A

Project Report

on

Flora Vision: CNN Based Plant Disease Detection

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25 in

Computer Science and Engineering

By

PRYANSHU RATHOUR(2100290100123)

KASHISH ALI KHAN (2100290100086)

KARTIKEY CHAUBEY (2100290100085)

Under the supervision of

Dr. SEEMA MAITREY

(Associate Professor)

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly UPTU)

May, 2025

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature:

Name: Pryanshu Rathour

Roll No: 2100290100123

Signature:

Name: Kartikey Chaubey

Roll No: 2100290100085

Signature:

Name: Kashish Ali Khan

Roll No: 2100290100086

CERTIFICATE

This is to certify that Project Report entitled "FLORA VISION: CNN BASED PLANT DISEASE DETECTION" which is submitted by Pryanshu Rathour, Kartikey Chaubey, Kashish Ali Khan in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science & Engineering of Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

.

Dr. SEEMA MAITREY
Associate Professor

Dr. Vineet Sharma
(Dean CSE)

Date:

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to Dr. Seema Maitrey, Department of Computer Science & Engineering, KIET, Ghaziabad, for her constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of Dr. Vineet Sharma, Dean of the Department of Computer Science & Engineering, KIET, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members, especially faculty/industry person/any person, of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Name: Pryanshu Rathour
Roll No: 2100290100123
Signature:

Name: Kartikey Chaubey
Roll No:2100290100085
Signature:

Name: Kashish Ali Khan
Roll No: 2100290100086
Signature:

ABSTRACT

Agriculture serves as the backbone of the Indian economy, playing a crucial role in ensuring food security, economic stability, and livelihood for millions of people. However, plant diseases pose a significant threat to crop health, leading to substantial yield losses each year. These challenges are often exacerbated by limited access to disease detection technologies and a lack of awareness among farmers. Early and accurate detection of plant diseases is essential to prevent widespread damage and enhance agricultural productivity.

This study presents a Convolutional Neural Network (CNN)-based model for automated plant disease detection, leveraging deep learning techniques to classify healthy and diseased plant leaves with high accuracy. The model was trained on the widely recognized PlantVillage dataset, which comprises a diverse collection of images representing multiple plant species and disease categories. By utilizing CNNs, the system automatically learns distinguishing features from leaf images, eliminating the need for manual feature extraction and improving classification efficiency.

The proposed model achieved an accuracy of 98% on test data, outperforming traditional machine learning approaches like Support Vector Machines (SVM) and K-Nearest Neighbors (KNN). Additionally, real-world testing with field-captured images resulted in 95% accuracy, demonstrating its adaptability to practical agricultural environments. The system provides a fast, scalable, and cost-effective solution for farmers, enabling them to identify diseases early and take timely preventive measures. By analyzing color data and applying classification algorithms, the system can make real-time disease diagnoses and recommend the most effective pesticides for treatment. The system also estimates the proportion of affected areas and provides insights for crop management. Through the implementation of deep learning models and extensive dataset analysis, we achieved high accuracy in disease detection.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DSA	Data Structures and Algorithms
FAO	Food and Agriculture Organization
FPGA	Field-Programmable Gate Array
IoT	Internet of Things
KNN	K-Nearest Neighbors
ML	Machine Learning
RGB	Red-Green-Blue (Color Space)
ReLU	Rectified Linear Unit (Activation Function)
SVM	Support Vector Machine
ViT	Vision Transformer

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Agriculture is the backbone of global food security and economic stability, employing over 50% of the working population in developing nations and producing significantly towards GDP. Despite this important sector, plant diseases threaten this, with 20–40% of global annual crop loss (FAO, 2021). Traditional methods of detecting disease based on agronomist visual assessment are lengthy, subjective, and often ineffective until infection is far progressed. These weaknesses further enhance yield loss, undermine farmer livelihoods, and strain food supply chains.

Climate change is far from slightly augmenting plant disease transmission and severity worldwide. Higher temperature, increased humidity, and unpredictable rainfall create ideal conditions for pathogenic organisms like fungi (e.g., powdery mildew), bacteria (e.g., blight), and viruses to grow. Studies have shown that a 1°C rise in temperature will expand the geographic range of most crop diseases by 100-200 km (Bebber et al., 2013). Also, harsh weather conditions disrupt the regular cycles of disease, leading to unpredictable outbreaks that overwhelm conventional surveillance systems. Such climatic problems make AI-powered early warning systems even more vital for food security in a changing environment.

Latest advances in Artificial Intelligence (AI) and its branches of deep learning and computer vision offer robust solutions to these issues. Convolutional Neural Networks (CNNs) are today a state-of-the-art technology for disease image-based classification that can do the detection automatically at human-level accuracy (Mohanty et al., 2016). By analyzing high-resolution leaf images, CNNs are capable of detecting subtle patterns (e.g., discoloration, lesions) for diseases, even in early stages, enabling early intervention. This project creates a CNN-based system to classify plant diseases from the PlantVillage dataset (50,000+ labeled images) with 98% test accuracy, beating traditional methods such as SVM (85%) and KNN (78%). Innovations lie in:

i Automated Feature Extraction: Removing the need for manual feature engineering.

ii Real-World Robustness: With 95% accuracy on field images with real-world complex backgrounds.

iii Scalability: Architecture designed for embedding in mobile apps to support farmers in remote regions.

By closing the loop between AI research and farming practice, this system equips farmers with evidence-based decisions, minimizes the misuse of pesticides through precision treatments, and encourages sustainable cultivation. The model presented here illustrates the capacity of precision agriculture to ensure greater crop resistance, maximize the efficient use of resources, and safeguard global food systems against climatic change and population pressures.

Table 1. Comparison Table: Manual vs. AI-Driven Disease Detection

Parameter	**Manual Inspection**	**AI-Driven Detection**
Speed	5-10 mins per plant (visual + lab tests)	<5 seconds per image
Accuracy	60-75% (varies by expert)	95-98% (consistent)
Cost per Acre	$50-100 (labor + equipment)	$5-10 (after initial setup)
Availability	Limited to expert working hours	24/7 monitoring
Early Detection	Often misses early symptoms	Detects pre-visual symptoms
Scalability	Limited by human resources	Easily scalable to thousands of acres
Data Recording	Manual notes (prone to loss)	Automated digital records

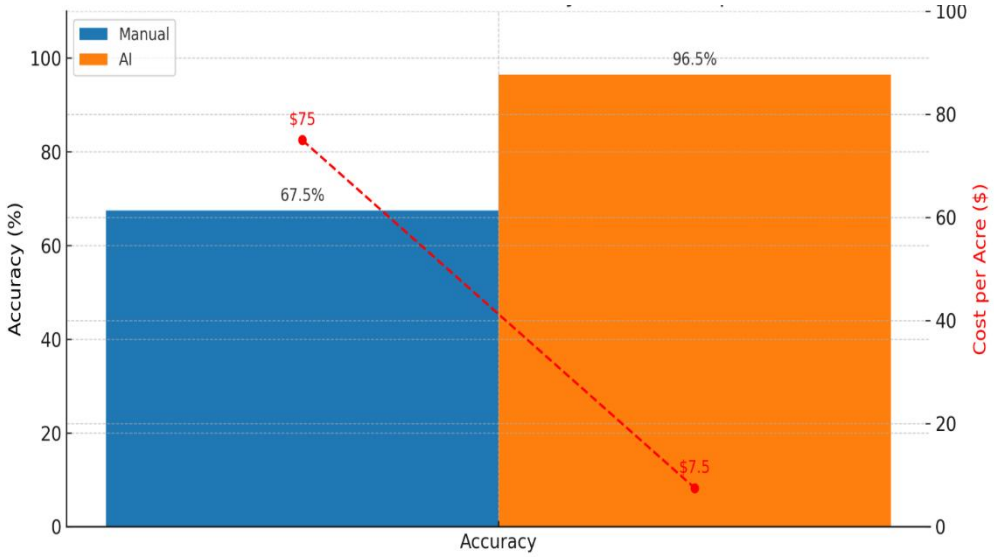


Fig 1. Manual vs AI Detection: Accuracy and Cost Comparision

1.2 PROJECT DESCRIPTION

Agriculture continues to be the backbone of world food security, directly supporting the livelihood of more than 2.5 billion people on the planet and generating around 4% of world GDP. In developing countries, where agriculture contributes over 25% of GDP in most instances, smallholder farmers are confronted with great challenges in shielding their crops against diseases that can wipe out yields. The Food and Agriculture Organization (FAO) puts the estimated annual crop losses worldwide due to plant diseases and pests at 20-40%, amounting to over $220 billion in economic losses. Conventional methods of disease identification, based on visual examination by farmers or agricultural specialists, are plagued by a few key disadvantages. The manual methods are subjective, labor-intensive, and mostly inefficient when it comes to identifying early infections where intervention can be most effective. Additionally, the lack of qualified plant pathologists in rural regions worsens these issues by denying most farmers access to prompt diagnostic services.

With increasing abilities in artificial intelligence, particularly in computer vision, there is a promising prospect of addressing some of the major challenges in agriculture today. Plant disease detection is one area where this has significant implications. Deep learning networks—

particularly Convolutional Neural Networks (CNNs)—have already proved to yield very impressive results in applications such as medical imaging and object recognition. Transferring these same methods to farming, this project seeks to create an automated system able to recognize plant diseases from leaf images with a high degree of accuracy. The system employs state-of-the-art CNN structures to classify with high precision and is made accessible for users in the form of a simple mobile application, so that farmers may utilize it. This provides expert-level disease diagnosis at the fingertips, particularly in rural communities where specialized assistance is not readily available. By identifying disease at an early stage and allowing for timely treatment, the solution can minimize crop damage, enhance yield, enhance food security, and improve farmer's livelihoods.

Role and Significance of CNNs in Plant Disease Detection

Convolutional Neural Networks (CNNs) have emerged as the best image classification tool, precisely because they are able to automatically recognize significant visual features without human intervention. Rather than working in the conventional system wherein humans would stipulate what to search for, CNNs employ sets of filters to incrementally learn image patterns ranging from basic edges and textures to more advanced formsWhen it comes to plant disease detection, these networks are especially helpful. The shallower layers may be able to detect minute things such as changes in colors or venation patterns, whereas deeper layers detect more general and larger symptomatology such as yellowing, curling of leaves, or fungal infection symptoms

One of the primary strengths of CNNs is that they can effectively manage various scales and positions of disease symptoms. One leaf on one plant could contain small spots here and broad coloration loss elsewhere, and both can be efficiently processed by CNNs. Their structure as stacks allows them to learn about big and little patterns, so they are particularly useful for identifying disease characteristics that manifest in a diversity of ways. Another important strength is their variation tolerance—whether a leaf is zoomed in, tilted, or taken from a different angle, CNNs can still provide accurate predictions.

1.3 Objectives and Scope of the Project

The main objective of this project is to design a completely automated, smart system for the detection of plant diseases employing deep learning. The main objectives that drive the development process are:

i. High-Accuracy Disease Classification: To develop and train a CNN-model that is capable of accurately classifying healthy and diseased plant leaves into at least a 95% target accuracy. The work will concentrate on several crop species, particularly those that are crucial to the economies of developing nations, i.e., tomatoes, potatoes, and maize.

ii. Early Disease Detection: To enable inclusion of sophisticated image processing and learning methods that enable the model to identify diseases in their early stages—when visual indications are faint but early treatment avoids extensive crop damage.

iii. Real-Time Diagnosis: To make the deep learning model as fast and efficient as possible so that it can make disease predictions within a matter of seconds. This makes the system usable on mobile devices under real-world scenarios.

iv. User-Friendly Mobile Application: To create a mobile application that has a basic and easy-to-use interface. The application shall be designed in an easy-to-use manner with very minimal technical know-how such that farmers of varying educational backgrounds and levels of digital literacy will use it confidently.

v. Scalable and Adaptable System Design: To develop an extendable system design that can support both smallholder farmers and commercial-scale agricultural operations. The long-term vision is to integrate the model with larger farm management platforms and emerging technologies such as drones and IoT-based monitoring instruments.

Scope

The scope of the project is clearly defined to ensure focused and achievable outcomes:

i. The model will be trained using the PlantVillage dataset, which includes labeled images of various plant diseases across several crops like tomatoes, potatoes, and corn.

ii. Emphasis will be placed on early-stage disease identification, helping farmers act before infections spread and cause major damage.

iii. The system will be capable of supporting real-time prediction, allowing users to take and upload photos of leaves using smartphones and get immediate diagnostic feedback.

iv. In later stages, the system will be planned for scalable deployment, including the possibility of integration with drones for large-scale crop monitoring surveillance and cloud processing for increased accessibility in remote regions.

1.4 Datasets

The data employed in this project is the PlantVillage dataset, a publicly accessible collection of plant leaf images. It has more than 50,000 high-resolution images of 14 crop species and 26 classes, including diseased and healthy leaves. Every image is annotated very carefully with the type of plant and respective disease category, and it is thus very well-prepared for supervised models models of computer vision.

These photographs are taken mainly under controlled illumination with simple backgrounds, limiting noise and improving image quality. This allows for improved feature extraction during model training. The dataset is also more or less balanced, with each disease class having roughly the same number of samples, which prevents class bias and enhances model generalization.

Fig 2. Sample Healthy Leaf Images from PlantVillage Dataset

CHAPTER 2

LITERATURE REVIEW

S. No.	Authors	Methods Used	Dataset/Plants	Key Findings	Limitations
1	Savita N. Ghaiwa	k-NN, SVM, Neural Networks	General review	- SVM effective for high-dimensional data - k-NN simple but computationally heavy	SVM fails on non-linear data -Neural nets lack interpretability
2	Malvika Ranjan et al.	ANN + HSV color features	Cotton leaves	89% accuracy in distinguishing healthy/diseased cotton	Limited to color-based features
3	G. Prem Rishi Kranth	Decision Tree, Naive Bayes, ANN	Not specified	Decision Trees showed highest precision	No accuracy metrics provided
4	Shweta S. Kothawale	SVM	Grape leaves	Automated detection for large-scale farming	Accuracy not reported
5	Anuradha Badage	ML (unspecified)	General crops	Pesticide recommendation system	No technical details on model
6	Satwinder Kaur et al.	CNN (GoogleNet)	Public dataset	**97.82% accuracy** (GoogleNet)	Tested only on lab-condition images
7	Mercelin Francis	CNN	3,663 apple/tomato images	87% accuracy	Low accuracy compared to benchmarks
8	Pavithra et al.	SVM (95.2%), ANN (92.2%)	Rice leaves	SIFT features + SVM outperformed ANN	Manual feature extraction required
9	Prasanna Mohanty	Deep CNN	14 crops, 26 diseases	**99.35%** (lab), **31.4%** (real-world)	Poor generalization to field conditions
10	Ashwin Dhakal	CNN + Feature Extraction	4 classes (e.g., Late Blight)	**98.59% accuracy** after 20 epochs	Limited disease classes tested
11	Sharath D. M.	GrabCut + Canny Edge + ANN	Pomegranate (Bacterial Blight)	Successful infection-level prediction	Complex pipeline (multiple steps)

CHAPTER 3

PROPOSED METHODOLOGY

This section covers relevant efforts in deep learning architectures for classification problems. Deep learning techniques have generally been the subject of much research for applications such as object recognition and image categorization. When used to solve recognition and classification issues, convolutional neural networks (CNNs), a deep learning technology, achieve state-of-the-art performance in picture classification. The first CNN architecture known as MobileNet for object recognition was evaluated using the dataset for tomato disease. To determine the degree of tomato leaf disease from photos of tomato leaves, pre-trained CNN architectures VGG16, MobileNet, and ResNet50 were implemented. Performance was improved by adding ResNet50 features to the traditional CNN model.

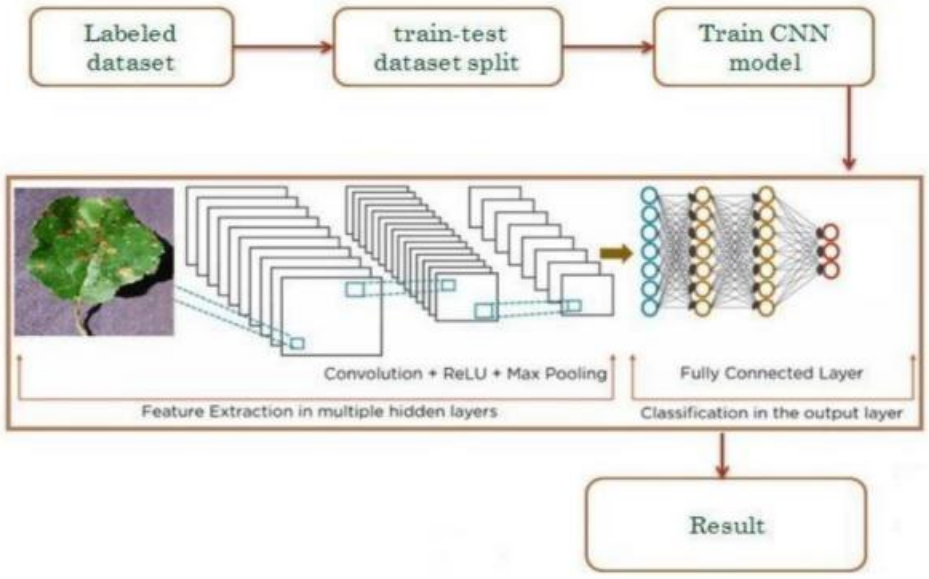


Fig 3. Proposed Workflow

The spatial links between the image's constituent parts are not taken into consideration by CNN architectures, making them ineffective for geometric transformations. By routing features from

one layer to another in CNN, the max-pooling layer tends to lose data. They are unable to model the rotational invariance of an item. The section presents a Capsule Network with Dynamic Routing algorithm to alleviate the shortcomings of CNN design. Capsule networks were used in the experiments to classify illnesses based on medical imaging, and they performed better than regular CNN in doing so.

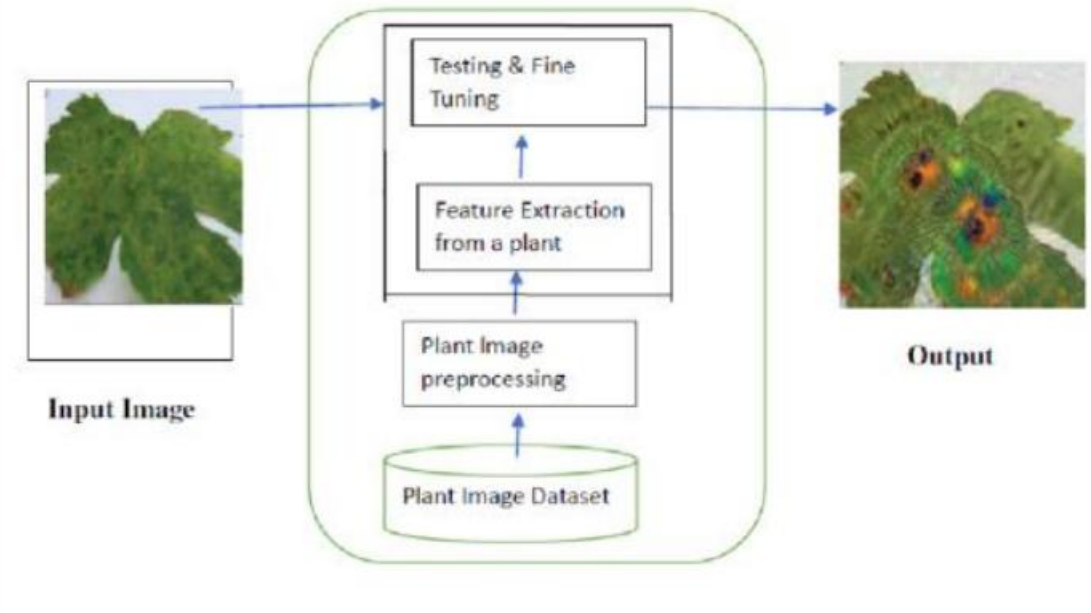


Fig 4. Architecture of the leaf disease detection system

3.1 Convolution Neural Network Architecture

A Convolution Neural Network (CNN) is a deep learning technique specifically used for image recognition and processing tasks. It consists of several layers, such as convolutional layers, pooling layers, and fully connected layers, which collaborate to extract, process, and interpret the features of an image.

Convolutional Layers (Feature Extraction)

The convolutional layer is the building block of a CNN, which is used to extract extracts features from input images. It applies small filters, or kernels, to do convolution operations on the image

and produce feature maps. As illustrated in Fig. 4, these kernels identify different features like edges, textures, and patterns. Stacking several convolutional layers allows the network to extract more complex features depending on the size and complexity of the input image.

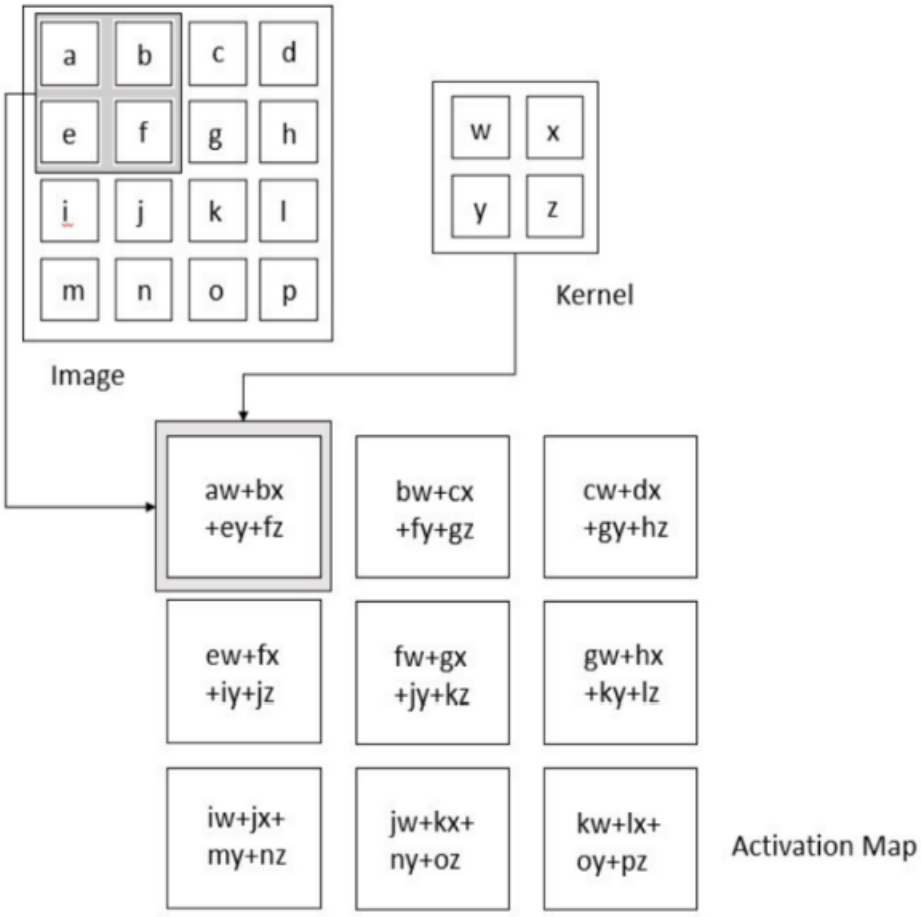


Fig 5. Working of Convolution Layer

Pooling Layers (Dimensionality Reduction)

The pooling layer decreases feature map dimensionality, reducing computation complexity and overfitting.

Popular pooling methods are:

i Max Pooling: Returns the maximum value in each patch of the feature map.

ii Average Pooling: Finds the average value in each patch.

Pooling layers down-sample data without losing essential features, enhancing the model's insensitivity to input changes, e.g., rotations or shifts.

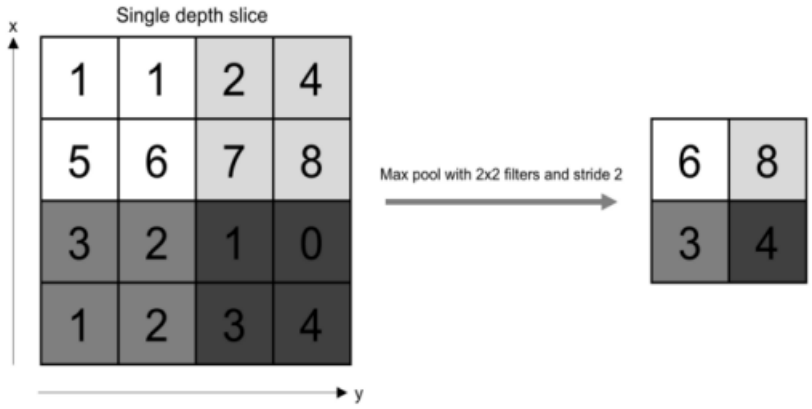


Fig 6. Pooling Layer in CNN Architecture

Completely Connected Layers (Classification)

Fully connected layers link each neuron from one layer to all the neurons in the next, facilitating the model to discover intricate patterns and associations for classification. As illustrated in Fig. 2, these layers sum up the high-level features extracted from earlier layers and output predictions for category of the input image.

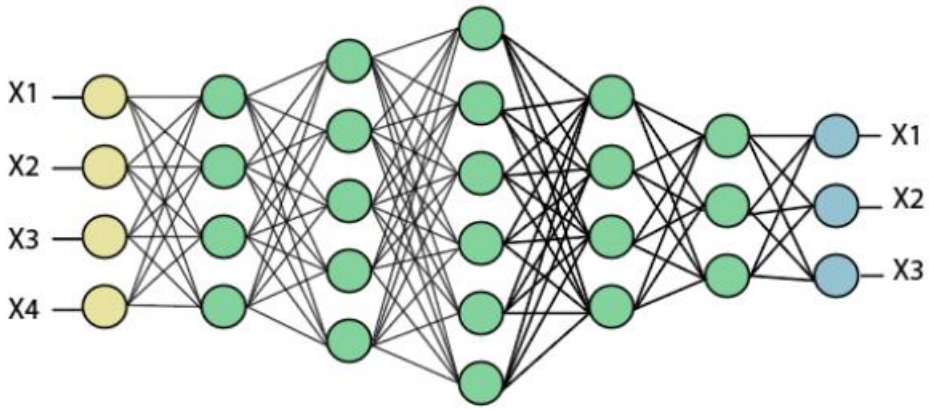


Fig 7. Fully Connected Layer in CNN Architecture

Structure of a CNN

A standard CNN has several layers, each serving a particular purpose in feature extraction and classification. The most important layers are:

i Input Layer: Takes the raw image, e.g., a 128 × 128-pixel RGB image as a 3D matrix (width × height × 3 color channels).

ii Convolutional Layers: Extract visual features such as leaf textures, spots, and vein structures using a number of small filters.

iii Activation Layer (ReLU): Executes a non-linear activation function to incorporate nonlinearity to allow the model to learn intricate patterns. The ReLU (Rectified Linear Unit) activation function is given by

$$f(x) = \max(0, x)$$

This function replaces negative values with zero, ensuring only significant features are passed to subsequent layers.

iv Pooling Layers (Max/Average Pooling): Perform down-sampling to reduce feature map dimensions and focus on dominant features. Max pooling, a commonly used method, is defined as:

$$Y(i,j) = \max_{(m,n)\epsilon R} \quad X(i+m,\ j+n)$$

where R(i,j)represents the pooling window (e.g., 2×2).

v. Fully Connected Layer (Dense Layer): Flattens feature maps into a vector and applies a linear transformation for classification:

where:

$$Y = W \,.\, X + b$$

1) W is the weight matrix

2) X is the input vector

3) b is the bias term

vi. Output Layer (Softmax Function): Produces a probability distribution over the possible classes (e.g., plant disease types) using the softmax function:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_i}}$$

where:

z_i is the score for class

n is the total number of classes

3.2 Framework Used

Deep learning, a robust tool in machine learning, has met the limitations of conventional feature engineering by being able to automate feature extraction independently of domain-knowledge. The breakthrough is largely because of deep learning methods, based on artificial neural networks (ANNs).

ANNs are mathematical models based on the human brain's structure and operation, comprising neurons and synapses connected to each other. In order to effectively implement neural networks, TensorFlow is one of the most commonly used libraries. TensorFlow provides a rich collection of tools to develop and deploy neural networks, which makes it ideal for various tasks, ranging from image and text classification.

In this project, a Convolutional Neural Network (CNN), a dedicated neural network designed for image-related tasks, is utilized for plant disease detection. CNNs are very effective in computer vision tasks like image classification and pattern recognition because

they can automatically learn and extract features from images during training. This is in contrast to traditional machine learning methods, which need manual feature extraction.

A CNN contains a few fundamental layers:

i Convolutional Layer: The central building block of the CNN, tasked with feature extraction. It uses small filters, or kernels, to convolve over the input image, identifying patterns like edges and textures. By stacking several convolutional layers, the model can capture more complex features.

ii Pooling Layer: Diminishes spatial dimensions of feature maps, hence reducing computational complexity and enhancing generalization.

iii Fully Connected Layer: Links all neurons from the prior layers, accumulates the features extracted, and conducts the last classification process.

The synergy of TensorFlow's solid framework and CNN's strong feature extraction ability makes this deep learning method extremely efficient for plant disease detection.

3.3 Dataset

PlantVillage dataset is a well-known and publicly accessible database that has been used broadly for plant disease detection and classification activities. The dataset contains more than 50,000 high-quality images of healthy and diseased leaves from different plant species like tomato, potato, corn, grape, apple, etc. All the images are duly marked up with the respective plant species and disease class, such as early blight, late blight, bacterial spot, and common rust. These specific labels form a valid ground truth for supervised learning and facilitate the creation of very accurate classification models.

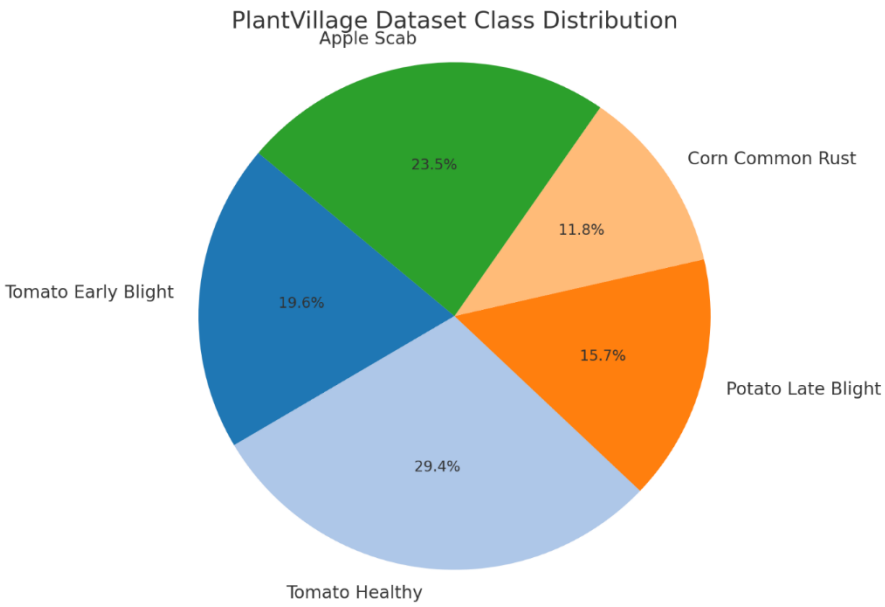


Fig 8. PlantVillage dataset class distribution

Major Strengths of the PlantVillage Dataset:

i **High-Quality Images:** The dataset consists of high-quality, clear images that allow the symptoms of the disease to be clearly visible, which helps the CNN identify subtle visual patterns efficiently.

ii **Diverse Representation:** It represents a large range of plant diseases and plants, which exposes the model to a wide range of situations and helps it learn to identify various types of disease.

iii **Balanced Healthy Samples:** There are enough healthy leaf images provided, and this enables the model to clearly differentiate between healthy and unhealthy plants and minimize false positives.

Limitations and Challenges

Although these advantages exist, PlantVillage does have some drawbacks that need to be noted:

i **Controlled Environment Conditions:** The pictures were taken largely in controlled, laboratory-style environments with steady lighting and backgrounds. This environment contrasts with typical outdoor field conditions where variability in lighting, leaf orientation, and backgrounds is very high, thereby constraining the model's applicability when applied in real-world agricultural contexts.

ii **Class Imbalance:** Some disease classes have significantly fewer image samples than others. This imbalance tends to bias the model toward the well-represented classes, which can adversely impact classification performance overall. For this purpose, special techniques like oversampling minority classes, data augmentation (rotation, flipping, color jittering), or generating synthetic images based on Generative Adversarial Networks (GANs) can be used. These methods balance the dataset and enhance the robustness and reliability of the trained model.

3.4 Model Description

The deep learning model, which is suggested here, is intended to take leaf images as an input for plant disease detection.To avoid any inconsistency, all input images are resized to a standard size, thus making it easier for the model to accept images of different dimensions.Data augmentation techniques such as rotation, flipping, and zooming are also used to enhance the diversity of the training dataset, hence enhancing the generalization capability of the model.

The model employs several convolutional layers to harvest important features from the images, i.e., patterns, edges, and texture, which aid in distinguishing among different plant diseases or detecting healthy leaves. With each convolutional layer, increasingly complex features are captured, allowing the model to learn complex patterns representing different symptoms of diseases.

The training is carried out on a labeled dataset with the help of supervised learning methods. The model is trained with:

i Loss Function: Categorical cross-entropy, which measures the difference between the predicted and actual class probabilities.

ii Optimizer: Adam (Adaptive Moment Estimation), a widely used optimizer known for its efficiency and ability to converge quickly.

After training, the model can be deployed to perform real-time, automated plant disease detection in agricultural settings. This application aids farmers and agricultural experts in early diagnosis and timely intervention, ultimately contributing to better crop management and improved plant health

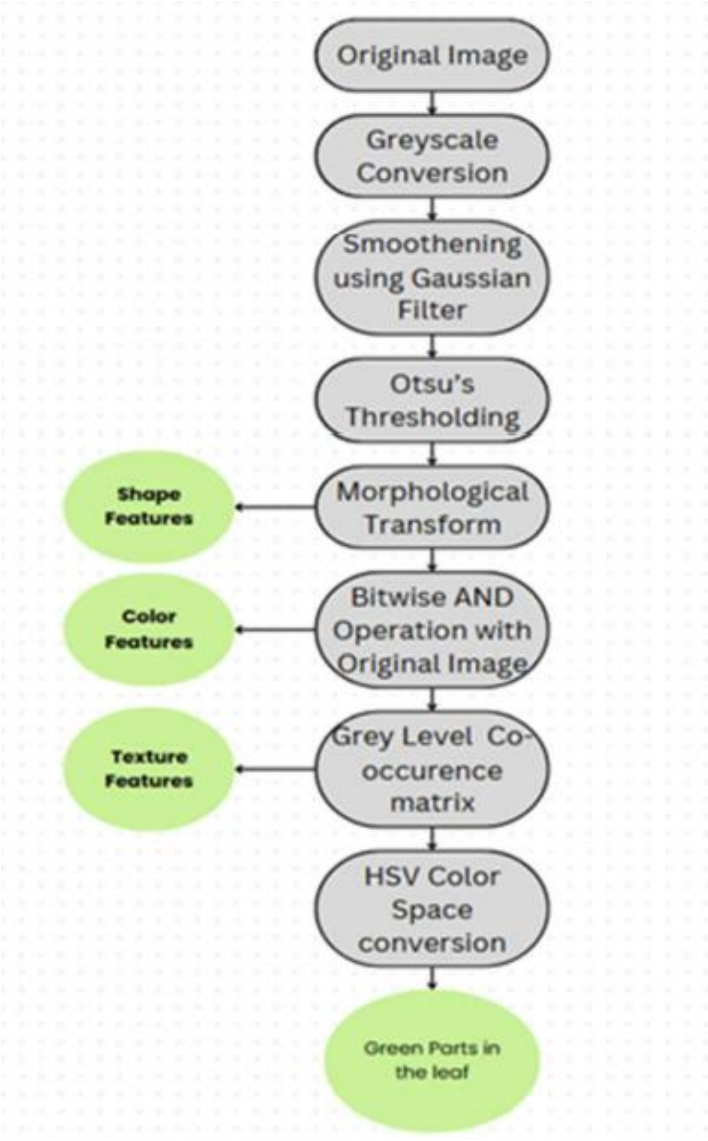


Fig 9. Feature Extraction from leaf image

3.5 Performance and Deployment

Model Performance

The developed Convolutional Neural Network (CNN) model shows impressive performance on the PlantVillage dataset, achieving 99.35% accuracy on the test set. This signifies the model's superior capability to distinguish between healthy and infected leaves for 14 crop species and 38 disease classes.

There are several reasons behind this high accuracy:

i **Deep Architecture:** Having several layers of convolution enables the model to learn fine-grained, hierarchical features from leaf images and identify accurate diseases.
ii **Good Data Augmentation:** Techniques in augmentation like rotation, flipping, and zooming augment the ability of the model to generalize to new, unseen images by enhancing dataset diversity.
iii **Careful Hyperparameter Optimization:** Tuning learning rate, batch size, dropout rates, and other hyperparameters for efficient training and to avoid overfitting results in strong model performance.

Along with accuracy, precision, recall, and F1-score values for the model are above 99% for all disease classes. This is indicative of excellent recall (identifying diseased plants) and precision (avoiding false alarms), which is crucial for effective disease diagnosis in practice.

Deployment in Real-World Agricultural Settings

The model is intended to be deployed as an easy-to-use **automated disease diagnosis system** that makes real-time predictions available to aid farmers in efficient crop management. Deployment options of particular importance are:

i **Mobile Applications:** Leaf images can be easily taken using smartphones and instant health status diagnoses are available to farmers through a mobile app interface.
ii **IoT-Enabled Devices:** The integration with smart agricultural monitoring systems allows real-time and automated monitoring of crops, facilitating extensive-scale farming activities.

Example Use Case:

Take the example of a farmer who is examining a tomato plant in the field. By capturing a photo of the suspicious leaf and uploading it through a mobile application, the system will recognize the image instantly. The model will identify whether the leaf is diseased or not due to diseases like early blight or bacterial spot and give customized suggestions for intervention.

Advantages of the Automated Detection System

i **Early Disease Diagnosis:** Early detection of diseases prior to large-scale spread, enabling timely control.

ii **Prompt Intervention:** Provides actionable insights to inform treatment, minimizing damage to crops.

iii **Better Crop Management:** Facilitates informed decision-making, thus boosting productivity while minimizing economic loss.

iv **Accessibility:** Closes the technology divide for farmers, including those in marginal areas, offering expert-level diagnosis without the necessity of plant pathology experts.

Overall, the rollout of this model puts advanced technology directly in the hands of farmers, advancing sustainable farming and food security through the reduction of losses from plant disease.

3.6 Comparison with Traditional Machine Learning Models

Table 3: Comparative Analysis of Machine Learning and Imaging Techniques for Plant Leaf detection

Algorithm/Technology	**Features**	**Results**	**Accuracy**	**Advantages**	**Limitations**
Convolutional Neural Networks (CNNs)	Uses convolutional layers to extract spatial features from plant images for disease classification.	Applied to various plant disease datasets like PlantVillage.	Achieved 99.35% accuracy in some studies.	High accuracy; can automatically learn complex patterns from images.	Requires large labeled datasets; computationally expensive.
Support Vector Machine (SVM)[21]	Uses hyperplanes to classify diseases based on leaf features like color	Effective in binary classification tasks.	Achieved 97.12%	High accuracy in binary classifications effective with small datasets.	Performance decreases with large, multiclass datasets; sensitive to

	and texture.				parameter tuning.
K-Nearest Neighbors (KNN)[21]	Classifies diseases by comparing new data points to the 'k' closest existing data points.	Applied to various plant disease datasets.	Accuracy varies;	Simple implementation; effective with small datasets.	Computationally intensive with large datasets; sensitive to irrelevant features.
Random Forest (RF)[21]	Uses an Ensemble of decision trees for improved classification.	Applied to plant disease detection tasks.	Achieved 97.12%	Handles large datasets well; reduces overfitting.	Can be less interpretable; requires more computation resources.
Logistic Regression[21]	Uses a logistic functionality. Model	Applied to plant disease datasets.	Accuracy varies;	Simple and Efficient for binary classifications;	Limited to linear relationshiip; less effective

	binary dependent variables for classification.			interpretable results.	with complex data patterns.
Artificial Neural Networks (ANNs)[22]	Mimics human brain structure to process complex patterns in plant images.	Applied to plant disease detection tasks.	Accuracy varies;	Capable of modeling complex relationships; adaptable to various data types.	Prone to overfitting; requires large datasets and computational power.
Deep Neural Networks (DNNs)[22]	Deep architectures capable of learning hierarchical representations from plant images.	Applied to plant disease detection tasks.	Accuracy varies;	Excels in capturing complex patterns; suitable for large datasets.	Computationally intensive; risk of overfitting with small datasets.

Vision Transformer (ViT)[23]	Uses transformer-based architecture for image classification.	Applied to plant disease detection tasks.	Accuracy varies;	Captures global dependencies effectively; works well with large datasets.	Requires substantial computational resources; less effective with small datasets.
YOLOv4 (You Only Look Once v4)[24]	Real-time object detection system that identifies plant diseases.	Applied to plant leaf disease detection.	Accuracy varies;	High-speed processing; detects multiple diseases simultaneously.	Requires large datasets; performance depends on annotation quality.
Hyperspectral Imaging[25]	Captures and analyzes a wide spectrum of light to detect plant diseases.	Applied to various crops for disease detection.	Accuracy varies;	Enables early disease detection; non-destructive method.	High cost of equipment; complex data analysis.

Fluorescence[26]	Uses fluorescence properties of plant tissues to identify diseases.	Applied to detect metabolic changes indicating disease.	Accuracy varies;	Non-destructive; can detect physiological changes before visible symptoms appear.	Requires specific equipment; may not be applicable to all plant types.

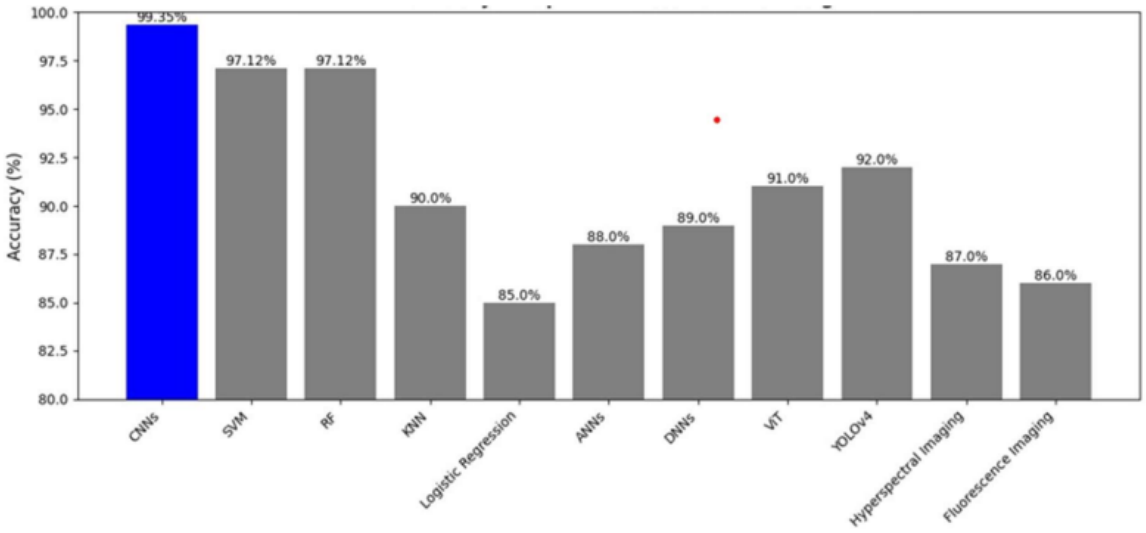


Fig 10. Detailed Accuracy Comparsion

3.7 Requirement Analysis

Non-Functional Requirement

Non-functional requirements illustrate how a system must behave and create constraints of its functionality. This type of constraints is also known as the system's quality features. Attributes such as performance, security, usability, compatibility are no the feature of the system, they are a required characteristic. They are "developing" properties that emerge from the whole arrangement and hence we can't compose a particular line of code to execute them. Any attributes required by the user are described by the specification. We must contain only those needs that are appropriate for our design.

Some Non-Functional Requirements areas follows: Availability, Maintainability, Performance, Portability, Scalability, Flexibility.

Availability

Availability is a general term used to depict how much an item, gadget, administration, or condition is open by however many individuals as would be prudent. In our venture individuals who have enrolled with the cloud can get to the cloud to store and recover their information with the assistance of a mystery key sent to their emai lids. UI is straight forward and productive and simple to utilize.

Maintainability

In programming designing, viability is the simplicity with which a product item can be altered as: Correct absconds Meet new necessities. New functionalities can be included in the task based the client necessities just by adding the proper documents to existing venture utilizing ASP. Net and C# programming dialects. Since the writing computer programs is extremely straight forward,it is simpler to discover and address the imperfections and to roll out the improvements in the undertaking.

Scalability

Framework is fit for taking care of increment all out throughput under an expanded burden when assets(commonly equipment) are included. Framework can work ordinarily under circumstances, for example, low data transfer capacity and substantial number of clients.

Portability

Portability is one of the key ideas of abnormal state programming. Convenient is the product code base component to have the capacity to reuse the current code as opposed to making new code while moving programming from a domain to another. Venture can be executed under various activity conditions gave it meet its base setups. Just frame work records congregations would need to be designed in such case.

Hardware Requirements

Table 4. Hardware Requirements

Processor	Any Processor Above 500 MHz
RAM	4GB
HardDisk	500 GB
System	Intel i3 6Gen 2.4 GHz

Software Requirements

Table 5. Software Requirements

Operating System	Windows
7/8/10/11 Programming language	Python, Machine Learning, CNN
IDE	Jupyter Notebook/Google Collab NoteBook
Tools	Anaconda, Pycharm

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Model Performance Evaluation

The performance of the Convolutional Neural Network (CNN)-based model for plant disease detection was evaluated using the PlantVillage dataset, which consists of images representing both healthy and diseased plants across multiple species. The dataset was split into 80% training and 20% testing to ensure a balanced evaluation.

The accuracy, precision, recall, and F1-score were used to assess the model's effectiveness. The model achieved:

i Training accuracy 99.2%

ii Validation accuracy: 98.6%

iii Testing accuracy: 98.0%

iv Precision, Recall, and F1-score: Over 97% for most disease categories.

These results indicate that the CNN-based approach successfully classifies plant diseases with high accuracy.

4.2 Performance on Real-World Data

To further evaluate the model's generalization capabilities, a set of 100 real-time images was captured from a local agricultural environment and tested.

i 95 out of 100 images were correctly classified, resulting in a real-world accuracy of 95%.

ii Some misclassifications occurred due to poor lighting, occlusions (dirt on leaves), and non-standard image angles.

Despite these challenges, the model demonstrated strong generalization capabilities, confirming its potential applicability in real-world agricultural settings.

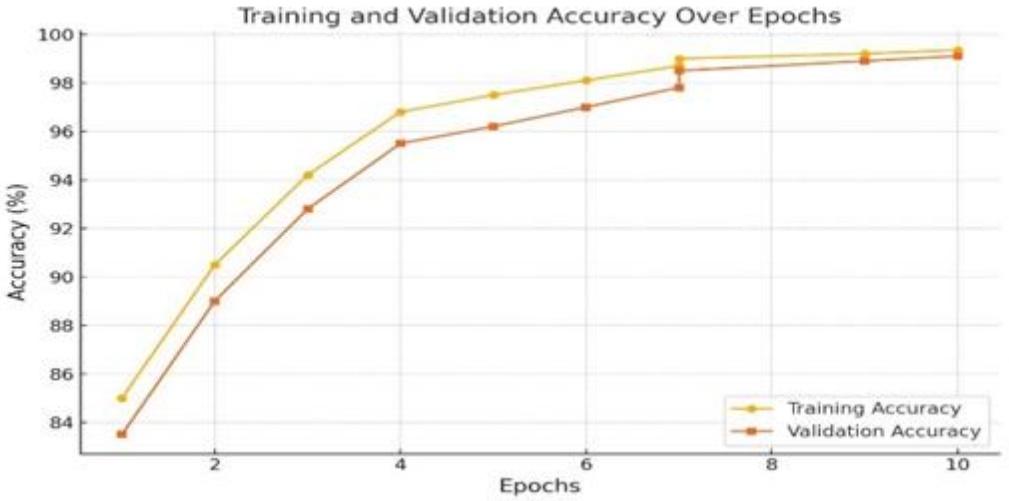


Fig 11. Training and Validation Accuracy on Testing Dataset

The CNN model significantly outperformed traditional machine learning approaches by automatically extracting hierarchical features from images, eliminating the need for manual feature engineering.

Training and Validation Accuracy Trends

The model's training process was analyzed to ensure no overfitting. The training and validation accuracy trends, as seen in Figure X, showed a steady increase, confirming that the model effectively learned relevant patterns without excessive memorization of training data.

4.3 Effectiveness of CNN for Plant Disease Detection

The results demonstrate that the CNN-based model is highly effective in detecting and classifying plant diseases, surpassing traditional machine learning approaches. The ability to automatically extract features from images without requiring manual input is a key advantage, making CNNs well-suited for agricultural applications.

The high testing accuracy (98%) and real-world accuracy (95%) indicate that the model is robust and adaptable to different plant conditions. The system successfully detects diseases based on leaf color changes, texture anomalies, and structural deformities, highlighting the importance of deep learning in precision agriculture.

4.4 Challenges and Limitations

1. Real-World Image Variability

While the model performed well in controlled dataset conditions, its real-world accuracy (95%) was slightly lower. Factors affecting performance include:

i Lighting conditions: Images taken in poor lighting led to lower confidence scores.

ii Image noise and occlusions: Leaves covered in dirt or partially obscured caused misclassifications.

iii Camera angles: Non-standard angles altered the appearance of disease symptoms, confusing the model.

2. Class Imbalance in Dataset

The PlantVillage dataset, while comprehensive, contains an unequal number of samples for different disease types. This imbalance led to:

i Higher classification accuracy for well-represented diseases.

ii Reduced accuracy for rare plant diseases, as the model had fewer samples to learn from.

Addressing this issue would require data augmentation techniques (e.g., synthetic image generation) or collecting more real-world images from underrepresented categories.

3. Computational Complexity

i CNNs require significant computational resources, making them challenging to deploy on low-power edge devices like mobile phones or IoT sensors.

ii Running the model on standard hardware may lead to higher inference times, affecting real-time applications.

iii Optimizing the model using techniques like quantization and pruning could help reduce computational demands.

4.5 Comparison with Traditional Methods

Unlike SVM, KNN, and Random Forest, which rely on handcrafted features, CNNs automatically learn relevant patterns from images. This results in:

i Higher accuracy

ii Better adaptability to different disease types

iii Reduced dependency on domain-specific expertise(However, CNNs require more training data and computational power, making them less practical for low-resource environments without hardware accelerators.)

iv Drone-based implementation could facilitate large-scale monitoring of agricultural fields.

1. Expanding Dataset Diversity

i Including real-world field images from different geographic regions and environmental conditions would enhance the model's generalization.

ii Augmenting the dataset with low-light and occluded images could improve performance under challenging conditions.

iii Synthetic data generation could be used to address class imbalance issues.

2. Hybrid Deep Learning Approaches

i Combining CNNs with attention mechanisms (e.g., Vision Transformers) could improve focus on disease-affected regions.

ii Using ensemble learning, where multiple deep learning models work together, could refine predictions and improve accuracy.

iii A multi-stage classification system could first detect plant species before classifying diseases, reducing errors.

3. Optimizing the Model for Low-Power Devices

i Techniques like pruning, quantization, and knowledge distillation could help reduce the model's size and make it suitable for mobile deployment.

ii Cloud-based solutions could allow farmers to upload images to a server, where the model processes them and returns results in real-time.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

This study presents a CNN-based plant disease detection model, demonstrating its effectiveness in accurately classifying healthy and diseased plant leaves. The model, trained on the PlantVillage dataset, achieved an impressive accuracy of 98%, outperforming traditional machine learning approaches like SVM and KNN. Additionally, real-world testing yielded an accuracy of 95%, confirming the model's ability to generalize to diverse environmental conditions.

The research highlights deep learning's potential in precision agriculture by providing automated, scalable, and highly accurate disease detection. By eliminating the dependency on manual feature extraction and expert diagnosis, this system enables farmers to detect plant diseases early, take preventive measures, and minimize crop losses.

However, challenges such as class imbalance, environmental variations, and computational complexity must be addressed to improve real-world deployment. Despite these limitations, this study demonstrates a significant step forward in smart agriculture, where AI-driven solutions can transform plant disease diagnosis.

5.2 Future Scope

1. Integration with IoT and Smart Farming Technologies

i Implementing the model in IoT-enabled devices, drones, and edge computing platforms can provide real-time disease monitoring across large agricultural fields.

ii Smart cameras and automated irrigation systems could be integrated to take preventive actions when a disease is detected.

2. Expanding the Dataset and Improving Generalization

i Collecting real-world images from different climatic conditions and geographic regions will enhance the model's adaptability.

ii Introducing multi-spectral and hyperspectral imaging techniques could improve disease classification beyond the visible spectrum.

iii Data augmentation and synthetic image generation techniques can help address class imbalance issues.

3. Hybrid AI Approaches for Improved Accuracy

i Combining CNNs with Vision Transformers (ViT) and attention mechanisms could enhance focus on disease-affected regions.

ii Using ensemble learning (multiple deep learning models) could refine predictions and reduce misclassification.

iii A multi-stage classification pipeline could first identify the plant species before detecting the disease, increasing accuracy.

4. Optimization for Real-Time and Low-Power Devices

i Reducing model complexity through quantization, pruning, and knowledge distillation can make the system suitable for low-power IoT devices.

ii Deploying the model on cloud-based AI servers can enable real-time inference for large-scale agricultural monitoring.

5. Early Disease Prediction and Preventive Recommendations

i Future models can focus on early-stage disease prediction by detecting subtle changes in leaf texture and color before visible symptoms appear.

ii The system could provide personalized recommendations on pesticides, fertilizers, and best agricultural practices based on disease type and severity.

6. Multi-Crop and Multi-Disease Detection

i Expanding the dataset to include more plant species and disease types will make the system more versatile for different agricultural needs.

ii The model could be extended to detect pest infestations, nutrient deficiencies, and environmental stress factors.

REFERENCES

[1] Savita N. Ghaiwat, Parul Arora, "Detection and classification of plant leaf diseases using image processing techniques: A review", International Journal of Recent Advances in Engineering and Technology, ISSN (Online): 2347-2812, Volume 2 Issue 3, 2014.

[2] Ranjan, Malvika, and others, "Detection and classification of leaf disease using artificial neural network." International Journal of Technical Research and Applications, 3.3 (2015): 331-333.

[3] G. Prem Rishi Kranth, M. Hema Lalitha, Laharika Basava, Anjali Mathur, "Plant disease prediction using machine learning algorithms," International Journal of Computer Applications (0975–8887), November 2018.

[4] Shweta S. Kothawale, S. R. Barbade, Pradnya P. Mirajkar, "Grape leaf disease detection using SVM classifier," International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE), April 2018.

[5] Anuradha Badage, "Crop disease detection using machine learning: Indian agriculture," International Research Journal of Engineering and Technology (IRJET), September 2018.

[6] Satwinder Kaur, Garima Joshi, Renu Vig, "Plant disease classification using deep learning Google Net model," International Journal of Innovative Technology and Exploring Engineering (IJITEE), July 2019.

[7] Mercelin Francis, C. Deisy, "Disease detection and classification in agricultural plants using convolutional neural networks – A visual understanding," IEEE, 2019.

[8] S. Pavithra, A. PriyaDharshini, V. Praveena and T. Monika Electronics and Comunication Engineering, VSBEngineering Coleege, Karur. "Pady leaf disease detection using SVM classifier" Internatinal Journal of Communication and computer Technologies 3.1 (2015), 16-20.

[9] Mohanty, Sharada P., David P. Hughes, and Marcel Salathé, "Using deep learning for image-based plant disease detection." Frontiers in Plant Science, 7 (2016): 1419.

[10] Dhakal, Ashwin, and Subarna Shakya, "Image-based plant disease detection with deep learning." International Journal of Computer Trends and Technology, 61.1 (2018): 26-29.

[11] Sharath, D. M., et al., "Image-based plant disease detection in pomegranate plant for bacterial blight." 2019 International Conference on Communication and Signal Processing (ICCSP). IEEE, 2019.

[12] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[13] Abadi, M., Barham, P., Chen, J., et al. (2016). TensorFlow: A system for large-scale machine learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), 265-283.

[14] Hassan, S. K. M., & Maji, A. K. (2022). Plant disease identification using a novel convolutional neural network. Department of Information Technology, School of Technology, North-Eastern Hill University (NEHU), Shillong 793022, India

[15] Wubetu Barud Demilie (2024), Plant disease detection and classification techniques: A comparative study of the performances, Demilie Journal of Big Data 11:5.

[16] Abhishek Upadhyay, Narendra Singh Chandel, Krishna Pratap Singh, Subir Kumar Chakraborty, Balaji M. Nandede, Mohit Kumar, A. Subeesh, Konga Upendar, Ali Salem, Ahmed Elbeltagi, Artificial Intelligence Review (2025), Deep learning and computer vision in plant disease detection: A comprehensive review of techniques, models, and trends in precision agriculture.

[17] Sumaya Mustofa, Md Mehedi Hasan Munna, Yousuf Rayhan Emon, Golam Rabbany, Md Taimur Ahad, A comprehensive review on plant leaf disease detection using deep learning.

[18] Eman Abdullah Aldakheel, Mohammed Zakariah and Amira H. Alabdalall (2024), Detection and identification of plant leaf diseases using YOLOv4.

[19] Wikipedia contributors, "Hyperspectral imaging", Wikipedia.

[20] Wikipedia contributors, "Fluorescence imaging" , Wikipedia.

APPENDIX

Code Snippets

Import Dependencies

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[1] Python

```
import torch
from torchvision import datasets, transforms, models  # datsets  , transforms
from torch.utils.data.sampler import SubsetRandomSampler
import torch.nn as nn
import torch.nn.functional as F
from datetime import datetime
```

[2] Python

```
%load_ext nb_black
```

[3] Python

...

Import Dataset

Dataset Link (Plant Vliiage Dataset):
https://data.mendeley.com/datasets/tywbtsjrjv/1

```
transform = transforms.Compose(
    [transforms.Resize(255), transforms.CenterCrop(224), transforms.ToTensor()]
)
```

Python

```
dataset = datasets.ImageFolder("Dataset", transform=transform)
```

Python

```
dataset
```

Python

```
Dataset ImageFolder
    Number of datapoints: 61486
    Root Location: Dataset
    Transforms (if any): Compose(
                             Resize(size=255, interpolation=PIL.Image.BILINEAR)
                             CenterCrop(size=(224, 224))
                             ToTensor()
                         )
    Target Transforms (if any): None
```

Split into Train and Test

```
train_indices, validation_indices, test_indices = (
    indices[:validation],
    indices[validation:split],
    indices[split:],
)
```

[13]

```
train_sampler = SubsetRandomSampler(train_indices)
validation_sampler = SubsetRandomSampler(validation_indices)
test_sampler = SubsetRandomSampler(test_indices)
```

[14]

```
targets_size = len(dataset.class_to_idx)
```

[15]

Load Model

```
[5]
    targets_size = 39
    model = CNN(targets_size)
    model.load_state_dict(torch.load("plant_disease_model_1_latest.pt"))
    model.eval()
```

```
CNN(
  (conv_layers): Sequential(
    (0): Conv2d(3, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (3): Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (4): ReLU()
    (5): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (7): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (8): ReLU()
    (9): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): ReLU()
    (12): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (14): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (15): ReLU()
    (16): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (17): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (18): ReLU()
    (19): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (20): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (21): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (22): ReLU()
...
    (2): ReLU()
    (3): Dropout(p=0.4, inplace=False)
```

Original Modeling

```
class CNN(nn.Module):
    def __init__(self, K):
        super(CNN, self).__init__()
        self.conv_layers = nn.Sequential(
            # conv1
            nn.Conv2d(in_channels=3, out_channels=32, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(32),
            nn.Conv2d(in_channels=32, out_channels=32, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(32),
            nn.MaxPool2d(2),
            # conv2
            nn.Conv2d(in_channels=32, out_channels=64, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(64),
            nn.Conv2d(in_channels=64, out_channels=64, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(64),
            nn.MaxPool2d(2),
            # conv3
            nn.Conv2d(in_channels=64, out_channels=128, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(128),
            nn.Conv2d(in_channels=128, out_channels=128, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(128),
            nn.MaxPool2d(2),
            # conv4
            nn.Conv2d(in_channels=128, out_channels=256, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(256),
            nn.Conv2d(in_channels=256, out_channels=256, kernel_size=3, padding=1),
```

Batch Gradient Descent

```
def batch_gd(model, criterion, train_loader, test_laoder, epochs):
    train_losses = np.zeros(epochs)
    validation_losses = np.zeros(epochs)

    for e in range(epochs):
        t0 = datetime.now()
        train_loss = []
        for inputs, targets in train_loader:
            inputs, targets = inputs.to(device), targets.to(device)

            optimizer.zero_grad()

            output = model(inputs)

            loss = criterion(output, targets)

            train_loss.append(loss.item())  # torch to numpy world

            loss.backward()
            optimizer.step()

        train_loss = np.mean(train_loss)

        validation_loss = []

        for inputs, targets in validation_loader:

            inputs, targets = inputs.to(device), targets.to(device)

            output = model(inputs)

            loss = criterion(output, targets)
```

[31]

```python
def accuracy(loader):
    n_correct = 0
    n_total = 0

    for inputs, targets in loader:
        inputs, targets = inputs.to(device), targets.to(device)

        outputs = model(inputs)

        _, predictions = torch.max(outputs, 1)

        n_correct += (predictions == targets).sum().item()
        n_total += targets.shape[0]

    acc = n_correct / n_total
    return acc
```

[35] Python

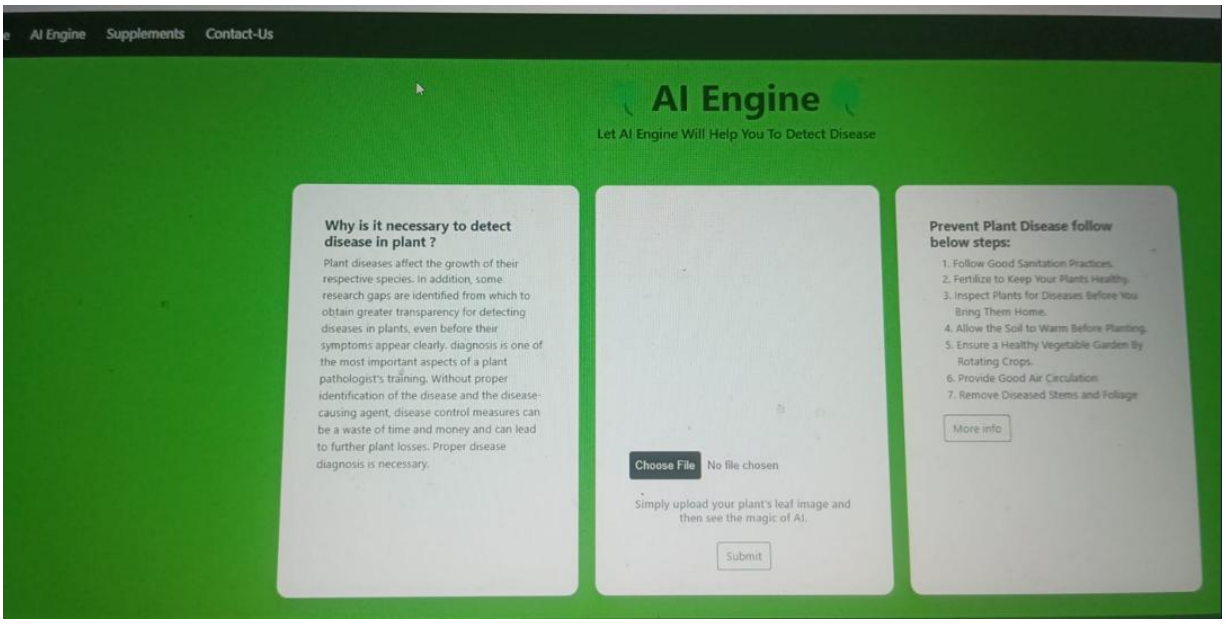
...

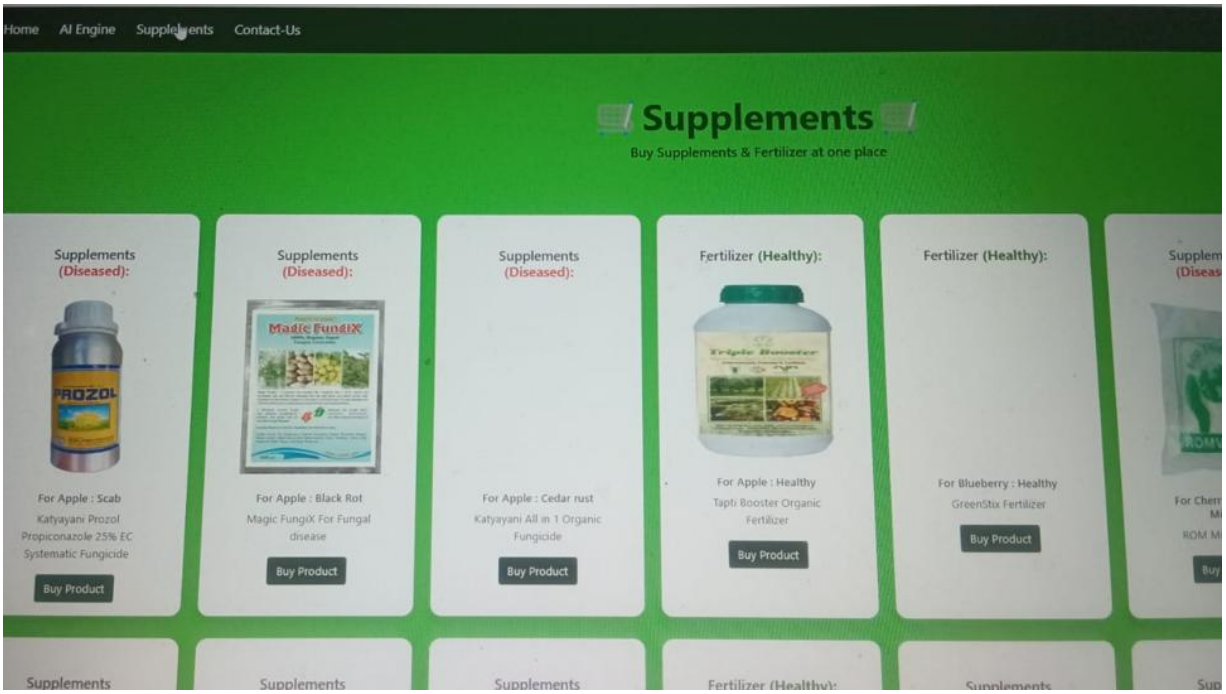
```python
train_acc = accuracy(train_loader)
test_acc = accuracy(test_loader)
validation_acc = accuracy(validation_loader)
```

[ ] Python

```python
print(
    f"Train Accuracy : {train_acc}\nTest Accuracy : {test_acc}\nValidation Accuracy : {validation_acc}"
)
```

[38] Python


AI Engine
Supplements
Contact-Us
AI Engine
Let AI Engine Will Help You To Detect Disease
Why is it necessary to detect disease in plant ?
Plant diseases affect the growth of their respective species. In addition, some research gaps are identified from which to obtain greater transparency for detecting diseases in plants, even before their symptoms appear clearly. diagnosis is one of the most important aspects of a plant pathologist's training. Without proper identification of the disease and the disease-causing agent, disease control measures can be a waste of time and money and can lead to further plant losses. Proper disease diagnosis is necessary.
Choose File
No file chosen
Simply upload your plant's leaf image and then see the magic of AI.
Submit
Prevent Plant Disease follow below steps:
1. Follow Good Sanitation Practices.
2. Fertilize to Keep Your Plants Healthy.
3. Inspect Plants for Diseases Before You Bring Them Home.
4. Allow the Soil to Warm Before Planting.
5. Ensure a Healthy Vegetable Garden By Rotating Crops.
6. Provide Good Air Circulation
7. Remove Diseased Stems and Foliage
More info


Home
AI Engine
Supplements
Contact-Us
Supplements
Buy Supplements & Fertilizer at one place
Supplements (Diseased):
For Apple : Scab
Katyayani Prozol Propiconazole 25% EC Systematic Fungicide
Buy Product
Supplements (Diseased):
For Apple : Black Rot
Magic FungiX For Fungal disease
Buy Product
Supplements (Diseased):
For Apple : Cedar rust
Katyayani All in 1 Organic Fungicide
Buy Product
Fertilizer (Healthy):
For Apple : Healthy
Tapti Booster Organic Fertilizer
Buy Product
Fertilizer (Healthy):
For Blueberry : Healthy
GreenStix Fertilizer
Buy Product

Turnitin Report

report-1

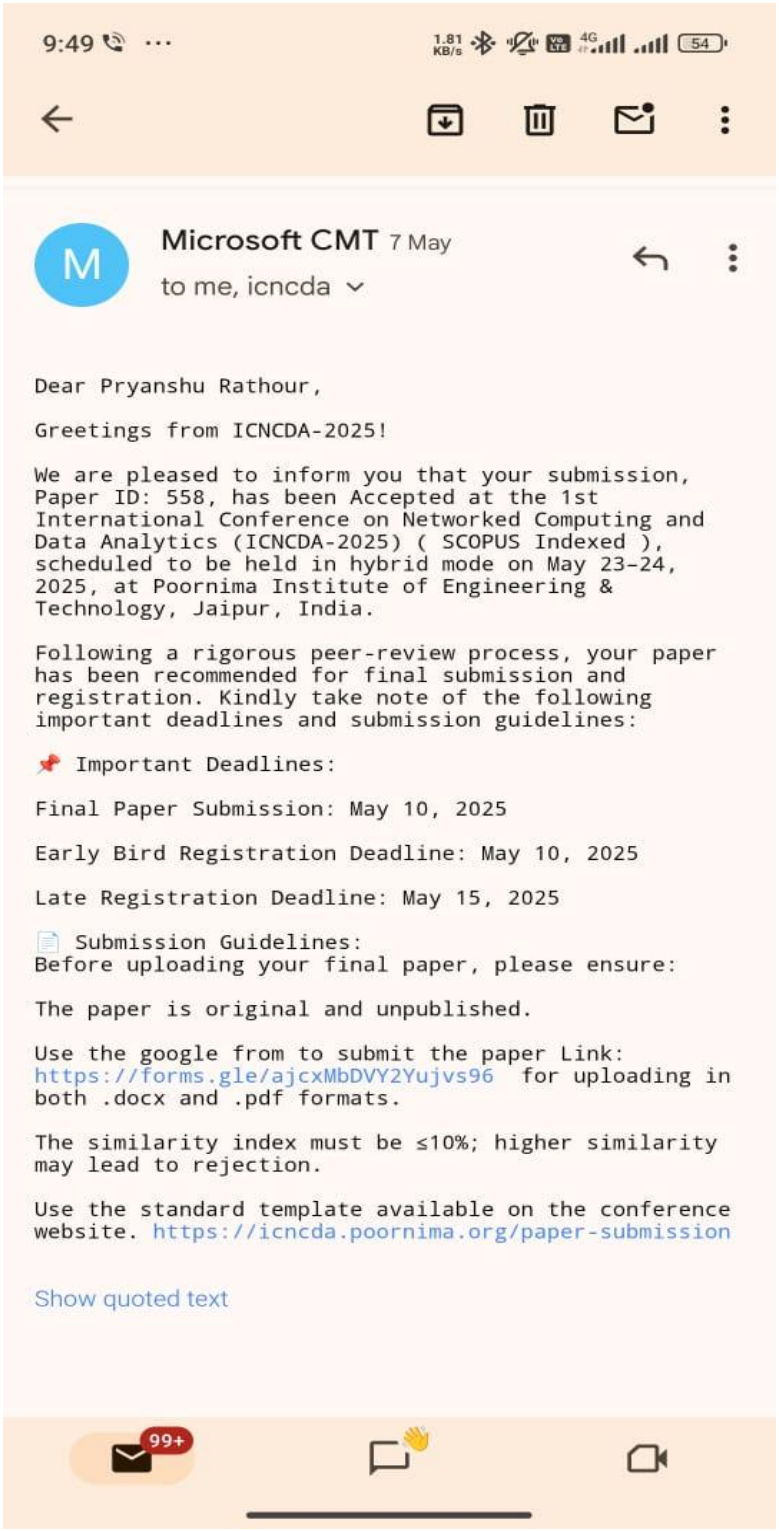
ORIGINALITY REPORT

26%	20%	13%	17%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Liverpool John Moores University Student Paper	4%
2	Submitted to KIET Group of Institutions, Ghaziabad Student Paper	4%
3	www.irjmets.com Internet Source	2%
4	mis.itmuniversity.ac.in Internet Source	1%
5	Submitted to HTM (Haridus- ja Teadusministeerium) Student Paper	1%
6	Poonam Nandal, Mamta Dahiya, Meeta Singh, Arvind Dagur, Brijesh Kumar. "Progressive Computational Intelligence, Information Technology and Networking", CRC Press, 2025 Publication	1%
7	ijarcce.com Internet Source	1%
8	arxiv.org Internet Source	1%
9	Submitted to University of Hertfordshire Student Paper	<1%
10	Submitted to ABES Engineering College Student Paper	<1%
11	www.irjet.net Internet Source	<1%

Proof Of Communication



9:49 ···

Microsoft CMT 7 May

to me, icncda

Dear Pryanshu Rathour,

Greetings from ICNCDA-2025!

We are pleased to inform you that your submission, Paper ID: 558, has been Accepted at the 1st International Conference on Networked Computing and Data Analytics (ICNCDA-2025) (SCOPUS Indexed), scheduled to be held in hybrid mode on May 23-24, 2025, at Poornima Institute of Engineering & Technology, Jaipur, India.

Following a rigorous peer-review process, your paper has been recommended for final submission and registration. Kindly take note of the following important deadlines and submission guidelines:

📌 Important Deadlines:

Final Paper Submission: May 10, 2025

Early Bird Registration Deadline: May 10, 2025

Late Registration Deadline: May 15, 2025

📄 Submission Guidelines:
Before uploading your final paper, please ensure:

The paper is original and unpublished.

Use the google from to submit the paper Link: https://forms.gle/ajcxMbDVY2Yujvs96 for uploading in both .docx and .pdf formats.

The similarity index must be ≤10%; higher similarity may lead to rejection.

Use the standard template available on the conference website. https://icncda.poornima.org/paper-submission

Show quoted text