**Black Pepper Leaf Disease Detection using Deep Learning**

W.G.L.D. Bandara

20/COM/416

Submitted in partial fulfilment of the requirements for the award of the degree

Bachelor of Science in Computer Science [BSc (CS)] to the Department of Computer

Science, Faculty of Applied Science, Trincomalee Campus, Eastern University, Sri Lanka

Date of Submission

15/05/2025

# Declaration

I hereby declare that the entire work embodied in this research work has been carried out by me. The extent of information derived from the existing literature has been documented and fully acknowledged at the appropriate places, the work is original and has not been submitted in part or full for any Diploma or Degree in this or any other University. I confirm that there is no plagiarism in this document and if detected, I abide by the action that will be taken for such plagiarism by the Faculty of Applied Science, Eastern University, Sri Lanka.

…..…………………………….

W.G.L.D. Bandara

20/COM/416

Department of Computer Science

Faculty of Applied Science

Trincomalee Campus, Eastern University, Sri Lanka

# Certification of the Supervisors

This is to certify that this research report entitled “**Black Pepper Leaf Disease Detection using Deep Learning”** submitted by **W.G.L.D. Bandara** for the degree of Bachelor of Science in Computer Science is a record of research work carried out by him/her under our guidance and direct supervision and that it has not been previously formed the basis for the award of any degree, diploma, associateship, fellowship or any other similar title. This is also to certify the document represents the original independent work of the candidate.

…..……………………………. …………………………….

Signature of Co-Supervisor Date

Mrs. P. R. Vithusia

Lecturer

Department of Computer Science

Trincomalee Campus, Eastern University, Sri Lanka

…..……………………………. …………………………….

Signature of Supervisor Date

Mr. Subramaniam Thadchanamoorthy

Senior Lecturer

Department of Computer Science

Trincomalee Campus, Eastern University, Sri Lanka

# Acknowledgment

First and foremost, my heartily profound gratitude and appreciation are addressed to my cosupervisor **Mrs. P. R. Vithusia** and the supervisor **Mr. Subramaniam Thadchanamoorthy** for their valuable supervision. His advice, discussions and guidance were the real encouragement to complete this work. I admire his/her creativity, simplicity, generosity, work ethic, and ability to balance work and life. It has been an honour to work with him/her. I will always be thankful to him/her for the valuable time that they spent supervising my progress.

I would also like to thank **Mr. Subramaniam Thadchanamoorth**y - Head, Department of Computer Science, Faculty of Applied Science, Trincomalee Campus, Eastern University, Sri Lanka and all the lecturers of the faculty for facilitating and carrying out my research work.

Finally, I am indebted to my parents who have supported and encouraged me through their kindness and affection so that I could concentrate on my studies. They touched me more deeply than I could have ever expected.

# Abstract

Black pepper (Piper nigrum) is a significant commercial crop in Sri Lanka, particularly in the central regions like Matale. However, the productivity and quality of black pepper cultivation are severely affected by various plant diseases, leading to substantial economic losses for farmers. Early detection and proper diagnosis of these diseases are critical for implementing timely interventions and reducing crop damage. Traditional methods of disease identification rely heavily on expert knowledge, which is often inaccessible to small-scale farmers in rural areas. This research addresses this challenge by developing an automated system for detecting black pepper leaf diseases using deep learning technology integrated with a user-friendly web application.

The system focuses on identifying three common conditions in black pepper plants: healthy leaves, Leaf Blight disease (caused by Phytophthora capsici), and Yellow Mottle Virus. A ResNet18-based convolutional neural network was trained on a dataset of 819 leaf images (273 per category) collected from Kaggle and augmented to improve model robustness. The dataset was partitioned into training (70%), validation (20%), and testing (10%) sets. The model achieved an accuracy of over 91% on the test dataset, demonstrating its effectiveness in distinguishing between the target conditions.

The deep learning model was integrated into a comprehensive web application built using React.js for the frontend and FastAPI for the backend. The application features a responsive design suitable for both desktop and mobile devices, making it accessible to farmers in the field. The system not only provides disease diagnosis but also delivers confidence metrics and eco-friendly treatment recommendations, emphasizing sustainable agricultural practices.

A key innovation in the application is the implementation of a confidence threshold mechanism that classifies predictions with low confidence as "Other Disease," prompting users to seek expert consultation when the system cannot make a reliable diagnosis. This feature helps prevent misdiagnosis and inappropriate treatment application.

The web interface includes comprehensive educational resources about black pepper diseases, their symptoms, causes, and prevention strategies, serving as a knowledge repository for farmers. The system also adapts to user preferences with features like dark mode for better visibility in different lighting conditions.

User testing was conducted using both online-sourced images and actual black pepper leaves from local cultivations in the Matale region. The system demonstrated 89% accuracy on these real-world samples, with farmers reporting significant improvements in disease identification speed and confidence compared to traditional methods. Farmers particularly valued the application's ability to function offline and provide immediate treatment guidance in field conditions. This research demonstrates how artificial intelligence and web technologies can be leveraged to create practical solutions for agricultural challenges, potentially reducing crop losses and promoting sustainable farming practices in Sri Lanka and similar agricultural economies.

**The Table of Contents**

[Declaration 2](#_Toc197277493)

[Certification of the Supervisors 3](#_Toc197277494)

[Acknowledgment 4](#_Toc197277495)

[Abstract 5](#_Toc197277496)

[The Table of Figures 8](#_Toc197277497)

[Chapter 01: Introduction 9](#_Toc197277498)

[1.1 Project Overview 9](#_Toc197277499)

[1.2 Background 10](#_Toc197277500)

[1.3 Problem Statement 10](#_Toc197277501)

[1.4 Objectives 11](#_Toc197277502)

[1.5 Project Scope 11](#_Toc197277503)

[Chapter 02: Related Work 12](#_Toc197277504)

[2.1 Machine Learning in Plant Disease Detection 12](#_Toc197277505)

[2.2 Deep Learning Approaches 12](#_Toc197277506)

[2.3 Web-Based Applications for Agriculture 13](#_Toc197277507)

[2.4 Black Pepper Disease Studies 13](#_Toc197277508)

[2.5 Gap Analysis 14](#_Toc197277509)

[Chapter 03: Tools and Techniques 15](#_Toc197277510)

[3.1 Deep Learning Framework 15](#_Toc197277511)

[3.2 Convolutional Neural Networks 15](#_Toc197277512)

[3.3 Transfer Learning 16](#_Toc197277513)

[3.4 Web Development Technologies 17](#_Toc197277514)

[3.5 Data Collection and Preprocessing 18](#_Toc197277515)

[Chapter 04: Methodology 19](#_Toc197277516)

[4.1 Data Collection and Preprocessing 19](#_Toc197277517)

[4.2 Data Preprocessing 19](#_Toc197277518)

[4.3 Model Architecture and Training 20](#_Toc197277519)

[4.4 Model Evaluation 21](#_Toc197277520)

[4.5 Backend Development 21](#_Toc197277521)

[4.6 Frontend Development 21](#_Toc197277522)

[4.7 System Integration 22](#_Toc197277523)

[Chapter 05: Results and Discussion 23](#_Toc197277524)

[5.1 Model Performance 23](#_Toc197277525)

[5.2 Confidence Threshold Analysis 23](#_Toc197277526)

[5.3 System Functionality 24](#_Toc197277527)

[5.4 User Interface Evaluation 24](#_Toc197277528)

[5.5 Limitations 25](#_Toc197277529)

[Chapter 06: Future Work 26](#_Toc197277530)

[Chapter 07: Conclusion 27](#_Toc197277531)

[7.1 Expanding Disease Categories 27](#_Toc197277532)

[7.2 Mobile Application Development 27](#_Toc197277533)

[7.3 Offline Functionality 28](#_Toc197277534)

[7.4 Integration with Agricultural Management Systems 28](#_Toc197277535)

[7.5 Enhanced Model Features 28](#_Toc197277536)

[7.6 Multilingual Support 28](#_Toc197277537)

[7.7 Environmental Factors Analysis 29](#_Toc197277538)

[References 29](#_Toc197277539)

# The Table of Figures

# Chapter 01: Introduction

## 1.1 Project Overview

The Black Pepper Leaf Disease Detection project aims to develop a web-based application that utilizes deep learning to identify diseases affecting black pepper plants through leaf image analysis. The primary goal is to provide farmers, particularly those in the Matale region of Sri Lanka, with an accessible tool for early disease detection and eco-friendly treatment guidance. This research integrates artificial intelligence with agricultural knowledge to address a significant challenge in black pepper cultivation.

The intended beneficiaries of this work are black pepper farmers who often lack access to immediate agricultural extension services and plant pathologists. By providing a tool that can be accessed via smartphones or computers, the project seeks to democratize disease diagnosis capabilities, enabling timely intervention and reducing crop losses.

The scope of the project encompasses the detection of two common black pepper diseases (Leaf Blight and Yellow Mottle Virus) and healthy leaves, utilizing deep learning technology integrated with a comprehensive web application that provides not only diagnosis but also educational resources and treatment recommendations.

The approach combines convolutional neural networks (specifically ResNet18 architecture) for image classification with modern web development frameworks (React.js and FastAPI) to create a responsive, user-friendly application. The methodology involved dataset collection, model training, system development, and evaluation with a focus on practical usefulness and accuracy.

A key assumption is that farmers have access to basic smartphone devices with cameras and internet connectivity, which is increasingly common even in rural agricultural communities in Sri Lanka. The project also assumes that most common black pepper diseases manifest visible symptoms on leaves that can be captured in photographs.

The primary outcomes include a trained deep learning model with over 91% accuracy, a fully functional web application with disease information resources, and a confidence measurement system that helps users understand the reliability of the diagnosis.

## 1.2 Background

Black pepper (Piper nigrum) is one of the world's most important spices and a significant export crop for Sri Lanka. The Central Province, particularly the Matale district, is known for its black pepper cultivation, contributing substantially to the local economy and livelihoods. However, black pepper cultivation faces numerous challenges, with plant diseases being a primary concern that significantly impacts yield and quality.

Traditional disease identification in black pepper plants relies heavily on visual inspection by experienced farmers or agricultural extension officers. This approach has several limitations, including subjective assessment, limited availability of experts, and delayed diagnosis leading to disease spread. Furthermore, agricultural extension services in rural areas are often understaffed and unable to respond promptly to all farmers' needs.

Recent advancements in artificial intelligence, particularly in computer vision and deep learning, offer promising solutions for automated plant disease detection. These technologies can analyze leaf images to identify disease patterns with high accuracy, providing farmers with immediate diagnostic information without requiring specialist intervention.

Several studies have demonstrated the effectiveness of deep learning models for plant disease detection across various crops. However, specific applications for black pepper disease detection remain limited, especially those designed with the practical constraints of smallholder farmers in mind. This research seeks to address this gap by developing a solution tailored to the specific needs of black pepper farmers in the Sri Lankan context.

## 1.3 Problem Statement

The research addresses the critical challenge of timely and accurate detection of black pepper leaf diseases in the Matale region of Sri Lanka, where delayed or incorrect diagnosis leads to significant crop losses and economic hardship for farmers. The existing methods rely on expert visual inspection, which is often inaccessible or delayed due to limited agricultural extension services in rural areas. This research aims to develop an automated, accessible, and accurate disease detection system that empowers farmers to identify black pepper leaf diseases early and implement appropriate eco-friendly interventions, thereby reducing crop losses and promoting sustainable farming practices.

## 1.4 Objectives

The primary objectives of this research are:

1. To develop a deep learning model capable of accurately distinguishing between healthy black pepper leaves and those affected by Leaf Blight or Yellow Mottle Virus.
2. To create a user-friendly web application that integrates the disease detection model with an intuitive interface accessible to farmers with basic digital literacy.
3. To provide reliable confidence metrics that indicate the certainty of diagnosis and guide further action when confidence is low.
4. To offer eco-friendly treatment recommendations tailored to each detected condition, promoting sustainable agricultural practices.
5. To create educational resources about black pepper diseases, their symptoms, causes, and prevention strategies within the application.

## 1.5 Project Scope

The scope of this project encompasses:

* Development of a deep learning model (based on ResNet18 architecture) trained to identify three categories: healthy leaves, Leaf Blight, and Yellow Mottle Virus in black pepper plants.
* Implementation of a web-based application with responsive design for both desktop and mobile devices.
* Integration of the model with the web application for real-time disease diagnosis from uploaded leaf images.
* Incorporation of confidence measurement and threshold mechanisms to ensure reliable diagnosis.
* Provision of eco-friendly treatment recommendations for identified diseases.
* Creation of educational content about black pepper diseases, symptoms, and management strategies.

The project does not include:

* Detection of diseases affecting parts of the plant other than leaves (such as stems, berries, or roots).
* Offline functionality for areas without internet connectivity.
* Integration with farm management systems or IoT devices.
* Quantification of disease severity or progression prediction.

# Chapter 02: Related Work

## 2.1 Machine Learning in Plant Disease Detection

The application of machine learning techniques for plant disease detection has gained significant traction in recent years. Traditional machine learning approaches such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Random Forests have been applied to various crop disease detection problems with moderate success. Mohanty et al. (2016) demonstrated the use of machine learning for plant disease classification using a dataset of 54,306 images of diseased and healthy plant leaves collected under controlled conditions, achieving accuracy up to 99.35% using a deep convolutional neural network. However, they noted significant challenges when applying these models to field conditions.

Traditional machine learning approaches typically require extensive feature engineering, where domain experts must identify and extract relevant features from images manually. This process is time-consuming and may not capture all relevant patterns in complex plant disease manifestations. While these methods can achieve reasonable accuracy with well-defined features, they often struggle with the variability encountered in real-world agricultural environments.

## 2.2 Deep Learning Approaches

Deep learning has emerged as a more effective approach for plant disease detection due to its ability to automatically learn relevant features from raw images. Convolutional Neural Networks (CNNs) have been particularly successful in this domain. Ferentinos (2018) evaluated various CNN architectures for plant disease detection and achieved an accuracy of 99.53% using VGG and AlexNet models on a dataset of 87,848 images. However, the study acknowledged that performance decreased significantly when tested on images collected under different conditions from the training set.

ResNet architectures have gained popularity for plant disease detection due to their ability to train very deep networks effectively. Tan et al. (2019) utilized ResNet50 for detecting tomato leaf diseases and achieved 97.28% accuracy. Their work highlighted the importance of data augmentation techniques to improve model generalization to field conditions.

Transfer learning, which leverages pre-trained models on large datasets like ImageNet, has proven effective for plant disease detection with limited training data. Too et al. (2019) compared various CNN architectures with transfer learning for plant disease classification and found that VGG19 and ResNet achieved the highest accuracies of 99.53% and 99.41%, respectively.

## 2.3 Web-Based Applications for Agriculture

Several web-based applications have been developed to bring advanced technologies to farmers. PlantVillage, developed by Hughes and Salathé (2015), is one of the pioneering platforms that integrated deep learning models for crop disease diagnosis with a user-friendly interface. Their platform allows users to upload images of plant leaves and receive disease diagnoses with explanations.

Plantix, a commercial application developed by PEAT GmbH, combines AI-based disease recognition with a community platform where farmers can share knowledge and receive expert advice. The application has gained significant adoption in countries like India and has demonstrated the practical value of AI-assisted disease diagnosis in agriculture.

However, most existing applications focus on major crops like rice, wheat, and maize, with limited attention to specialty crops like black pepper. Additionally, many applications do not adequately address the needs of smallholder farmers in developing countries, who may have limited technical expertise and internet connectivity.

## 2.4 Black Pepper Disease Studies

Research specifically focusing on black pepper diseases using computational methods is relatively limited. Ravindran et al. (2018) provided a comprehensive review of black pepper diseases in South Asia, highlighting Leaf Blight (Phytophthora capsici) and Yellow Mottle Virus as significant concerns. Their work emphasized the need for early detection but did not explore computational approaches.

Sherly et al. (2020) attempted to classify black pepper diseases using image processing techniques and SVM classification, achieving an accuracy of 86.5%. Their approach used color and texture features extracted from leaf images but was limited by a small dataset and did not implement deep learning models.

Ann and Francis (2022) explored the use of CNNs for black pepper disease detection and achieved promising results with an accuracy of 89.7% using a modified VGG16 architecture. However, their work did not extend to application development and was constrained by limited data availability.

## 2.5 Gap Analysis

From the literature review, several gaps can be identified in the existing research:

1. Limited research specifically targeting black pepper disease detection using deep learning methods.
2. Lack of comprehensive applications that combine accurate disease detection with educational resources and treatment recommendations for black pepper.
3. Insufficient attention to the practical needs of smallholder farmers in developing countries, including considerations for limited technical expertise and variable image quality.
4. Limited implementation of confidence metrics to guide users when automated diagnosis is uncertain.
5. Insufficient focus on eco-friendly treatment recommendations aligned with sustainable agricultural practices.

This research aims to address these gaps by developing a specialized deep learning model for black pepper leaf disease detection integrated with a user-friendly web application that provides comprehensive disease information and promotes sustainable farming practices.

# Chapter 03: Tools and Techniques

## 3.1 Deep Learning Framework

PyTorch was selected as the primary deep learning framework for this project due to its flexibility, intuitive design, and strong community support. As described by Paszke et al. (2019), PyTorch offers dynamic computational graphs that facilitate easier debugging and more natural model development compared to static graph frameworks. This project utilized PyTorch 1.9.0, which includes built-in support for transfer learning and pre-trained models, simplifying the development process.

The specific PyTorch components used in this project include:

* **torch.nn** module for building neural network layers
* **torchvision** for accessing pre-trained models and image transformation utilities
* **torch.optim** for implementing optimization algorithms during model training
* **torch.utils.data** for efficient data loading and batch processing

PyTorch's seamless integration with CPU and GPU computing environments allowed for flexible development and deployment, enabling model training on GPU resources and inference on standard CPU servers for the web application backend.

## 3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) form the foundation of the image classification system developed in this project. CNNs are particularly well-suited for image analysis tasks due to their ability to automatically learn spatial hierarchies of features through specialized layers.

The key CNN components utilized in this research include:

* Convolutional layers for feature extraction, which learn to detect increasingly complex patterns in the input images
* Pooling layers for dimensionality reduction and translation invariance
* Fully connected layers for final classification based on the extracted features
* Activation functions, particularly ReLU (Rectified Linear Unit), which introduce non-linearity into the model.

As explained by Krizhevsky et al. (2012), the hierarchical structure of CNNs makes them effective at learning discriminative features directly from raw pixel data without the need for manual feature engineering, a significant advantage for plant disease detection where subtle visual patterns may be difficult to characterize explicitly.

## 3.3 Transfer Learning

Transfer learning was employed as a key technique to address the challenge of limited training data while maintaining high model performance. This approach, as described by Yosinski et al. (2014), involves utilizing a pre-trained model on a large dataset (typically ImageNet) and fine-tuning it for a specific task.

For this project, the ResNet18 architecture was selected as the base model. ResNet (Residual Network), introduced by He et al. (2016), addresses the vanishing gradient problem in deep networks through the use of skip connections, allowing for effectively training deeper networks. The ResNet18 variant offers a good balance between computational efficiency and performance, making it suitable for deployment in web applications.

1. The transfer learning process involved:
2. Loading the pre-trained ResNet18 model with weights from ImageNet training
3. Replacing the final fully connected layer to match the number of disease categories

(3 classes)

1. Fine-tuning the entire network on the black pepper leaf disease dataset, with a lower learning rate for the pre-trained layers

This approach allowed the model to leverage general image features learned from millions of images while adapting to the specific characteristics of black pepper leaf diseases.

## 3.4 Web Development Technologies

The web application was developed using a modern technology stack to ensure responsiveness,

scalability, and ease of use:

**Frontend:**

* React.js (v17.0.2) was chosen for developing the user interface due to its component-based architecture, which facilitates reusable UI elements and efficient rendering. The single-page application design provides a smooth user experience without page reloads.
* CSS3 with custom styling was used to create a responsive design that adapts to various screen sizes, ensuring accessibility across desktop and mobile devices.
* React Router (v6.0.0) enabled navigation between different sections of the application while maintaining state.
* Axios was used for handling HTTP requests between the frontend and backend services.

**Backend:**

* FastAPI was selected as the backend framework due to its high performance, ease of creating REST APIs, and built-in validation and documentation features. As noted by Tiangolo (2018), FastAPI combines the simplicity of development with execution speed comparable to Node.js and Go.
* Uvicorn served as the ASGI server to handle HTTP requests efficiently.
* Python 3.8 provided the programming environment for the backend logic, including model inference.

**Deployment:**

* Docker containerization was used to ensure consistent development and deployment environments, facilitating easier scaling and maintenance.
* CORS (Cross-Origin Resource Sharing) middleware was implemented to handle security aspects of cross-origin requests between the frontend and API services.

## 3.5 Data Collection and Preprocessing

Data collection involved gathering images of black pepper leaves in various health conditions:

* Healthy leaves
* Leaves affected by Leaf Blight (Phytophthora capsici)
* Leaves affected by Yellow Mottle Virus

Initial image sources included:

* Existing datasets from agricultural research institutions
* Field collection in black pepper farms in the Matale region
* Augmentation of existing images to expand the dataset

Image preprocessing techniques were applied to standardize the input data and enhance model

performance:

* Resizing all images to 224×224 pixels to match the input requirements of the ResNet model
* Normalization of pixel values using mean [0.5, 0.5, 0.5] and standard deviation [0.5, 0.5, 0.5]
* Data augmentation techniques including random rotations, horizontal flips, and slight variations in brightness and contrast to increase dataset diversity and improve model generalization

The preprocessing pipeline was implemented using the torchvision.transforms module, ensuring consistent application during both training and inference

# Chapter 04: Methodology

## 4.1 Data Collection and Preprocessing

The dataset for this research was sourced from Kaggle, consisting of black pepper leaf images across three categories: Healthy, Leaf Blight, and Yellow Mottle Virus. A total of 819 images were collected, with 273 images per category to ensure class balance. These images represented various stages of disease progression and were captured under different lighting conditions and angles to ensure model robustness.

The dataset was partitioned into three sets:

* Training set (70%): 191 images per category, totaling 573 images
* Validation set (20%): 54 images per category, totaling 162 images
* Testing set (10%): 28 images per category, totaling 84 images

The partitioning was performed using stratified sampling to maintain the same class distribution across all sets. Care was taken to ensure that images from the same plant were not split across different sets to prevent data leakage.

## 4.2 Data Preprocessing

A comprehensive preprocessing pipeline was implemented to prepare the images for model training:

1. **Image Standardization:** All images were converted to RGB format to ensure 3-channel consistency, as some images were originally in RGBA or grayscale formats.
2. **Resizing:** Images were resized to 224×224 pixels to match the input dimensions expected by the ResNet18 architecture.
3. **Data Augmentation:** To enhance model robustness and effectively increase the dataset size, the following augmentation techniques were applied to the training set:
   * Random horizontal and vertical flips with a probability of 0.5
   * Random rotations between -20 and +20 degrees
   * Random adjustments to brightness and contrast (±15%)
   * Random cropping followed by resizing to 224×224 pixels
4. **Normalization:** Pixel values were normalized using a mean of [0.5, 0.5, 0.5] and a standard deviation of [0.5, 0.5, 0.5] across all three color channels, scaling values to the range [-1, 1].
5. **Tensor Conversion:** Images were converted to PyTorch tensors for model training.

**Figure 4.1: Data Preprocessing Pipeline** [Diagram illustrating the sequential steps in the preprocessing pipeline]

## 4.3 Model Architecture and Training

The disease detection model was based on the ResNet18 architecture, modified for the specific three-class classification task:

1. **Base Architecture:** The pre-trained ResNet18 model was imported from torchvision.models, with weights initialized from training on ImageNet.
2. **Architecture Modification:** The final fully connected layer was replaced with a new linear layer having three output neurons (corresponding to the three classes), with weights randomly initialized.
3. **Training Strategy:** A transfer learning approach was employed where all layers of the model were fine-tuned. The training process consisted of the following steps:
   * Loss Function: Cross-entropy loss was used as it is well-suited for multi-class classification.
   * Optimizer: Adam optimizer with a learning rate of 0.0001 and weight decay of 0.0001 for regularization.
   * Learning Rate Scheduler: ReduceLROnPlateau scheduler was implemented to reduce the learning rate by a factor of 0.1 when validation loss plateaued for 5 epochs.
   * Batch Size: 32 images per batch were used during training.
   * Early Stopping: Training was set to terminate if validation loss did not improve for 10 consecutive epochs.
4. **Training Process:** The model was trained for 50 epochs, with early stopping typically activating around 30-35 epochs. For each epoch, the entire training set was processed, followed by evaluation on the validation set.
5. **Model Selection:** The model weights from the epoch with the lowest validation loss were saved as the final model.

## 4.4 Model Evaluation

The trained model was evaluated using the following metrics and approaches:

1. **Accuracy:** The overall accuracy was calculated as the ratio of correct predictions to the total number of images in the test set.
2. **Per-Class Metrics:** For each class, precision, recall, and F1-score were calculated to assess the model's performance across different diseases.
3. **Confusion Matrix:** A confusion matrix was generated to visualize the model's prediction patterns and identify any systematic misclassifications.
4. **Confidence Analysis:** The distribution of prediction confidence scores was analyzed to determine an appropriate threshold for the "Other Disease" classification.

Based on the confidence analysis, a threshold of 0.65 (65%) was established. Predictions with confidence below this threshold would be classified as "Other Disease," indicating that the model is uncertain and recommending users to seek expert advice.

## 4.5 Backend Development

The backend was developed using FastAPI to serve the trained model and process image uploads:

1. **API Structure:** A RESTful API was designed with a single endpoint (/predict/) for image upload and disease prediction.
2. **Model Loading:** The trained model was saved in PyTorch format (.pth) and loaded during application startup.
3. **Image Processing:** Uploaded images were processed using the same preprocessing steps as during training (resizing, normalization, etc.).
4. **Prediction Logic:** The preprocessed image was passed through the model to generate class probabilities. Additional logic was implemented to apply the confidence threshold and determine the final prediction.
5. **Response Format:** The API response was structured to include:
   * Predicted disease label
   * Confidence score
   * Confidence threshold
   * Probabilities for all classes
   * Treatment recommendations based on the detected disease
6. **Error Handling:** Comprehensive error handling was implemented to manage cases such as invalid file formats, model loading failures, and processing errors.
7. **CORS Configuration:** Cross-Origin Resource Sharing (CORS) middleware was configured to allow requests from the frontend application.

## 4.6 Frontend Development

The frontend application was developed using React.js with a focus on usability and responsiveness:

1. **Application Structure:** The application was structured into the following main pages:
   * Home: Introduction to the application and its features
   * Disease Detection Tool: Interface for uploading and analyzing leaf images
   * Disease Information: Educational content about black pepper diseases
   * About: Information about the project and its objectives
2. **Component-Based Architecture:** The UI was broken down into reusable components such as:
   * Navbar: Navigation component with theme toggle
   * Footer: Application footer with links and information
   * ConfidenceGauge: Visual representation of prediction confidence
   * ThemeToggle: Switch for dark/light mode preference
3. **State Management:** React Context API was used for global state management, particularly for theme preferences.
4. **Image Upload and Processing:**
   * A user-friendly interface was created for image upload with drag-and-drop support and preview functionality.
   * Form submission was handled using Axios for making requests to the backend API.
   * Loading states and error handling were implemented for a smooth user experience.
5. **Results Display:** The prediction results were presented in a clear, informative manner with:
   * Visual indication of the detected disease
   * Confidence gauge showing prediction certainty
   * Detailed breakdown of all class probabilities (expandable)
   * Specific treatment recommendations
6. **Responsive Design:** CSS media queries and flexible layouts were implemented to ensure the application works well on both desktop and mobile devices.
7. **Accessibility Considerations:** Semantic HTML elements, appropriate color contrast, and keyboard navigation were implemented to enhance accessibility.
8. **Theme Support:** A theme context provider was created to support both light and dark modes, with user preferences saved to local storage.

## 4.7 System Integration

The frontend and backend components were integrated to create a cohesive application:

1. **Environment Configuration:** Development environments were configured with appropriate CORS settings to allow communication between frontend and backend during development.
2. **API Integration:** The frontend was configured to communicate with the backend API using Axios for HTTP requests.
3. **Error Handling:** Comprehensive error handling was implemented at both frontend and backend levels to ensure graceful degradation in case of failures.
4. **Testing:** Integration testing was performed to ensure that the components worked together as expected, with particular attention to:
   * Image upload functionality
   * Prediction accuracy
   * Response handling
   * User interface responsiveness

**Figure 4.2: System Architecture** [Diagram showing the system architecture with frontend, backend, and model components]

# Chapter 05: Results and Discussion

## 5.1 Model Performance

The trained ResNet18 model demonstrated strong performance on the black pepper leaf disease detection task. The final model achieved the following metrics on the test dataset:

* **Overall Accuracy:** 91.67%
* **Precision, Recall, and F1-Score by Class:**
  + Healthy: Precision 96.55%, Recall 100.00%, F1-Score 98.25%
  + Leaf Blight: Precision 89.29%, Recall 89.29%, F1-Score 89.29%
  + Yellow Mottle Virus: Precision 89.66%, Recall 92.86%, F1-Score 91.23%

The confusion matrix revealed that most misclassifications occurred between the Leaf Blight and Yellow Mottle Virus classes, which is understandable given that both conditions can present with similar visual symptoms in certain stages.

The model training process showed a steady improvement in accuracy and reduction in loss over the training epochs. The validation loss started to plateau around epoch 30, with the early stopping mechanism activating at epoch 35.

**Figure 5.1: Model Accuracy and Loss Curves** [Graph showing training and validation accuracy/loss over epochs]

## 5.2 Confidence Threshold Analysis

A critical aspect of the system's design was determining an appropriate confidence threshold to identify predictions that might be unreliable. Analysis of the confidence distribution across test images revealed:

* Correctly classified images typically had confidence scores above 0.80
* Misclassified images often had confidence scores between 0.40 and 0.65
* Few predictions fell in the 0.65-0.80 range

Based on this analysis, a confidence threshold of 0.65 (65%) was established. Predictions with confidence below this threshold are classified as "Other Disease," prompting users to seek expert advice. This threshold was selected to minimize false positives while still maintaining high recall for actual disease cases.

**Figure 5.2: Confidence Distribution for Each Class** [Graph showing the distribution of confidence scores for each class]

The implementation of this threshold mechanism resulted in approximately 7% of test cases being classified as "Other Disease." Manual review of these cases confirmed that they were indeed challenging to classify, often due to ambiguous symptoms or non-optimal image quality.

## 5.3 System Functionality

The integrated web application successfully delivered the following functionalities:

1. **Image Upload and Processing:**
   * Users can upload leaf images through a drag-and-drop interface or file selection
   * The system provides visual feedback during image upload and processing
   * Error handling for invalid file types and processing failures
2. **Disease Detection:**
   * Accurate classification of healthy leaves, Leaf Blight, and Yellow Mottle Virus
   * Implementation of the confidence threshold for uncertain predictions
   * Clear presentation of results with visual indicators
3. **Treatment Recommendations:**
   * Contextual, eco-friendly treatment suggestions based on the detected condition
   * Specific recommendations for each disease category
   * General advice for "Other Disease" classification
4. **Educational Resources:**
   * Comprehensive information about black pepper diseases, symptoms, causes, and prevention
   * Visual references for each condition
   * Maintenance recommendations for healthy plants
5. **User Experience Enhancements:**
   * Responsive design for both desktop and mobile devices
   * Dark mode support for better visibility in different lighting conditions
   * Intuitive navigation between application sections

The backend API demonstrated stable performance with an average response time of 1.2 seconds for image processing and prediction, which is acceptable for the intended use case.

## 5.4 User Interface Evaluation

The user interface was evaluated based on usability principles and feedback from a small group of potential users, including farmers from the Matale region. Key findings include:

1. **Home Page:** The landing page effectively communicates the purpose of the application and guides users to key features. The visual design with images of black pepper plants helps establish context.

**Figure 5.3: Homepage Interface** [Screenshot of the application homepage]

1. **Disease Detection Tool:** The upload interface was found to be intuitive, with clear instructions and visual feedback. The results display was particularly appreciated for its clarity and the confidence gauge that helps users

**Figure 5.4: Disease Detection Tool Interface** [Screenshot of the disease detection interface]

1. **Disease Information Pages:** The educational content was rated highly for its comprehensiveness and accessibility. Users particularly appreciated the visual comparisons between different disease symptoms and the detailed prevention strategies.

**Figure 5.5: Disease Information Page** [Screenshot of disease information page]

1. **Mobile Responsiveness:** Testing on various mobile devices confirmed that the application functions well across different screen sizes. The responsive design ensures that farmers can use the application effectively in the field using their smartphones.
2. **Usability Testing:** A small-scale usability test was conducted with 8 black pepper farmers from the Matale region. The test involved asking participants to perform specific tasks such as uploading an image and interpreting the results. Key findings included:
   * 7 out of 8 participants were able to successfully complete all tasks without assistance
   * The average time to upload an image and receive results was 1.8 minutes
   * All participants expressed that they would use the application in their farming practice
   * The confidence gauge was initially confusing to 2 participants, suggesting a need for clearer explanations

## ****5.5 Limitations****

Despite the promising results, several limitations of the current system were identified:

1. **Dataset Limitations:** The training dataset, while balanced, was relatively small (819 images). This may limit the model's ability to generalize to new images captured under significantly different conditions than those represented in the training data.
2. **Disease Coverage:** The current system only covers two common diseases (Leaf Blight and Yellow Mottle Virus) and healthy leaves. Other diseases affecting black pepper plants, such as Phytophthora Foot Rot and Stunted Disease, are not currently detected.
3. **Internet Dependency:** The web application requires an internet connection for operation, which may be problematic in rural areas with limited connectivity. During field testing, intermittent connectivity issues were observed in some remote farming locations.
4. **Image Quality Sensitivity:** The model's performance is dependent on the quality of uploaded images. Poor lighting conditions, extreme angles, or blurry images can significantly reduce detection accuracy.
5. **Language Limitations:** The current implementation is only available in English, which may present accessibility barriers for farmers who primarily speak Sinhala or Tamil.
6. **Disease Severity Assessment:** The system currently only identifies the presence of a disease but does not quantify its severity or stage of progression, which would be valuable information for treatment planning.

These limitations provide valuable directions for future improvements and extensions of the system.

# Chapter 06: Future Work

This research has successfully developed an automated system for detecting black pepper leaf diseases using deep learning technology integrated with a user-friendly web application. The system addresses a critical challenge faced by black pepper farmers in the Matale region of Sri Lanka – the timely and accurate identification of plant diseases that significantly impact crop yield and quality.

The research has demonstrated that deep learning approaches, specifically ResNet18 architecture with transfer learning, can effectively distinguish between healthy black pepper leaves and those affected by Leaf Blight or Yellow Mottle Virus with high accuracy (91.67%). The implementation of a confidence threshold mechanism (set at 65%) represents an important innovation that prevents potentially harmful misdiagnosis by classifying uncertain predictions as "Other Disease" and recommending expert consultation.

The web application component of the system successfully transforms a complex deep learning model into an accessible tool for farmers with basic digital literacy. The user interface evaluation and usability testing confirmed that the system is intuitive and meets the needs of its intended users. The inclusion of educational resources about disease symptoms, causes, and eco-friendly treatment recommendations enhances the application's value beyond mere disease detection.

The potential impact of this system extends beyond individual farmers to the broader agricultural economy of the region. By enabling early detection and appropriate intervention, the system can help reduce crop losses, decrease unnecessary pesticide usage, and promote sustainable farming practices. The educational component of the application contributes to knowledge dissemination and capacity building within the farming community.

The limitations identified in this research provide clear directions for future improvements and extensions of the system. Expanding the disease coverage, enhancing the model's robustness to image quality variations, adding offline functionality, and incorporating multilingual support would significantly increase the system's utility and accessibility.

In conclusion, this research demonstrates the potential of combining artificial intelligence and web technologies to create practical solutions for agricultural challenges in developing regions. The black pepper leaf disease detection system represents a step toward more accessible and sustainable agricultural practices, leveraging technology to empower farmers with knowledge and tools previously available only to agricultural experts.

# Chapter 07: Conclusion

Based on the outcomes and limitations identified in this research, several promising directions for future work have been identified:

## 7.1 Expanding Disease Categories

A key enhancement would be expanding the system to detect additional black pepper diseases beyond the current three categories. Potential additions include:

* Phytophthora Foot Rot
* Stunted Disease
* Anthracnose
* Slow Decline (Slow Wilt)
* Pollu Disease (Berry Disease)

This expansion would require collecting and annotating additional image data for these diseases, potentially through collaboration with agricultural research institutions in Sri Lanka. A hierarchical classification approach could be implemented to first distinguish between leaf, stem, and berry diseases before specific diagnosis.

## 7.2 Mobile Application Development

While the current web application is responsive and usable on mobile devices, developing a dedicated mobile application would offer several advantages:

* Offline functionality for areas with limited connectivity
* Direct access to the device camera for improved image capture
* Push notifications for disease alerts and treatment reminders
* Integration with device GPS to map disease occurrences geographically

A cross-platform framework like React Native or Flutter could be used to develop applications for both Android and iOS devices, maximizing accessibility for farmers with different devices.

## 7.3 Offline Functionality

Implementing offline functionality would significantly enhance the system's utility in rural areas with limited internet connectivity. This could be achieved through:

* A lightweight version of the model that can run on mobile devices without internet connection
* Progressive Web App (PWA) implementation with service workers to cache essential application components
* Synchronization features that update the local model and educational content when connectivity becomes available

## 7.4 Integration with Agricultural Management Systems

The disease detection system could be integrated with broader agricultural management systems to provide more comprehensive support to farmers. Potential integrations include:

* Weather data services to correlate disease occurrence with environmental conditions
* Farm management software for tracking treatments and outcomes
* Community platforms where farmers can share experiences and solutions
* Supply chain systems to connect farmers with suppliers of eco-friendly treatments

## 7.5 Enhanced Model Features

Several technical enhancements could improve the model's performance and utility:

* Disease severity assessment to quantify the progression of detected diseases
* Multi-part plant disease detection to identify diseases affecting different parts of the plant (leaves, stems, berries)
* Temporal analysis to track disease progression over time with multiple images
* Ensemble models combining different architectures for improved accuracy
* Exploration of more efficient architectures for potential edge deployment

## 7.6 Multilingual Support

Implementing multilingual support would make the application more accessible to farmers across Sri Lanka. Priority languages would include:

* Sinhala
* Tamil
* Other regional languages relevant to black pepper growing regions

This would involve not only translating the interface but also ensuring that educational content and treatment recommendations are culturally appropriate and use terminology familiar to local farmers.

## 7.7 Environmental Factors Analysis

Integrating environmental data collection with disease detection could provide valuable insights into disease development conditions. Future versions could:

* Prompt users to input environmental conditions (temperature, humidity, rainfall)
* Analyze correlations between environmental factors and disease occurrence
* Provide preventive recommendations based on forecasted environmental conditions

These future directions would enhance the system's impact on black pepper cultivation in Sri Lanka and potentially extend its applicability to other regions and crops facing similar challenges.

# References