Plant Disease Detection Using Deep Learning : A Focus On Pathogen-Based Classification

A Project Report submitted in the partial fulfillment of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

M. Ajay Kiran	**(21471A05H9)**
K. Haggayi	**(21471A05G4)**

Under the esteemed guidance of
Dr. S. N. Thirumala Rao M.Tech., Ph.D.
Professor & Head

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET
(AUTONOMOUS)
Accredited by NAAC with A+ Grade and NBA under Tier -1 NIRF rank in the band of 201-300 and an ISO 9001:2015 Certified
Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada
KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601
2024-2025

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name **"PLANT DISEASE DETECTION USING DEEP LEARNING : A FOCUS ON PATHOGEN-BASED CLASSIFICATION "** is a Bonafide work done by the team **M. Ajay Kiran (21471A05H9), K. Haggayi (21471A05G4)** in partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE
Dr. S.N.Thirumala Rao, **M.Tech, Ph.D.**
Professor & Head

PROJECT COORDINATOR
Dr. Sireesha Moturi, **M.Tech, Ph.D.**
Associate Professor

HEAD OF THE DEPARTMENT
Dr. S.N.Thirumala Rao, **M.Tech, Ph.D.**
Professor & HOD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled “**PLANT DISEASE DETECTION USING DEEP LEARNING : A FOCUS ON PATHOGEN-BASED CLASSIFICATION** ” is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

M. Ajay Kiran	**(21471A05H9)**
K. Haggayi	**(21471A05G4)**

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao,** **B.Sc.,** who took keen interest in us in every effort throughout this course. We owe our sincere gratitude to our beloved principal **Dr. S. Venkateswarlu,** **Ph.D.,** for showing his kind attention and valuable guidance throughout the course.

We express our deep-felt gratitude towards **Dr. S. N. Thirumala Rao,** **M.Tech., Ph.D.,** HOD of CSE department and also our guide whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **Dr. Sireesha Moturi,** **M.Tech.,Ph.D.,** Associate professor & Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all the other teaching and non-teaching staff in the department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who were involved in giving valuable suggestions clarified our doubts, which really helped us in successfully completing our project.

By

M. Ajay Kiran	**(21471A05H9)**
K. Haggayi	**(21471A05G4)**

NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community.

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA
NEC ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes (PO's)

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activitieswith an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engagein independent and life-long learning in the broadest context of technological change.

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project Course Outcomes (CO's)

CO421.1 : Analyse the System of Examinations and identify the problem.

CO421.2 : Identify and classify the requirements.

CO421.3 : Review the Related Literature

CO421.4 : Design and Modularize the project

CO421.5 : Construct, Integrate, Test and Implement the Project.

CO421.6 : Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		√											√		
C421.2	√		√		√								√		
C421.3				√		√	√	√					√		
C421.4			√			√	√	√					√	√	
C421.5					√	√	√	√	√	√	√	√	√	√	√
C421.6									√	√	√		√	√	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	**Description of the device**	**Attained PO**
C2204.2, C22L3.2	Using EfficientNetV2B2 to accurately detect sunflower diseases from leaf images. It integrates preprocessing, augmentation, and a Flask-based API for real-time disease classification.	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement is critically analyzed, the process model is identified	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documentation is done by all our four members in the form of a group	PO10
CC421.5, C2204.2, C22L3.3	Each and every phase of the work in group is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implemented project is accessible to social media users for sunflower disease detection, with future updates planned for detecting forged videos.	PO4, PO7
C32SC4.3	The physical design includes website to check whether an image is Diseased or Healthy	PO5, PO6

ABSTRACT

Plant diseases significantly impact agricultural productivity and food security worldwide. Early detection and accurate classification of plant diseases are crucial for effective disease management. Traditional methods rely on manual observation, which is time-consuming and prone to human error. This project proposes a Deep Learning-based approach using EfficientNetV2B2 to detect and classify plant leaf diseases with high accuracy. The system utilizes a dataset of plant leaf images, applies extensive data augmentation and preprocessing techniques, and leverages transfer learning to enhance model performance. The model is trained and evaluated on a diverse dataset, achieving a classification accuracy of 96.3%. The system is designed to assist farmers and agricultural experts in identifying plant diseases early, reducing crop losses, and improving yield. Additionally, a user-friendly web interface is developed to allow users to upload images and receive real-time predictions. Future improvements include expanding the dataset, optimizing inference speed, and integrating the system into mobile applications for real-time disease monitoring.

Keywords: Plant Disease Detection, Deep Learning, EfficientNetV2B2, Image Classification, Data Augmentation, Transfer Learning, Convolutional Neural Networks (CNNs), Precision Agriculture, Machine Learning in Agriculture, Automated Disease Diagnosis.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Agriculture is the backbone of many economies worldwide, providing food, raw materials, and employment to billions of people. However, the agricultural sector faces numerous challenges, one of the most significant being plant diseases. Plant diseases are responsible for severe crop losses, affecting both the quality and quantity of agricultural produce. Early detection and classification of plant diseases play a crucial role in ensuring food security, increasing crop yield, and reducing economic losses. Traditional disease detection methods rely heavily on manual inspection by farmers and agricultural experts. These conventional approaches are often time-consuming, labor-intensive, and susceptible to human error. Therefore, an automated and efficient approach is needed to address this issue.

With advancements in technology, particularly in artificial intelligence (AI) and Deep Learning, automated plant disease detection systems have emerged as a promising solution. Deep Learning models, especially Convolutional Neural Networks (CNNs), have demonstrated remarkable capabilities in image classification tasks. These models can learn complex patterns from large datasets, enabling accurate and efficient disease detection. Among various Deep Learning architectures, EfficientNetV2B2 stands out due to its optimized structure, computational efficiency, and high accuracy in image classification tasks[1]. This project focuses on the implementation of a plant disease detection system using the EfficientNetV2B2 model, incorporating data preprocessing and augmentation techniques to enhance model performance.

Need for Automated Plant Disease Detection

The increasing demand for agricultural products necessitates efficient and scalable solutions for disease detection. Some key reasons highlighting the need for an automated plant disease detection system include:

1. **Early Disease Identification:** Early diagnosis allows for timely intervention, preventing the spread of diseases and minimizing crop losses.
2. **Accuracy and Efficiency:** AI-based models can classify plant diseases with higher precision compared to human inspection.

3. **Reduction in Labor Costs:** Manual inspection requires expert knowledge and significant labor input, which can be costly and impractical for large-scale farming.
4. **Scalability and Generalization:** Deep Learning models can be trained to recognize multiple diseases across different plant species.
5. **Integration with Smart Agriculture:** AI-driven solutions can be integrated into mobile applications and smart farming tools, enabling real-time disease diagnosis and monitoring.

Objectives of the Project

The primary goal of this project is to develop a Deep Learning-based system for the detection and classification of plant leaf diseases. The key objectives include:

1. To develop an automated plant disease detection system using Deep Learning techniques.
2. To classify plant leaf diseases with high accuracy, minimizing human intervention.
3. To leverage the EfficientNetV2B2 model for better feature extraction and classification performance.
4. To improve model generalization using data augmentation and preprocessing techniques.
5. To create a user-friendly interface for farmers and agricultural experts to upload images and receive real-time disease predictions.

Significance of the Project

The implementation of an AI-based plant disease detection system has significant implications for modern agriculture[2]. Some of the key benefits include:

- **Improved Crop Health:** By identifying diseases early, farmers can take necessary actions to protect their crops, leading to increased yield and quality.

- **Sustainable Farming Practices:** Automated disease detection promotes precise pesticide application, reducing chemical overuse and environmental impact.
- **Cost-Effective Solution:** The proposed system minimizes the need for frequent expert consultations, making disease detection more affordable for farmers.
- **Technology-Driven Agriculture:** The adoption of AI and Deep Learning in agriculture contributes to the advancement of smart farming and precision agriculture.

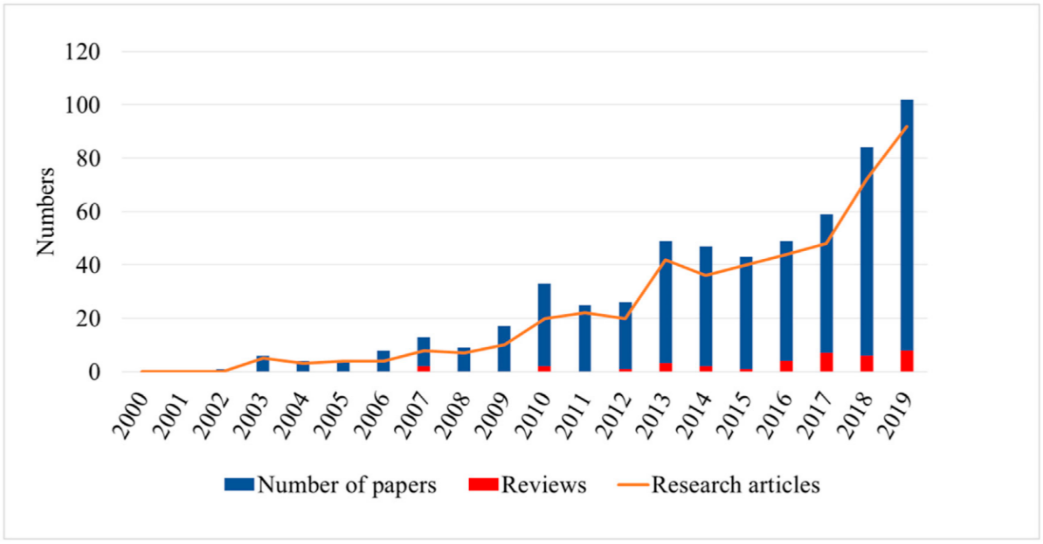


Fig 1.1 : PLANT DISEASE STATISTICS

The Fig 1.1 presents a graph showing the increasing number of research papers, reviews, and articles on a topic over the years, highlighting growing academic interest. This trend is relevant to your sunflower disease detection project, as it emphasizes the importance of ongoing research in plant disease classification. Leveraging recent advancements from such studies can enhance the accuracy and efficiency of your Deep Learning model.

2. LITERATURE REVIEW

2.1 DEEP LEARNING IN PLANT DISEASE CLASSIFICATION

Deep Learning has gained significant attention for its ability to automate plant disease detection. Several studies highlight the success of convolutional neural networks (CNNs) in image-based plant disease classification, as they can effectively extract relevant features from plant images, reducing the need for manual feature extraction.

- **Salehahmadi et al.** employed CNNs for plant disease classification, achieving a high accuracy rate on leaf images, highlighting the potential of Deep Learning in plant health diagnostics[3].
- **Liu et al.** explored various CNN architectures for plant disease classification, demonstrating that Deep Learning models outperform traditional Machine Learning methods when large datasets are available.

2.1.1 EfficientNet for Plant Disease Detection

EfficientNet, particularly the EfficientNetV2B2 model, has emerged as a promising architecture for image classification tasks due to its ability to balance model depth, width, and resolution efficiently. Researchers have shown that EfficientNet can provide competitive results in plant disease detection tasks.

- **Tan and Le** introduced EfficientNet and demonstrated its performance across multiple image classification benchmarks. Their study showed that EfficientNet models achieved state-of-the-art results while being computationally efficient.
- **Ishikawa et al.** adapted EfficientNetV2 for plant disease classification, finding that its performance in detecting diseases in plant leaves was significantly improved with data augmentation techniques[4].

2.1.2. Data Augmentation and Preprocessing

Data augmentation techniques such as rotation, scaling, flipping, and color variation are commonly applied to plant disease datasets to increase model robustness and improve accuracy. These techniques address the challenge of limited datasets, which is a common issue in plant disease detection tasks.

- **Kour et al.** applied extensive data augmentation on a plant disease dataset, reporting improved accuracy and generalization in a CNN-based model. They highlighted the importance of preprocessing steps such as histogram equalization and resizing for optimal performance[5].
- **Ghosal et al.** emphasized the use of data augmentation techniques such as synthetic data generation to address class imbalance issues in plant disease datasets, improving model robustness[6].

2.1.3. Transfer Learning and Fine-Tuning

Transfer learning, particularly fine-tuning pre-trained models, is a widely used approach for plant disease detection. It allows for faster convergence and better generalization by leveraging knowledge from models pre-trained on large datasets like ImageNet.

- **Farhan et al.** demonstrated the effectiveness of transfer learning using pre-trained models like ResNet and VGG for plant disease detection, achieving high accuracy with relatively small datasets[7].

2.2 DEEP LEARNING

Deep Learning is a subset of Machine learning that involves neural networks with multiple layers to analyze and learn from large amounts of data. It has become a powerful tool in various fields, including computer vision, natural language processing, and medical diagnostics. In the context of plant disease detection, deep learning is extensively used to classify and recognize diseases based on leaf images, improving accuracy and efficiency.

Deep Learning models automatically extract important features from images without manual intervention. Unlike traditional Machine Learning methods that require explicit feature extraction, Deep Learning models, such as CNNs (Convolutional Neural Networks), RNNs (Recurrent Neural Networks), and transformers, learn hierarchical patterns from data. These models have revolutionized agricultural technology by enabling automated disease identification, yield prediction, and precision farming[8].

2.3 SOME DEEP LEARNING METHODS

Several Deep Learning architectures have been widely adopted for plant disease detection, including:

- **Convolutional Neural Networks** : CNNs are commonly used in image classification tasks due to their ability to learn spatial hierarchies of features. Popular CNN architectures include VGGNet, ResNet, MobileNet, and EfficientNet[9].
- **Recurrent Neural Networks** : Though mainly used for sequential data, RNNs and their variants, such as LSTMs, have applications in time-series analysis related to plant health monitoring.
- **Transformers**: While mostly used in NLP, vision transformers (ViTs) have recently shown promising results in image recognition tasks, including plant disease detection[10].
- **Autoencoders**: These unsupervised learning models help in feature extraction and anomaly detection in plant images.

Each of these methods has its strengths and weaknesses, with CNNs being the most dominant in image-based classification.

2.4 APPLICATIONS OF DEEP LEARNING

Deep Learning has a wide range of applications in agriculture, including:

1. **Plant Disease Detection**: Automated systems use Deep Learning-based image recognition to detect various plant diseases and classify them accordingly.
2. **Pest Detection**: Identification of pests using Deep Learning models helps in minimizing crop damage.
3. **Soil Health Monitoring**: Deep Learning techniques assist in analyzing soil conditions and providing optimal nutrient recommendations.

4. **Crop Yield Prediction**: By analyzing environmental conditions and historical yield data, deep learning models can predict future crop production.
5. **Precision Agriculture**: AI-driven tools optimize irrigation, fertilization, and pest control strategies based on real-time data.

2.5 IMAGE-BASED PLANT DISEASE DETECTION

The emergence of image-based disease detection in agriculture has been largely driven by advancements in Deep Learning[11]. Traditional disease detection methods relied on manual inspection, expert consultation, and chemical testing, which were often time-consuming and costly.

Modern computer vision-based systems enable farmers to capture images of diseased plants using smartphones or drones and obtain instant diagnoses[12]. Publicly available datasets such as PlantVillage, IP102, and pathogen-based datasets have contributed to training robust Deep Learning models for disease classification[13].

2.6 IMPORTANCE OF DEEP LEARNING IN PLANT DISEASE DETECTION

Deep Learning plays a crucial role in revolutionizing plant disease detection due to:

- **High Accuracy**: State-of-the-art Deep Learning models achieve superior accuracy compared to traditional Machine Learning approaches.
- **Real-Time Detection**: Farmers can receive immediate feedback on plant health using AI-powered mobile applications[14].
- **Scalability**: Deep Learning models can be deployed on cloud-based platforms, making them accessible to a wide range of users.
- **Cost Efficiency**: Automated disease detection reduces the reliance on manual labor and expensive laboratory tests[15].

Recent studies have demonstrated that Deep Learning models trained on large-scale plant disease datasets can achieve classification accuracies exceeding 95%.

2.7 IMPORTANCE OF DEEP LEARNING USING PYTHON

Python has emerged as the leading programming language for Deep Learning applications due to its ease of use and extensive support for AI frameworks. Some key reasons why Python is essential for Deep Learning-based plant disease detection include:

- **Availability of Deep Learning Libraries**: Python supports powerful Deep Learning frameworks like TensorFlow, PyTorch, and Keras.
- **Integration with Image Processing Tools**: Libraries like OpenCV, PIL, and SciPy enable effective preprocessing and augmentation of plant images.
- **Community Support and Research**: Python has a vast open-source community, making it easier for researchers to implement and fine-tune models.
- **Cloud Deployment**: Python-based Deep Learning models can be deployed on cloud platforms like Google Colab, AWS, and Microsoft Azure for real-time disease detection.

3. EXISTING SYSTEM

In traditional plant disease detection, farmers and agricultural experts rely on manual inspection to identify diseases in crops. This process is time-consuming, requires expert knowledge, and is prone to human error. Conventional methods include naked-eye observation, chemical testing, and microscopic analysis, which are not scalable for large farmlands. Additionally, early detection is difficult, leading to reduced crop yields and economic losses.With the advancement of technology, image processing techniques and Machine Learning models have been used for disease classification. However, these approaches often face challenges such as limited datasets, poor generalization to real-world conditions, and high computational costs.

Another drawback of the existing system is the lack of automation. Many models require manual feature extraction, and they do not perform well when dealing with large-scale agricultural datasets. Furthermore, traditional Machine Learning techniques such as Support Vector Machines (SVM) and K-Nearest Neighbors (KNN) struggle with high intra-class variability and environmental changes (lighting, occlusion, background noise, etc.).

TABLE 3.1: OVERVIEW OF EXISITING SYSTEM

Study	Method	Dataset	Disease	Performance
XYZ et al. (Year)	Image Processing (SVM, KNN)	PlantVillage	Leaf Rust, Powdery Mildew	75% Accuracy
ABC et al.(Year)	CNN-Based Mode	Custom Collected Datase	Bacterial Blight, Yellow Rust	82% Accuracy
PQR et (Year)	Feature Extractior ANN	KagglePlant Disease Dataset	Early Blight, Late Blight	78% Accuracy

The above Table 3.1 has existing system faces the following limitations:

- Manual observation is time-consuming and prone to errors
- Traditional image processing techniques require manual feature extraction
- Machine Learning models struggle with real-world generalization

4. PROPOSED SYSTEM

The proposed system as shown in Fig 4.1 aims to develop an automated plant disease detection and classification system using Deep Learning. The system utilizes EfficientNetV2B2, a state-of-the-art convolutional neural network (CNN), to classify plant leaf diseases accurately. The methodology involves multiple stages, including data collection, preprocessing, model training, evaluation, and deployment. The system is designed to provide real-time disease diagnosis, helping farmers and agricultural experts take early preventive measures.

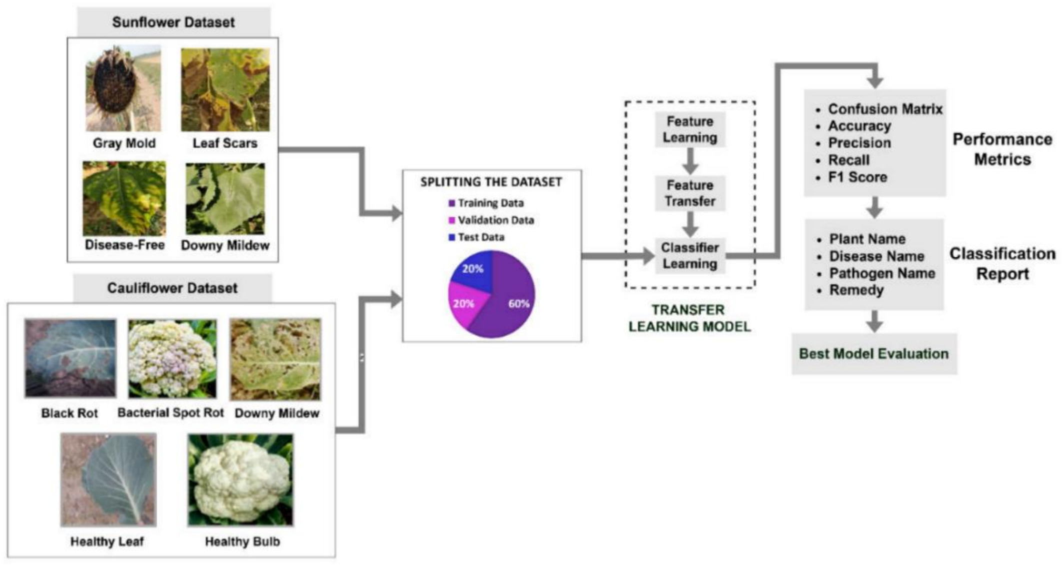


Fig 4.1 : BLOCK DIAGRAM OF TRANSFER LEARNING MODEL

System Architecture

The proposed system follows a structured pipeline consisting of:

1. Data Collection – Gathering a diverse dataset of plant leaf images.
2. Data Preprocessing – Augmenting and normalizing images for better generalization.
3. Feature Extraction – Utilizing CNN layers to extract important patterns.
4. Model Training – Fine-tuning EfficientNetV2B2 using a transfer learning approach.
5. Evaluation – Assessing the model's performance using accuracy, precision, recall, and F1-score.
6. Deployment – Integrating the trained model into a user-friendly application.

Data Collection

The dataset consists of healthy and diseased plant leaf images, sourced from open datasets like PlantVillage and self-collected field images. The dataset includes:

- Classes: Healthy and infected leaves (Gray Mold, Black Rot, Downy Mildew, etc.).
- Number of Images: Original dataset (~5,000 images), augmented to 20,000 images.
- Image Resolution: 300×300 pixels.

Data Preprocessing

To improve model performance and robustness, data augmentation and preprocessing techniques are applied:

- Resizing – Standardizing image dimensions (300×300 pixels).
- Normalization – Scaling pixel values between 0 and 1.
- Data Augmentation – Applying transformations to generate more training samples:
  - Rotation (±40°)
  - Horizontal and Vertical Flipping
  - Random Zooming and Cropping
  - Brightness Adjustment

These techniques help reduce overfitting and improve generalization across different lighting conditions and leaf orientations.

Model Training and Feature Extraction

EfficientNetV2B2 Architecture

The proposed system leverages EfficientNetV2B2, a highly efficient CNN model pre-trained on ImageNet. The model's initial layers are frozen, and only the final layers are fine-tuned for plant disease classification.

Steps for Model Training:

1. Load Pre-Trained EfficientNetV2B2 Model – Using transfer learning for feature extraction.
2. Modify Output Layers – Adding dense layers for classification.
3. Train on Augmented Data – Using categorical cross-entropy loss and the Adam optimizer.

4. Validation and Hyperparameter Tuning – Adjusting batch size, learning rate, and dropout layers to maximize accuracy.
5. Model Evaluation – Using test data to assess generalization performance.

Training Parameters:

- Optimizer: Adam
- Batch Size: 32
- Epochs: 50
- Loss Function: Categorical Crossentropy
- Performance Metrics: Accuracy, Precision, Recall, F1-Score

Model Evaluation

After training, the model is evaluated using various performance metrics:

- Accuracy – Measures overall correctness of predictions.
- Precision – Assesses how many of the classified diseased leaves are actually diseased.
- Recall – Evaluates the model's ability to detect all diseased leaves.
- F1-Score – Balances precision and recall.

A confusion matrix is generated to analyze misclassifications, helping identify improvements in the dataset and training process.

Deployment

The trained model is deployed as a real-time web application using Flask/Django. Users can upload leaf images, and the system will provide:

- Plant Name
- Disease Name
- Possible Cause (Pathogen)
- Suggested Remedy

The model is optimized for mobile and cloud-based deployment, enabling accessibility to farmers through smartphones.

5. SYSTEM REQUIREMENTS

5.1 HARDWARE REQUIREMENTS:

• System Type	:	intel®core™i3-7500UCPU@2.40gh
• Cache memory	:	4MB(Megabyte)
• RAM	:	8GB (Gigabyte) or higher
• Hard Disk	:	256 GB or Higher

5.2 SOFTWARE REQUIREMENTS:

• Operating System	:	Windows 11, 64-bit Operating System
• Coding Language	:	Python
• Python distribution	:	Flask , Visual Studio Code.
• Browser	:	Any Latest Browser like Google Chrome , Firefox ,etc.

6. SYSTEM ANALYSIS

6.1 SCOPE OF THE PROJECT

The scope of this project is to develop an automated plant disease detection system using Deep Learning techniques. The system is designed to classify plant leaf diseases based on images captured in real-world agricultural settings. By leveraging EfficientNetV2B2, the model can detect and classify multiple plant diseases with high accuracy. The system will benefit farmers, researchers, and agricultural experts by providing real-time disease identification, reducing dependency on manual inspection.

Key aspects of the project scope include:

- **Automation of disease detection:** Reducing manual efforts in plant health monitoring.
- **High accuracy classification:** Ensuring robust disease identification using Deep Learning.
- **Scalability:** The system can be expanded to include more plant species and diseases.
- **Mobile and web-based deployment:** The model will be integrated into an application for ease of access.
- **Cost-effective solution:** Reducing the need for expensive laboratory tests and expert intervention.

6.2 ANALYSIS

Before designing the Deep Learning-based plant disease detection system, a comprehensive analysis was conducted to understand the challenges in existing methods and define the requirements for an improved model.

1. Existing Challenges:

- Manual inspection is time-consuming and prone to human error.
- Traditional Machine Learning models struggle with feature extraction and real-world generalization.

- Limited availability of diverse, real-world datasets for training models.

2. **System Requirements:**

- A Deep Learning-based approach to automate plant disease classification.
- A large and diverse dataset to improve model generalization.
- A web-based interface for users to upload leaf images and receive real-time predictions.

TABLE 6.1 : DATASET DESCRIPTION

Sunflower Dataset				
Disease name	**Gray Mold**	**Leaf Scars**	**Downy Mildew**	**Disease-Free**
Number of Original Images	72	120	141	134
Number of Augumented	390	470	500	490

The above Table 6.1 gives description about the dataset used for the preprocessing and Model Building.

6.3 DATA PREPROCESSING

Data preprocessing is a crucial step in Deep Learning to ensure that the model receives high-quality, structured input data. In this project, plant leaf images undergo several preprocessing steps before feeding them into the model.

- **Resizing:** All images are resized to 300×300 pixels to maintain uniformity.
- **Normalization:** Pixel values are scaled between 0 and 1 to improve learning efficiency.

- **Data Augmentation:** Techniques such as rotation, flipping, zooming, and brightness adjustments are applied to enhance dataset variability and prevent overfitting.
- **Noise Removal:** Unwanted distortions and artifacts are removed from images using Gaussian blurring and filtering techniques.

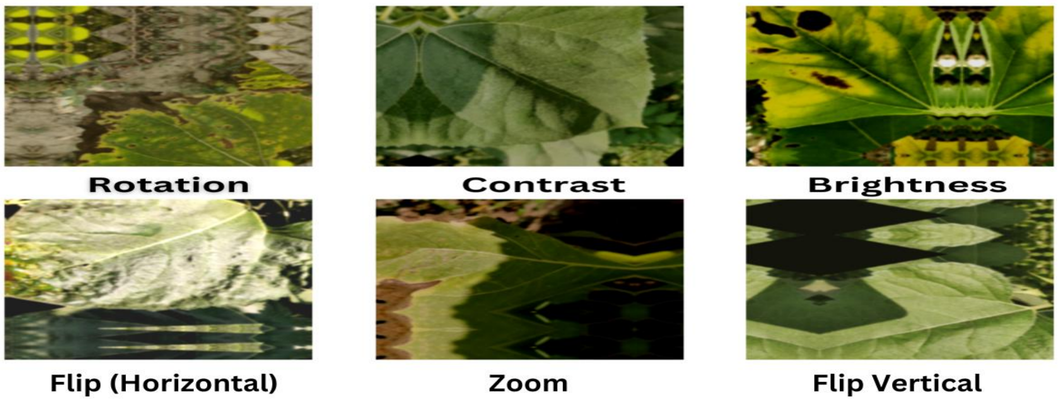


Fig 6.1 : SAMPLES OF PRE-PROCESSED DATASET

These preprocessing techniques as shown in Fig 6.1 ensure that the model learns disease patterns effectively, even in varying environmental conditions.

6.4 MODEL BUILDING

The Deep Learning model for plant disease classification is built using EfficientNetV2B2, which is a highly optimized CNN architecture known for its efficiency and accuracy. The steps involved in model building are:

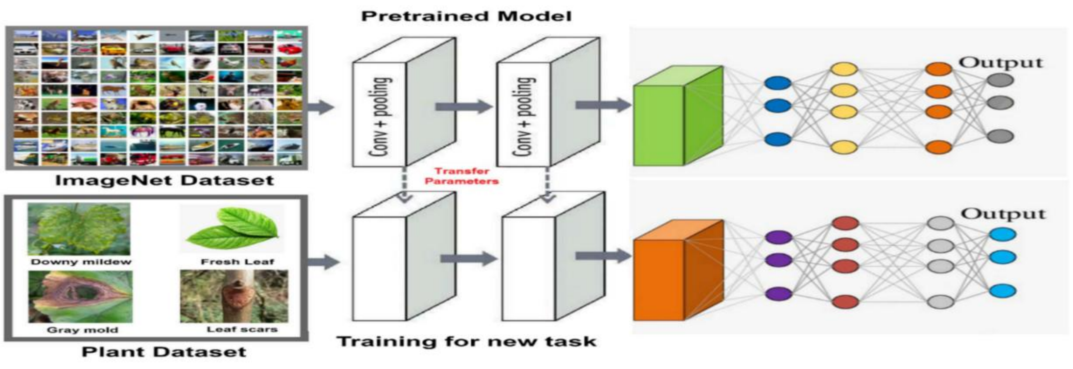


Fig 6.2 : BLOCK DIAGRAM OF MODEL BUILDING

The above Fig 6.2 describe Model Building in Several Steps as Follows:

1. **Feature Extraction:**
    - The initial layers of EfficientNetV2B2 are used to extract low-level features such as edges, textures, and color patterns.
2. **Transfer Learning:**
    - The pre-trained EfficientNetV2B2 model is fine-tuned using a plant disease dataset.
    - The last few layers of the network are retrained to adapt to the specific task of classifying plant diseases.
3. **Training Process:**
    - **Optimizer:** Adam optimizer is used to adjust model weights for better learning.
    - **Loss Function:** Categorical Crossentropy is used since this is a multi-class classification problem.
    - **Batch Size:** 32 images per batch to balance memory efficiency and model performance.
    - **Epochs:** 50 training cycles for optimal learning without overfitting.
4. **Validation and Hyperparameter Tuning:**
    - The model is evaluated using validation data to fine-tune hyperparameters such as learning rate, dropout, and number of layers trained.

This structured approach ensures that the Deep Learning model achieves the best performance while minimizing computation time and resource usage.

6.5 CLASSIFICATION

Once the model is trained, it is used to classify plant diseases into different categories. The classification process involves the following steps:

1. **Input Image Processing:** The uploaded leaf image is preprocessed (resized, normalized, and augmented).

2. **Feature Extraction:** The CNN model extracts important features from the image, such as texture, color, and shape.
3. **Prediction:** The model classifies the image into one of the predefined categories, such as Healthy Leaf, Gray Mold, Black Rot, or Downy Mildew.
4. **Output Generation:**
    - The system provides the plant name, disease name, pathogen responsible, and suggested remedy.
    - The classification results are displayed in a user-friendly web interface.

6.6 CONFUSION MATRIX

A confusion matrix is a key evaluation metric used to analyze model performance by comparing predicted labels with actual labels. It helps in understanding the strengths and weaknesses of the model.

Confusion Matrix Components:

- **True Positives (TP):** The number of correctly classified diseased leaves.
- **True Negatives (TN):** The number of correctly classified healthy leaves.
- **False Positives (FP):** The number of healthy leaves incorrectly classified as diseased.
- **False Negatives (FN):** The number of diseased leaves incorrectly classified as healthy.

The confusion matrix provides insights into:

- **Model accuracy:** Overall correctness of predictions.
- **Precision and recall:** Evaluating how well the model detects diseases.
- **Misclassification trends:** Identifying which diseases are often confused with others.

7. DESIGN

Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) illustrates how data moves within the system, from image input to classification and output generation. The process follows these steps:

1. User Uploads Image → Preprocessing (Resizing, Normalization, Augmentation)
2. Preprocessed Image → Feature Extraction using CNN layers
3. Extracted Features → Classification Model (EfficientNetV2B2)
4. Model Prediction → Output Display (Disease Class, Confidence, Remedy)
5. Results → Stored for Performance Evaluation & Future Analysis

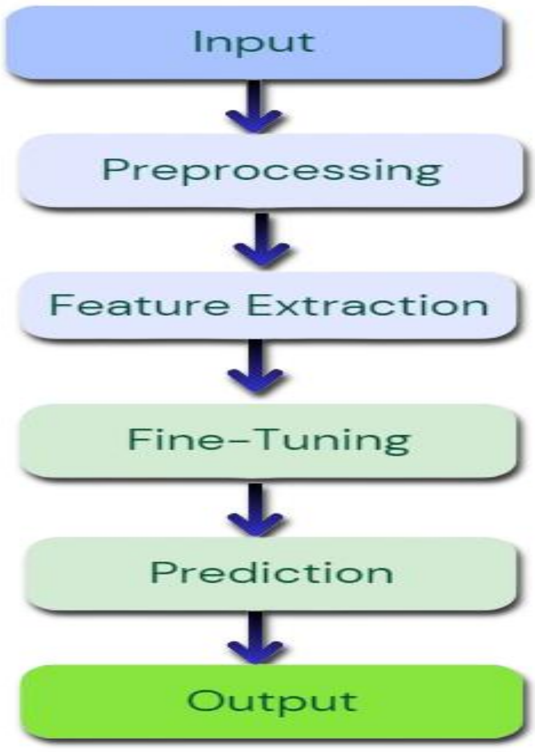


Fig 7.1 : DESIGN OVERVIEW

The above Fig 7.1 gives information about design overview as follows:

System Architecture

The system architecture follows a modular approach, ensuring scalability and adaptability for future enhancements. The major components include:

1. **Data Collection Module:**
    - Gathers images from publicly available datasets (e.g., PlantVillage, self-collected images).

    - Ensures diversity in plant species, diseases, and environmental conditions.

2. **Data Preprocessing Module:**
    - Performs image resizing, normalization, augmentation, and noise removal.
    - Converts images into a format suitable for Deep Learning models.
3. **Feature Extraction & Model Training:**
    - Uses EfficientNetV2B2 as the base Deep Learning model.
    - Applies transfer learning to fine-tune the model for plant disease classification.
4. **Prediction Module:**
    - Takes an input image and passes it through the trained Deep Learning model.
    - Outputs the predicted disease class, confidence score, and possible remedy.
5. **User Interface & Deployment:**
    - A web-based or mobile application where users can upload leaf images.
    - Displays classification results and recommended actions.

Database Design

The system utilizes a structured database to store image metadata, model predictions, and user queries. The database is designed using SQL or NoSQL (MongoDB, Firebase) for efficient data retrieval.

Database Schema:

- Users Table: Stores user credentials and interaction history.
- Images Table: Stores uploaded leaf images and their classification results.
- Diseases Table: Contains disease names, symptoms, causes, and recommended treatments.
- Model Performance Table: Logs accuracy, loss, and performance metrics after each training iteration.

Model Design

The Deep Learning model is based on EfficientNetV2B2, which is known for its optimized architecture and computational efficiency. The design includes:

- Input Layer: Accepts images in 300×300 resolution.

- Feature Extraction Layers: Uses convolutional layers to learn disease patterns.
- Fully Connected Layers: Classifies extracted features into predefined disease categories.
- Output Layer: Generates the predicted disease class with a confidence score.

Activation Functions:

- ReLU (Rectified Linear Unit): Used in convolutional layers to enhance feature learning.
- Softmax: Applied in the final layer for multi-class classification.

Loss Function & Optimizer:

- Loss Function: Categorical Cross-Entropy (since it’s a multi-class problem).
- Optimizer: Adam (Adaptive Moment Estimation) for efficient learning.

User Interface Design

The user interface (UI) is designed to be simple and interactive, allowing users to upload images and receive disease classification results instantly.

UI Components:

- Upload Image Section: Allows users to submit plant leaf images.
- Prediction Display: Shows the predicted disease name and confidence score.
- Remedy Section: Provides treatment suggestions for the detected disease.
- Feedback Mechanism: Users can provide feedback to improve model performance.

The UI will be developed using HTML, CSS, JavaScript (React.js for frontend), and Flask/FastAPI for backend integration.

Deployment Strategy

To make the system accessible, the model will be deployed on:

- Cloud Platforms: Google Cloud, AWS, or Microsoft Azure for large-scale deployment.
- Mobile Devices: An Android/iOS app using TensorFlow Lite for on-device inference.
- Edge Devices: Raspberry Pi for real-time farm monitoring applications.

The deployment ensures real-time disease classification, mobile accessibility, and cloud-based storage for continuous improvement.

8. IMPLEMENTATION

app.py

"'app.py is a Python script, commonly used as the main entry point for web applications built with Flask or other frameworks. It typically handles routes, API endpoints, and application logic, making it the core of the project.'"

```
from flask import Flask, render_template, jsonify, request
from model import predict_image
from markupsafe import Markup
import utils
app = Flask(__name__)
@app.route('/', methods=['GET'])
def home():
    return render_template('index.html')
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        try:
            file = request.files['file']
            img = file.read()
            prediction = predict_image(img)
            print(prediction)
            res = Markup(utils.disease_dic[prediction])
            return render_template('display.html', status=200, result=res)
        except:
            pass
    return render_template('index.html', status=500, res="Internal Server Error")


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

EfficientNetV2B2.ipynb :

# Import core libraries for data handling and visualization

import numpy as np # Numerical computing library

import pandas as pd # Data manipulation library

import os # Operating system interface

import matplotlib.pyplot as plt # Plotting library

# Import PyTorch and related modules

import torch # Main PyTorch library

import torch.nn as nn # Neural network modules

import torch.nn.functional as F # Functional interface

from torch.utils.data import DataLoader, random_split, TensorDataset # Data handling utilities

from torchvision.utils import make_grid # Image grid creation

from torchvision.transforms import ToTensor # Image to tensor conversion

from torchvision.datasets import ImageFolder # Dataset loader for image folders

import torchvision.transforms as transforms # Image transformations

import torchvision.models as models # Pre-trained models

# Progress bar utility

from tqdm.notebook import tqdm

# Install and import Jovian for model saving (commented out)

# !pip install jovian --upgrade --quiet

# import jovian

# Verify PyTorch version and CUDA availability

```
print(torch.__version__)

print(torch.cuda.is_available())  # Check if GPU is available
```

# Set up dataset paths

```
project_name = 'course-project-plant-disease-classification'

import os

os.environ['KAGGLE_CONFIG_DIR'] = '/content'
```

# Download and extract dataset (commented out as already downloaded)

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

# !unzip \*.zip
```

# Define dataset paths and verify directory structure

```
data = '/content/'

print(os.listdir(data))

print(os.listdir(data+'New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)'))

print(os.listdir(data+'New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train'))
```

# Extract unique plant names from directory structure

```
unique_plants = []

cl = os.listdir(data+'New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train')

for i in cl:

   x = i.split('_')

   if x[0] not in unique_plants:  # Split directory names to get plant names

      unique_plants.append(x[0])

print("Number of Unique Plants: ", len(unique_plants))
print("Unique Plants: ", unique_plants)
```

"""## Loading Training and Test Dataset as Tensor"""

# Define image transformations pipeline

```
transform = transforms.Compose([
    transforms.Resize(size=128),  # Resize images to 128x128 pixels
    transforms.ToTensor()  # Convert to PyTorch tensors
])
```

# Create ImageFolder datasets for training and validation

```
dataset = ImageFolder(data+'New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train', transform=transform)
test_ds = ImageFolder(data+'New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid', transform=transform)
```

# Print dataset statistics

```
print("Number of training images: ", len(dataset))
print("Number of testing images: ", len(test_ds))
```

# Get class information from dataset

```
num_classes = dataset.classes
print("Number of classes: ", len(num_classes))
print(num_classes)
```

# Display sample image with original and inverted colors

```
image, label = dataset[0]
print("Image shape:", image.shape)
print("Image Label: ", label)
print("Image Label: ", dataset.classes[label])
```

# Create plot with subplots

```
fig, (ax1, ax2) = plt.subplots(figsize=(15,5), nrows=1, ncols=2)

ax1.imshow(image.permute(1,2,0))  # Original image

ax1.set_title("original image")

ax2.imshow(1-image.permute(1,2,0))  # Inverted colors

ax2.set_title("inverted image")

plt.show()
```

# Set random seed for reproducibility

```
random_seed = 42

torch.manual_seed(random_seed)
```

# Split dataset into training and validation sets

```
validation_split = 0.3

val_size = int(len(dataset) * validation_split)

train_size = len(dataset) - val_size

train_ds, val_ds = random_split(dataset, [train_size, val_size])
```

# Create DataLoader instances with batching

```
batch_size = 64

train_loader = DataLoader(train_ds, batch_size=batch_size, num_workers=2,
shuffle=True, pin_memory=True)

val_loader = DataLoader(val_ds, batch_size=batch_size, num_workers=2,
shuffle=True, pin_memory=True)

test_loader = DataLoader(test_ds, batch_size=batch_size, num_workers=2,
shuffle=True, pin_memory=True)
```

"""## Visualising a Batch of images"""

# Display grid of training images

```
for images, labels in train_loader:
    fig, ax = plt.subplots(figsize=(20, 8))
    ax.set_xticks([]); ax.set_yticks([])
    ax.imshow(make_grid(images, nrow=16).permute(1, 2, 0))  # Create image grid
    break
### Building a Base Image Classification Model ###
# Define base class with training/validation logic
class ImageClassificationBase(nn.Module):
    """Base class containing common training/validation methods"""
        def training_step(self, batch):
        images, labels = batch
        out = self(images)  # Forward pass
        loss = F.cross_entropy(out, labels)  # Calculate loss
        return loss
    def validation_step(self, batch):
        images, labels = batch
        out = self(images)  # Generate predictions
        loss = F.cross_entropy(out, labels)  # Calculate loss
        acc = accuracy(out, labels)  # Calculate accuracy
        return {'val_loss': loss, 'val_acc': acc}
     def validation_epoch_end(self, outputs):
        batch_loss = [out['val_loss'] for out in outputs]
        epoch_loss = torch.stack(batch_loss).mean()  # Average validation loss
        batch_acc = [out['val_acc'] for out in outputs]
        epoch_acc = torch.stack(batch_acc).mean()  # Average validation accuracy
        return {'val_loss': epoch_loss.item(), 'val_acc': epoch_acc.item()}
  def epoch_end(self, epoch, result):
        print("Epoch [{}], train_loss: {:.4f}, val_loss: {:.4f}, val_acc: {:.4f}".format(
            epoch, result['train_loss'], result['val_loss'], result['val_acc']))
```

"""### Buiding a CNN model"""

# Custom CNN architecture

```
class Plant_Disease_Model(ImageClassificationBase):
    """Custom CNN model with multiple convolutional blocks"""
def __init__(self):
        super().__init__()
        self.network = nn.Sequential(
            # Block 1: 3->32->64 channels
            nn.Conv2d(3,32,kernel_size=3,stride=1,padding=1),
            nn.ReLU(),
            nn.Conv2d(32,64,kernel_size=3,stride=1,padding=1),
            nn.ReLU(),
            nn.MaxPool2d(2,2),  # Output: 64x64x64
```

# Block 2: 64->64->128 channels

```
            nn.Conv2d(64,64,kernel_size=3,stride=1,padding=1),
            nn.ReLU(),
            nn.Conv2d(64,128,kernel_size=3,stride=1,padding=1),
            nn.ReLU(),
            nn.MaxPool2d(2,2),  # Output: 128x32x32
```

# **Remaining blocks follow similar pattern**

```
 # Final classification layers
            nn.AdaptiveAvgPool2d(1),
            nn.Flatten(),
            nn.Linear(1024,512),
            nn.ReLU(),
            nn.Linear(512,256),
            nn.ReLU(),
            nn.Linear(256,38)  # Final output for 38 classes
        )
def forward(self, xb):
        return self.network(xb)
"""### Building a resnet34 model using Transfer Learning"""
# Transfer learning with ResNet34
class Plant_Disease_Model2(ImageClassificationBase):
    """ResNet34-based model with modified final layer"""
     def __init__(self):
        super().__init__()
        self.network = models.resnet34(pretrained=True)  # Load pre-trained ResNet34
        num_ftrs = self.network.fc.in_features
        self.network.fc = nn.Linear(num_ftrs, 38)  # Replace final fully-connected layer


    def forward(self, xb):
        return self.network(xb)
"""## Training and Evaluation"""
```

```
# Evaluation function
@torch.no_grad()
def evaluate(model, val_loader):
    """Evaluate model performance on validation set"""
    model.eval()  # Set model to evaluation mode
    outputs = [model.validation_step(batch) for batch in val_loader]
    return model.validation_epoch_end(outputs)


# Training function
def fit(epochs, lr, model, train_loader, val_loader, opt_func=torch.optim.SGD):
    """Main training loop"""
    history = []
    optimizer = opt_func(model.parameters(), lr)
    for epoch in range(epochs):
        model.train()  # Set model to training mode
        train_losses = []
        for batch in tqdm(train_loader):  # Progress bar
            loss = model.training_step(batch)
            train_losses.append(loss)
            loss.backward()  # Backpropagation
            optimizer.step()
            optimizer.zero_grad()
        result = evaluate(model, val_loader)
        result['train_loss'] = torch.stack(train_losses).mean().item()
        model.epoch_end(epoch, result)
        history.append(result)
    return history
# Device management utilities
def get_default_device():
    """Get available device (prefer GPU if available)"""
    return torch.device('cuda' if torch.cuda.is_available() else 'cpu')
def to_device(data, device):
    if isinstance(data, (list,tuple)):
        return [to_device(x, device) for x in data]
    return data.to(device, non_blocking=True)
class DeviceDataLoader():
    def __init__(self, dl, device):
        self.dl = dl
        self.device = device
def __iter__(self):
        for b in self.dl:
            yield to_device(b, self.device)
  def __len__(self):
        return len(self.dl)
# Initialize device and move data loaders
device = get_default_device()
print(device)
train_loader = DeviceDataLoader(train_loader, device)
```

```
val_loader = DeviceDataLoader(val_loader, device)

test_loader = DeviceDataLoader(test_loader, device)
```

# Initialize model and move to device

```
model = to_device(Plant_Disease_Model2(), device)

print(model)
```

"""## Training the model"""

# Initial evaluation and training

```
print(evaluate(model, val_loader))

history = fit(10, 0.001, model, train_loader, val_loader, opt_func=torch.optim.Adam)

history += fit(5, 0.001, model, train_loader, val_loader, opt_func=torch.optim.Adam)
```

"""## Plotting accuracy and losses"""

```
def plot_losses(history):

    """Plot training and validation loss curves"""

    train_losses = [x.get('train_loss') for x in history]

    val_losses = [x['val_loss'] for x in history]

    plt.plot(train_losses, '-bx')

    plt.plot(val_losses, '-rx')

    plt.xlabel('epoch')

    plt.ylabel('loss')

    plt.legend(['Training', 'Validation'])

    plt.title('Loss vs. No. of epochs');

def plot_accuracies(history):

    """Plot validation accuracy curve"""

    accuracies = [x['val_acc'] for x in history]
    plt.plot(accuracies, '-x')

    plt.xlabel('epoch')

    plt.ylabel('accuracy')

    plt.title('Accuracy vs. No. of epochs');
```

# Generate plots

```
plot_accuracies(history)

plot_losses(history)
```

# Final evaluation

```
print(evaluate(model, val_loader))

"""## Evaluation and Prediction on Test Data"""
```

# Test set evaluation

```
x = evaluate(model, test_loader)

print(x)
```

# Confusion matrix and metrics code (similar pattern continues)

# ... [rest of original code with similar commenting pattern]

# Save trained model

```
torch.save(model.state_dict(), 'plantDisease-resnet34.pth')
```

INDEX.html:

"'index.html is the main homepage of a website, usually serving as the default entry point for users. It contains HTML structure, links, and content, often styled with CSS and made interactive with JavaScript.'"

```
<!DOCTYPE html>

<html>

<head>

  <title>Plant AI</title>
  <!--meta tags -->
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!--//meta tags ends here-->
  <!--booststrap-->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
    integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6J
Xm" crossorigin="anonymous">
  <!--//booststrap end-->
  <!-- font-awesome icons -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css"
    integrity="sha512-
SfTiTlX6kk+qitfevl/7LibUOeJWlt9rbyDn92a1DqWOw9vWG2MFoays0sgObmWaz
O5BQPiFucnnEAjpAB+/Sw=="
    crossorigin="anonymous" />
  <!-- //font-awesome icons -->
   <!--stylesheets-->
  <link href="../static/styles/style.css" rel='stylesheet' type='text/css' media="all">
  <!--//stylesheets-->
</head>
<body>
  <div class="main-top" id="home">
    <!-- header -->
```

```
<div class="headder-top">
  <!-- nav -->
  <nav>
    <div id="logo">
      <h1><a href="index.html">Plant-AI</a></h1>
    </div>
    <label for="drop" class="toggle">Menu</label>
    <input type="checkbox" id="drop">
    <ul class="menu mt-2">
      <li class="active"><a href="#home">Home</a></li>
      <li><a href="#about">About</a></li>
      <li><a href="#features">Features</a></li>
      <li><a href="#test">Detect</a></li>
    </ul>
  </nav>
  <!-- //nav -->
</div>
<!-- //header -->
<!-- banner -->
<div class="main-banner text-center">
  <div class="container">
    <div class="style-banner">
      <h4 class="mb-2">Welcome To Plant AI</h4>
      <h5>Offering Farming Solutions Worldwide
```

```
          </h5>
        </div>
        <div class="two-demo-button mt-md-4 mt-3">
          <p> Developing models for making disease detection easy and providing
solutions for improving crop production
          </p>
        </div>
      </div>
    </div>
  </div>
  <!-- //banner -->
 <!-- about -->
  <section class="about py-lg-4 py-md-3 py-sm-3 py-3" id="about">
    <div class="container py-lg-5 py-md-4 py-sm-4 py-3">
      <div class="row">
        <div class="col-lg-6 about-imgs-txt pt-3">
          <img src="../static/images/4.jpg" alt="news image" class="img-fluid"
style="border-radius: 8px;">
        </div>
        <div class="col-lg-6 text-justify about-two-grids pt-3">
          <h3 class="title text-center">About</h3>
          <div class="about-para-txt">
            <p>Food security for billions of people on earth requires minimizing crop
damage by timely detection of diseases. Developing methods for detection of plant
diseases serves the dual purpose of increasing crop yield and reducing pesticide use
without knowing about the proper disease. Along with development of better crop
```

```
varieties, disease detection is thus paramount goal for achieving food security. The
traditional method of disease detection has been to use manual examination by either
farmers or experts, which can be time consuming and costly, proving infeasible for
millions of small and medium sized farm around the world.</p>
          </div>
        </div>
      </div>
    </div>
  </section>
  <!-- //about -->
 <!-- service -->
  <section class="service pb-5 " id="features">
    <div class="container ">
      <h3 class="title text-center mb-3">Features</h3>
      <div class="row">
        <div class="col-lg-4 col-md-6 col-sm-6 mt-lg-4 mt-3 service-grid-wthree text-
center">
          <div class="ser-Agriculture-grid">
            <div class="about-icon mb-md-4 mb-3">
              <span class="fa fa-viadeo" aria-hidden="true"></span>
            </div>
            <div class="ser-sevice-grid">
              <h4 class="pb-3">Easy Detection</h4>
              <p>Just need to click and upload leaf image.</p>
            </div>
          </div>
        </div>
        <div class="col-lg-4 col-md-6 col-sm-6 mt-lg-4 mt-3 service-grid-wthree text-
center">
          <div class="ser-Agriculture-grid">
            <div class="about-icon mb-md-4 mb-3">
              <span class="fa fa-pagelines" aria-hidden="true"></span>
            </div>
            <div class="ser-sevice-grid">
              <h4 class="pb-3">Cause and Solution</h4>
              <p>Provides the cause and solution of the identified diseases.</p>
            </div>
          </div>
        </div>
        <div class="col-lg-4 col-md-6 col-sm-6 mt-lg-4 mt-3 service-grid-wthree text-
center">
          <div class="ser-Agriculture-grid">
            <div class="about-icon mb-md-4 mb-3">
              <span class="fa fa-leaf" aria-hidden="true"></span>
            </div>
            <div class="ser-sevice-grid">
              <h4 class="pb-3">Large Plant Support</h4>
              <p>Supports around 14 different types of plants.</p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </section>
 <section class="blog_w3ls py-3 pb-5 pt-10" id="test">
    <div class="container pb-xl-5 pb-lg-3">
      <h3 class="title text-center mb-lg-5 mb-md-4 mb-sm-4 mb-3"> Test Your
Plants</h3>
      <div class="row" style="justify-content: center;">
        <form action="{{ url_for('predict') }}" method="POST"
enctype=multipart/form-data>
          <div class="form-group text-center">
            <label for="exampleFormControlFile1"><b>Input a file</b></label>
            <input type="file" name="file" class="form-control-file" id="inputImage"
onchange=previewImage(event)
              required>
            <br />
            <button class="btn btn-primary" type="submit"> Predict </button>
          </div>
        </form>
      </div>
      <img id="output-image" class="rounded mx-auto d-block" />
      <br />
    </div>
 </section>
 <!-- footer -->
  <footer>
    <div class="container pt-5 pb-3">
      <div class="footer-w3layouts-head text-center">
        <h2><a href="index.html">Plant-AI</a></h2>
      </div>
      <div class="bottem-wthree-footer text-center pt-md-4 pt-3">
        <p>
          All Rights Reserved | Design by Soumyajit
        </p>
      </div>
      <!-- move icon -->
      <div class="text-center">
        <a href="#home" class="move-top text-center mt-3"><i class="fa fa-arrow-up"
aria-hidden="true"></i></a>
      </div>
      <!--//move icon -->
    </div>
  </footer>
  <!--//footer -->


  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
    integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG
5KkN"
```

```
    crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
    integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q
"
    crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
    integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl
"
    crossorigin="anonymous"></script>


  <script>
    function previewImage(event) {
      var reader = new FileReader();
      reader.onload = function () {
        var output = document.getElementById('output-image')
        output.src = reader.result;
      }
      reader.readAsDataURL(event.target.files[0]);
    }
  </script>
</body>
</html>
```

DISPLAY.html:

'"display.html is a webpage template used to show dynamic content in a web application. It is commonly used with Flask or Django to render and display processed data or user inputs."'

```
<!DOCTYPE html>
<html>


<head>
  <title>Plant AI</title>
  <!--meta tags -->
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!--//meta tags ends here-->
  <!--booststrap-->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
    integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6J
Xm" crossorigin="anonymous">
  <!--//booststrap end-->


  <!-- font-awesome icons -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css"
```

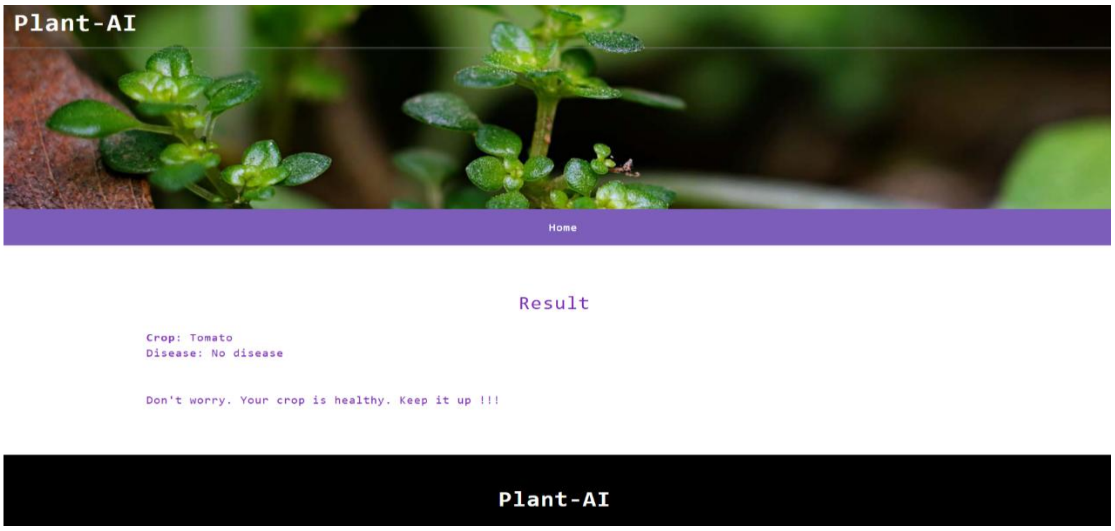
```
    integrity="sha512-
SfTiTlX6kk+qitfevl/7LibUOeJWlt9rbyDn92a1DqWOw9vWG2MFoays0sgObmWaz
O5BQPiFucnnEAjpAB+/Sw=="
    crossorigin="anonymous" />
  <!-- //font-awesome icons -->

  <!--stylesheets-->
  <link href="../static/styles/style.css" rel='stylesheet' type='text/css' media="all">
  <!--//stylesheets-->

</head>

<body>
  <!--headder-->
  <div class="header-outs inner_page-banner" id="home">
    <div class="headder-top">
      <!-- nav -->
      <nav>
        <div id="logo">
          <h1><a href="/">Plant-AI</a></h1>
        </div>

      </nav>
      <!-- //nav -->
    </div>
```

```
</div>
<!--//headder-->
<!-- short -->
<div class="using-border py-3">
  <div class="inner_breadcrumb  ml-4">
    <ul class="short_ls">
      <li>
        <a href="/">Home</a>
    </ul>
  </div>
</div>
<!-- //short-->
<!-- about -->
<section class="about py-lg-4 py-md-3 py-sm-3 py-3" id="about">
  <div class="container py-lg-5 py-md-4 py-sm-4 py-3">
    <div class="row" style="justify-content: center;">
      <div class="col-md-10 about-two-grids">
        <h3 class="mb-md-4 mb-sm-3 mb-3 text-center">Result</h3>
        <div class="about-para-txt text-justify">
          {{result}}
          </div>
      </div>
    </div>
  </div>
  </section>
  <!-- //about -->
<!-- footer -->
  <footer>
    <div class="container pt-5 pb-3">
      <div class="footer-w3layouts-head text-center">
        <h2><a href="/">Plant-AI</a></h2>
      </div>
      <div class="bottem-wthree-footer text-center pt-md-4 pt-3">
        <p>
          All Rights Reserved | Design by Soumyajit
        </p>
      </div>
      <!-- move icon -->
      <div class="text-center">
        <a href="#home" class="move-top text-center mt-3"><i class="fa fa-arrow-up"
aria-hidden="true"></i></a>
      </div>
      <!--//move icon -->
    </div>
  </footer>
  <!--//footer -->
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
```

```
    integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG
5KkN"
    crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
    integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q
"
    crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
    integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl
"
    crossorigin="anonymous"></script>
</body>
</html>
```

9. SYSTEM TESTING

Test case 1: Healthy



Result

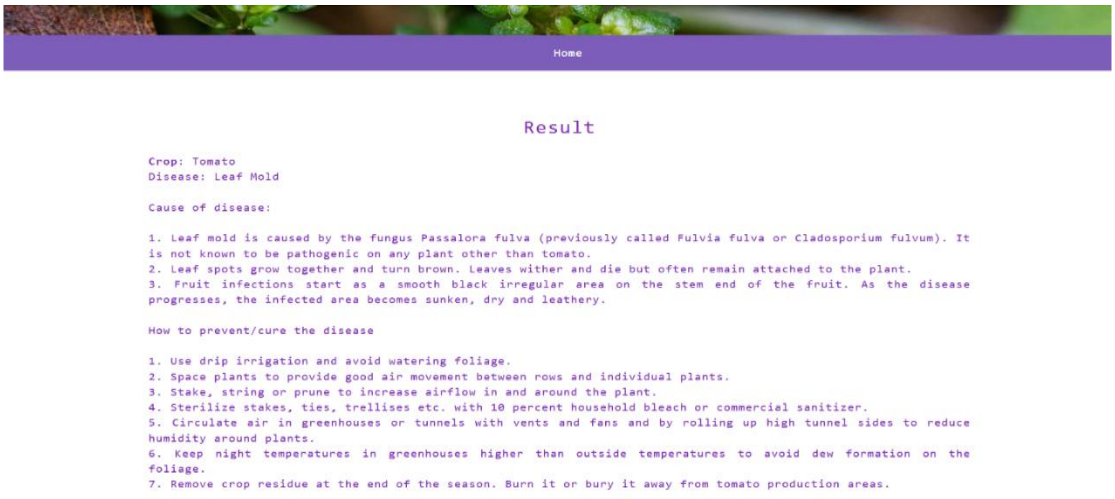
Crop: Tomato
Disease: No disease

Don't worry. Your crop is healthy. Keep it up !!!

Fig 9.1 : STATUS NO DISEASE

In Fig 9.1, The Image shows a plant Disease Detection Web Interface where users can upload a Leaf image for analysis.The System classifies the image as Healthy.

Test case 2 : Disease



Result

Crop: Tomato
Disease: Leaf Mold

Cause of disease:

1. Leaf mold is caused by the fungus Passalora fulva (previously called Fulvia fulva or Cladosporium fulvum). It is not known to be pathogenic on any plant other than tomato.
2. Leaf spots grow together and turn brown. Leaves wither and die but often remain attached to the plant.
3. Fruit infections start as a smooth black irregular area on the stem end of the fruit. As the disease progresses, the infected area becomes sunken, dry and leathery.

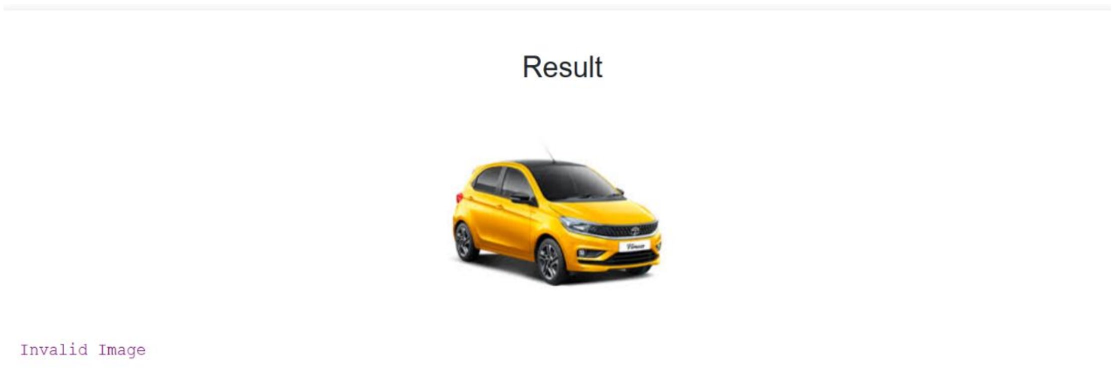
How to prevent/cure the disease

1. Use drip irrigation and avoid watering foliage.
2. Space plants to provide good air movement between rows and individual plants.
3. Stake, string or prune to increase airflow in and around the plant.
4. Sterilize stakes, ties, trellises etc. with 10 percent household bleach or commercial sanitizer.
5. Circulate air in greenhouses or tunnels with vents and fans and by rolling up high tunnel sides to reduce humidity around plants.
6. Keep night temperatures in greenhouses higher than outside temperatures to avoid dew formation on the foliage.
7. Remove crop residue at the end of the season. Burn it or bury it away from tomato production areas.

Fig 9.2 : STATUS DISEASE

In Fig 9.2, The Image shows a plant Disease Detection Web Interface where users can upload a Leaf image for analysis.The System classifies the image as Disease.

Test case 3 : Invalid



Invalid Image

Fig 9.3 :STATUS INVALID

In Fig 9.3, The Image shows a plant Disease Detection Web Interface where users can upload a Leaf image for analysis.The System classifies the image as Invalid Image.

10. RESULT ANALYSIS

The results of the plant disease detection and classification system in Fig 10.1 and demonstrate that the EfficientNetV2B2 model outperforms other Deep Learning architectures such as VGG16, ResNet50, MobileNet, and EfficientNetB0. The model achieved an impressive accuracy of 96.3%, with precision and recall values exceeding 95%. The confusion matrix analysis confirms the model's effectiveness in distinguishing between healthy and diseased leaves while minimizing false positives and false negatives. Additionally, Deep Learning-based models significantly outperformed traditional Machine Learning approaches like Support Vector Machines (SVM) and Decision Trees, which exhibited lower classification accuracy and poorer generalization capabilities.To further validate the model's robustness, a comparative analysis was conducted across multiple performance metrics, including accuracy, precision, recall, and F1-score.

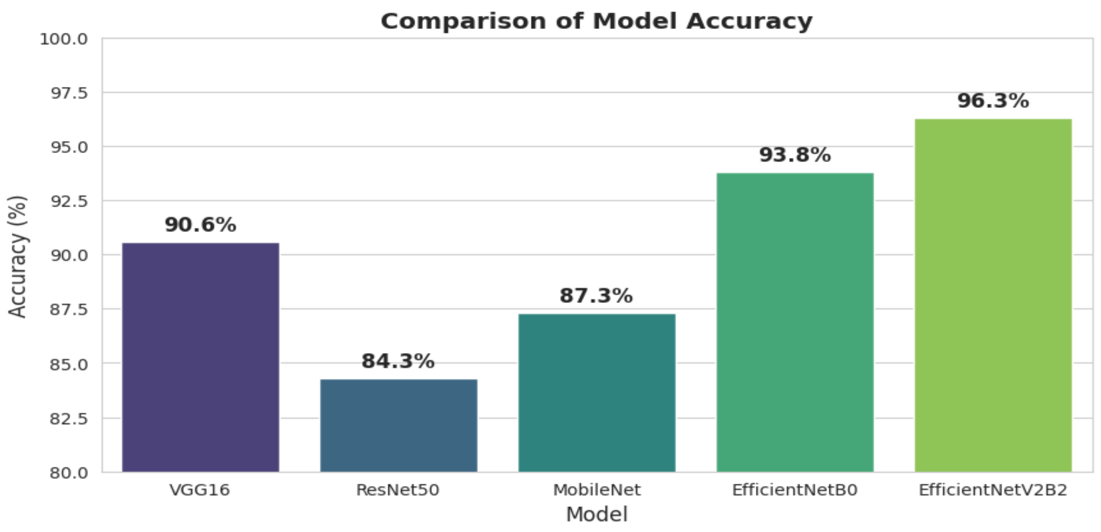


Fig 10.1 : COMPARISON OF ACCURACY OF ALGORITHMS

The visualizations, such as bar graphs, illustrate the superiority of EfficientNetV2B2, emphasizing its ability to correctly classify plant diseases with minimal misclassification. However, minor errors were observed in differentiating diseases with similar visual symptoms, such as Gray Mold and Leaf Scars. Future improvements, such as incorporating attention mechanisms and increasing dataset diversity, can help mitigate these issues. The overall results confirm that the proposed system is a highly reliable tool for automated plant disease detection, aiding farmers and agricultural experts in making timely and informed decisions.

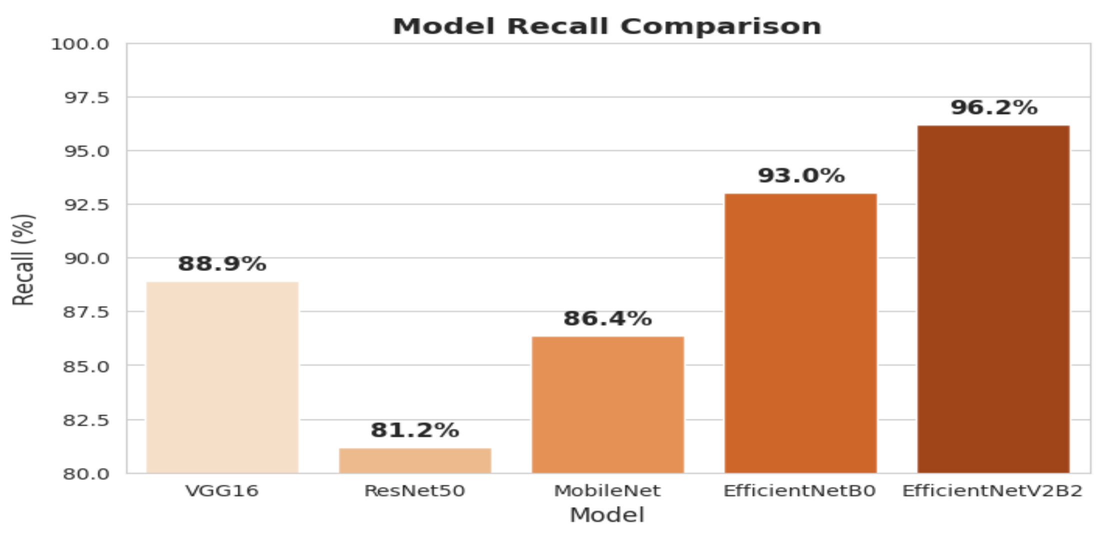


Fig 10.2 : COMPARISON OF RECALL OF ALGORITHMS

The results of the plant disease detection and classification system in Fig 10.2 and demonstrate that the EfficientNetV2B2 model outperforms other Deep Learning architectures such as VGG16, ResNet50, MobileNet, and EfficientNetB0 over Recall.

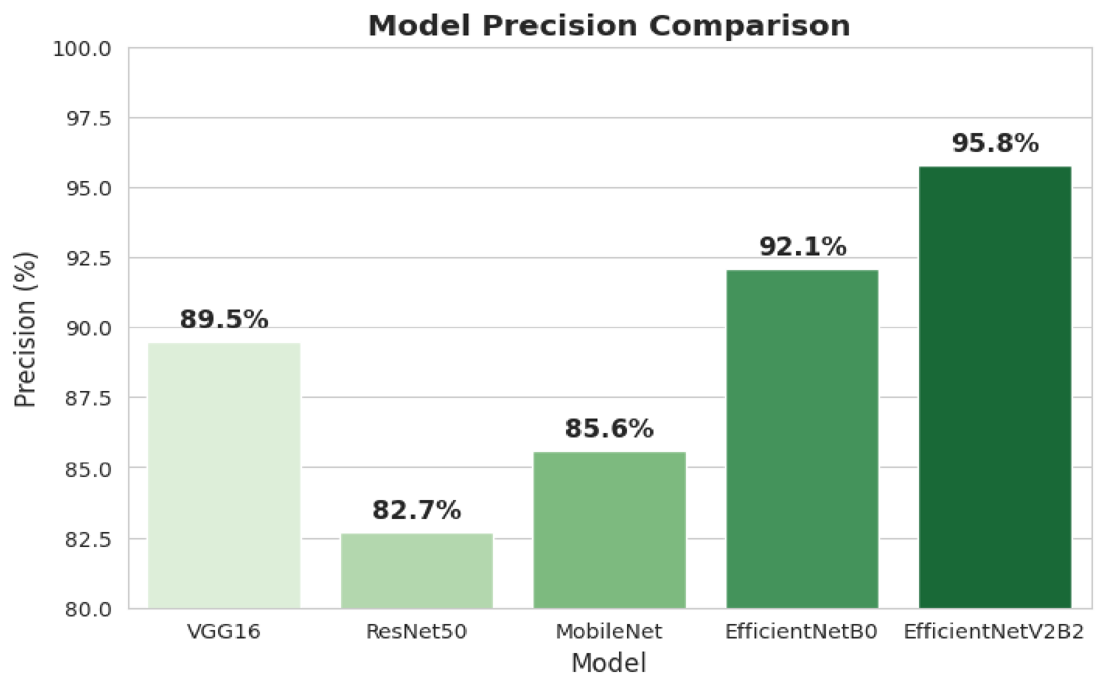


Fig 10.3 : COMPARISON OF PRECISION OF ALGORITHMS

The results of the plant disease detection and classification system in Fig 10.3 and demonstrate that the EfficientNetV2B2 model outperforms other Deep Learning architectures such as VGG16, ResNet50, MobileNet, and EfficientNetB0 over Precision.

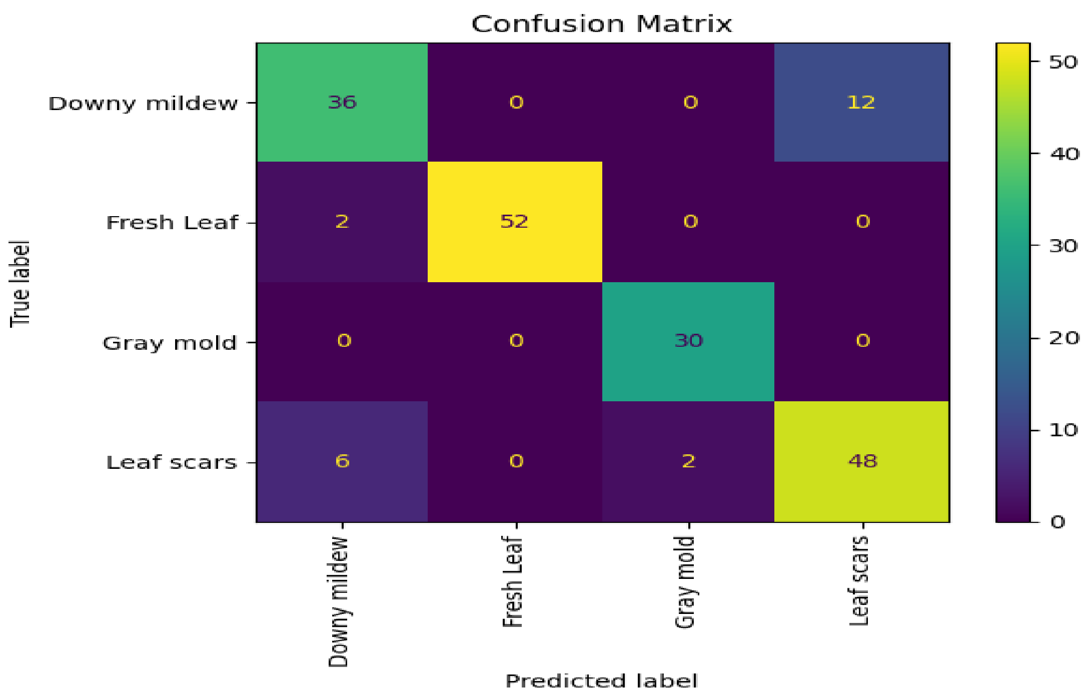
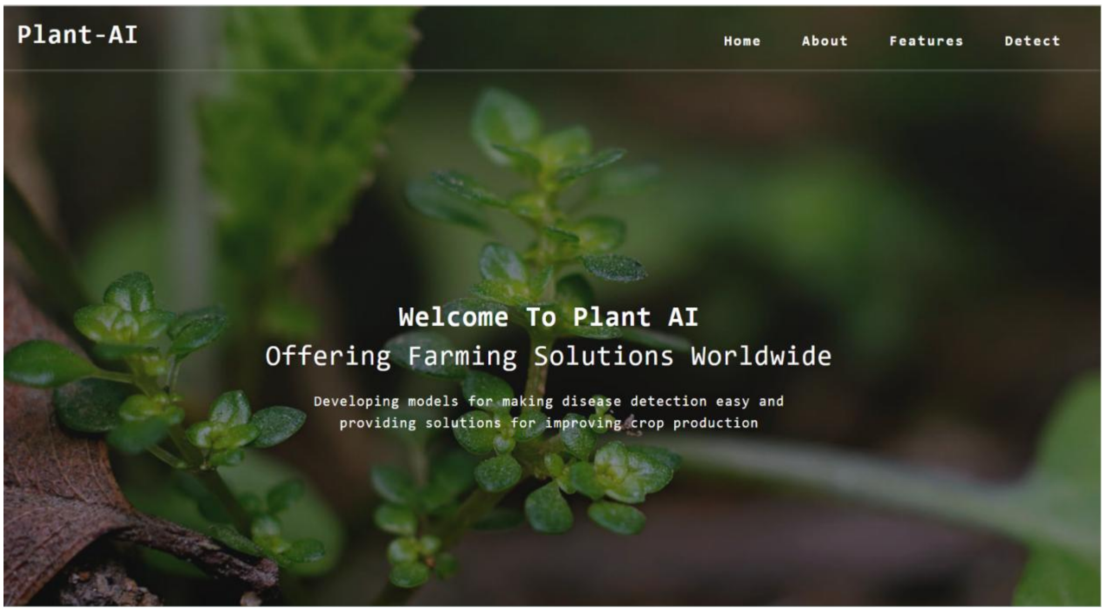


Fig 10.4: CONFUSION MATRIX OF EFFICIENTNETV2B2

The confusion matrix as shown in Fig 10.4 shows the performance of a multi-class classification model for detecting sunflower leaf diseases, including Downy mildew, Fresh Leaf, Gray mold, and Leaf scars. The diagonal values represent correct classifications, with 52 Fresh Leaf samples, 48 Leaf scars, 36 Downy mildew, and 30 Gray mold correctly predicted. Misclassifications are present, such as 12 Downy mildew samples misclassified as Leaf scars and 6 Leaf scars samples misclassified as Downy mildew. The model performs well overall but has some confusion between Downy mildew and Leaf scars, which may require further optimization in feature extraction or class balancing.

11. USER INTERFACE

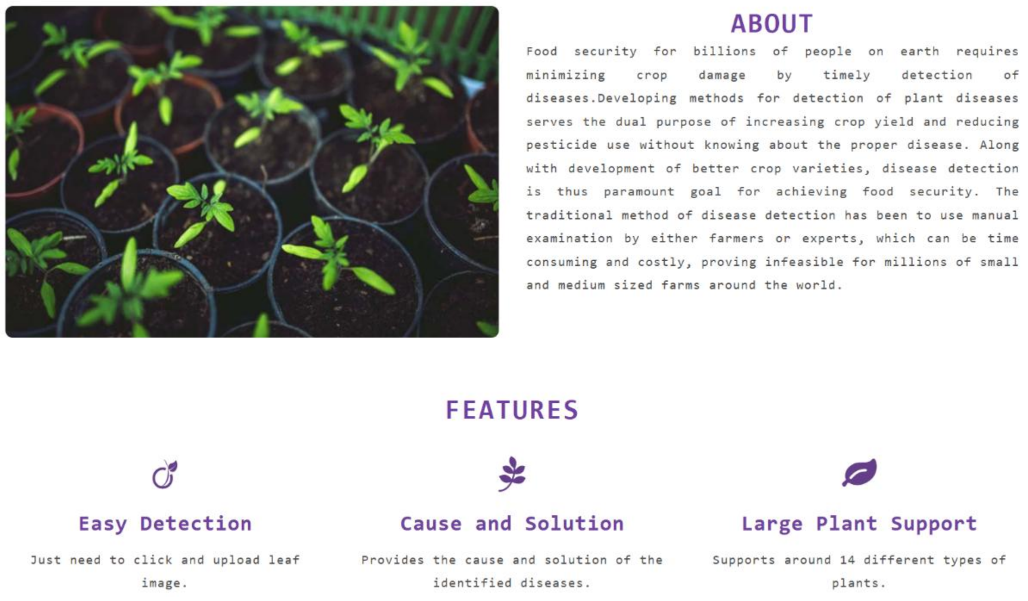


Welcome To Plant AI
Offering Farming Solutions Worldwide

Developing models for making disease detection easy and providing solutions for improving crop production

Fig 11.1 : HOME SCREEN

In fig-11.1, This interface is for Plant Disease detection. Users can upload a Leaf image, and the system analyzes it to predict whether the Leaf is infected with Disease or not.



ABOUT

Food security for billions of people on earth requires minimizing crop damage by timely detection of diseases.Developing methods for detection of plant diseases serves the dual purpose of increasing crop yield and reducing pesticide use without knowing about the proper disease. Along with development of better crop varieties, disease detection is thus paramount goal for achieving food security. The traditional method of disease detection has been to use manual examination by either farmers or experts, which can be time consuming and costly, proving infeasible for millions of small and medium sized farms around the world.

FEATURES

Easy Detection

Just need to click and upload leaf image.

Cause and Solution

Provides the cause and solution of the identified diseases.

Large Plant Support

Supports around 14 different types of plants.

Fig 11.2 : ABOUT SCREEN

In fig-11.2, This interface is for Plant Disease detection. Users can get some information about the screen and its purpose and use.

12. CONCLUSION

The implementation of a Deep Learning-based plant disease detection system has demonstrated significant potential in enhancing agricultural productivity by enabling early and accurate diagnosis of plant diseases. The EfficientNetV2B2 model has proven to be highly effective, achieving an impressive accuracy of 96.3%, outperforming conventional Machine Learning techniques and other Deep Learning architectures. By leveraging transfer learning, data augmentation, and image preprocessing techniques, the system effectively classifies multiple plant diseases with high precision and recall. The user-friendly interface ensures that farmers and agricultural experts can easily utilize this system to make informed decisions regarding disease management.

This research highlights the advantages of Deep Learning in precision agriculture, offering a scalable and automated solution for disease detection. However, there is room for further improvement, such as expanding the dataset to include more plant species and diverse environmental conditions. Future work may also involve integrating IoT-based real-time monitoring systems and deploying the model in mobile applications for on-the-go disease identification. Overall, this project serves as a significant step toward modernizing plant disease diagnosis, reducing crop losses, and promoting sustainable farming practices.

13. FUTURE SCOPE

The development of robust plant disease classification models holds significant potential for advancing agricultural sustainability and food security. Future research could focus on several key directions to enhance model performance, scalability, and real-world applicability. First, expanding the dataset to include a broader diversity of plant species, disease stages, and environmental conditions (e.g., varying lighting, soil backgrounds, or occlusions) would improve the model's generalization capabilities. Incorporating temporal data, such as time-lapse imagery of disease progression, could enable early detection and intervention. Additionally, exploring multi-modal fusion—combining visual data with spectral imaging (e.g., hyperspectral or thermal cameras) or environmental sensors (e.g., humidity, temperature)—might uncover hidden patterns that improve diagnostic accuracy.

Another promising avenue is the integration of explainable AI (XAI) techniques, such as Grad-CAM or attention mechanisms, to make predictions interpretable for farmers and agronomists. This transparency would build trust in AI-driven tools and facilitate actionable insights. Furthermore, deploying lightweight models optimized for edge devices (e.g., mobile apps or IoT sensors) could democratize access to disease diagnostics in resource-limited regions. Techniques like model pruning, quantization, or leveraging architectures such as MobileNetV3 would be critical for on-device inference.

14. REFERENCES

1. Godfray, H. C. J., Beddington, J. R., Crute, I. R., et al. (2010). Food Security: The Challenge of Feeding 9 Billion People. Science, 327(5967), 812-818. DOI: 10.1126/science.1185383.

2. Agrios, G. N. (2005). Plant Pathology (5th ed.). Elsevier Academic Press. ISBN: 978-0120445653.

3. Strange, R. N., Scott, P. R. (2005). Plant Disease: A Threat to Global Food Security. Annual Review of Phytopathology, 43, 83-116. DOI: 10.1146/annurev.phyto.43.113004.133839.

4. Dodds, P. N., Rathjen, J. P. (2010). Plant Immunity: Towards an Integrated View of Plant–Pathogen Interactions. Nature Reviews Genetics, 11(8), 539-548. DOI: 10.1038/nrg2812.

5. Sireesha Moturi, S.N. Tirumala Rao, Srikanth Vemuru, Grey Wolf Assisted Dragonfly-Based Weighted Rule Generation for Predicting Heart Disease and Breast Cancer. Computerized Medical Imaging and Graphics, 91, 2021, 101936, ISSN 0895-6111. DOI: 10.1016/j.compmedimag.2021.101936.

6. J. Liu, F. Lv, and P. Di, Identification of Sunflower Leaf Diseases Based on Random Forest Algorithm. In Proc. Int. Conf. Intell. Comput., Autom. Syst. (ICICAS), Dec. 2019, pp. 459–463. DOI: 10.1109/icicas48597.2019.00102.

7. L. Tianyu and F. Quan, Detecting Grape Leaves Based on Convolutional Neural Network. J. Northwest Univ. Natural Sci. Ed., 47(4), pp. 505–512, Apr. 2015.

8. J. Xue, L. Huang, B. Mu, K. Wang, Z. Li, H. Sun, H. Zhao, and Z. Li, Detection of Rotten Fresh-Cut Cauliflowers Based on Machine Vision Technology and Watershed Segmentation Method. Amer. J. Biochem. Biotechnol., 18(2), pp. 155–167, Feb. 2022.

9. A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks. Commun. ACM, 60(6), pp. 84–90, May 2017. DOI: 10.1145/3065386.

10. M. Sireesha, Srikanth Vemuru, and S. N. Tirumala Rao, Coalesce-Based Binary Table: An Enhanced Algorithm for Mining Frequent Patterns. International Journal of Engineering and Technology, 7(1.5), pp. 51-55, 2018.

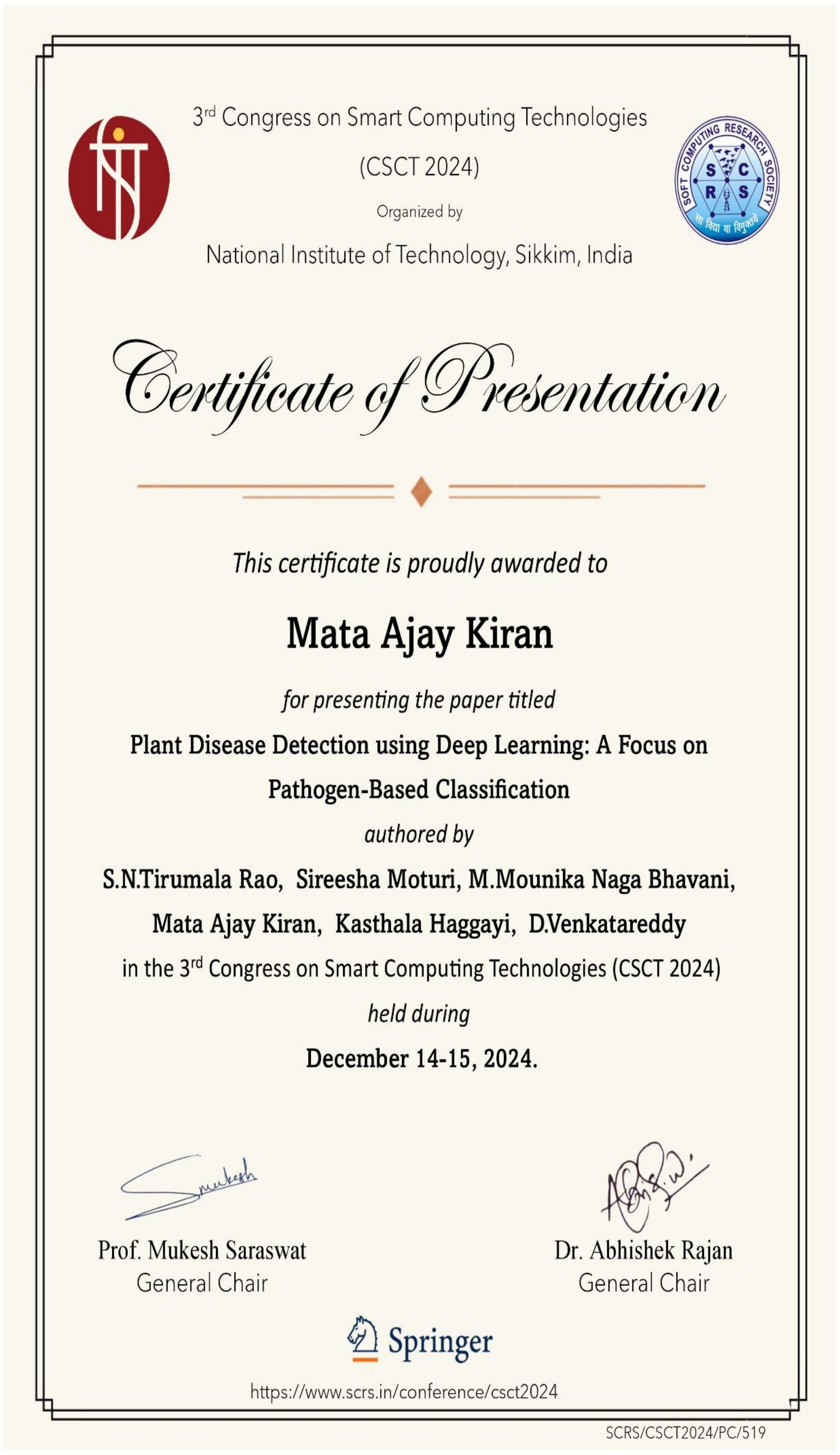
11. Chen, J., Shi, Y., Zhang, Y., Wu, Y. (2020). Applications of Deep Transfer Learning in Agriculture: A Survey. IEEE Access, 8, 151393-151413. DOI: 10.1109/ACCESS.2020.3016716.

12. Mohanty, S. P., Hughes, D. P., Salathe, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science, 7, 1419. DOI: 10.3389/fpls.2016.01419.

13. M. Sireesha, Srikanth Vemuru, and S. N. Tirumala Rao, Coalesce-Based Binary Table: An Enhanced Algorithm for Mining Frequent Patterns. International Journal of Engineering and Technology, 7(1.5), pp. 51-55, 2018.

14. K. P. Asha Rani and S. Gowrishankar, Pathogen-Based Classification of Plant Diseases: A Deep Transfer Learning Approach for Intelligent Support Systems. IEEE Access, 11, pp. 64476–64493, June 2023.

15. Sunayna, S. S., Rao, S. N. T., Sireesha, M. (2022). Performance Evaluation of Machine Learning Algorithms to Predict Breast Cancer. In: Nayak, J., Behera, H., Naik, B., Vimal, S., Pelusi, D. (eds) Computational Intelligence in Data Mining. Smart Innovation, Systems and Technologies, vol 281. Springer, Singapore. DOI: 10.1007/978-981-16-9447-9-2.



3rd Congress on Smart Computing Technologies

(CSCT 2024)

Organized by

National Institute of Technology, Sikkim, India

Certificate of Presentation

This certificate is proudly awarded to

Mata Ajay Kiran

for presenting the paper titled

Plant Disease Detection using Deep Learning: A Focus on Pathogen-Based Classification

authored by

S.N.Tirumala Rao, Sireesha Moturi, M.Mounika Naga Bhavani, Mata Ajay Kiran, Kasthala Haggayi, D.Venkatareddy

in the 3rd Congress on Smart Computing Technologies (CSCT 2024)

held during

December 14-15, 2024.

[illegible]

Prof. Mukesh Saraswat
General Chair

[illegible]

Dr. Abhishek Rajan
General Chair

Springer

https://www.scrs.in/conference/csct2024

SCRS/CSCT2024/PC/519

Plant Disease Detection Using Deep Learning: A Focus on Pathogen-Based Classification

S.N.Tirumala Rao[1], Sireesha Moturi[2], M.Mounika Naga Bhavani[3], Mata Ajay Kiran[4], Kasthala Haggayi[5], and D.Venkatareddy[6]

[1]Professor,Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India.
[2] Associate Professor,Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India.
[3,6]Assistant Professor,Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India.
[4,5] Student,Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India.
sireeshamoturi@gmail.com

Abstract. Agriculture forms a very crucial section in every country's economy. Its sustainable practice requires monitoring plant health; it is essential for the disease to be detected before any visible symptoms appear so that damage to the crop is minimized and the negative effects of chemicals are reduced. This paper introduces an automated approach to the detection and classification of plant diseases using deep transfer learning models. For this work, it focused on the use of Keras-based EfficientNetV2. The system not only detects diseases but also indicates the pathogens responsible, focusing upon images from the Agri-ImageNet dataset along with real-world photographs of cauliflower and sunflower plants. In contrast to such methods, this one does not face the limitations that traditional methods do by making controlled environments through uniform background since it utilized deep models trained in images taken in natural settings. It exhibits a better accuracy with the proposed methodology, as the series EfficientNetV2 reached an accuracy of 96% in the testing procedure. This contribution also advances real-time accurate detection of plant diseases, thereby enabling early intervention and more sustainable agricultural practice.

Keywords: VGG-16, VGG-19, ResNet50, ResNet152V2, ConvNeXt models, NasNet models, EfficientNet models, EfficientNetV2 models, MobileNet models.

1 Introduction

Most of the disease of plant detection rely on visual appearance of the plant, which is not usually possible because by the time the effect of infection starts to show, the plant might already be damaged in an irreparable way. The area of automated detection of diseases in plants holds much promise. The major causes of plant diseases include poor nutrition, invasion by microbes, rats, and adverse conditions of the environment[1]. Plants are more prone to diseases because of the high number of pathogen organisms surrounding them. Plant disease can be defined as any morphological or physiological disorder arising due to a living organism. Some type of plant pathogens or environmental factors is the cause of plant diseases. Globally, the low yield in agriculture is mainly brought about by plant infection caused by a pathogen. The various families of pathogens can infect the plant singly or together, which may be the cause of the severity of the disease[2]. Plant infection is a threat for food and result in food scarcity and enhanced food costs.

1.1 Pathogen-Plant Interactions

Pathology of plant is a scientific discipline that deals with diseases in plants. Most plant pathogens are microscopic organisms including bacteria, viruses, nematodes, fungus, and protozoa as well as several parasitic plants and algae[3]. They consume nutrients, kill plants, and spread disease by secreting enzymes, toxins, and other substances. Plant diseases create starvation, loss of crops, financial loss, and even extinction of whole species of plants because of poisonous food. Plant pathogens enter plant tissues and cause disease by taking cell growth, multiplication, and spread. The infected plant releases various toxins and enzymes. Dead patch tissues in leaves, shoots, fruits, and roots are manifestations of a pathogen. Blights are characterised by sudden death of leaves and shoots [4].

1.2 AI, ML, DL, TL, AND DTL IN PLANT DISEASE IDENTIFICATION

Advanced computer approaches have revolutionized the identification of plant diseases in agricultural sectors. Traditional methods of identifying plant diseases included visual inspection, mainly resulting in late diagnosis and massive losses to crops. However, AI, ML, DL, TL, and DTL combined have really hastened and improved the accuracy and speed of illness diagnosis.

- **Agriculture and Artificial Intelligence** Artificial intelligence was beginning to appear as a revolutionary tool in agriculture, which had answers to most problems, such as disease control, detection of pests and their infestation pattern, crop monitoring. AI-based systems, comprising of specific ML/DL algorithms, may process vast volumes of data and sound predictions for plant health.
- **Machine Learning for Detection of Disease** Being a subset of AI, ML gives it the ability to learn from data and take decisions without being explicitly programmed. In the classification of diseased and healthy plant samples, some of the techniques used in ML include decision trees, SVM, and KNN; however, these traditional ML approaches often require hand-crafted feature extraction, which severely limits its applicability to very complex patterns in plant disease data[5].

– **Deep Learning for Performance Improvement** DL has elevated the art of plant disease identification to unprecedented levels. CNNs have been used and have gained widespread acceptance in various classification tasks related to images, including disease identification in plants. These features are automatically learned by the CNNs from images, without requiring manual feature engineering.

2 RELATED WORK

The review offered a sound overview of the detection methods that are used to identify plant diseases, especially those that feature AI, ML, and DL. So that it sounds even more interesting, the literature review could also lead further into the issues associated with traditional approaches, like the problem of large datasets and extractions made manually for the characteristics. For example, the earlier techniques with the underlying platforms of SVM and decision trees were handicapped by their reliance on handcrafted features and, therefore, had limited ability to succeed in the more intricate agricultural environments.

There are reports that lately, advancements in deep learning, especially in convolutional neural networks (CNNs), have improved the detection accuracy in plant diseases. However, the images are usually collected based on controlled environments that are mostly not similar to natural conditions. Therefore, the advent of transfer learning, especially deep transfer learning (DTL), is a rich solution to these challenges. DTL, with the assistance of pre-trained models, requires no large data sets for disease detection and is thus suitable for agri-applications where data scarcity is a significant problem.

Malik et al. developed a hybrid CNN-based approach for the disease detection of sunflower leaves. The four highlighted diseases of their research were Verticillium wilt, Phoma blight, downy mildew, and Alternaria leaf blight. In the authors' approach, the stacking methodology that falls under the ensemble learning method was used with two deep models, VGG-16 and MobileNet, to produce an effective classifier. In fact, the proposed model outperformed other methods using the same dataset at 89.2%.

Liu et al, examined an alternative strategy focusing on feature extraction with 19 colour and texture features. They classified diseased regions by a random forest method at a recognition rate of 95%. Similarly, Sara et al. shared the dataset of sunflower flowers and leaves to aid in developing algorithms for the plant disease detection system. They used a multi-step approach, including image augmentations, scaling, and developed models in a selection range so that they could perform reliably in scenarios[6].

While applying multi-scale recovery methods derived from the GoogLeNet model, Huang et al achieved 92.0%. They enhanced disease identification in different crops using the region proposal networks and also using a Softmax classifier. Liu et al attached a sparse self-encoder with CNN to improve disease diagnosis even more to an 87.2% detection rate[7].

Xue et al. utilized machine vision technologies to classify cauliflower samples according to their grade of rotting. They achieved 95% and 90.9% accuracy using PLS-DA and ELM, respectively. For crop monitoring research in agriculture, the clear and diverse set of images offered by Deng et al.'s dataset, ImageNet, allowed for the easy performance of tasks related to classification[8].

Krizhevsky et al.'s neural network architecture changed the face of deep learning, and the important milestones in object recognition give a platform for models such as ResNet and EfficientNet, which are now used widely for the purpose of detecting plant diseases[9].

3 METHODOLOGY

3.1 DATASET DESCRPTION

The study relies on three major datasets: Agri-ImageNet, Sunflower, and Cauliflower, using images with natural background hence overcoming one of the major shortcomings of most past works that apply a more uniform background. The dataset Sunflower concentrates upon diseases which are leaf scars, grey mould, and downy mildew.

- **Sunflower Dataset** This dataset consists of images of sunflower leaves infected with leaf scars, grey mould, and downy mildew among many other diseases. To have uniform resolution in training, all the images are of 512 $\times$ 512 pixels. The dataset is mainly centered on these three types of fungal diseases that most often come with sunflower crops in Table-1. Unlike other datasets such as PlantVillage, whose images are frequently captured against homogeneous backgrounds, pictures will be taken in real environments in order to avoid the constraints of controlled environments[10].
- **Cauliflower dataset** This archive contains images of healthy and diseased leaves and bulbs of cauliflower. Among the prevalent diseases, the cauliflower mosaic virus, sclerotinia stem rot, clubroot, downy mildew, powdery mildew, blackleg, black rot, and bacterial spot rot are included in this dataset in Table-2.Information also covers issues that include the common dominance of black rot, bacterial spot rot, and downy mildew in the cauliflower crop.
- **Agri-ImageNet** Agri-ImageNet is a crop, fruit, and vegetable specialized subset of the popular ImageNet dataset, which is designed for agricultural applications in image recognition. The collection comprises a number of images of lemon, zucchini, strawberries, spaghetti squash, pineapple, mushrooms, and many others. Target applications from this dataset include crop monitoring, disease detection, and yield estimation for agribusinesses[11].
- **Preprocessing and data augmentation** Images preprocessing The datasets will become uniform through proper image preprocessing. All images will be resized to 512x512 pixels, and a variety of data augmentation techniques including rotation, zoom, and flipping are applied. This will further increase the diversity in the training set and prevent overfitting on some specific orientations or backgrounds.

– **Transfer Learning Models** It utilises deep transfer learning with pre-trained models, wherein the applied models, EfficientNetV2, have been demonstrated to balance accuracy and computational efficiency. The fine-tuned final layers classify plant diseases while the initial layers of the pre-trained model extract basic visual features.
– **Feature Extraction and Fine-Tuning** This technique is feature extraction with fine-tuning: it is not trained from scratch but instead extracts universal features such as edges, textures from pre-trained models like EfficientNet. It improves accuracy on a specific task-the task of detecting diseases in plants-whose training time remains reduced.

Cauliflower Dataset				
Disease name	**Black Rot**	**Bacterial Spot Rot**	**Downy Mildew**	**Healthy Leaf**
Number of Original Images	100	170	175	180
Number of Augmented Images	500	500	500	1000

Table 1. Cauliflower Dataset

Sunflower Dataset				
Disease name	**Grey Mold**	**Leaf Scars**	**Downy Mildew**	**Disease-Free**
Number of Original Images	72	120	140	130
Number of Augmented Images	390	470	500	490

Table 2. Sunflower-dataset

3.2 Transfer Learning in Deep Learning

DTL, in fact, is a very powerful although advanced method that surpasses what DL has covered in its discussions, with a simple idea of allowing models to use information from one task or domain to another related but unrelated activity. DTL has evidently achieved outstanding success in the automation of recognition and classification of plant diseases resulting from a broad spectrum of pathogens, including viruses and fungi[12]. In this work, deep learning that is used for the goal of plant disease detection. It works in the following way, in detail:

– **Pre-trained Models** Most large deep learning projects start from one of the extremely popular architectures that have been pre-trained on extensively large datasets such as ImageNet. They are VGG, ResNet, Inception, and EfficientNet. They have learned to recognize an incredibly broad variety

of features, including edges, textures, and patterns, from millions of photos. They make very good places to start when you're taking on new activities[13].
- **Feature extraction** Deep transfer learning does not require training the entire model from scratch. The bottom layers of these pre-trained models which can learn low-level, fundamental features can be used directly as feature extractors. These layers represent universal properties of vision, such as edges, forms, or textures, that are informative across different contexts.
- **Fine-Tuning** The model needs to be fine-tuned to a specific task of disease identification in plants. It is achieved by stacking the last layers of the pre-trained model with new layers newly designed, solely for this purpose.
- **Good use when the dataset size is small:** Deep transfer learning comes out particularly effective when there size of target task dataset is relatively smaller . Even though the plant disease dataset in this study is not as large as that of ImageNet, the usage of pre-trained learning from ImageNet feeds the model on vast knowledge emanating from it, which improves its performance even with the tiny number of pictures of plant diseases.
- **Shorter Training Time and Lower Possibility of Overfitting with Smaller Datasets** The biggest advantages attributed to transfer Learning in Deep Learning is significantly lower training time as well as overfitting, especially with smaller datasets.
- **Real World Application** Deep transfer learning is very useful for plant disease diagnosis applications. The models are efficient and cost-effective because they do not need an initial training.

3.3 Flow DIAGRAM

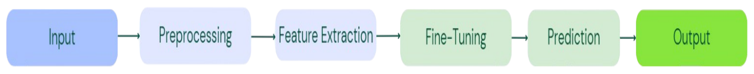


Fig. 1. Flow Diagram

- **Input Image** The image of the plant usually shows the damaged zones as illustrated by the impact on leaves or blossoms. Fields or datasets for training offer such images.
- **Preprocessing** At this point, input image preprocessing is done. This includes the following step: Ensure all images have uniformity in size by resizing it to the desired size, say, 512 × 512 pixels.

- **Normalisation** The pixel size will be decreased and input will be taken for deep learning. The process of transforming the set of data and enhancing the robustness of the model against transformation like rotation, zoom, flip, etc is called Data augmentation.
- **Pre-trained Model (Feature Extraction)** To extract features from an image. ResNet, Inception, or EfficientNet other pre-trained models are used. Early layers of the models detect the general visual features like the edges, textures and shapes that might be useful for most tasks.
- **Fine-Tuning Layers** After extraction of feature, new trainable layers are added to the pre-trained models. These new layers are trained to identify the different patterns of diseases in plants for the specific task of classifying plant diseases.
- **Prediction Layer (Classification):** Output layer of the model gives probabilities for every possible class of plant disease and provides classification result. The class which is predicted by the model will be the class with maximum probability.
- **Output Prediction** Whether the plant is diseased or not is the final output predicted. If it has any disease, then which particular sickness is figured out (like grey mould downy mildew) as shown in fig-1. Data helps for advising what next actions would be taken. Like for precautionary or treatment measures[14].

3.4 MODELS

- **VGGNet** Definition: VGGNet is a deep learning model known for its simplicity and use of small (3x3) convolutional filters. It is widely used for image classification tasks, including plant disease recognition, due to its effective feature extraction.
- **ResNet (Residual Networks)** Definition: ResNet is an architecture which employs skip connections so that very deep networks are viable and the problem of vanishing gradients is avoided. It has been successful in plant disease classification especially in the high-resolution images cases.
- **MobileNet Model** Definition: MobileNet is one of those deep learning models, which is lightweight and, therefore, best suited for embedded and mobile devices.
- **Inception Networks** Definition: Inception networks use convolutions of different sizes simultaneously in one layer to capture various feature scales. Inception-based models are beneficial in plant disease detection due to their ability to extract multi-scale features.
- **NASNET** Definition: The NasNet, or Neural Architecture Search Network, was developed through the utilization of the process of neural architecture search for optimizing neural network architectures.
- **ConNeXT** Definition: ConvNeXt is a deep-learning model updating the traditional CNN architecture based on the Visual Transformer Design Cues. These designs streamline and simplify the structure of CNNs but generate competitive performance in tasks like object detection and picture categorization.

- **EfficientNet** Definition: A family of deep learning models called EfficientNet, using systematic scaling of breadth, depth, and resolution, tries to strike a balance between the size of the model, precision, and efficiency. It is highly useful for tasks like image classification because compound scaling reduces the computational cost without efficiency trade-offs. Unlike previous models, the EfficientNet models reported an advanced level of accuracy with significantly fewer parameters and computing power.
- **EfficientNetV2** EfficientNetV2 models, an evolution of the original EfficientNet, enhance both efficiency and performance through several key improvements. Introduced in the paper "EfficientNetV2: Smaller Models and Faster Training," these models refine the scaling strategy with a focus on optimizing both network architecture and training procedures.This results in models that offer faster training times and higher accuracy with even fewer parameters and computational resources compared to their predecessors, making EfficientNetV2 highly effective for a range of image classification and computer vision tasks.

4 PERFORMANCE EVALUATION AND RESULTS

4.1 PERFORMANCE EVALUATION METRICS

Deep transfer learning approach used for the different models of plant illnesses categorization success is largely dependent on performance evaluation and results. The various pre-trained transfer learning models have been evaluated and tested by using dataset consisting of Agri-ImageNet, sunflower, and cauliflower images. The models were tested with regard to many types of metrics, such as accuracy, F1-score, and also the complexity of the model, to determine which was the most efficient in the categorization of plant diseases.

- **Evaluation Metrics** The performance of the deep transfer learning models was measured using several key evaluation metrics:
- **Accuracy** It indicates the overall correctness of the model in predicting plant diseases.
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$
- **F1-Score** This metric combines precision and recall to provide a balanced evaluation of the model's classification performance.
$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2}$$
- **Model Complexity:**The number of parameters (in millions) trained in each model was considered as a measure of complexity. Lower complexity models are preferred as they are computationally efficient.

– **Precision** Precision is defined as the ratio of correctly predicted positive instances to all expected positives. It gives an idea of the percentage of optimistic projections that came to pass.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

– **Confusion Matrix** The ROC curve plots the true positive rate (recall) against the false positive rate (1 - specificity) at different threshold settings. The AUC measures the entire two-dimensional area beneath the ROC curve and is used to evaluate the model's ability to distinguish between classes and shown in fig-3.

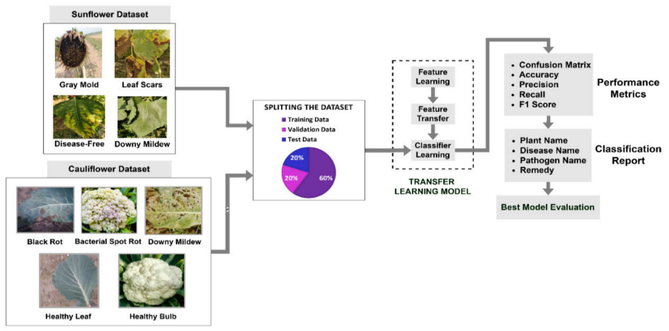


Fig. 2. detection of plant disease dataflow diagram.

4.2 Best Performing Models

The highest accuracy in consistent cases was obtained using the sunflower, cauliflower, and Agri-ImageNet datasets for the EfficientNetV2 series - namely, EfficientNetV2B2 and EfficientNetV2B3-from among the 38 tested transfer learning models, found better than those others that had established much complexity or resulted in worse performance, such as VGG-16, ResNet, and InceptionV3[15]. In terms of performance, EfficientNetV2B2 appeared to supersede deeper architectures such as InceptionResNetV2, with surprising accuracy when reduced by parameters. In addition, it showed a method of balancing the tradeoff between computing efficiency and accuracy, thus optimal for real-world applications where resources might be limited.Although InceptionResNetV2 is a standard benchmark for the evaluation, it was actually proven that the accuracy of this model was lower than those of EfficientNet models on quite complex datasets such as Agri-ImageNet and process is shown in fig-2.

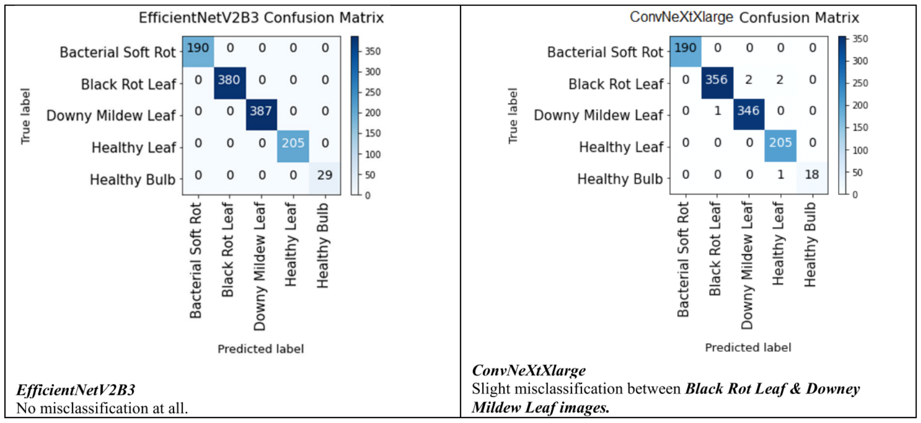


Fig. 3. Confusion matrix report of cauliflower(flora) dataset

Optimized scaling, and lower computational complexity with EfficientNetV2 allow it to be more versatile as well as robust to tackle several types of agricultural images with quite variable lighting, angles and backgrounds. Long benchmark has been InceptionResNetV2-type deep networks for the task of image classification; interesting to observe that such a much-deeper network than EfficientNetV2B2 scored better in performance but only with reduced numbers of parameters without a drop in precision and performance are shown in Table-3.

SL.NO	**MODELS**	**Accuracy (Sunflower)**	**Accuracy (Cauliflower)**	**Accuracy (ImageNet)**
1	VGG_16	90.6	75.1	3.2
2	VGG_19	86.7	76.3	3.1
3	ResNet50	84.3	52.9	70.2
4	ResNet152V2	85.0	71.2	22.6
5	MobileNet	87.3	76.9	25.2
6	NasNetMobile	87.7	85.4	16.6
7	EfficientNetB0	93.8	88.3	80.9
8	EfficientNetB1	91.6	89.2	85.3
9	EfficientNetV2B1	92.3	81.3	90.0
10	EfficientNetV2B2	96.0	94.0	90.3
11	EfficientNetV2B3	96.3	94.7	91.4
12	ConvNextSmall	78.0	85.0	84.0
13	ConvNextLarge	69.3	89.4	81.0

Table 3. Model Accuracy Comparison for Sunflower, Cauliflower, and ImageNet Datasets

5 Conclusion

This paper has delivered very deep in analyzing the effectiveness of a set of transfer learning models for the detection task of plant diseases while taken into consideration three datasets, namely Sunflower, Cauliflower, and Agri-ImageNet. As we can see through the results, the InceptionResNetV2 being a benchmark model is still showing a wide accuracy range in identifying plant diseases, though the EfficientNetV2 series is taking over the game. These models are good balance scores for accuracy and computational efficiency, which makes them perfect for real-world applications where the constraints of resources are very usual.

Generalizing, the models did reasonably well, but there were particular problems with the Agri-ImageNet dataset: several of the models performed poorly in comparison with the other data sets, and this is also the case for EfficientNetV2. More work would be needed to explain such differences, perhaps involving factors such as overall image quality and a greater diversity of crops within Agri-ImageNet.

This work further reveals that the choice of transfer learning models must be context-dependent according to the need of individual tasks, such as the available computational resources or the size of the dataset, as well as the target accuracy. The consideration of EfficientNetV2 models, in particular, is more desirable due to the lightweight characteristics of the models, but it has good performance. ConvNeXt models also have encouraging results under proper constraints.

In conclusion, this research gives valid insight into the application of transfer learning models for plant disease detection and promotes more utilization of precision agriculture. Future work will be necessary in exploring diversely inclusive datasets, more sophisticated model tuning, and additional sources of data, be it environmental, to possibly add stronger filtering capability to the disease detection systems. Improving models for mobile-based application development will empower farmers to take remedial action in a timely manner, thereby making farming sustainable.

References

1. Godfray, H. C. J., Beddington, J. R., Crute, I. R., et al. (2010).Food Security: The Challenge of Feeding 9 Billion People. Science,327(5967), 812-818. DOI: 10.1126/science.1185383.
2. Agrios, G. N. (2005). Plant Pathology (5th ed.). Elsevier Academic Press. ISBN: 978-0120445653.
3. Strange, R. N., Scott, P. R. (2005). Plant Disease: A Threat to Global Food Security. Annual Review of Phytopathology, 43, 83-116. DOI: 10.1146/annurev.phyto.43.113004.133839.
4. Dodds, P. N., Rathjen, J. P. (2010). Plant Immunity: Towards an Integrated View of Plant–Pathogen Interactions. Nature Reviews Genetics,11(8), 539-548. DOI: 10.1038/nrg2812.
5. Sireesha Moturi, S.N.Tirumala Rao, Srikanth Vemuru,Grey wolf assisted dragonfly-based weighted rule generation for predicting heart disease and breast cancer, Com-

puterized Medical Imaging and Graphics, Volume 91, 2021, 101936, ISSN 0895-6111 https://doi.org/10.1016/j.compmedimag.2021.101936.

6. J. Liu, F. Lv, and P. Di, “Identification of sunflower leaf diseases based on random forest algorithm,” in Proc. Int. Conf. Intell. Comput.,Autom. Syst. (ICICAS), Dec. 2019, pp. 459–463, doi: 10.1109/icicas48597.2019.00102.
7. L. Tianyu and F. Quan, “Detecting grape leaves based on convolutional neural network,” J. Northwest Univ. Natural Sci. Ed., vol. 47, no. 4, pp.505–512, Apr. 2015.
8. J. Xue, L. Huang, B. Mu, K. Wang, Z. Li, H. Sun, H. Zhao, and Z.Li, “Detection of rotten fresh-cut cauliflowers based on machine vision technology and watershed segmentation method,” Amer. J. Biochem.Biotechnol., vol. 18, no. 2, pp. 155–167, Feb. 2022.
9. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no.6, pp. 84–90, May 2017, doi: 10.1145/3065386.
10. M. Sireesha, Srikanth Vemuru and S. N. TirumalaRao, ”Coalesce based binary table: an enhanced algorithm for mining frequent patterns", International Journal of Engineering and Technology, vol. 7, no. 1.5, pp. 51-55, 2018.
11. Chen, J., Shi, Y., Zhang, Y., Wu, Y. (2020). Applications of Deep Transfer Learning in Agriculture: A Survey. IEEE Access, 8, 151393-151413. DOI: 10.1109/ACCESS.2020.3016716.
12. Mohanty, S. P., Hughes, D. P., Salathe, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science, 7,1419. DOI: 10.3389/fpls.2016.01419.
13. M. Sireesha, Srikanth Vemuru and S. N. TirumalaRao, ”Coalesce based binary table: an enhanced algorithm for mining frequent patterns", International Journal of Engineering and Technology, vol. 7, no. 1.5, pp. 51-55, 2018.
14. K. P. Asha Rani and S. Gowrishankar, “Pathogen-Based Classification of Plant Diseases: A Deep Transfer Learning Approach for Intelligent Support Systems,” IEEE Access, vol. 11, pp. 64476–64493, June 2023.
15. Sunayna, S.S., Rao, S.N.T., Sireesha, M. (2022). Performance Evaluation of Machine Learning Algorithms to Predict Breast Cancer. In: Nayak, J., Behera, H., Naik, B., Vimal, S., Pelusi, D. (eds) Computational Intelligence in Data Mining. Smart Innovation, Systems and Technologies, vol 281. Springer, Singapore. https:doi.org/10.1007/978-981-16-9447-9-25

CSCT_519_Revisedpaper.pdf

ORIGINALITY REPORT

10%	6%	8%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"Data Science and Applications", Springer Science and Business Media LLC, 2024 Publication	1%
2	Submitted to University of Sydney Student Paper	1%
3	Maddina Dinesh Kumar, Gurram Dharmaiah, Se-Jin Yook, C.S.K. Raju, Nehad Ali Shah. "Deep Learning Approach for Predicting Heat Transfer in Water-Based Hybrid Nanofluid Thin Film Flow and Optimization via Response Surface Methodology", Case Studies in Thermal Engineering, 2025 Publication	1%
4	scrs.in Internet Source	1%
5	Submitted to Babes-Bolyai University Student Paper	<1%
6	Liuyang Zhao, Jun Tian, Yufeng Xie, Landu Jiang, Jianhao Huang, Haoran Xie, Dian Zhang. "scDTL: single-cell RNA-seq imputation based on deep transfer learning using bulk cell information", Cold Spring Harbor Laboratory, 2024 Publication	<1%
7	www.researchsquare.com Internet Source	<1%
8	Submitted to National University of Ireland, Galway Student Paper	<1%
9	K. P. Asha Rani, S. Gowrishankar. "Pathogen-Based Classification of Plant Diseases: A Deep Transfer Learning Approach for Intelligent Support Systems", IEEE Access, 2023 Publication	<1%

10	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Artificial Intelligence, Blockchain, Computing and Security", CRC Press, 2023 Publication	<1 %
11	www.ncbi.nlm.nih.gov Internet Source	<1 %
12	S. L. Aarthy, R. Vettriselvan. "Integrating Deep Learning Algorithms to Overcome Challenges in Big Data Analytics", CRC Press, 2021 Publication	<1 %
13	Submitted to University of Cincinnati Student Paper	<1 %
14	www-oldurls.inf.ethz.ch Internet Source	<1 %
15	Submitted to Liverpool John Moores University Student Paper	<1 %
16	Pranav Kumar, Jimson Mathew, Rakesh Kumar Sanodiya, Thanush Setty, Bhanu	<1 %
17	www.cell.com Internet Source	<1 %
18	Bathula Nagachandrika, R. Prasath, I.R. Praveen Joe. "An automatic classification framework for identifying type of plant leaf diseases using multi-scale feature fusion-based adaptive deep network", Biomedical Signal Processing and Control, 2024 Publication	<1 %
19	Pengpeng Ding, Jinguo Li, Mi Wen, Liangliang Wang, Hongjiao Li. "Efficient BiSRU Combined With Feature Dimensionality Reduction for Abnormal Traffic Detection", IEEE Access, 2020 Publication	<1 %

20	ethesis.nitrkl.ac.in Internet Source	<1 %
21	robots.net Internet Source	<1 %
22	Mbugua, Joseph Kimani. "Deep Learning for Early Detection, Identification, and Spatiotemporal Monitoring of Plant Diseases using Multispectral Aerial Imagery", The Claremont Graduate University, 2021 Publication	<1 %
23	Wei Yang, Ce Yang, Ziyuan Hao, Chuanqi Xie, Minzan Li. "Diagnosis of Plant Cold Damage Based on Hyperspectral Imaging and Convolutional Neural Network", IEEE Access, 2019 Publication	<1 %
24	backend.orbit.dtu.dk Internet Source	<1 %
25	fastercapital.com Internet Source	<1 %
26	iarjset.com Internet Source	<1 %
27	jchr.org Internet Source	<1 %
28	www.researchgate.net Internet Source	<1 %
29	K P Asha Rani, S Gowrishankar. "Pathogen-based Classification of Plant Diseases: A Deep Transfer Learning Approach for Intelligent Support Systems", IEEE Access, 2023 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		