PLANT DISEASE DETECTION

A PROJECT REPORT

Submitted by

AKSHARA T K (TKM23MCA-2009)

to

TKM College of Engineering

Affiliated to

The APJ Abdul Kalam Technological University

In partial fulfilment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATIONS

NOVEMBER 2024

DECLARATION

I undersigned hereby to declare that the project report on **PLANT DISEASE DETECTION**, submitted for partial fulfilment of the requirements for the award of degree of Master of Computer Application of the APJ Abdul Kalam Technological University, Kerala is a Bonafide work doneby me under supervision of **Dr. Nadera Beevi S.** This submission represents my ideas in my own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. I also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data oridea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

KOLLAM AKSHARA T K

11/11/24

DEPARTMENT OF COMPUTER APPLICATIONS

TKM COLLEGE OF ENGINEERING

(Government Aided and Autonomous)

KOLLAM - 5

CERTIFICATE

This is to certify that, the report entitled **Plant Disease Detection** submitted by **AKSHARA T K (TKM23MCA-2009)** to the **APJ Abdul Kalam Technological University** in partial fulfilment of the requirements for the award of the Degree of **Master of Computer Application** is a Bonafide record of the project work carried out by him/her under my/our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

----------------------------------	----------------------------------
Internal Supervisor(s)	Mini Project Co-ordinator

ACKNOWLEDGEMENT

First and foremost, I thank GOD almighty and our parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time andknowledge for the successful completion of my project.

I am extremely grateful to Prof. Natheera Beevi M, Head of the Department, Department of Computer Application, for providing us with best facilities.

I extend my sincere gratitude to my project guide, Dr. Nadera Beevi S., for her invaluable guidance, encouragement and support throughout the course of this project.

I would like to thank my project coordinator Prof. Sheera Shamsu, Department of Computer Applications, who motivated me throughout the project.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study.

I owe thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

AKSHARA T K

ABSTRACT

In modern agriculture, the accurate and timely identification of plant diseases and pests is crucial for maintaining healthy crop yields and ensuring food security. This mini-project aims to develop an advanced plant disease identification system utilizing deep learning techniques, specifically Convolutional Neural Networks (CNN) with architectures such as AlexNet and ResNet. Leveraging the comprehensive PlantVillage dataset, which encompasses a diverse range of crops, the system is trained to detect and recognize various plant diseases and pests with high precision. The models developed in this project demonstrate significant potential as practical tools for early disease detection and advisory services in agriculture, enabling farmers to make informed decisions and implement timely interventions.

The exceptional performance of the CNN models demonstrates their capability to generalize across different crops, making the system versatile and applicable to various agricultural contexts. By providing an automated solution for plant disease and pest identification, this project addresses a significant challenge in large-scale farming, where traditional methods can be labor-intensive and time-consuming. The implementation of such a system can substantially enhance agricultural productivity and sustainability, especially in regions like India, where pest and disease management are critical issues. The success of this project lays a solid foundation for the development of an integrated plant disease identification framework that can be further refined and expanded to support a holistic agricultural advisory system, ultimately contributing to improved crop management and food security.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the evolving world of agricultural technology, plant disease detection using deep learning has emerged as a transformative innovation, enabling the rapid identification of diseases through automated image analysis. This approach empowers farmers, researchers, and agricultural experts to diagnose plant ailments accurately and efficiently, minimizing the impact of disease on crop yields. By leveraging deep learning, plant disease detection transforms traditional diagnostics into a streamlined, data-driven solution accessible to a wide audience, regardless of expertise in plant pathology.

Traditional plant disease identification often requires expert knowledge and hands-on examination, which can be time-intensive and challenging to access, particularly in rural or under-resourced areas. Real-time plant disease detection using deep learning bypasses these limitations by utilizing advanced image classification models to provide fast, accurate predictions. By analyzing leaf images in real time, these systems detect diseases based on complex patterns and visual cues that are difficult to discern manually.

For this project, I use the PlantVillage dataset, which includes 28 plant diseases across 9 crop types, as the foundation for training and evaluating three key deep learning architectures: Convolutional Neural Network (CNN), AlexNet, and ResNet50. After extensive testing, the CNN model achieved the highest accuracy and was selected to power a Flask-based user interface, allowing users to upload leaf images and instantly receive diagnostic results. This dynamic approach to plant disease detection opens new possibilities for scalable, real-time disease monitoring in agriculture.

Through this project, I aim to bridge the gap between technology and agriculture, demonstrating how deep learning can revolutionize plant disease detection with a fast, reliable, and accessible solution that supports sustainable farming practices.

1.1 Existing System

Existing plant disease detection systems have often relied on traditional machine learning models or basic image processing techniques to identify diseases in plants. These systems typically employ classical methods like color analysis, texture extraction, and basic classifiers such as Support Vector Machines (SVM) or Random Forests. While these models have some success, they face significant challenges, especially when dealing with large and diverse datasets. Variations in lighting, plant morphology, and environmental conditions can make it difficult for these systems to consistently identify diseases. Moreover, they often struggle with complex disease symptoms that may not be easily distinguishable from healthy plant parts, leading to misclassifications.

In terms of deep learning, Convolutional Neural Networks (CNNs) have shown improved performance in detecting plant diseases, yet traditional CNN models can be limited by their inability to generalize well to unseen data. These models may also face issues with overfitting, especially when the training dataset is imbalanced or when certain diseases are underrepresented. Furthermore, traditional models often require separate classifiers for each disease type, making them less efficient and harder to deploy at scale in real-world agricultural settings. Handling complex, multi-class classifications with these methods can be resource-intensive and may require significant computational power, which limits their accessibility for broader use in agriculture.

Major drawbacks of currently existing models are:

- **Class Imbalance:** Many datasets contain imbalanced distributions of disease types, where certain diseases are underrepresented. This results in models that are biased and perform poorly on less frequent classes.
- **Dependence on Traditional Machine Learning Techniques:** Existing systems often rely on manual feature extraction, which is both time-consuming and prone to errors. These traditional models have limited performance due to their reliance on handcrafted features and may struggle to generalize well to new, unseen datasets.
- **Poor Robustness In Real World Conditions:** Traditional models are vulnerable to variations in environmental conditions such as lighting, angle, and plant appearance, which can reduce detection accuracy.

- **Resource Intensive:** Separate models are often required for each plant disease, leading to inefficiency and high computational resource demands.
- **Inability to Handle Multi-Class Classification:** Many existing models require separate classifiers for different disease types, making them complex and not scalable for handling multiple diseases simultaneously.

1.2 Proposed System

The proposed system addresses the limitations of traditional plant disease detection by employing deep learning models, such as CNNs, ResNet50, and AlexNet, to accurately classify diseases from plant leaf images. These models offer enhanced accuracy and robustness, especially under varying conditions like lighting and angle.

- **End-to-End Deep Learning:** The system eliminates manual feature extraction, allowing models to learn directly from images, improving efficiency and accuracy.
- **Data Augmentation:** Techniques like rotation and zooming are used to diversify training data, helping to address class imbalance and improve model generalization.
- **Real-World Robustness:** Trained on a diverse dataset, the system can handle environmental variations, providing higher accuracy than traditional models.
- **Multi-Class Classification:** A single model can classify multiple diseases, streamlining deployment and improving scalability.
- **Scalability and Efficiency:** Pre-trained models like ResNet50 offer high performance on large datasets, reducing the need for separate classifiers.
- **Real-Time Detection:** Through a user-friendly interface, users can upload leaf images for instant disease predictions, assisting with quicker diagnostics.

1.3 Objectives

- **Develop a Deep Learning Model for Disease Classification**: Create a Convolutional Neural Network (CNN) architecture specifically designed to classify plant diseases based on leaf images. This model aims to focus solely on disease identification, eliminating the need for separate models for different plant species and simplifying the process.
- **Implement Enhanced Feature Extraction with Pre-trained Models**: Leverage pre-trained models such as ResNet50 for feature extraction to improve the accuracy and generalization ability of the disease detection system. This approach enhances the model's ability to detect diseases across various plant types without the need for extensive training from scratch.
- **Improve Model Performance on Real-World Data**: Apply preprocessing techniques such as resizing, normalization, and data augmentation to address challenges posed by environmental factors like lighting, image quality, and leaf orientation. This will ensure the model is robust and performs effectively on diverse real-world plant images.
- **Create an Accessible User Interface**: Develop a user-friendly, web-based interface where users can easily upload plant leaf images to receive real-time disease predictions. This system will cater to agricultural experts, researchers, and farmers, making disease detection accessible to a broader audience.
- **Achieve High Accuracy and Efficiency**: Optimize the deep learning models to ensure high accuracy in detecting plant diseases while maintaining efficient processing. This will make the system suitable for real-world agricultural applications, where quick diagnosis and minimal computational resources are crucial for large-scale deployment.

CHAPTER 2

LITERATURE REVIEW

A literature survey, also known as a literature review, is a comprehensive study and evaluation of existing research and literature on a specific topic or subject. It involves identifying, analyzing, and synthesizing relevant sources such as books, scholarly articles, and other publications to provide a comprehensive overview of the current state of knowledge on the topic. The purpose of a literature survey is to identify gaps in the existing literature, establish the significance of the research, and provide a theoretical framework for the study. It is commonly conducted as part of the research process in academic and scientific fields.

2.1 Purpose of the Literature Review

- Providing a background to the research problem or question by summarizing existing knowledge on the topic.
- Establishing the context in which the current study fits within the broader academic or research landscape.
- Identifying areas where previous research has left gaps or unanswered questions.
- Highlighting areas where new research can contribute to the existing body of knowledge. Helping to construct a theoretical framework by presenting and analyzing relevant theories and concepts.
- Formulating a clear rationale for the current study based on the shortcomings or limitations found in the existing literature.
- Providing insights into the methodologies used in previous studies, helping researchers make informed decisions about their own research design.
- Summarizing and synthesizing information from various sources to provide a comprehensive overview of the topic.
- Analyzing trends, patterns, and contradictions in the existing literature.
- Ensuring that the proposed research does not duplicate efforts already made by other researchers.Justifying why the current study is necessary despite previous work in the

field. Offering a historical perspective on the development of ideas and theories related to the research topic.

2.2 Related Works

1. The paper *"Toward Generalization of Deep Learning-Based Plant Disease Identification Under Controlled and Field Conditions"* (2023) by Aanis Ahmad, Aly el Gamal, and Dharmendra Saraswat examines the challenges of applying deep learning models to plant disease identification in both controlled and real-world field conditions. The authors compare the performance of several deep learning models, including VGG16, ResNet50, InceptionV3, DenseNet169, and Xception, trained on datasets with controlled backgrounds like PlantVillage and tested on field images. The study highlights the difficulties in generalizing models from controlled environments to real-world conditions, where field images often contain complex backgrounds and environmental variations. Their findings emphasize the need for diverse datasets to improve model robustness and generalization, showing a significant drop in accuracy when models trained on controlled datasets were applied to field conditions.
2. The paper *"A Comprehensive Review on Detection of Plant Disease Using Machine Learning and Deep Learning Approaches"* by C. Jackulin and S. Murugavalli, published in 2022, provides an extensive overview of machine learning (ML) and deep learning (DL) techniques used in plant disease detection. It covers various algorithms and methods, including traditional ML approaches like Support Vector Machines (SVM), Random Forests, and Decision Trees, as well as deep learning methods such as Convolutional Neural Networks (CNNs). The review examines the strengths and limitations of these methods, highlighting their ability to improve accuracy in detecting plant diseases. The paper emphasizes the importance of dataset quality, feature extraction techniques, and model generalization, particularly when deploying models in real-world agricultural environments. Additionally, it discusses challenges like the need for large, annotated datasets and the difficulty in adapting models from controlled lab conditions to diverse field conditions. The authors call for further research into hybrid models and data augmentation techniques to enhance performance and generalization in plant disease detection systems.

3. The paper *"Plant Disease Detection and Classification by Deep Learning: A Review"* by Lili Li, Shujian Zhang, and Bin Wang, published in 2021, provides an in-depth analysis of the application of deep learning techniques, particularly Convolutional Neural Networks (CNNs), in the detection and classification of plant diseases. The review discusses the evolution of plant disease identification, from traditional methods like visual inspection and expert knowledge to more automated, deep learning-based approaches. It highlights the significant improvements in accuracy and efficiency brought about by CNNs in identifying plant diseases from images of plant leaves. The authors examine various datasets used in the field, common preprocessing techniques, and the challenges involved in generalizing models across different environments and plant species. The paper also explores the integration of deep learning models with mobile and IoT technologies, enabling real-time disease detection in the field. Additionally, it addresses the need for large, diverse datasets and suggests directions for future research to improve model robustness and adaptability to real-world agricultural conditions.
4. The paper *"Real-Time Plant Disease Dataset Development and Detection of Plant Disease Using Deep Learning"* (2024) by Diana Susan Joseph, Kaustubh Chakradeo, and Pranav M. Pawar focuses on developing a real-time plant disease detection system through deep learning. The authors highlight the importance of accurate early disease detection for ensuring food security, especially given the global crop loss rate of around 14.1%. To overcome challenges in real-world environments, they developed specialized datasets for rice, wheat, and maize, addressing common bacterial and fungal diseases. The study applies CNN-based models, including Xception, MobileNet, and Inception V3, which achieved impressive accuracies. Notably, a new CNN model was trained on these datasets, achieving testing accuracies of up to 98%. This research aims to enhance plant disease identification, ultimately improving crop management and contributing to global food security.
5. The paper *"A Survey on Using Deep Learning Techniques for Plant Disease Diagnosis and Recommendations for Development of Appropriate Tools"* (2022) by Aanis Ahmad, Dharmendra Saraswat, and Aly El Gamal provides an in-depth review of deep learning techniques used in plant disease detection. The authors analyze the various methods employed for accurate disease classification and diagnosis in plants, including convolutional neural networks (CNNs) and other machine learning models. They also highlight the need for more robust, real-time diagnostic tools to help farmers manage crop health efficiently. The paper

stresses the importance of addressing the challenges of limited datasets and the variation in environmental factors, which complicate disease identification. The authors propose the development of more accessible, scalable, and user-friendly tools to aid farmers in preventing disease outbreaks and improving overall crop yields. They suggest that further research into hybrid models and the use of larger, more diverse datasets could enhance the accuracy and applicability of plant disease detection systems.

6. The paper *"Image-based Plant Disease Detection using Deep Learning"* (2023) by Adesh V. Panchal et al. investigates the application of deep learning, specifically convolutional neural networks (CNNs), for detecting plant diseases through image analysis. The study focuses on automating the plant disease detection process, which traditionally relies on manual inspection and is prone to errors and inefficiency. By leveraging CNNs to classify and identify diseases from plant images, the authors aim to enhance the speed and accuracy of disease diagnosis, critical for crop health management. The research demonstrates that deep learning techniques can overcome many limitations of traditional methods and emphasizes their scalability and potential in real-world applications. Despite showing promising results, the study also points out the need for optimization to address challenges like varying environmental conditions and handling real-time image data for effective disease detection in agricultural practices.

CHAPTER 3

METHODOLOGY

The plant disease detection system leverages deep learning models, particularly Convolutional Neural Networks (CNN), to identify and classify diseases from plant leaf images. The dataset used in this study consists of various plant species and a wide range of diseases. For preprocessing, the images are standardized by resizing and normalizing, ensuring consistency across the dataset. Additionally, data augmentation techniques such as rotation, shifting, and flipping are applied to increase the dataset's diversity and enhance model robustness, mitigating overfitting and improving generalization. The models are trained using well-established deep learning frameworks with an emphasis on minimizing classification errors. Optimizers like Adam are typically employed for efficient model training, and performance is evaluated using metrics like accuracy, precision, and recall. Figure 3.1 provides the block diagram of the project flow, illustrating the stages from data preprocessing to model deployment. The final trained model is deployed in a user-friendly interface that allows for real-time disease detection by simply uploading images of plant leaves. This method aims to provide an automated and accurate solution for early disease detection, helping to improve agricultural productivity and plant health management.

3.1 Block Diagram

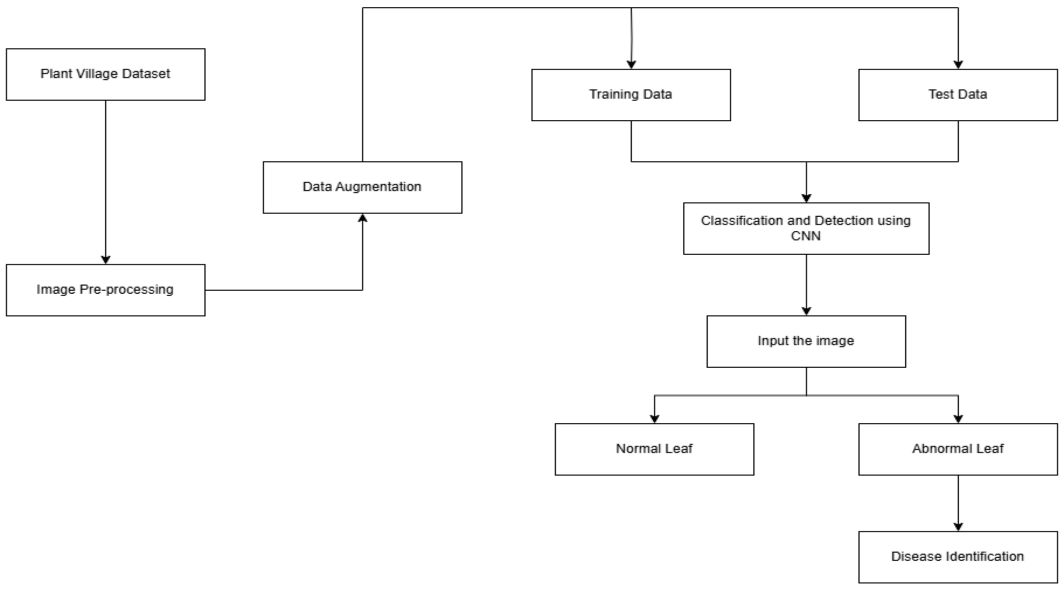
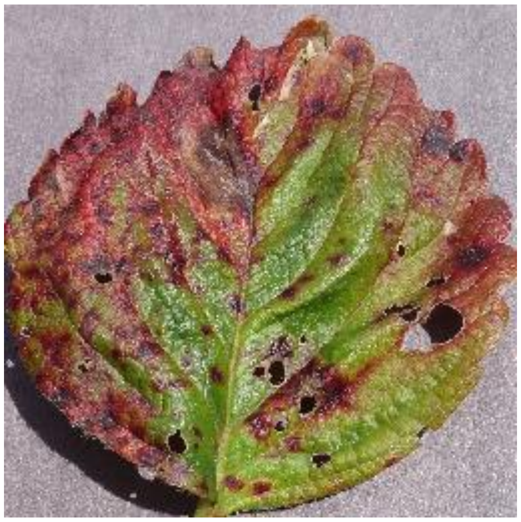


Figure 3.1: Block Diagram

3.1.1 Data Collection

The dataset used in this project is the PlantVillage dataset, which contains over 70,000 images of healthy and diseased plant leaves, covering 29 classes across 9 plant species. These species include common crops like tomato, potato, and apple, with images representing various bacterial and fungal diseases. The dataset is designed for plant disease classification and includes high-resolution images that capture detailed leaf textures and disease symptoms. Figure 3.2 shows sample images of healthy and disease-affected leaves, highlighting the visual differences across conditions. Preprocessing steps such as image resizing and augmentation are applied to improve model accuracy and generalization. The dataset is structured with clear labels for each class (e.g., healthy or infected by a specific disease), and it includes high-resolution images that capture detailed leaf textures and colours. This makes it a robust resource for training models that can perform real-time, accurate disease detection.

Leaf Scorch **Strawberry Healthy**



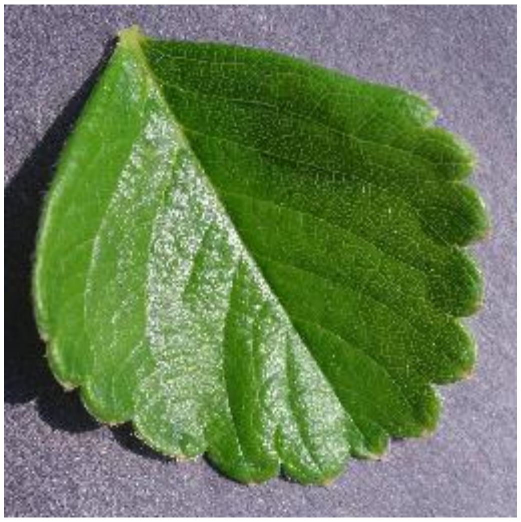


Figure 3.2: Sample Images from Dataset

3.1.2 Image Preprocessing

Preprocess the collected data, ensuring consistency and model readiness, images undergo several preprocessing steps:

- **Color Conversion**: Images are converted to color format RGB to standardize the input for the model.

- **Resizing and Normalization**: Each image is resized to a fixed input dimension 64x64 pixels, and pixel values are normalized to enhance model efficiency.
- **Data Augmentation**: Data augmentation techniques, such as horizontal and vertical flipping, rotation, shearing and zooming, are applied to increase image diversity within the dataset. This reduces the risk of overfitting and improves the model's ability to generalize across varied real-world inputs.

3.1.3 Model Architecture Design

1. **Convolutional Neural Network (CNN):** Convolutional Neural Networks, or CNNs, are a type of deep learning algorithm widely used in computer vision and other domains where

structured grid-like input, particularly images, is processed. Using convolutional layers to apply learnable filters and extract features with translation-invariant capabilities, CNNs automatically learn the spatial hierarchies of the features present in the data. Layers of pooling further reduce the sample size of feature maps while retaining pertinent data. ReLU and other nonlinear activation functions introduce the complexity required to understand complicated relationships and patterns. High-level reasoning based on the extracted features is facilitated by the fully connected layer. Figure 3.3 shows the architecture diagram, illustrating the flow from input to output through multiple layers of convolution, pooling, and activation. CNNs are trained using backpropagation, where weights are adjusted to minimize a loss function, typically categorical cross-entropy for classification tasks. Due to their hierarchical architecture and effective feature extraction capabilities, CNNs are highly effective at tasks like object identification, image segmentation, and image classification.

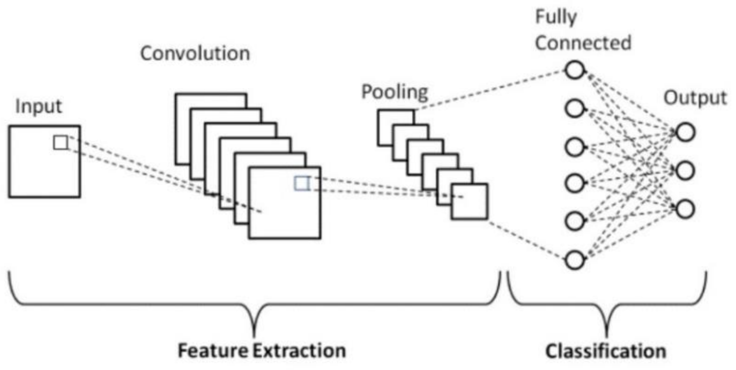


Figure 3.3: CNN Architecture

2. **AlexNet:** In the field of computer vision, AlexNet—a groundbreaking Convolutional Neural Network (CNN) architecture consisting of five convolutional layers followed by three fully connected layers—introduced significant advancements that propelled the deep learning field forward. AlexNet addressed the vanishing gradient problem by using rectified linear units (ReLU) as activation functions, which accelerated convergence during training. Additionally, the inclusion of local response normalization, which provided contrast normalization within local neighborhoods, facilitated improved generalization. Dropout regularization further enhanced the network's robustness by reducing overfitting through the random removal of units

during training. Figure 3.4 illustrates the architecture diagram, highlighting the layer structure from input to output. AlexNet's groundbreaking performance in image classification tasks, especially on large-scale datasets like ImageNet, demonstrated the immense potential of deep learning.

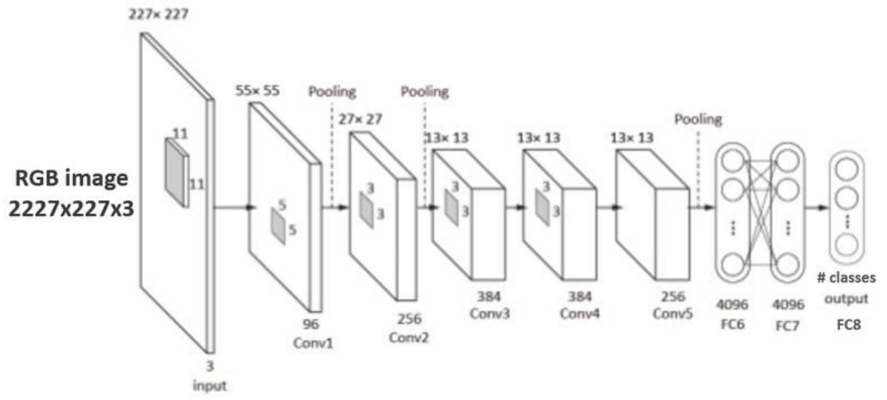


Figure 3.4: AlexNet Architecture

3. **ResNet50:** ResNet50 (Residual Networks with 50 layers) is a deep Convolutional Neural Network (CNN) architecture designed to address the limitations of very deep networks, such as vanishing gradients and poor convergence. Introduced by Kaiming He et al. in 2015, ResNet50 uses residual connections, or skip connections, which enable the network to learn residual mappings rather than direct mappings. This design facilitates the flow of gradients and allows for the training of very deep networks. The architecture includes 50 layers, with 48 convolutional layers and two fully connected layers, making it highly efficient in feature extraction from images. Key elements like batch normalization and ReLU activations are incorporated to improve performance, accelerate training, and prevent overfitting. Figure 3.5 provides the architecture diagram, illustrating the flow and structure of residual blocks and convolutional layers. ResNet50 has achieved high accuracy in various computer vision tasks and is widely used for applications like image classification, object detection, and medical imaging, due to its robust and scalable design.

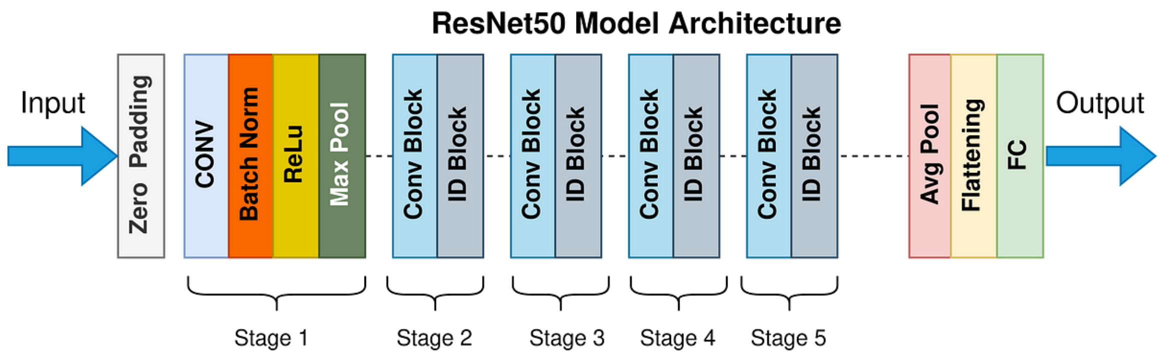


Figure 3.5: ResNet50 Architecture

3.1.4 Model Training

In this project, the plant disease detection model was trained using a Convolutional Neural Network (CNN), with AlexNet and ResNet50 also implemented for comparison. The training process for each model was carried out using the Adam optimizer and categorical cross-entropy loss. The models were trained over 75 epochs with early stopping in place to prevent overfitting by halting the training process when performance on the validation set ceased to improve.

The CNN model employed a series of convolutional layers to extract relevant features from the input images, followed by max-pooling, batch normalization, and dropout layers to improve performance and mitigate overfitting. The training of AlexNet followed a similar approach, using its well-established deep architecture of convolutional layers, ReLU activation functions, and fully connected layers. The ResNet50 model, with its residual connections, was also trained in the same manner, leveraging deeper layers and skip connections that allow for better gradient flow during training.

Each model was trained using the same training dataset, and their performances were evaluated based on accuracy on a separate test set. After training, the best-performing model, based on accuracy, was selected for deployment and used to make predictions on new images. This approach ensures the development of a robust and generalizable model for plant disease detection.

3.1.5 Model Evaluation

In the plant disease detection project, the models (CNN, AlexNet, and ResNet50) were evaluated using the test dataset to measure their classification accuracy. The CNN model achieved the highest validation accuracy of 93%, outperforming AlexNet and ResNet50. Evaluation focused on accuracy, which reflects the proportion of correct predictions, and the confusion matrix to identify misclassifications. Early stopping was used to prevent overfitting by halting training when validation performance ceased improving. The results indicated that the CNN model, though simpler, was the most effective for this task, providing reliable predictions for plant disease detection.

3.1.6 Deployment

For the deployment of the plant disease detection model, a Flask-based web application was developed, allowing users to interact with the model through a simple interface. The application includes an option to upload an image of a plant leaf, which is then processed by the trained model to predict the disease. The interface is styled using CSS, ensuring a user-friendly and visually appealing design. Once the image is uploaded, the model predicts the disease and displays the result on the webpage. This deployment approach enables easy accessibility of the model for real-world use, allowing users to check for plant diseases with minimal technical expertise.

3.2 Software Requirements and Specifications

3.2.1 Operating System

The project is designed to run on modern operating systems like Windows 10/11 or Linux (Ubuntu). This ensures compatibility across different development environments and allows users to work with the software on their preferred platform, whether for local development or deployment. The system's flexibility supports a broad range of user preferences and makes it adaptable to various hardware configurations.

3.2.2 Python 3.11

Python is a high-level, general-purpose programming language known for its simplicity and readability, making it a popular choice for data science, artificial intelligence, and web development. The version used in this project is Python 3.11, which offers several improvements over previous versions, including enhanced performance and new features for developers. Python’s versatility and rich ecosystem of libraries like TensorFlow, NumPy, and Pandas make it ideal for implementing complex machine learning and computer vision tasks such as those required in this project.

3.2.3 Visual Studio Code (VS Code)

The project utilizes Visual Studio Code (VSCode), a free, open-source code editor developed by Microsoft. VSCode is lightweight yet powerful, supporting a wide range of programming languages, including Python. Key features include:

- Built-in Git support for version control.
- Debugging tools to help identify and resolve code issues.
- An extension marketplace to enhance functionality with additional tools.
- Integrated terminal, which facilitates executing scripts directly from the editor.
- Code completion and IntelliSense for better productivity and reduced coding errors.

These features make VSCode an ideal choice for development, enabling developers to efficiently manage their codebase, troubleshoot issues, and streamline their workflow.

3.2.4 Jupyter Notebook

Jupyter Notebook provides an interactive development environment that is widely used in data science and machine learning. It allows you to create and share documents that contain live code, equations, visualizations, and narrative text. In your project, Jupyter Notebooks are used for exploring the dataset, experimenting with preprocessing techniques, and visualizing results. It's also useful for running and documenting the iterative process of model training.

Key Features:

- Cell-based execution for modular code.
- Supports rich text, LaTeX, and visualizations within the notebooks.
- Easy integration with Python libraries such as NumPy, Pandas, and Matplotlib.

3.2.5 Libraries

The following libraries are essential for the successful implementation of the project:

- **TensorFlow/Keras**: These libraries are crucial for building and training deep learning models like CNN, AlexNet, and ResNet50. TensorFlow provides powerful tools for model construction, training, and evaluation, while Keras offers high-level APIs for creating neural networks efficiently.
- **OpenCV**: Used for image processing tasks like reading, resizing, and augmenting images. It's especially useful for preparing the dataset before feeding it into the models.
- **NumPy**: This library is fundamental for handling numerical data, such as image arrays, and performing various operations like resizing and normalizing the image data.
- **Matplotlib**: Employed for visualizing model training progress, such as plotting training and validation accuracy and loss curves.
- **scikit-learn**: Provides tools for splitting the dataset into training and testing sets and evaluating model performance using metrics like accuracy.

3.2.6 Flask

Flask is a lightweight Python web framework used to create web applications. In this project, Flask is used to develop the web interface that allows users to upload images for demographic classification and age prediction. The framework's simplicity and flexibility make it an ideal choice for rapid prototyping and deployment. Flask supports the creation of RESTful APIs, which can be used to interact with the trained models and process predictions.

It also integrates well with front-end technologies such as HTML, CSS, and JavaScript, enabling the development of a user-friendly interface for displaying the results of demographic predictions.

3.2.7 Google Chrome

Google Chrome is a widely-used web browser known for its speed, security, and performance. It is essential for testing and deploying the web application developed in this project. Chrome provides excellent developer tools, including the JavaScript console, network monitoring, and performance analysis features. These tools assist developers in debugging issues with the web interface, optimizing performance, and ensuring smooth deployment of the application. Additionally, Chrome's extension support and compatibility with modern web standards make it a suitable browser for testing the web application's functionality.

CHAPTER 4

RESULTS AND DISCUSSION

The plant disease detection project demonstrated that the custom CNN model outperformed both AlexNet and ResNet50, achieving a validation accuracy of 93%, compared to 89% and 70% for AlexNet and ResNet50, respectively. The CNN's performance highlights its suitability for plant disease detection due to its efficient feature extraction and customized architecture. While AlexNet and ResNet50 showed competitive results, they faced challenges such as overfitting and the need for more fine-tuning. The trained models were deployed in a Flask web interface, enabling users to upload images for disease predictions, making the tool practical and accessible for real-world agricultural applications.

4.1. Testing

Testing in web application development is a critical process that involves verifying the functionality, performance, security, and usability of the web application. It ensures the application works as intended and meets user requirements. Testing helps in identifying and addressing potential issues before the application is deployed to users, thereby improving the overall quality and user experience. For this project, various testing methodologies such as unit testing, integration testing, and functional testing will be implemented to ensure the reliability and performance of the web application.

4.1.1 Unit Testing

● Verify that individual components or functions of the code work as intended

● Developers write tests for specific functions or modules to ensure they produce the expected output.

4.1.2 Integration Testing

● Test the interaction between different components, modules, or services to ensure they work together correctly.

● Verify that integrated components collaborate as expected, checking for issues such as data flow and communication.

4.1.3 Functional Testing

● Validate that the web application's features and functions work as intended from an end-user perspective.

● Conduct tests to ensure that user interactions, data input, and expected outputs align with the specified requirements.

4.2 Output Screens and Results

The following section describes the output and how the system performs during testing, including the output screens displayed to the user:

4.2.1 Steps to Use the System

- Visit the website: Navigate to the hosted website demographic AI.
- Upload Image: User can upload a facial image to processed by the trained models.
- Results Displayed: Predictions for Gender, Race, and Age displayed in a readable format on the screen.

4.2.2 Output Screens

Web Interface:

The web interface, developed using Flask, provides a user-friendly platform for interacting with the plant disease detection system. When an image of a plant leaf is uploaded, the system processes it and displays the predicted disease status, whether healthy or affected by a specific disease alongside the processed image. Figure 4.1 (a) shows a screenshot of the user interface, illustrating the layout and features available to the user. Figure 4.1 (b) captures a sample prediction for a healthy leaf, demonstrating how the interface presents this information, while Figure 4.1 (c) displays a screenshot of a disease prediction, showing how the system communicates the disease label to the user. This interface is designed to streamline the detection process, making it accessible and efficient for end-users seeking real-time plant health insights.

Screenshots

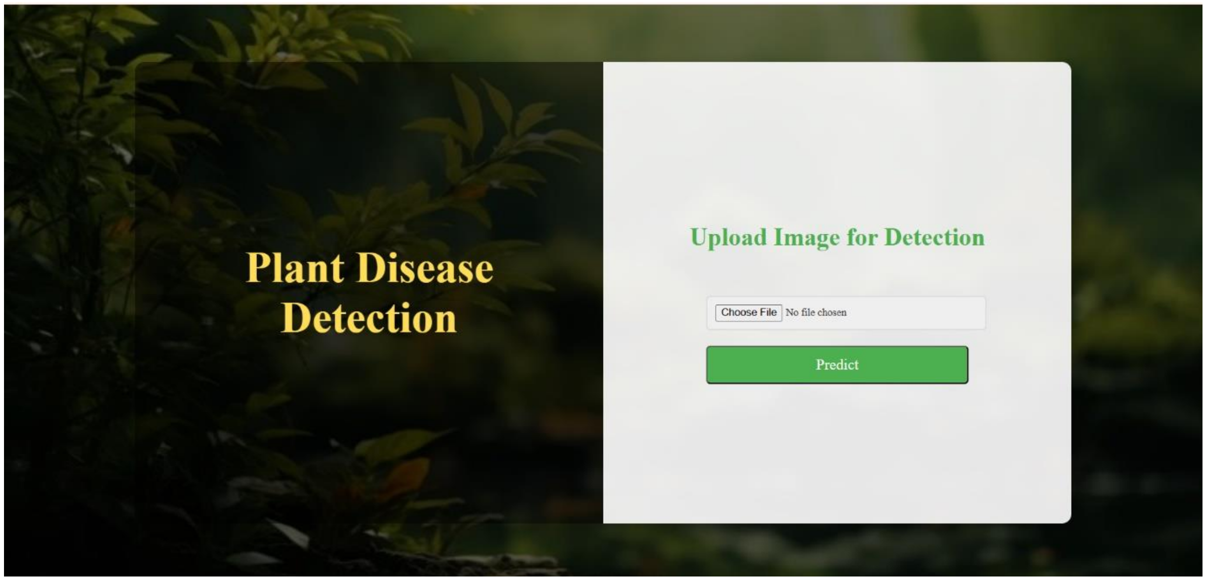
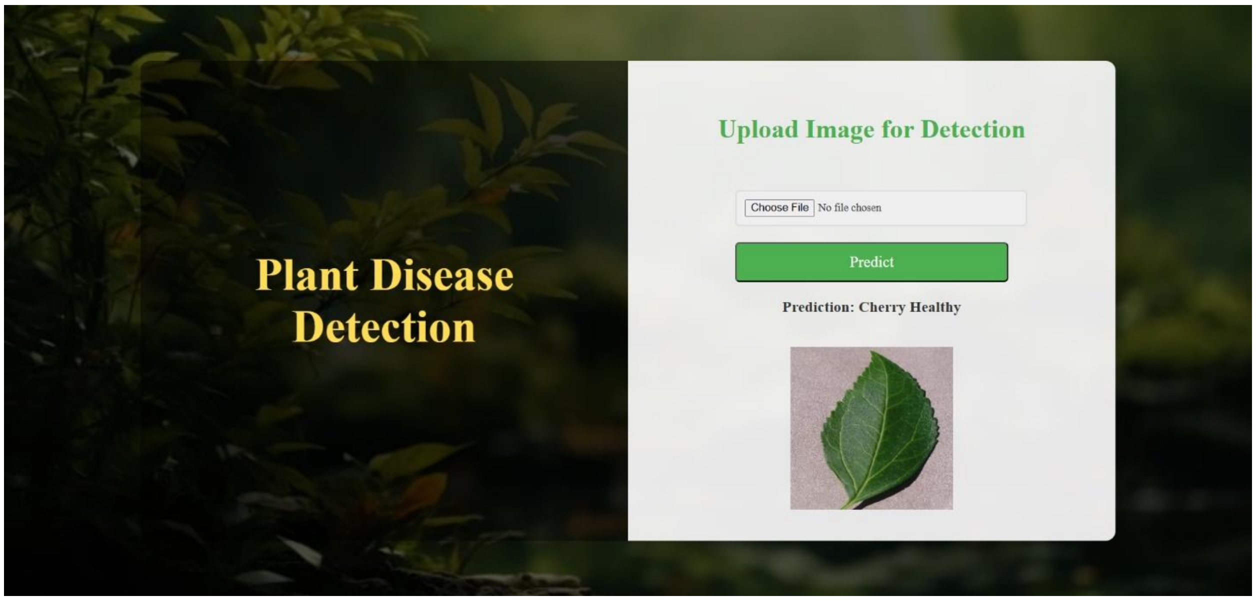
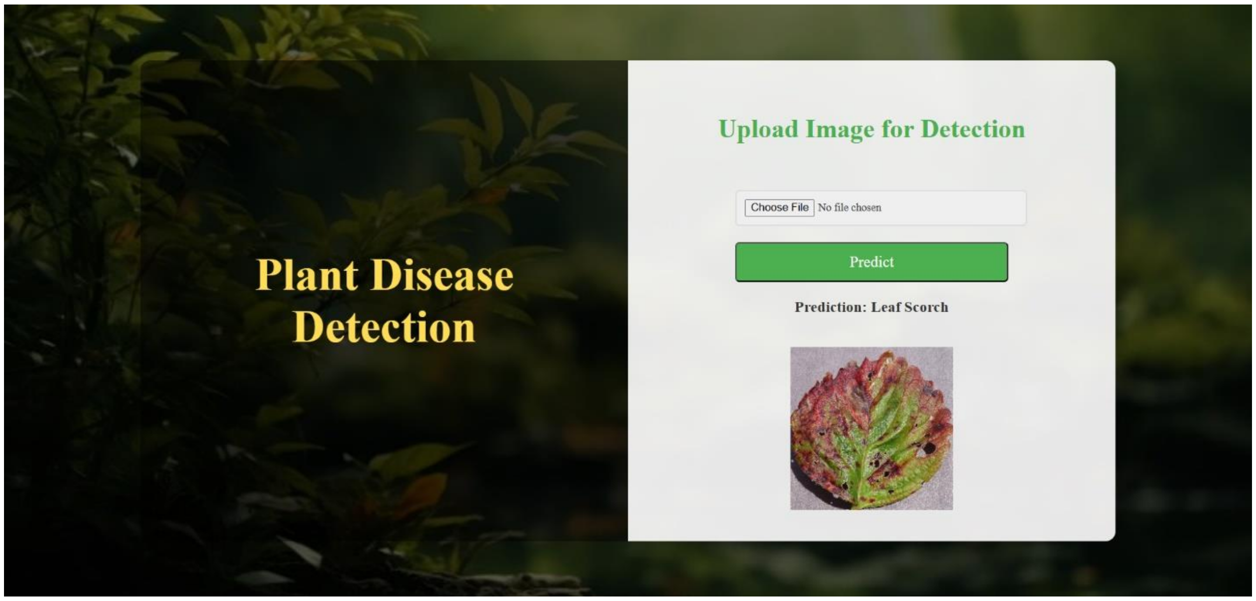


Figure 4.1(a): Screenshot of UI



Prediction: Cherry Healthy

Figure 4.1(b): Screenshot of Healthy Prediction



Prediction: Leaf Scorch

Figure 4.1(c): Screenshot of Disease Prediction

4.3 Results and Performance Evaluation

The performance of the plant disease detection model was evaluated using the validation dataset, comparing the effectiveness of three models: CNN, AlexNet, and ResNet50.

- CNN Model: The custom CNN model achieved the highest validation accuracy of 93%, demonstrating its strong ability to detect plant diseases across 28 diseases and multiple crops. Figure 4.2(a) shows the accuracy and loss plots for the CNN model, illustrating its performance and effective learning over training epochs.
- AlexNet Model: The AlexNet model performed well, achieving a validation accuracy of 89%. While slightly lower than the CNN model, it still showed robust performance in identifying plant diseases. Figure 4.2(b) presents the accuracy and loss plots for the AlexNet model, highlighting its learning progress and effectiveness across training epochs.
- ResNet50 Model: The ResNet50 model showed the lowest performance, with a validation accuracy of 70%. Despite being a deep architecture, it faced challenges in fine-tuning and overfitting on the dataset. Figure 4.2(c) presents the accuracy and loss plots for the

ResNet50 model, highlighting its learning progress and effectiveness across training epochs.

All three models performed well, with the CNN model emerging as the most effective for plant disease detection. The results highlight the importance of architecture customization and fine-tuning in achieving high performance, particularly in specialized tasks like plant disease detection.

4.3.1 Evaluation Curves

- **CNN**

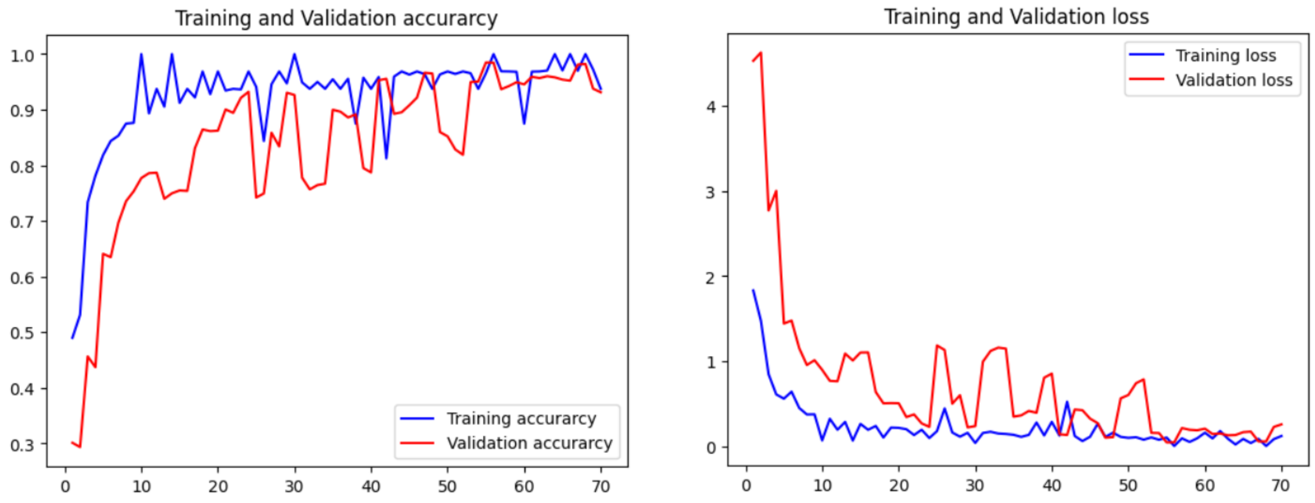


Figure 4.2(a): CNN Accuracy Curve

- **AlexNet**

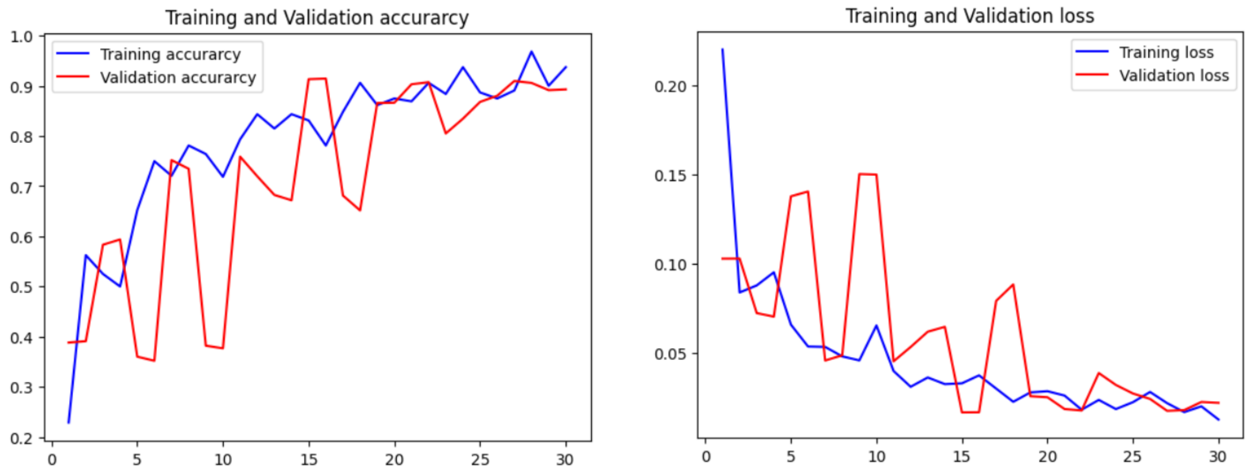


Figure 4.2(b): AlexNet Accuracy Curve

- **ResNet50**

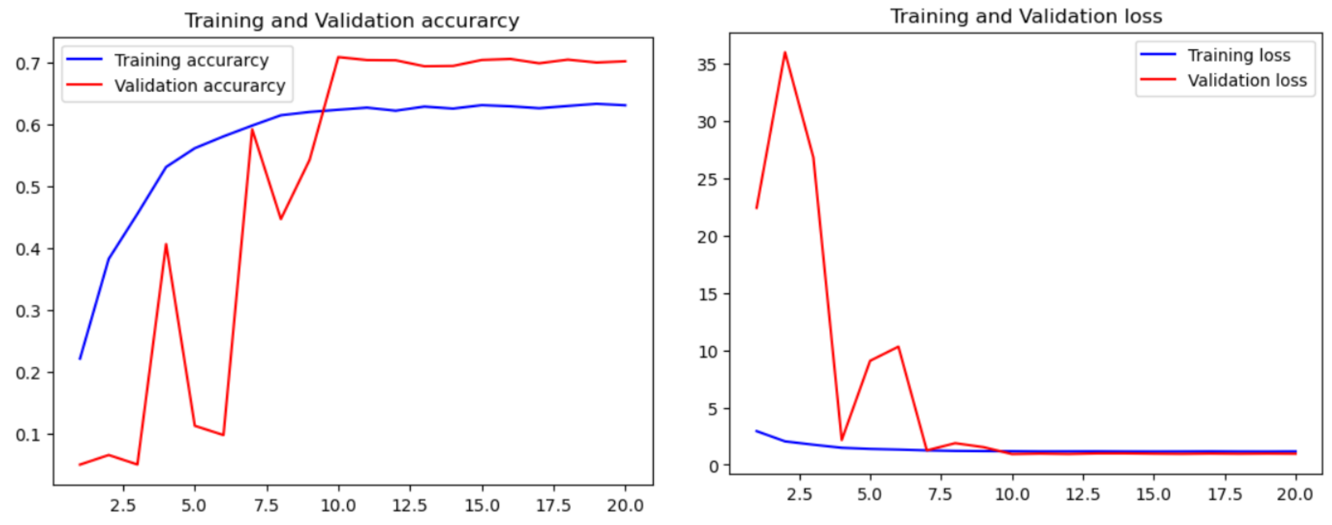


Figure 4.2(c): ResNet50 Accuracy Curve

CHAPTER 5

CONCLUSION

In conclusion, this plant disease detection project successfully demonstrated the potential of deep learning models, including CNN, AlexNet, and ResNet50, in identifying plant diseases. The custom CNN model outperformed the other models, achieving the highest validation accuracy of 93%, indicating its strong ability to generalize and accurately classify plant leaf images. While AlexNet and ResNet50 showed good performance, they were not as effective in this specific context, with AlexNet reaching 89% accuracy and ResNet50 at 70%. The project highlights the importance of model selection and fine-tuning for specialized tasks like plant disease detection. Additionally, the development of a Flask-based web interface provided a user-friendly platform for easy deployment, enabling users to upload images and receive disease predictions in real time. This work contributes to the growing field of agricultural technology, offering a promising tool for early disease detection and prevention, which can ultimately help in managing crop health more efficiently.

5.1 Future Enhancements

For future enhancements of the Plant Disease Detection System, several directions can be explored to improve its performance and applicability:

- **Mobile and IoT integration**: Develop a mobile app or deploy the model on IoT devices (e.g., Raspberry Pi) for real-time, offline plant disease detection in the field.
- **Multi-Disease and Early Detection**: Extend the model to detect multiple diseases in a single leaf image and identify early-stage symptoms for proactive intervention.
- **Disease localization and Severity Estimation**: Implement image segmentation to highlight infected areas on leaves and predict the percentage of disease-affected regions, aiding in precise severity estimation and treatment recommendations
- **Real-Time Detection**: Develop a real-time disease detection system that could be integrated with drones or cameras to autonomously monitor plant health in the field, providing instant disease detection feedback.

REFERENCES

[1] Aanis Ahmed, Aly L Gamal and Dharmendra Saraswat, 2023. Towards Generalization of Deep Learning- based plant disease identification under controlled and Field conditions. *IEEE Access, 11:9042–9057*

[2] C Jackulin and S Murugavalli, 2022. A comprehensive review on detection of plant disease using machine learning and deep learning approaches. *Measurement: Sensors, 24:100441*

[3] Lili Li, Shujuan Zhang, and Bin Wang, 2021 Plant disease detection and classification by deep learning—a review. *IEEE Access, 9:56683–56698.*

[4] Emmanuel Moupojou, Appolinaire Tagne, Florent Retraint, Anicet Tadonkemwa, Dongmo Wilfried, Hyppolite Tapamo, and Marcellin Nkenlifack. Fieldplant, 2023: A dataset of field plant images for plant disease detection and classification with deep learning. *IEEE Access, 11:35398 35410*