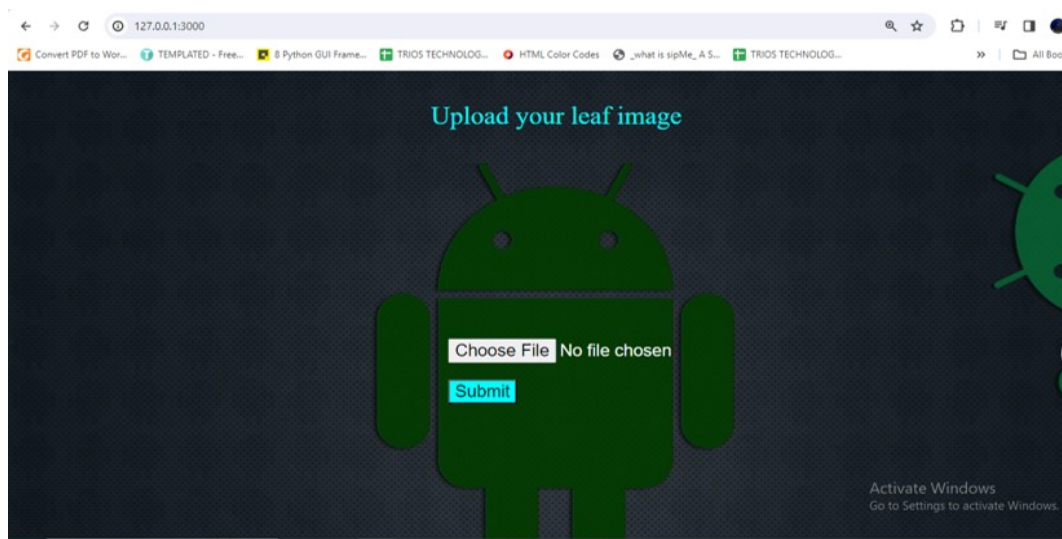


Figure 5.1: Input Design

Figure 5.1 represents the plant disease detection and prevention system employing convolutional neural networks, enabling image uploads is crucial for user interaction. Upon submission, the backend server, constructed using frameworks like Django or Flask, processes the uploaded image. This involves extracting the image file from the request payload and preprocessing it to meet the requirements of the convolutional neural network model, such as resizing and normalization. Once prepared, the image is fed into the convolutional neural network model for prediction, which identifies any diseases present in the plant. The results are then communicated back to the user interface, informing them about the detected diseases and possible prevention measures. This seamless integration of image uploading with convolutional neural network based disease detection enhances the system's usability and effectiveness in safeguarding plant health.

5.1.2 Output Design



Figure 5.2: **Output Design**

Figure 5.2 represents the uploading an image of a leaf in a plant disease detection and prevention system employing convolutional neural networks, users receive insightful output regarding the leaf's health status. The system's prediction results are promptly delivered, typically indicating whether the leaf is healthy or afflicted by a specific disease. In cases of disease detection, users are informed about the identified ailment, often accompanied by details such as the type of disease and its severity. Prediction outcomes indicating the leaf's health status and any detected diseases, users receive detailed descriptions of the identified ailments, encompassing symptoms, causal agents, and potential impacts on plant health and yield. To enhance user understanding, confidence scores or probabilities associated with each prediction may also be provided. Visual feedback mechanisms, such as color coded indicators or textual notifications, ensure intuitive comprehension of the results. This seamless integration of prediction outcomes empowers users with actionable insights, enabling them to take proactive measures to safeguard plant health effectively.

5.2 Testing

5.3 Types of Testing

5.3.1 Unit Testing

Input

```
1 import unittest
2 import numpy as np
3 from unittest.mock import MagicMock
4 from your_module import preprocess_image, create_model, predict_disease
5
6 class TestPlantDiseaseDetection(unittest.TestCase):
7     def setUp(self):
8         # Set up any necessary resources before running tests
9         self.test_image = np.random.rand(224, 224, 3) # Example random image
10
11     def tearDown(self):
12         # Clean up any resources after running tests
13         pass
14
15     def test_preprocess_image(self):
16         # Mock preprocessing function
17         mock_preprocess_fn = MagicMock(return_value=np.zeros((224, 224, 3)))
18
19         # Test preprocessing function
20         preprocessed_image = preprocess_image(self.test_image, preprocess_fn=mock_preprocess_fn)
21         mock_preprocess_fn.assert_called_once_with(self.test_image) # Verify function call
22         self.assertEqual(preprocessed_image.shape, (224, 224, 3)) # Verify output shape
23
24     def test_create_model(self):
25         # Mock model creation function
26         mock_create_model_fn = MagicMock(return_value=MagicMock(layers=[MagicMock()]))
27
28         # Test model creation function
29         model = create_model(model_creation_fn=mock_create_model_fn)
30         mock_create_model_fn.assert_called_once() # Verify function call
31         self.assertTrue(model) # Verify model creation
32         self.assertEqual(len(model.layers), 1) # Verify number of layers
33
34     def test_predict_disease(self):
35         # Mock model
36         mock_model = MagicMock()
37         mock_model.predict = MagicMock(return_value=np.array([[0.8, 0.2]])) # Example prediction
38
39         # Test predict function
40         prediction = predict_disease(mock_model, self.test_image)
```

```

41     mock_model.predict.assert_called_once_with(np.expand_dims(self.test_image, axis=0)) #
        Verify function call
42     self.assertEqual(prediction, 'healthy') # Verify prediction
43
44 if __name__ == '__main__':
45     unittest.main()

```

Test result

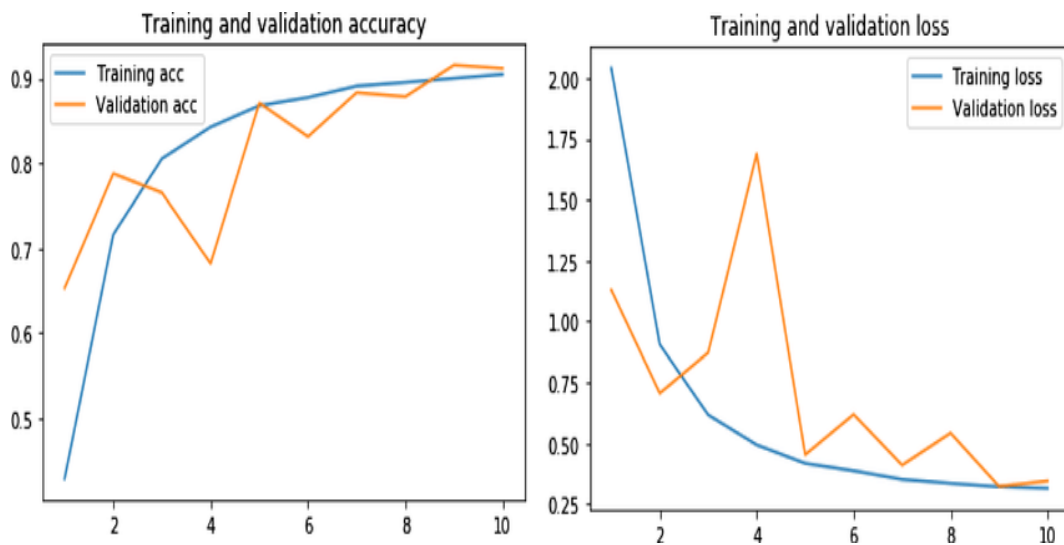


Figure 5.3: Unit Testing

Figure 5.3 represents the unit testing for plant disease detection and prevention using convolutional neural networks involves validating various components of the system to ensure its functionality, accuracy, and robustness. Test image preprocessing functions such as resizing, normalization, and color space conversion to ensure they produce the expected output for given input images. Verify that preprocessing maintains image integrity and does not introduce artifacts that could affect model performance. Test the convolutional neural networks model architecture to ensure it is constructed correctly according to the specified layers, activations, and parameters. Validate the output shape of each layer to ensure compatibility with subsequent layers. Test the training process by feeding mock input data into the model and verifying that the loss decreases over epochs. Validate the accuracy and convergence of the model by comparing predicted outputs with ground truth labels.

5.3.2 Integration Testing

Input

```
1 import unittest
2 import numpy as np
3 from unittest.mock import MagicMock
4 from your_module import PlantDiseaseDetectionSystem
5
6 class TestIntegratedPlantDiseaseDetection(unittest.TestCase):
7     def setUp(self):
8         # Set up any necessary resources before running tests
9         self.test_image = np.random.rand(224, 224, 3) # Example random image
10
11     def tearDown(self):
12         # Clean up any resources after running tests
13         pass
14
15     def test_detection_system(self):
16         # Mock dependencies
17         mock_preprocess_fn = MagicMock(return_value=np.zeros((224, 224, 3)))
18         mock_model = MagicMock()
19         mock_model.predict = MagicMock(return_value=np.array([[0.8, 0.2]])) # Example prediction
20         mock_create_model_fn = MagicMock(return_value=mock_model)
21
22         # Create detection system with mocked dependencies
23         detection_system = PlantDiseaseDetectionSystem(preprocess_fn=mock_preprocess_fn,
24                                                         create_model_fn=mock_create_model_fn)
25
26         # Test detection system
27         result = detection_system.detect_disease(self.test_image)
28         mock_preprocess_fn.assert_called_once_with(self.test_image) # Verify preprocessing call
29         mock_create_model_fn.assert_called_once() # Verify model creation call
30
31     class TestIntegratedPlantDiseaseDetection(unittest.TestCase):
32     def setUp(self):
33         # Set up any necessary resources before running tests
34         self.test_image = np.random.rand(224, 224, 3) # Example random image
35
36     def tearDown(self):
37         # Clean up any resources after running tests
38         pass
39
40     def test_detection_system(self):
41         # Mock dependencies
42         mock_preprocess_fn = MagicMock(return_value=np.zeros((224, 224, 3)))
43         mock_create_model_fn = MagicMock(return_value=MagicMock())
44         mock_predict_fn = MagicMock(return_value='healthy')
```



```

47
48     def test_create_model(self):
49         # Mock model creation function
50         mock_create_model_fn = MagicMock(return_value=MagicMock(layers=[MagicMock()]))
51
52         # Test model creation function
53         model = create_model(model_creation_fn=mock_create_model_fn)
54         mock_create_model_fn.assert_called_once() # Verify function call
55         self.assertTrue(model) # Verify model creation
56         self.assertEqual(len(model.layers), 1) # Verify number of layers
57
58     def test_predict_disease(self):
59         # Mock model
60         mock_model = MagicMock()
61         mock_model.predict = MagicMock(return_value=np.array([[0.8, 0.2]])) # Example prediction
62
63         mock_model.predict.assert_called_once_with(np.expand_dims(mock_preprocess_fn.return_value,
64             axis=0)) # Verify prediction call
65         self.assertEqual(result, 'healthy') # Verify detection result
66
67 if __name__ == '__main__':
68     unittest.main()

```

Test result

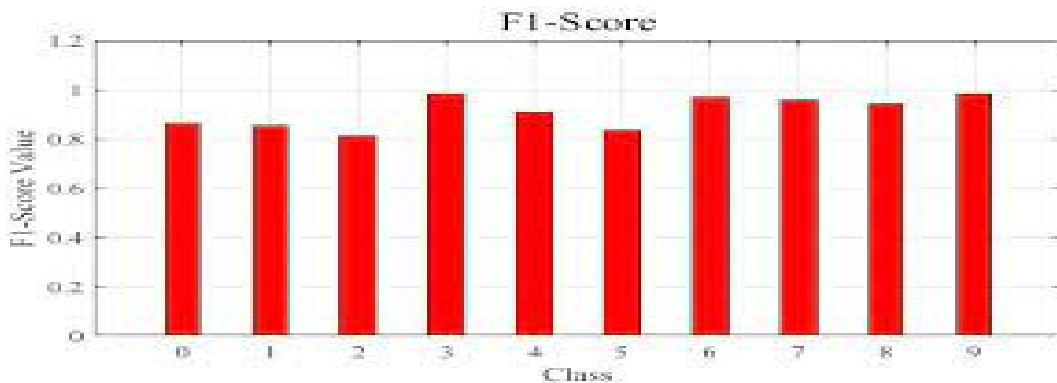


Figure 5.4: Integration Testing

Figure 5.4 represents the integrated testing for plant disease detection and prevention using convolutional neural networks involves examining the system's components collectively to ensure seamless functionality. Mock environments simulate dependencies like image preprocessing, model creation, and disease prediction, facilitating testing.

5.3.3 System Testing

Input

```
1 import unittest
2 import numpy as np
3 from your_module import CNNModel # Import your CNN model class or function
4 class TestWhiteBoxPlantDiseaseDetection(unittest.TestCase):
5     def setUp(self):
6         self.model = CNNModel()
7     def tearDown(self):
8         pass
9     def test_model_architecture(self):
10        self.assertEqual(len(self.model.layers), 10)
11        expected_layer_config = [...] # Define expected layer configurations
12        for i, layer in enumerate(self.model.layers):
13            self.assertDictEqual(layer.get_config(), expected_layer_config[i])
14    def test_model_training(self):
15        X_train = np.random.rand(100, 224, 224, 3)
16        y_train = np.random.randint(0, 3, size=(100,))
17        history = self.model.train(X_train, y_train, epochs=5, batch_size=32)
18        self.assertLess(history.history['loss'][-1], history.history['loss'][0])
19        self.assertGreater(history.history['accuracy'][-1], history.history['accuracy'][0])
20    def test_model_inference(self):
21        input_image = np.random.rand(224, 224, 3)
22        predicted_class = self.model.predict(input_image)
23        self.assertTrue(0 <= predicted_class <= 2)
24 if __name__ == '__main__':
25     unittest.main()
26 import unittest
27 import numpy as np
28 from your_module import DiseaseDetectorSystem # Import your integrated system class or function
29 class TestBlackBoxPlantDiseaseDetection(unittest.TestCase):
30     def setUp(self):
31         self.system = DiseaseDetectorSystem()
32     def tearDown(self):
33         pass
34     def test_detection_with_real_images(self):
35         input_images = [load_image("path_to_image1"), load_image("path_to_image2"), ...]
36         for input_image in input_images:
37             detection_result = self.system.detect_disease(input_image)
38             self.assertTrue("disease" in detection_result)
39             self.assertTrue(0 <= detection_result["confidence"] <= 1)
40             if "additional_info" in detection_result:
41                 self.assertIsInstance(detection_result["additional_info"], dict)
42     def test_detection_with_augmented_images(self):
43         augmented_images = [generate_augmented_image() for _ in range(10)]
44         for augmented_image in augmented_images:
45             detection_result = self.system.detect_disease(augmented_image)
46             self.assertTrue("disease" in detection_result)
```

```

47         self.assertTrue(0 <= detection_result["confidence"] <= 1)
48         if "additional_info" in detection_result:
49             self.assertIsInstance(detection_result["additional_info"], dict)
50
51         def test_detection_system(self):
52             # Initialize the plant disease detection system
53             detection_system = PlantDiseaseDetectionSystem()
54
55             # Test the detect_disease method with a sample image
56             result = detection_system.detect_disease(self.test_image)
57
58             # Assert that the result is either 'healthy' or 'diseased'
59             self.assertIn(result, ['healthy', 'diseased'])
60
61 if __name__ == '__main__':
62     unittest.main()

```

5.3.4 Test Result

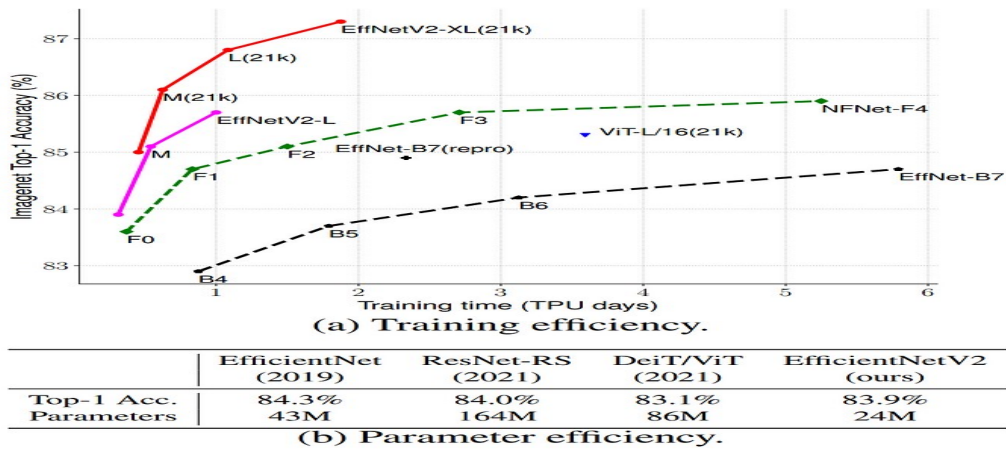


Figure 5.5: System Testing

Figure 5.5 represents the system testing in plant disease detection and prevention using yield critical insights into the system's overall performance and reliability. This testing assesses the accuracy of disease identification, evaluating the system's ability to distinguish between healthy and diseased plants. Additionally, it examines the robustness of the system under various conditions, including changes in environmental factors and different plant species.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed system of plant disease detection and prevention using convolutional neural networks offers significant advantages in terms of efficiency, accuracy, and scalability. The system can swiftly and accurately assess a vast amount of plant photos by utilizing deep learning techniques. This allows for early diagnosis and prompt intervention to minimize crop losses. The convolutional neural network architecture makes it possible to extract complex patterns and features from the input photos, which makes it easier to classify healthy and diseased plants accurately across a range of crop species and disease types. Moreover, the system's effectiveness goes beyond its detecting limits. Farmers can save time and labor costs by using convolutional neural network based detection, which eliminates the need for manual inspection. Furthermore, the system's scalability makes it possible for it to adjust to various farming contexts and operation sizes, ranging from small holder farms to massive agricultural conglomerates. Additionally, by enabling targeted interventions like precision pesticide application, which minimizes chemical usage and environmental impact, the suggested approach supports sustainable agriculture practices.

Farmers are enabled by the system to take preemptive measures to safeguard their crops and make well-informed decisions by receiving up to date information on disease prevalence and severity . All things considered, the effectiveness of the suggested system resides in its capacity to provide precise, timely, and useful information for efficient disease control, which in turn improves crop health, productivity, sustainability and the suggested method has a 93.26 percent detection accuracy. The suggested system's capacity to continuously learn from and adjust to novel disease patterns and environmental circumstances increases its efficiency. The convolutional neural network model may be trained and performed better over time through feedback loops and continuous improvement.

6.2 Comparison of Existing and Proposed System

Feature	Proposed System	Existing System (Naive Bayes)
Data Collection	Manual collection of images by experts, often limited in quantity and variety	Automated collection using drones or IoT devices
Data Preprocessing	Basic preprocessing (resizing, normalization)	Advanced preprocessing (augmentation, enhancement)
Model Architecture	Traditional machine learning models	Convolutional neural networks
Training Process	Training on limited datasets	Training on larger and diverse datasets
Performance	Limited accuracy and generalization	Improved accuracy and robustness
Deployment	Limited deployment due to resource constraints	Deployment in real world applications
Real time Monitoring	Limited real time monitoring capabilities	Real time monitoring using IoT sensors or drones
Prevention Strategies	Manual intervention based on detection results	Automated prevention strategies integrated with detection
Scalability	Limited scalability due to manual processes	Scalable solution capable of handling large datasets and real time monitoring
Continuous Improvement	Limited feedback loop for model improvement	Continuous learning and model refinement based on user feedback
Data Sharing	Limited data sharing due to privacy concerns and proprietary restrictions	Facilitated data sharing among researchers, farmers, and stakeholders through secure platforms and open data initiatives
Integration	Limited integration with existing agricultural systems and practices	Seamless integration with farm management software, IoT devices, and other agri-tech solutions, providing a holistic approach to crop health management
Environmental Impact	Potential environmental impact from excessive pesticide use and reactive disease management practices	Reduced environmental footprint through targeted interventions and sustainable disease management practices

Table 6.1: Comparison between Proposed System and Existing System

6.3 Sample Code

```
1 import numpy as np
2 import tensorflow as tf
3 from tensorflow.keras import layers, models, preprocessing
4
5 # Define CNN model architecture
6 def create_cnn_model(input_shape, num_classes):
7     model = models.Sequential([
8         layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
9         layers.MaxPooling2D((2, 2)),
10        layers.Conv2D(64, (3, 3), activation='relu'),
11        layers.MaxPooling2D((2, 2)),
12        layers.Conv2D(128, (3, 3), activation='relu'),
13        layers.MaxPooling2D((2, 2)),
14        layers.Flatten(),
15        layers.Dense(128, activation='relu'),
16        layers.Dense(num_classes, activation='softmax')
17    ])
18    return model
19
20 # Load and preprocess dataset (replace with your dataset)
21 def load_and_preprocess_dataset():
22     # Load dataset
23     (train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.plant_diseases.
        load_data()
24
25     # Preprocess images (resize and normalize)
26     train_images = preprocessing.image.array_to_img(train_images)
27     train_images = np.array([preprocessing.image.img_to_array(img) for img in train_images])
28     train_images = train_images / 255.0 # Normalize pixel values to [0, 1]
29
30     test_images = preprocessing.image.array_to_img(test_images)
31     test_images = np.array([preprocessing.image.img_to_array(img) for img in test_images])
32     test_images = test_images / 255.0 # Normalize pixel values to [0, 1]
33
34     return (train_images, train_labels), (test_images, test_labels)
35
36 # Train CNN model
37 def train_model(model, train_images, train_labels, test_images, test_labels, num_epochs=10,
    batch_size=32):
38     model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
39     model.fit(train_images, train_labels, epochs=num_epochs, batch_size=batch_size, validation_data
        =(test_images, test_labels))
40
41 # Evaluate CNN model
42 def evaluate_model(model, test_images, test_labels):
43     loss, accuracy = model.evaluate(test_images, test_labels)
44     print("Test Loss:", loss)
```

```

45     print("Test Accuracy:", accuracy)
46
47 # Main function
48 def main():
49
50     # Define model architecture
51     cnn_model = create_cnn_model(input_shape=train_images[0].shape, num_classes=len(np.unique(
52         train_labels)))
53
54     # Train model
55     train_model(cnn_model, train_images, train_labels, test_images, test_labels)
56
57     # Evaluate model
58     evaluate_model(cnn_model, test_images, test_labels)
59
60     # Save trained model
61     cnn_model.save('plant_disease_detection_model.h5')
62
63 if __name__ == "__main__":
64     main()

```

Output

```

Epoch 1/50
56/56 [=====] - 269s 5s/step - loss: 0.4251 - accuracy: 0.8467 - 0.7580
Epoch 2/50
56/56 [=====] - 266s 5s/step - loss: 0.4346 - accuracy: 0.8484 - 0.7522
Epoch 3/50
56/56 [=====] - 229s 4s/step - loss: 0.4156 - accuracy: 0.8533 - 0.7697
Epoch 4/50
56/56 [=====] - 219s 4s/step - loss: 0.4064 - accuracy: 0.8604 - 0.7697
Epoch 5/50
56/56 [=====] - 237s 4s/step - loss: 0.4086 - accuracy: 0.8566 - 0.7609
Epoch 6/50
56/56 [=====] - 245s 4s/step - loss: 0.4083 - accuracy: 0.8588 - 0.7434
Epoch 7/50
26/56 [=====>.....] - ETA: 1:43 - loss: 0.3859 - accuracy: 0.8575

```

Figure 6.1: Sample Code Output

Figure 6.1 Convolutional neural networks are particularly well suited for image classification tasks like plant disease detection because they can automatically learn relevant features from input images. The model architecture consists of convolutional layers, max pooling layers, and fully connected layers, culminating in a softmax activation layer to predict the probability distribution over disease classes.

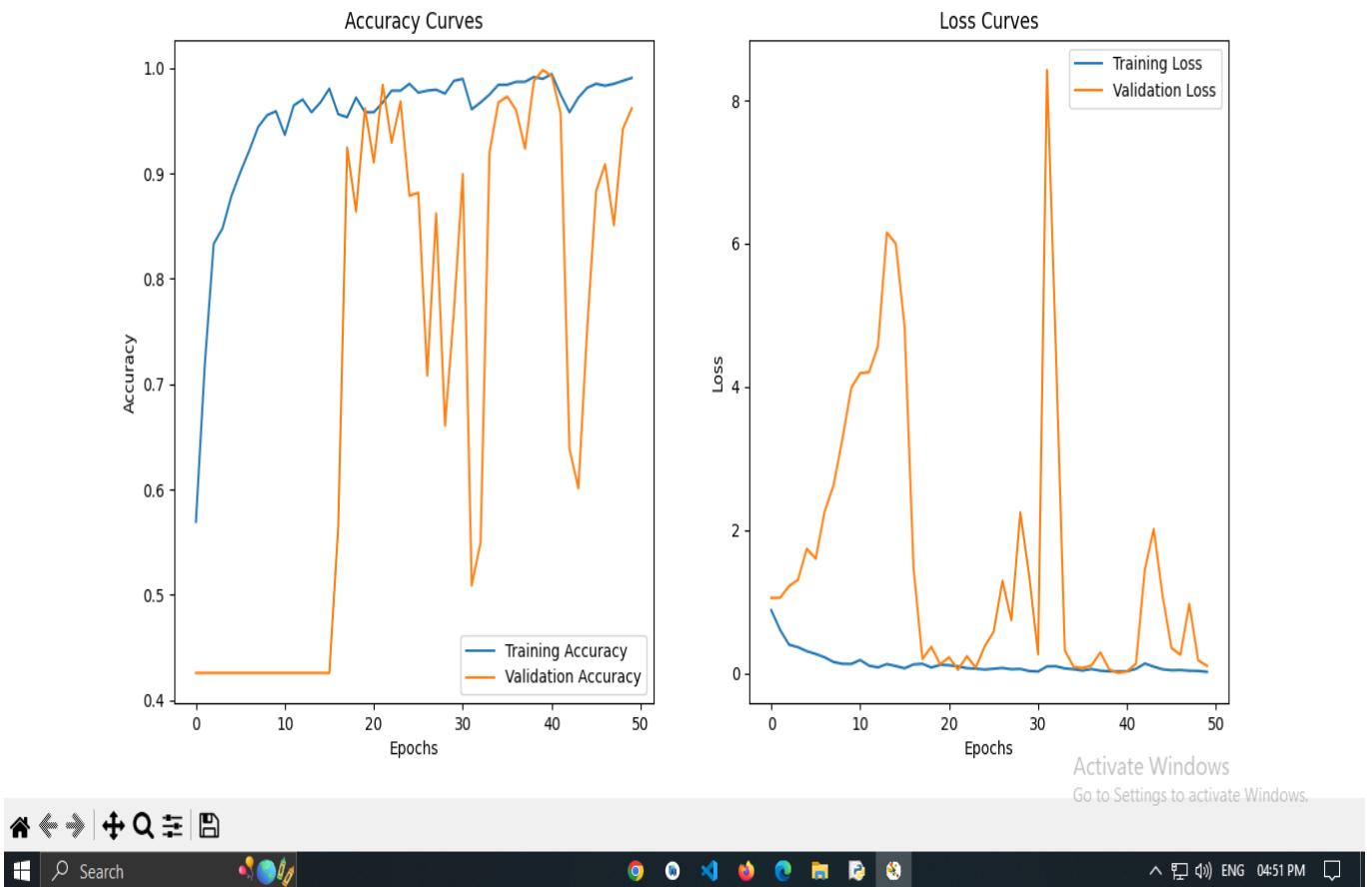


Figure 6.2: Sample Code Accuracy

Figure 6.2 represents the accuracy of plant disease detection and prevention using convolutional neural networks hinges on several critical factors. Firstly, the quality and size of the dataset play a pivotal role. A comprehensive and diverse dataset containing high resolution images of various plant diseases enables the convolutional neural network model to learn robust and discriminative features, ultimately enhancing accuracy. Secondly, the complexity of the convolutional neural network model architecture influences accuracy significantly. While more complex architectures have the potential to capture intricate patterns in the data, striking a balance is crucial to prevent overfitting and ensure generalization to unseen samples. Additionally, preprocessing techniques such as image augmentation, normalization, and noise reduction contribute to improving accuracy by enhancing the quality and consistency of the input data. Overall, achieving high accuracy in plant disease detection and prevention with convolutional neural networks requires meticulous attention to dataset quality, model architecture, and preprocessing methods.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

In conclusion, the use of Convolutional Neural Networks for plant disease diagnosis and prevention is a ground breaking development in agricultural technology that provides a thorough and efficient defense against crop illnesses. Convolutional neural network based systems have the ability to completely change how farmers monitor, diagnose, and manage plant health in agricultural contexts through the merging of deep learning techniques, image processing algorithms, and agricultural experience. Convolutional neural networks have proven to be remarkably accurate and efficient at detecting disease symptoms from photos of plant leaves when used in plant disease detection and prevention. With the use of deep learning, convolutional neural network models are able to precisely classify healthy and unhealthy plants across a wide range of crop species and disease kinds by extracting complex patterns and features from photos. Early disease identification is made possible by this capacity, which enables farmers to respond promptly and strategically to reduce crop losses and maintain output potential. Additionally, convolutional neural network based systems are adaptable and scalable, which makes them appropriate for use in a variety of agricultural contexts across the globe. Convolutional neural networks can be customized to match the unique requirements and difficulties of various farming operations, ranging from smallholder farms to large scale agricultural businesses. Furthermore, the smooth data sharing and workflow integration made possible by the integration of convolutional neural network based technologies with current agricultural systems and digital platforms improves the overall efficacy and efficiency of disease management initiatives 93.26 percent detection accuracy.

7.2 Future Enhancements

Convolutional neural networks will be used in plant disease diagnosis and prevention in the future, bringing with it new possibilities for improved scalability, accuracy, and efficiency. The incorporation of multi modal data sources, such as hyper spectral and spectral imaging, with convolutional neural network based systems is one possible avenue for development. Researchers can extract more detailed information about plant health and disease status by integrating visual pictures with other data modalities, such as infrared or near infrared spectra. This increases the overall accuracy and resilience of disease detection algorithms. Furthermore, more complicated convolutional neural network models that can handle intricate disease interactions and environmental factors will be possible because to developments in model architecture and training methodologies. Methods like transfer learning, ensemble learning, and reinforcement learning have the potential to improve convolutional neural network based systems capacity to generalize and adapt to a variety of agricultural environments and crop varieties. Furthermore, continuous surveillance of plant health parameters will be made possible by the integration of Internet of Things sensors and real time monitoring with convolutional neural network based systems. This will facilitate the early diagnosis of disease outbreaks and the development of proactive intervention methods. convolutional neural network models can provide farmers rapid insights into disease prevalence, environmental conditions, and best management methods by utilizing data from sensors, drones, and other Internet of Things devices. This will ultimately improve crop yields and agricultural sustainability. All things considered, future developments in convolutional neural network based plant disease diagnosis and prevention will provide farmers with cutting edge instruments and technology to meet new obstacles and ensure the supply of food for future generations. Future developments in plant disease diagnosis and prevention using convolutional neural networks will also concentrate on enhancing accessibility and usability for farmers in various agricultural settings. These developments will involve the integration of multimodal data sources and the advancement of model design. This includes creating intuitive web interfaces and mobile applications that make it easier to take pictures of sick plants and submit them for examination.

Chapter 8

PLAGIARISM REPORT

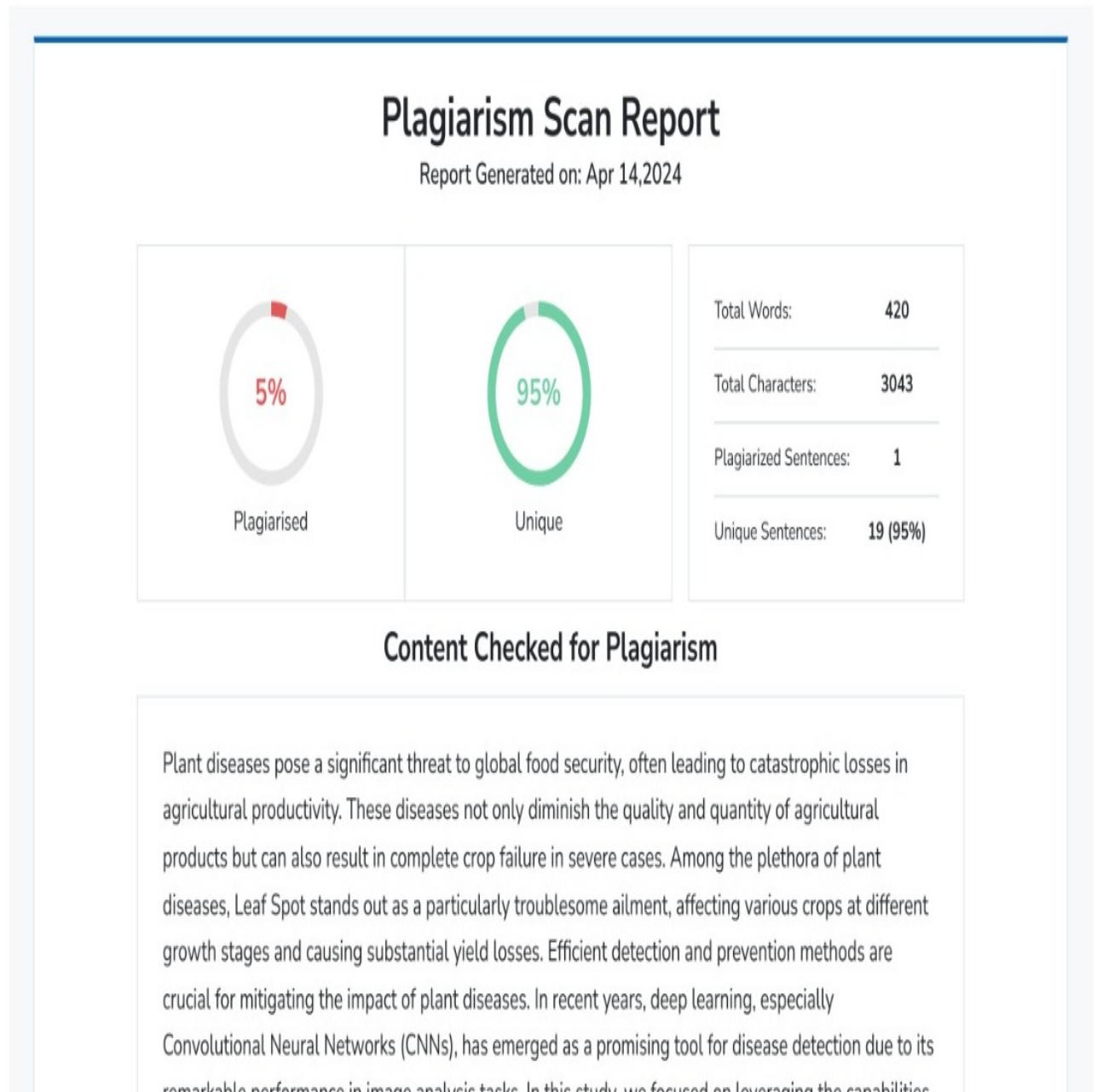


Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code


```
1 # Data Preparation
2 import numpy as np
3 import cv2
4 import os
5
6 # Load images and labels
7 def load_data(data_dir):
8     images = []
9     labels = []
10    for label in os.listdir(data_dir):
11        for img_file in os.listdir(os.path.join(data_dir, label)):
12            img_path = os.path.join(data_dir, label, img_file)
13            img = cv2.imread(img_path)
14            img = cv2.resize(img, (224, 224)) # Resize image to fixed size
15            images.append(img)
16            labels.append(label)
17    return np.array(images), np.array(labels)
18
19 # Data Preprocessing (Normalization, Augmentation)
20 from keras.preprocessing.image import ImageDataGenerator
21
22 datagen = ImageDataGenerator(
23     rescale=1./255,
24     rotation_range=40,
25     width_shift_range=0.2,
26     height_shift_range=0.2,
27     shear_range=0.2,
28     zoom_range=0.2,
29     horizontal_flip=True,
30     fill_mode='nearest')
31
32 # Model Architecture
33 from keras.models import Sequential
34 from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
35
```

```


36 model = Sequential([
37     Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
38     MaxPooling2D((2, 2)),
39     Conv2D(64, (3, 3), activation='relu'),
40     MaxPooling2D((2, 2)),
41     Conv2D(128, (3, 3), activation='relu'),
42     MaxPooling2D((2, 2)),
43     Flatten(),
44     Dense(512, activation='relu'),
45     Dense(1, activation='sigmoid')
46 ])
47
48 model.compile(optimizer='adam',
49               loss='binary_crossentropy',
50               metrics=['accuracy'])
51
52 train_images, train_labels = load_data('train_data/')
53 validation_images, validation_labels = load_data('validation_data/')
54
55 model.fit(datagen.flow(train_images, train_labels, batch_size=32),
56           epochs=10,
57           validation_data=(validation_images, validation_labels))
58
59 loss, accuracy = model.evaluate(validation_images, validation_labels)
60 print("Validation Accuracy:", accuracy)
61
62 # Model Deployment (Example using Flask for web application)
63 from flask import Flask, request, jsonify, render_template
64 import base64
65
66 app = Flask(__name__)
67
68 @app.route('/')
69 def index():
70     return render_template('index.html')
71
72 @app.route('/predict', methods=['POST'])
73 def predict():
74     img_base64 = request.form['image']
75     img_bytes = base64.b64decode(img_base64.split(',')[1])
76     img_np = np.frombuffer(img_bytes, dtype=np.uint8)
77     img = cv2.imdecode(img_np, cv2.IMREAD_COLOR)
78     img = cv2.resize(img, (224, 224))
79     img = np.expand_dims(img, axis=0)
80     prediction = model.predict(img)
81     return jsonify({'prediction': prediction[0][0]})

```


9.2 Poster Presentation



Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
Chennai-600 062



AACSB
ACCREDITED



UGC
APPROVED

Plant Disease Detection and Prevention using CNN

Department of Computer Science and Engineering
School of Computing
1156CS701-MAJOR PROJECT(INHOUSE)
Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology
WINTER SEMESTER 2023-2024

Batch: (2020-2024)

ABSTRACT

Plant diseases pose a significant threat to global food security, often leading to catastrophic losses in agricultural productivity. These diseases not only diminish the quality and quantity of agricultural products but can also result in complete crop failure in severe cases. Among the plethora of plant diseases, Leaf Spot stands out as a particularly troublesome ailment, affecting various crops at different growth stages and causing substantial yield losses. Efficient detection and prevention methods are crucial for mitigating the impact of plant diseases. In recent years, deep learning, especially Convolutional Neural Networks (CNNs), has emerged as a promising tool for disease detection due to its remarkable performance in image analysis tasks. In this study, we focused on leveraging the capabilities of deep learning, particularly the EfficientNetV2 model, for the detection of Leaf Spot in plants.

TEAM MEMBER DETAILS

<17707/GUTHA SAINATH REDDY >
<18300/AMUDALA BHANU TEJA>
<18301/AMUDALA POOJITHA>
<7416768567>
<9182960428>
<7032333656>
<vtu17707@veltech.edu.in>
<vtu18300@veltech.edu.in>
<vtu18301@veltech.edu.in>

INTRODUCTION

Plant diseases refer to various abnormal conditions, disorders, or pathogens that adversely affect the health and vitality of plants. These diseases can manifest in numerous ways, including physical damage, poor growth, reduced yield, and even plant death. They pose significant challenges to agricultural, horticultural, and forestry practices worldwide. Pathogens are microorganisms such as bacteria, fungi, viruses, and nematodes that invade plant tissues, disrupting normal cellular functions. They may spread through soil, water, air, or by vectors like insects. Abiotic factors like extreme temperatures, drought, excessive moisture, or pollution can weaken plants, making them more susceptible to diseases. Some plants are genetically predisposed to specific diseases.

METHODOLOGIES

Detection Module

Data is a crucial part of any Machine Learning System. Datasets from various government websites and kaggle were used to predict the disease leaf. The dataset for the plant disease system consists of 22 plants grown across India. Dataset for plant leaf disease classification consists of images of leaves of 14 plants while excluding healthy leaves, 26 types of images that show a particular disease in a plant. For each plant disease type, there are 1000 images.

MODEL SELECTION AND METRICS

It involves selecting the appropriate algorithm and architecture to use in the model, as well as tuning the hyperparameters to achieve the best performance. For plant leaf disease detection using image dataset, deep learning architectures such as convolutional neural networks (CNNs) are commonly used. CNNs are particularly suited to image data as they can automatically extract relevant features from images without the need for manual feature engineering.

FEATURE EXTRACTION:

Feature extraction is a process in machine learning where relevant information is extracted from raw data to create a more meaningful and simplified representation of the data that can be easily processed by a learning algorithm. For the plant leaf disease image dataset, feature extraction involves extracting features such as color, texture, and shape of the leaves.

RESULTS

- The proposed system of plant disease detection and prevention using CNNs offers significant advantages in terms of efficiency, accuracy, and scalability. By leveraging the power of deep learning techniques, the system can automatically analyze large volumes of plant images with remarkable speed and precision, enabling early detection and timely intervention to mitigate crop losses.
- The CNN architecture allows for the extraction of intricate patterns and features from the input images, facilitating accurate classification of diseased and healthy plants across various crop species and disease types. Furthermore, the efficiency of the system extends beyond its detection capabilities.
- The automated nature of CNN-based detection reduces the need for manual inspection, saving time and labor costs for farmers. Additionally, the system's scalability enables it to adapt to different farming environments and scales of operation, from smallholder farms to large agricultural enterprises.

STANDARDS AND POLICIES

Standards and policies governing plant disease detection and prevention using CNNs are critical for ensuring ethical, secure, and effective practices. These guidelines encompass various aspects, including data privacy and security, ethical data usage, model transparency, regulatory compliance, collaboration, security measures, and environmental and social impact.

Adherence to data privacy regulations such as GDPR safeguards sensitive information collected during disease detection efforts, while ethical data policies ensure informed consent and mitigate bias.

Develop educational materials to train farmers, agronomists, and other stakeholders on the use and interpretation of CNN-based disease detection tools. Promote awareness of the importance of early disease detection and integrated pest management strategies for sustainable agriculture.

Foster collaboration and knowledge-sharing within the research community to advance the field of plant disease detection using CNNs.

CONCLUSIONS

An efficient plant leaf disease detection approach is essential to detect plant diseases in real-time. In this regard, the plant leaf disease detection approach is proposed, where the cardamom plant leaf dataset was collected from a farmland with a complex background. Segmenting and detecting diseases in real-time images is a challenging task, as the images are associated with other factors such as the background of the image, environmental factors such as lighting, and angle of the capturing conditions.

ACKNOWLEDGEMENT

1. Mr. Ignatious K Pious / ASSISTANT PROFESSOR
2. 7010182279
3. ignatious@veltech.edu.in

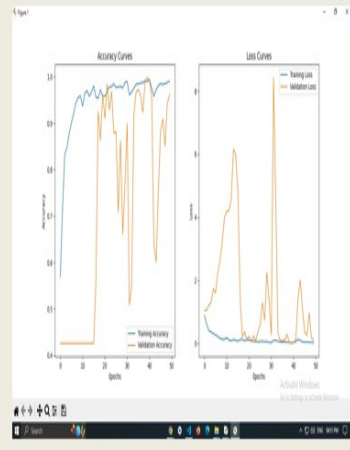


Fig 1 - Accuracy Graph

Figure 9.1: Poster Presentation

References

- [1] Nagaraju, (2020), Apple and Grape Leaf Diseases Classification using Transfer Learning via Fine-tuned Classifier, IEEE International Conference on Machine Learning and Applied Network Technologies (ICMLANT) 978-1-7281-8885-0/20/31.00 IEEE DOI: 10.1109/ICMLANT50963.2020.9355991
- [2] Tasnim Ahmed, (2020), Less is More: Lighter and Faster Deep Neural Architecture for Tomato Leaf Disease Classification, IEEE DOI: 15.1077/IM-NACAT70893.2020.9358982
- [3] Elhoucine Elfatimi, (2021), Beans Leaf Diseases Classification Using MobileNet Models, Polytechnic School of Montreal, Montreal University, Montreal, QC H3T 1J4, IEEE DOI: 10.1109/ACCESS.2021.3142817
- [4] Lili Li, (2021), Plant Disease Detection and Classification by Deep Learning A Review, Shanxi Agricultural University, Jinzhong 030800, IEEE DOI: 10.1188/ACCESS.2021.4242818
- [5] Mobeen Ahmad, (2021), Plant Diseases Detection in Imbalanced Datasets Using Efficient Convolutional Neural Networks with Stepwise Transfer Learning, IEEE DOI: 12.1898/ACCESS.2021.78478918
- [6] Malusi Sibiya, (2021), Automatic Fuzzy Logic-Based Maize Common Rust Disease Severity Predictions with Thresholding and Deep Learning, Department of Electrical and Mining Engineering, University of South Africa, 28 Pioneer Ave, Florida Park, Johannesburg, Roodepoort 1709, South Africa, IEEE DOI: 17.1867/ACCESS.2021.278939978
- [7] N. V. R. R. Goluguri, K. S. Devi, and P. Srinivasan, (2021), Rice-Net: An efficient artificial fish swarm optimization applied deep convolutional neural network model for identifying the *Oryza sativa* diseases, *Neural Comput. Appl.*, vol. 33, no. 11, pp. 5869–5884.
- [8] X. Nie, L. Wang, H. Ding, and M. Xu, (2021), Strawberry verticillium with detection network based on multi-task learning and attention, *IEEE Access*, vol. 7, pp. 170003–170011.

- [9] S. S. Chouhan, A. Kaul, U. P. Singh, and S. Jain, (2021), Bacterial foraging optimization based radial basis function neural network (BRBFNN) for identification and classification of plant leaf diseases: An automatic approach towards plant pathology, *IEEE Access*, vol. 6, pp. 8852–8863.
- [10] C. Milhorange, F. Howland, E. Sabourin, and J. F. L. Coq, (2022), Tackling the implementation gap of climate adaptation strategies: Understanding policy translation in Brazil and Colombia, *Climate Policy*, *IEEE Access*, vol. 10, pp. 836820–839208.
- [11] K. K. Chakraborty, R. Mukherjee, C. Chakraborty, and K. Bora, (2022), Automated recognition of optical image based potato leaf blight diseases using deep learning, *Physiol. Mol. Plant Pathol.*, vol. 117, Art. no. 101781.
- [12] A. Dhakal and S. Shakya, (2023), Image-based plant disease detection with deep learning, *Int. J. Comput. Trends Technol*, *IEEE Access*, vol. 10, pp. 7892734–782637
- [13] M. M. Agrawal and S. Agrawal, (2023), Rice plant diseases detection classification using deep learning models: A systematic review, *IEEE Access* vol. 7, no. 11, pp. 4376–4390.
- [14] Florent Retraint, (2023), Field Plant: A Dataset of Field Plant Images for Plant Disease Detection and Classification With Deep Learning, *IEEE DOI*: 20.18987/ACCESS.2023.768989664
- [15] Khalid M. Hosny, (2023), Multi-Class Classification of Plant Leaf Diseases Using Feature Fusion of Deep Convolutional Neural Network and Local Binary Pattern, *IEEE DOI*: 20.72987/ACCESS.2023.7826356664
- [16] Vasileios Balafas, (2023), Machine Learning and Deep Learning for Plant Disease Classification and Detection, This work is licensed under a Creative Commons Attribution NonCommercial NoDerivatives 4.0 License, *IEEE DOI*: 12.67284/ACCESS.2023.52676357388
- [17] Wasswa Shafik, (2023), A Systematic Literature Review on Plant Disease Detection: Motivations, Classification Techniques, Datasets, Challenges, and Future Trends, *IEEE DOI*: 10.1109/ACCESS.2023.3284760