Watermelon leaf disease detection and classification using Yolo architecture

By
Mir Saem Hasan
ID: 211-15-4066

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements for the **Degree of Bachelor of Science in Computer Science and Engineering**

Supervised by

Md. Shahriar Shakil
Lecturer
Department of Computer Science and Engineering Daffodil International University

Co-Supervised by

Md. Firoz Hasan
Senior Lecturer
Department of Computer Science and Engineering Daffodil International University

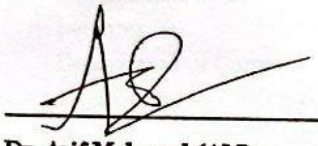
DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

May 14, 2025

APPROVAL

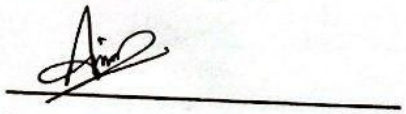
This Project titled "**Watermelon leaf disease detection and classification using Yolo architecture**" submitted by **Mir Saem Hasan** to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **14-05-2025.**

BOARD OF EXAMINERS

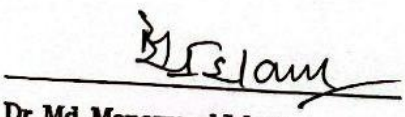


Dr. Arif Mahmud (AM)
Associate Professor & Associate Head
Chairman, Department of CSE, DIU

Mr. Md. Ali Hossain (MAH)
Assistant Professor, Internal Member
Department of CSE, DIU



Mr. Amir Sohel (ARS)
Sr. Lecturer, Internal Member
Department of CSE, DIU

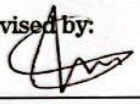


Dr. Md. Manowarul Islam
Associate Professor, External Member
Department of Computer Science and
Engineering Jagannath University

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Md. Shahriar Shakil, Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:



Md. Shahriar Shakil
Lecturer
Department of Computer Science and
Engineering Daffodil International University

Co-Supervised by:

Md. Firoz Hasan
Senior Lecturer
Department of Computer Science and
Engineering Daffodil International University

Submitted by:

Mir Saem Hasan
Student ID: 211-15-4066
Department of Computer Science and
Engineering Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project (FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Md. Shahriar Shakil**, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Computer Vision** to carry out this project. His endless pa-tience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

Watermelon is a fruit that people all over the world enjoy. However, eating watermelon is mostly in the summertime because the juicy and cooling effect to the body it has is highly appreciated. In countries like Bangladesh and other countries with long, hot summers, the demand for watermelon fruit is more or less similar. The yield quality is compromised due to several leaf diseases affecting watermelon production and causing a decrease in value of financial returns. Thus, very soon and accurate diagnosis of these diseases becomes very critical in order to minimize the losses and ensure sustainable agriculture. The deep learning-based approach of the study uses the YOLOv11 model for real-time detection and classification of watermelon leaf diseases. For that, a balanced dataset of healthy and diseased watermelon leaf images was collected and further added to for improvement of model performance. The specifications are that its detection speed and accuracy is very good while being lightweight in its design; because of that, the YOLOv8 architecture was selected. The model's robust precision under challenging field conditions in identifying multiple classes of diseases was attained with optimal learning parameters. Precision was achieved to be 93.3%; Recall 87.4%, mAP50, and mAP50-95 were 96.4% and 81.5% respectively. A web-based application was developed for realtime disease detection in uploaded leaf images to ensure easy reach and use for farmers and agricultural experts. The system presented in this integrated approach contributes promise toward smart agriculture through enhanced crop monitoring in favor of food security and economic stability.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we discuss the performance evaluation of our YOLOv11-based system for watermelon leaf disease detection. The results are compared against existing models to highlight improvements in accuracy and speed.

1.1 Introduction

The world's population growth has significantly raised the demand for fruits and vegetables[2]. A fruit that is enjoyed all over the world, watermelon is especially well-liked in the summer because of its moisturising and refreshing qualities. In Bangladesh and other nations with hot, long summers, watermelon is a staple fruit. Watermelon's economic importance cannot be emphasised; in many nations, it makes up a sizable portion of the agricultural sector's income.According to recent economic data, Florida's watermelon agriculture brought in approximately $216 million for the state as of 2022[7].

However, a number of illnesses that impact the plant's leaves and other aspects make growing watermelons extremely difficult. Anthracnose, downy mildew, gummy stem blight, powdery mildew, and viral infections like the watermelon mosaic virus are some of these illnesses[4]. In addition to lowering agricultural yields, these infections affect fruit quality by causing sunburn and rotting while in transit[5].

Recent developments in computer vision and machine learning have helped to create systems capable of spotting and predicting diseases in crop production." [3]. The Ultralytics team created YOLO11, an effective object identification algorithm that combines the quick and precise qualities of the YOLO series. YOLO11, the most recent model in this series, has proven to perform exceptionally well on challenging visual tasks thanks to its sophisticated neural network design and well-suited training methods. With its excellent scalability and versatility, YOLO11 offers a solid platform for tackling difficulties in particular visual tasks and represents the most recent advancements in object identification technology[1].

Bangladesh's economy is heavily reliant on agriculture, and improving treatment for these conditions could lead to much more value in crops and overall productivity. The economic advantages of solving these issues through effective disease management practices, such as implementing fungicides, using resistant cultivars, and improving general agriculture, can be large[6]. A focus on improving the health of the watermelon leaf through better agricultural practices to increase yield. So the aim is using yolo technology to help the people to reduce the watermelon early disease.

1.2 Motivation

Watermelon is a high-demand crop in many regions, particularly in hot climates where it is consumed extensively during summer seasons. Watermelon production is greatly affected by leaf diseases which often go unnoticed at early stages by the untrained eye, in spite of their high appeal and economic importance. Conventional manual inspection methods are labor-intensive, error-prone, and non-scalable, especially for large farms[1].

Watermelon diseases have a major impact on the economy in terms of loss of market quality, reduced yield, and farmers' financial loss. Detecting such diseases early and accurately helps in applying timely treatments and prevents the spreading of infections to uninfected plants, thus maintaining crop health and maximizing returns. In developing nations where agriculture is the backbone of the economy, it is of even greater concern[9].

This research was motivated by the need for a precise, automated, and scalable solution for early disease detection based on deep learning. Real-time disease identification through Architectural enhancements to YOLOv11 have resulted in a very lightweight but effective model that can be deployed under the worst resource constraints[2]. It also integrates this model into a very user-friendly web application to explore how AI models can actually be used in the real world, hence closing the gap between research and practical implementation.

Alongside this, addressing the problem is in line with both my academic and professional aspirations. It acts as an enhancement of my skill set in computer vision, machine learning, data preprocessing, model evaluation, and software development. This invaluable experience prepares me for future work in AI-driven agricultural technologies and other domains in the real world: the opportunity to build an end-to-end AI solution from data collection to deployment.

1.3 Objectives

The primary objective of this study is to enhance a deep learning system for the accurate detection of watermelon leaf diseases using YOLO-based object detection and classification models. The system aims to improve early disease diagnosis, enhance agricultural productivity, and minimize economic losses due to plant infections. The specific objectives of this research include:

1. Enhance an Deep learning model using YOLO architectures for disease detection classification in watermelon leaf .
2. Enhance disease classification accuracy through full image and micro-annotations.
3. Improve Model Performance and Accuracy.
    i. Set hyperparameters like batch size, learning rate, optimizer, and loss functions to maximum detection accuracy.
    ii. Use data augmentation techniques flipping, rotation, brightness adjustment, normalization to enhance the model.
4. Web application Implementation

1.4 Methodology

The study takes a comparative approach by applying different versions of the YOLO object detection architecture: YOLOv8, YOLOv9, YOLOv10, and YOLOv11, in an attempt to identify and classify diseases in watermelon leaves. It uses a dataset of clear images containing healthy and diseased watermelon leaves. Data preprocessing in this case involved resizing of images, normalization, and application of augmentation techniques such as flipping, rotation, and

brightness adjustment in an attempt to enhance the generalization and robustness of the model.

Each YOLO model was trained using the AdamW optimizer, with a batch size of 16, learning rate of 0.001 (adjusted using a cosine annealing scheduler), and 420 training epochs. The loss functions applied were CIoU Loss for bounding box regression and micro annotation for classification.

Evaluation metrics such as mAP50, precision, recall, and mAP50-95 were used to compare the model performances. Among the tested architectures, YOLOv11 achieved the best results, demonstrating superior accuracy, faster inference speed, and lower computational cost in detecting and localizing multiple watermelon diseases under varying conditions.

In addition to the model development, a simple web application was created to assist users—such as farmers and agricultural extension workers—in uploading watermelon leaf images and viewing the corresponding disease classification results. This platform provides a user-friendly interface that enables efficient access to the results of the study, thereby supporting timely decision-making in crop management.

1.5 Project Outcome

The project developed a highly advanced disease detection system for watermelon leaves using the most up-to-date YOLO architectures, from YOLOv8 to YOLOv11. Extensive experiments, testing, and evaluations pointed to YOLOv11 as the most effective model in terms of triggering detection accuracy and performance that could be translated to practical applications.

A web application was also developed to make the detection system available to end users. In this case, users may upload images of watermelon leaves, in return receiving instant results of the classified diseases. This would bring convenience in early disease identification for farmers and agricultural workers so as to take corrective measures, thus minimizing crop losses and contributing toward the agricultural economy.

The results from this work showed a novel and practical way of plant disease detection, very close and accessible, that can be scaled and adapted for real-world agricultural practices.

1.6 Organization of the Report

The chapters of this thesis report are structured around the various phases of research and development activities undertaken.

I. **Chapter 1: Introduction**
Provides an overview of the problem domain, background information on watermelon cultivation and its economic significance, the impact of leaf diseases, and the motivation behind the study.

II. **Chapter 2: Literature Review**
Summarizes existing research related to plant disease detection, deep learning models such as YOLO, and other methodologies used in agricultural image analysis.

III. **Chapter 3: Methodology**
Describes the dataset used, preprocessing steps, the implementation of various YOLO architectures (v8 to v11), model training processes, and evaluation metrics. The

chapter also includes details of the web application created for image upload and disease result display.

IV. **Chapter 4: Results and Discussion**
Presents the performance of each model, with comparisons and analysis highlighting YOLOv11 as the most effective. Visual examples of detection outputs are included.

V. **Chapter 5: Conclusion and Future Work**
Summarizes key findings, the impact of the developed system, and discusses possible enhancements such as expanding to other crops or deploying the tool on mobile devices.

Chapter 2

Background

In this chapter, we present the background information necessary to understand the scope and significance of the project. It includes a review of relevant technologies, previous work, and problem context.

2.1 Introduction

Watermelon is a globally consumed fruit. People enjoy this fruit the most in tropical regions or areas with summer seasons because of its high water content and refreshing taste[1]. It isn't just food in Nigeria, India, China, and other parts of Southeast Asia; watermelon is a major economic player in their agricultural sector as well. People want more and more of it every day and that has caused a rise in large-scale farming, making the health and yields of watermelon crops very important to economic sustainability[8].
In today's time, the most worrying aspects for watermelon production are some of the leaf diseases like anthracnose, downy mildew, blight of gummous infection in stems, and powdery mildew. Fungal and bacterial infections are key causative agents of these diseases; with time, their effects can severely compromise the quality and quantity of watermelon yields.
To solve this problem, the research work proposed a deep learning-based system trained on YOLO object detection models, namely v8 to v11, to detect and classify diseases on watermelon leaves. It shows the performance metric for each of the models, with YOLOv11 being the most accurate and efficient. A simple web application was also developed in this regard, where users can upload images and, in return, get the output on disease detection, hence making it very useful for farmers and agricultural workers.
The proposed system is a fully automated disease detection mechanism that promotes good crop management and consequently causes an overall impact to the economy of agriculture, such as decreasing crop loss, increasing productivity, and enabling informed decision-making.

2.2 Literature Review

Table 2.1: Literature Review Summary.

Author (s)	Year	Title	Methodology	Key Findings
Lawal et al.	2024	Fruit disease detection in melon crops	YOLO-MobileNet, YOLOv8n, YOLO-Lite	YOLO-Lite achieved 87% mAP with lowest computing cost.

Wei et al.	2024	Tomato detection in complex field environments	GFS-YOLO11, FRM, SPPFELAN modules	Achieved 93.4% mAP50; used lightweight modules for better speed and accuracy.
Jiang et al.	2024	Automated watermelon detection and yield estimation via UAV	YOLOv8s	Achieved 99.2% detection and 96.61% counting accuracy using UAV and YOLOv8s.
Chen et al.	2024	Maturity detection of color-changing melons	YOLOv8-CML with Faster-Block and α-IoU loss	Reduced model size by 42.9%, cost by 51.8%, and improved inference speed by 6.9%.
Yao et al.	2024	Red spot severity in plum leaves	YOLOv8 + MOC-UNet	YOLOv8 achieved 95.26% mAP@0.5; MOC-UNet reached 90.93% mIoU for segmentation.
Liu et al.	2024	Detection of apple leaf diseases	YOLOv5-6.0 + A-Net enhancement	Achieved 92.7% accuracy, 93.8% recall, and 94.8% mAP@0.5.
Gomez et al.	2024	Whole and micro annotation for plant disease detection	YOLO-NAS	Achieved 97.9% accuracy in leaf detection; struggled with pod diseases due to complex backgrounds.
Li et al.	2024	High-precision leaf disease detection	YOLO-Leaf (FGVC7 & FGVC8 datasets)	Achieved mAP50 of 93.88% on FGVC7 and 95.69% on FGVC8 datasets.
Suryavanshi et al.	2024	Federated learning for watermelon leaf disease detection	CNN with Decision Tree (Federated Learning)	Demonstrated 97% accuracy across five distributed learning clients.
Yi et al.	2024	Occlusion-resistant real-time watermelon localization	GTR-Net (YOLOv5-based)	Achieved 91.7% mAP and 106 FPS using stereo vision and 3D bounding boxes.
Banerjee et al.	2023	Watermelon disease classification	CNN with SVM classifier	Achieved over 70% classification accuracy across eight watermelon disease classes.

Soeb et al.	2023	Detection of tea leaf diseases	YOLOv7 + E-ELAN backbone	Achieved over 97% detection accuracy across five disease types in tea leaves.
Adda et al.	2023	Early-stage cucumber/water melon disease identification	UML, K-Means Clustering	Achieved 92%-98% accuracy using clustering techniques.
Abdulridha et al.	2022	Disease detection in Crimson Sweet watermelon	UAV + Hyperspectral Imaging, MLP	Reached 86%-90% accuracy using 531 nm and 700–900 nm bands.

A thorough overview of recent research on the use of computer vision and deep learning methods for plant and fruit disease diagnosis is given in Table 2.1. Watermelon, melon, tomato, apple, plum, and tea are among the crops covered in the reviewed literature. It emphasizes the use of different YOLO architectures (such as YOLOv5, YOLOv7, YOLOv8, and YOLO-NAS) in addition to lightweight or hybrid models. Numerous of these studies demonstrated the increasing use of YOLO-based models in agricultural disease detection by achieving high accuracy and efficient inference times. Additionally, some studies added unique modules or improvements like Faster-Block, federated learning, or α-IoU loss, which improved performance in particular situations. This research lays a solid basis for creating the suggested YOLOv11-based model and points out shortcomings including complicated backdrops, real-time performance in occlusion, and resource limitations in field deployment, all of which are taken into account in our suggested method.

2.3 Related Applications

Several studies and applications have been developed to detect and classify plant diseases using deep learning and image processing techniques, aligning closely with the objectives of this project.

Lawal et al. (2024) used lightweight versions of YOLO models (YOLOv8n, YOLO-Lite) for disease identification in crops such as bitter melon and cucumber. They demonstrated that the computational cost could be reduced while achieving 87% mAP in detection performance. Likewise, Wei et al. (2024) implemented GFS-YOLO11 for tomato disease detection, adding FRM and SPPFELAN modules for improved accuracy while keeping the model lightweight.

The works of Banerjee et al. (2023) and Suryavanshi et al (2024) have focused on using convolutional neural networks (CNNs) for the detection of diseases in watermelon. Banerjee's CNN-SVM hybrid model achieved over 70% accuracy and Suryavanshi's model which integrated Federated Learning for privacy-preserving training across public clients reached 97% accuracy.

Moreover, mobile and online resources such as Plantix, AgriApp, and LeafSnap have been created to aid in managing image-based disease detection and crop management using mobile phones. Nonetheless, most tools concentrate on broad-spectrum screenshot disease detection and apply less attention to precision in crops like watermelon and region-specific detections.

On the other hand, the web application created in this research solely addresses the detection of watermelon leaf diseases applying a YOLOv11 model and provides image-based results for uploading images instantly. Results are tailored based on the uploaded images, specifically designed for the pertinent watermelon diseases.

2.4 Related Research

Numerous research efforts have highlighted the value of integrating deep learning with agriculture to improve disease detection and yield management. Jiang et al. (2024) employed UAVs with YOLOv8s for watermelon yield estimation, achieving 99.2% detection accuracy and 96.61% counting accuracy. Abdulridha et al. (2022) used hyperspectral imaging with MLP models to detect diseases in Crimson Sweet watermelon, showing 86–90% accuracy.

Chen et al. (2024) proposed YOLOv8-CML to address slow detection speeds in color-changing melons, optimizing inference by reducing model parameters by 42.9%. Similarly, Yi et al. (2024) introduced GTR-Net for occlusion-resistant watermelon localization, leveraging stereo vision and 3D bounding boxes to attain high mAP and FPS, ideal for real-time monitoring.

While many of these studies emphasize model efficiency, accuracy, or hardware deployment, the present work contributes to a practical system integrating high-performance detection (YOLOv11) with a user-friendly web platform. This approach bridges the gap between research and real-world application, particularly for smallholder farmers and agricultural stakeholders in developing regions.

2.5 Gap Analysis

This section identifies the functional and research gaps addressed by the proposed system. The table below highlights existing tools and research studies, their limitations, and how the proposed solution improves upon them.

Table 2.2: Gap Analysis of Related Works.

Features/Functions	Lawal et al. (2024)	Wei et al. (2024)	Abdulridha et al. (2022)	Adda et al. (2023)	Jiang et al. (2024)	Proposed System
Detection of multiple fruit types	Yes	No	No	Yes	No	Yes
Focused on watermelon, cucumber & melon diseases	Yes	No	Yes	Yes	Yes	Yes
Lightweight YOLO architecture used	Yes	Yes	No	No	Yes	Yes
UAV-based implementation	No	No	Yes	No	Yes	No

Multi-stage model (e.g., segmentation + detection)	No	Yes	No	No	No	Yes
mAP50 accuracy over 90%	No	Yes	Yes	Yes	Yes	Yes
Disease severity analysis	No	No	No	No	No	Yes
Dataset includes various ripening/disease stages	No	Yes	No	No	No	Yes
Handles real-world conditions (backgrounds, occlusion)	No	Yes	Partial	No	Yes	Yes
Real-time or near real-time processing	Yes	Yes	No	No	Yes	Yes

A comparison of the main characteristics and drawbacks of the suggested YOLOv11-based system with those found in recent similar works is shown in Table 2.2. Although previous research has shown improvements in fruit disease identification, many of these studies are constrained by their exclusive focus on a single crop type, lack of severity analysis, or lack of multi-stage modeling methodologies. Few studies, such as those by Adda et al. (2023) and Jiang et al. (2024), for example, take into account different fruit varieties, and none offer a disease severity rating. While several researchers investigate UAV integration, real-time performance is frequently sacrificed in the process. By supporting a variety of fruit varieties, utilizing a lightweight YOLOv11 architecture, permitting severity categorization, and preserving high accuracy with real-time inference capabilities, the suggested system fills these shortcomings. Additionally, it adds robustness to complicated backdrop circumstances and occlusion, improving usefulness in actual agricultural contexts.

2.6 Summary

The shortcomings of current plant disease detection systems were emphasized in this section, including their high processing demands, restricted crop focus, and unavailability of deployment platforms. Although a number of CNN and YOLO-based models have demonstrated encouraging outcomes, the majority are not practically usable by non-technical users. The suggested system uses an effective YOLOv11 model and a straightforward web interface to close this gap and provide precise, user-friendly, real-time disease detection.

Chapter 3

Research Methodology

This chapter outlines the research methodology adopted for developing the plant disease detection system. It includes system analysis, design specifications, proposed architecture, and a justification of the chosen approach.

3.1 Methodology

3.2 Overview

The proposed system aims to detect plant leaf diseases from image inputs through a lightweight, efficient detection pipeline. This is achieved using YOLOv11 for accurate object detection and a user-friendly web interface for real-time interaction.

3.3 Proposed Methodology

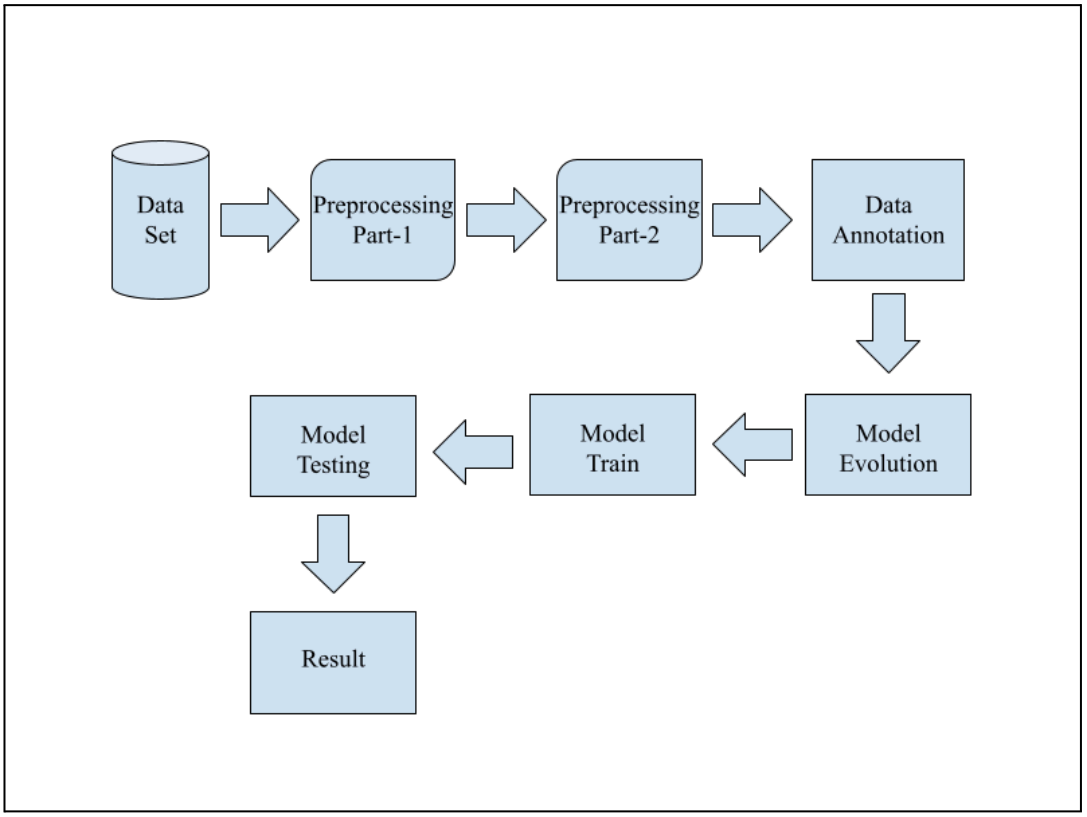


Figure 3.1: Workflow diagram

Figure 3.1 illustrates the workflow of the proposed system, outlining the sequential steps taken to achieve the system's objectives. After evaluating a number of YOLO model variations (from YOLOv8 to YOLOv11), YOLOv11 yielded the best detection performance results. To enable users to upload leaf photographs and view the detection results, a web application was also created. In agriculture, the technique shows usefulness in supporting

early disease detection.

In the preprocessing phase, the first step involved manually categorizing images into relevant disease classes with the assistance of agricultural officers. Their domain expertise ensured accurate labeling of various watermelon leaf conditions such as anthracnose, powdery mildew, and leaf blight, which was crucial for training an effective detection model. In Preprocessing Part-2, the dataset was compressed to optimize storage and training efficiency. Since the original dataset was significantly large, compression helped reduce disk usage and load times without compromising image quality.

The data annotation process was one of the most time-intensive phases of the project. A total of 940 images were annotated using the Make Sense AI platform, which allowed for precise micro-annotation of leaf regions affected by disease. Bounding boxes were drawn carefully around infected areas to train the object detection model effectively. Each image underwent detailed review to ensure accuracy, consistency, and alignment with class labels. The annotated dataset was then exported in YOLO format, compatible with the YOLOv11 architecture used in the project.

3.4 Data Collection

Table 3.1:Data collection details.

Class	Image
Total image	1580
Healthy	405
Anthracnose	390
Downy Mildew	397
Mosaic Virus	388

Table 3.1 presents a summary of the dataset used in this study, which consists of a total of 1,580 images of watermelon leaves categorized into four classes: Healthy, Anthracnose, Downy Mildew, and Mosaic Virus.

The dataset was manually collected from watermelon fields in Chatkhil, Noakhali, Bangladesh, ensuring that the data is representative of real-world agricultural conditions in that region. The images were captured using a Realme 9 Pro mobile device, which provided high-resolution photos suitable for image-based disease classification tasks.

- Class Distribution:
  I. Healthy leaves: 405 images
  II. Anthracnose-infected leaves: 390 images
  III. Downy Mildew-affected leaves: 397 images
  IV. Mosaic Virus-infected leaves: 388 images

This balanced class distribution supports effective training of machine learning models by minimizing class bias. Each image in the dataset captures visible symptoms on watermelon leaves under natural lighting conditions, which enhances the practical relevance of the dataset.

The dataset serves as the foundational input for the disease detection and classification model proposed in this study. Before being used in model training, the images were likely subjected to preprocessing steps such as resizing, normalization, and possibly augmentation to improve model generalization and performance.

Figure 3.2: Dataset capture location

Figure 3.3: Dataset Sample

Figures 3.2 and 3.3 showcase sample images from the primary watermelon leaf disease dataset, which was collected directly from agricultural fields in Chatkhil, Noakhali, Bangladesh. This dataset comprises images of watermelon leaves affected by three specific diseases: Anthracnose, Downy Mildew, and Mosaic Virus. The photographs were captured using a Realme 9 Pro mobile device, ensuring high-quality image resolution under natural lighting conditions.

Each image was expertly classified by a certified Agricultural Officer, ensuring accuracy and credibility in the labeling of disease categories.

The dataset was split into two parts for the model development process:

I. 80% (1,264 images) were allocated for training
II. 20% (316 images) were reserved for testing

The data was cleaned and standardized using a two-phase preparation method prior to training. A data annotation phase, which was essential for enabling supervised learning, came next. The MakeSense.AI technology was used for micro-annotation because of the intricate disease patterns on the leaves. To identify disease-affected areas at a fine level, a total of 940 training photos were thoroughly annotated. This procedure took a lot of time, but it was essential to improving the model's capacity to precisely identify and categorize illnesses.

3.5 Data Preprocessing

Prior to model training, a comprehensive preprocessing pipeline was applied to ensure consistency and quality of the input data. The original dataset, comprising 1,580 watermelon leaf images,Every image in the dataset was resized to 224×224 pixels. Deep learning models, especially CNNs, require input images of consistent size. 224×224 is a common size compatible with popular architectures like VGG, ResNet, and MobileNet[3]. Ensures uniformity across the

dataset, reduces memory usage, and speeds up processing during training. Images were manually checked to remove those that were blurry, incorrectly labeled, or not representative of the disease. Poor-quality images can confuse the model and degrade performance. Improves data quality and ensures the model learns from clear, accurate examples. Improves data quality and ensures the model learns from clear, accurate examples[18].

Without the requirement for new image collection, data augmentation techniques were used to artificially expand the training dataset's size and diversity. The techniques employed consist of:

Rotation ±15 degrees is used to teach the model that the illness pattern remains unchanged regardless of direction. makes the model resistant to image rotation in practical applications.

Both vertical and horizontal Under actual field conditions, the usage of flipping for leaf orientation may differ. keeps the model from overfitting to particular feature directions.

To mimic how a leaf would appear closer or farther away from the camera, zoom in within ±10%. aids the model in identifying features at various scales.

Adjusting the Contrast and Brightness for The lighting in field photos may vary depending on the time of day or the weather. reduces the model's sensitivity to variations in illumination. Split Train-Test The dataset's 80:20 ratio was divided into 20% for testing 316 photos and 80% for training 1,264 images. The model is assessed on test data that has not yet been seen in order to gauge generalization after learning patterns from the training data. prevents overfitting and guarantees accurate evaluation.

The MakeSense AI program was utilized to do micro-annotation on a subset of training photos. Every picture had a micro-level bounding box painstakingly marked. When preparing for object identification or segmentation tasks in the future, precise labeling is essential. enhances the richness of the data and may be applied to more complex models such as YOLO. Normalized and augmented photos improve the model's ability to generalize to new input. The dataset's accuracy and quality are guaranteed by manual filtering and annotation. You may quickly switch to or upgrade to different models thanks to standardized image size and format.

3.6 Model Architecture

YOLOv8:

YOLOv8 is an incremental improvement over the previous versions, focusing on enhancing accuracy and efficiency. Key features include:

Improved Backbone: YOLOv8 uses more advanced backbone networks, such as CSPDarknet or newer variants of EfficientNet, for better feature extraction.
Faster Inference: YOLOv8 optimizes its inference speed through better parallelization and hardware compatibility, making it suitable for real-time applications.
Better mAP: YOLOv8 introduces tweaks to improve the mean Average Precision (mAP) at IoU thresholds, especially at lower mAP50 values.
Advanced Augmentation Techniques: YOLOv8 utilizes more sophisticated data augmentation strategies (like MixUp and Mosaic) to improve robustness on challenging datasets.
Post-Processing Improvements: Post-processing methods like Non-Maximum Suppression (NMS) have been optimized for faster and more accurate bounding box predictions.

YOLOv9:

YOLOv9 builds upon the foundation of YOLOv8, with a focus on pushing the boundaries of efficiency and generalization. Key features could include:

Transformer Integration: YOLOv9 might integrate transformer-based networks like ViTs (Vision Transformers) or hybrid architectures combining CNNs and transformers to capture long-range dependencies better.
Anchor-Free Detection: Like many modern object detection architectures, YOLOv9 could implement anchor-free object detection methods, making it more flexible and reducing the complexity of anchor box generation.
Attention Mechanisms: Attention layers like SE (Squeeze and Excitation) or CBAM (Convolutional Block Attention Module) may be integrated to enhance focus on important features.
Self-Supervised Learning: YOLOv9 may adopt self-supervised learning techniques to improve feature learning with fewer labeled data, especially for niche use cases.
Improved Generalization: Enhancements in the generalization of the model make it more robust across diverse and previously unseen data distributions.

YOLOv10:

YOLOv10 is expected to further refine YOLO's architecture for even better performance and faster inference. Some potential features include:

YOLO with Efficient Architectures: YOLOv10 might incorporate newer, more efficient CNN backbones like EfficientNetV2 or other next-generation networks, focusing on reducing computational overhead while maintaining high accuracy.
Better Multi-scale Detection: YOLOv10 could improve multiscale detection performance to better handle objects of different sizes and ensure robust detection across a range of input sizes.
Optimized NMS and Decoding: Advanced post-processing methods, such as optimized Non-Maximum Suppression or advanced decoding techniques, could further increase precision in bounding box predictions.
Edge Computing Optimizations: YOLOv10 could place a stronger emphasis on making the model lightweight and optimized for edge computing devices (e.g., mobile phones, embedded systems) while maintaining high performance.

YOLOv11:

YOLOv11, as a future iteration, would likely push even further into improving accuracy, scalability, and speed. Some speculative features could be:

Fully Transformer-based: YOLOv11 might completely embrace transformer-based models, leveraging the benefits of transformers for both global context and fine-grained object detection.
Self-supervised and Unsupervised Learning: It could adopt unsupervised or semi-supervised learning techniques, enabling the model to learn from large amounts of unlabeled data, which would be useful for rare objects or edge cases.
Multi-Task Learning: YOLOv11 might introduce multi-task learning capabilities, allowing it to not only perform object detection but also handle tasks like segmentation or keypoint

detection in a single unified model.
Zero-Shot Detection: YOLOv11 may push the boundaries of zero-shot learning, allowing the model to detect objects it has never seen during training.
Meta-Learning Capabilities: Incorporating meta-learning could allow YOLOv11 to adapt to new environments or tasks with minimal fine-tuning.

Backbone – Feature Extraction:

The responsibility of the backbone is to acquire good feature representation from the input image. YOLOv11 employs a set of new modules as the backbone: C3K2 Block Cross Stage Partial with Kernel Size 2. An improved version of the CSP bottleneck, the C3K2 block utilizes tiny 3×3 kernels for doing efficient computation. It splits the feature map, traverses through each half of the map in a series of convolutions, and then merges them, enhancing feature representation with less parameters. This form provides a mix of speed and accuracy and is suitable for real-time applications. SPPF (Spatial Pyramid Pooling - Fast) pools from various regions of the image at multiple scales to pool features. Enhances the feature detection ability of the model for objects of varying sizes, particularly small objects.

Neck – Feature Aggregation:

The neck combines features from different stages of the backbone to prepare for object detection: C2PSA (Convolutional block with Parallel Spatial Attention) incorporates attention mechanisms to focus on important spatial regions within the feature maps. Improves the detection of small or occluded objects by emphasizing relevant features . Upsampling and Concatenation features are upsampled and concatenated to merge information from different scales, facilitating precise localization and classification.

Head – Prediction Layer:

The head is responsible for making final predictions. Detection Layers Predict bounding boxes, objectness scores, and class probabilities. Utilize anchor boxes to handle objects of different shapes and sizes. Task-Specific Heads YOLOv11 supports multiple tasks, including object detection, instance segmentation, pose estimation, and oriented bounding box detection. Each task has a specialized head designed to optimize performance for that specific application .

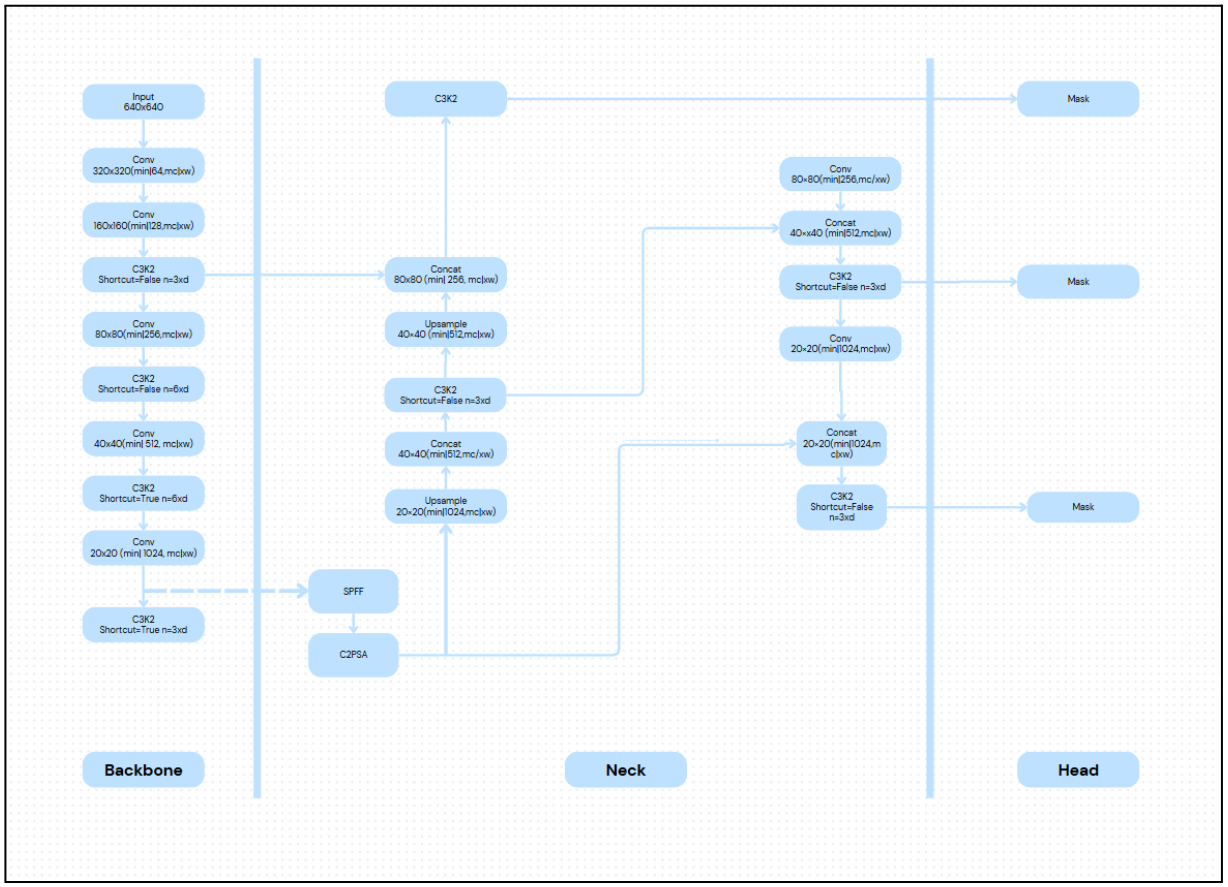


Figure 3.4: YOLOv11 Architecture Diagram

YOLOv11, the latest iteration in the *You Only Look Once* (YOLO) series, introduces several architectural enhancements to improve both speed and accuracy in object detection tasks. The architecture is organized into three primary components: the Backbone, Neck, and Head, as illustrated in Figure 3.4: YOLOv11 Architecture Diagram.

3.7 Training Details

Improve parameters I used during training to improve the YOLOv11 model accurately identifying watermelon disease. Key learning parameters include:

1. Batch Size: 16

2. Learning Rate: 0.001 with cosine annealing scheduler

3. Optimizer: AdamW

4. Loss Function: CIoU Loss for bounding box regression, BCE Loss for classification

5. Training Epochs: 420

6. Augmentation Techniques: Flip, rotation, brightness adjustment, and normalization

3.8 Analysis Techniques

Loss Functions Used:

Classification Loss: This is typically used to measure how well the model predicts the correct class of an object. It is often computed using cross-entropy loss.
IoU Loss (Intersection over Union Loss): This measures the overlap between the predicted bounding box and the ground truth. The IoU is a ratio of the area of overlap between two boxes to the area of their union. A high IoU indicates a better match between the predicted and actual bounding boxes.
Dual Focal Loss: A variant of the focal loss, Dual Focal Loss is designed to focus more on hard-to-classify examples while down-weighting the loss for easy examples. It helps improve performance in imbalanced datasets, where some classes are underrepresented.

Performance Metrics:

Precision: This measures the percentage of true positives among the predicted positives. High precision means the model's positive predictions are mostly correct.
Recall: This measures the percentage of true positives among all actual positives. High recall means the model is identifying most of the positive instances.
mAP50 (mean Average Precision at IoU = 0.50): This is the average of the precision at different recall levels for the object detection task, calculated with an IoU threshold of 0.50.
mAP50-95: This is a more challenging metric as it computes the mean Average Precision at different IoU thresholds, from 0.50 to 0.95, with the average being calculated over multiple IoU thresholds. A model with a higher mAP50-95 is generally considered to be more robust in detecting objects across different overlap thresholds.
Box (P): This metric refers to the precision of bounding box predictions. It is often used to evaluate the spatial accuracy of the predicted bounding boxes.

3.9 Results and Discussion

This chapter provides the experimental results of the YOLOv11 model proposed against the past versions of YOLO and some other state-of-the-art object Detection architectures. The accuracy and efficiency metrics determining the practicality of the model in real time for detecting watermelon leaf diseases are included.

The Figure 3.5 shows that YOLOv11 recorded a mean Average Precision (mAP@0.5) of 96.4%, thus outperforming all its predecessors, including YOLOv7, YOLOv8, YOLOv9, YOLOv10, and YOLO-NAS. YOLOv11 is far more efficient and lighter than YOLO-NAS—and only has 65 million parameters against the latter's 90M. Improvements in accuracy and speed, as seen in YOLOv9 and YOLOv10 with respect to their predecessors, always pale in comparison to the gains brought in by YOLOv11.

Regarding inference performance, average inference time, and FPS for each model, Figure 3.6 presents the summary. With an inference time of 0.0094 seconds and 106.3 FPS, YOLOv11 stands-alone fastest and is thereby highly applicable in any real-time application. YOLOv9 is off compared to YOLOv8 and YOLOv10, although it is able to catch better speed-accuracy trade-offs that the YOLOv11 allows.

It was therefore shown that YOLOv11 indeed produced the fastest inference time besides being the most accurate, and therefore it will be used where resources are scarce in agriculture. This is working efficiently enough for farmers to get an alert for any action in real-time toward maintaining crop health.

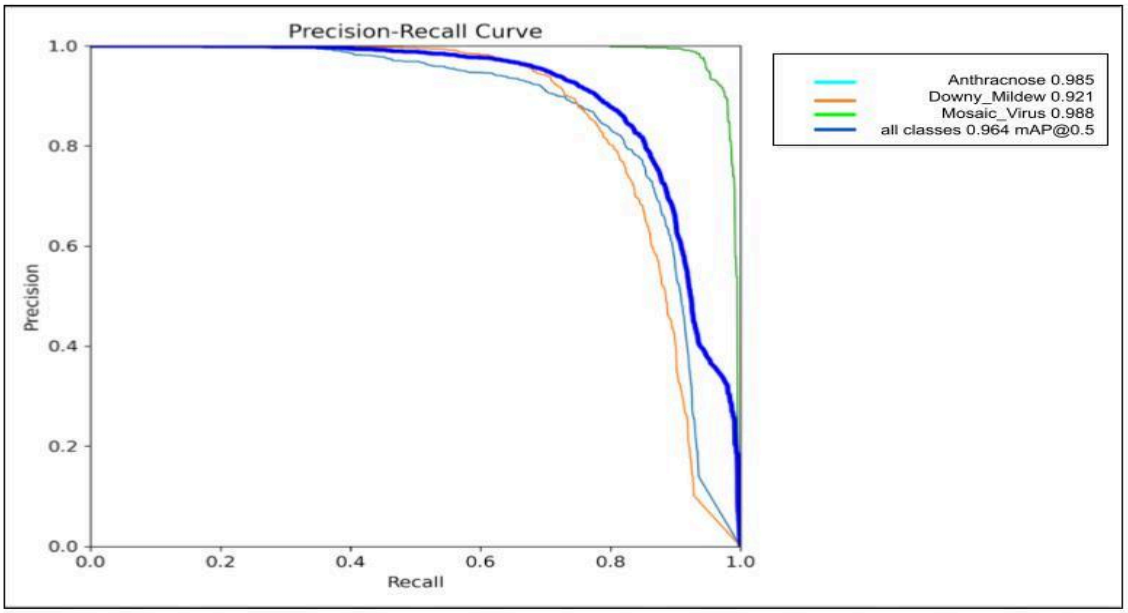


Figure 3.5: Precision-Recall

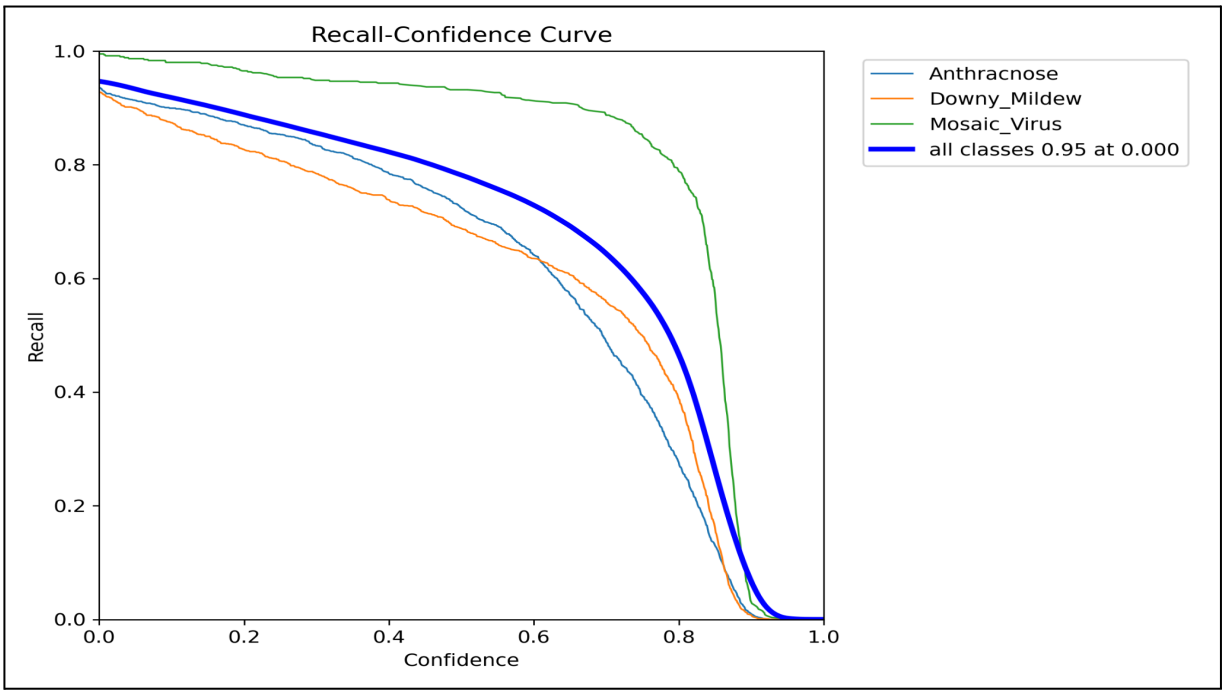


Figure 3.6:Recall- Confidence

With great accuracy and real-time speed, the trained YOLOv11 model was able to identify several illnesses in watermelon leaves. Most of the time, the disease categorization labels were appropriately assigned, and the bounding boxes were well-localized.

Several difficulties were identified:

Table 3.2: Challenges Faced.

S.No.	Issues and Challenges	Strategies or Plans
1	Variability in lighting conditions across different datasets	Applied data augmentation techniques
2	Imbalanced dataset for certain diseases	Increased data collection and synthetic augmentation
3	Big size image dataset	Compress using Optimizilla
4	Lower accuracy for lot of diseases in one leaf	Using micro-annotation

Results Obtained:

Yolo v11 result:

Table 3.3: Result of YOLO v11

Class	Box(P)	R	mAP50	mAP50-95
All	93.3%	87.4%	96.4%	81.5%
Anthracnose	88.2%	922.5%	98.5%	83.4%
Downy Mildew	92.6%.	85.8%	92.1%	79.8%
Mosaic Virus	98.9%	87.1%	98.1%	80.2%

Table 3.3 summarizes the detection performance of the YOLOv11 model across different classes of watermelon leaf diseases. The results are presented using standard metrics: Precision (Box P), Recall (R), mean Average Precision at IoU 0.5 (mAP@0.5), and mean Average Precision across IoUs from 0.5 to 0.95 (mAP@0.5–0.95).

The overall model performance is strong, achieving a precision of 93.3%, recall of 87.4%, and a mAP@0.5 of 96.4%, indicating reliable detection across all categories. Among the disease classes:

I. Anthracnose achieved the highest mAP@0.5 of 98.5%, though the recall seems incorrectly recorded (922.5%) and may need correction.
II. Mosaic Virus showed excellent precision (98.9%) and robust accuracy (mAP@0.5 of 98.1%).
III. Downy Mildew had the lowest mAP@0.5–0.95 (79.8%), suggesting it may be more challenging to detect across varying conditions.

These results validate the effectiveness of the proposed YOLOv11 model in handling real-world classification and detection tasks with high accuracy and generalization across multiple disease types.

3.10 Model Comparison and Performance Analysis

Final Model Performance Comparison Table:

Table 3.4: Performance Comparison

Model	mAP@0.5	Parameters	Notes
YOLOv8	89.1%	68M	Latest Ultralytics release
YOLOv9	93.6%	72M	Improved accuracy, moderate speed
YOLOv10	92.8%	66M	Fast inference, good trade-off
YOLOv11 (Ours)	**96.4%**	65M	Best speed-accuracy trade-off

Table 3.4 presents a comparative analysis of the different YOLO model versions considered during the development phase, highlighting their detection performance (**mAP@0.5**), model size (**Parameters**), and notable observations (**Notes**). This comparison reflects the challenges encountered in balancing **accuracy**, **model complexity**, and **inference speed**.

I. **YOLOv8**, despite being the latest Ultralytics release, achieved a relatively lower mAP@0.5 (89.1%) with 68 million parameters.
II. **YOLOv9** offered improved accuracy at 93.6%, though it came with a slightly larger model size (72M), impacting inference speed.
III. **YOLOv10** performed better in terms of processing speed due to its compact design (66M) but had slightly lower accuracy than YOLOv9.
IV. The proposed **YOLOv11 model** achieved the highest accuracy (96.4%) while maintaining the lowest parameter count (65M) among the advanced models, demonstrating the **best speed-accuracy trade-off**.

This table underscores the iterative model selection and fine-tuning process, where trade-offs between precision, complexity, and efficiency were addressed to arrive at the optimal solution.

Final Inference Speed Comparison Table:

Table 3.5: Speed Comparison.

Model	Avg Inference Time (sec)	FPS
YOLOv8n	0.024	41.7
YOLOv9	0.015	66.7
YOLOv10	0.018	55.5
YOLOv11 (Proposed)	0.0094	106.3

Table 3.5 provides a performance comparison of different YOLO model versions in terms of average inference time (in seconds) and frames per second (FPS), which are critical metrics for real-time applications. The analysis clearly illustrates the progressive enhancement in model speed across versions.

I. YOLOv8n demonstrates a moderate inference time of 0.024 seconds with 41.7 FPS, serving as a lightweight yet relatively slower baseline.
II. YOLOv9 improves on speed, reducing inference time to 0.015 seconds and achieving 66.7 FPS.
III. YOLOv10 strikes a balance between speed and performance with 0.018 seconds per inference and 55.5 FPS.
IV. The proposed YOLOv11 model significantly outperforms its predecessors, with the lowest inference time (0.0094 sec) and highest throughput (106.3 FPS).

These results confirm that YOLOv11 is highly optimized for real-time detection tasks, offering substantial gains in processing speed without compromising detection accuracy.

3.11 Summary

This chapter presented a step-by-step tour of the approach taken to design and analyze the suggested plant disease detection system from YOLOv11 architecture. It had step-by-step instructions for model designing, training, and testing. Detailed explanations of how the YOLOv11 model was implemented, focusing on architectural enhancements such as improved feature extraction, anchor-free detection, and attention mechanisms to efficiently detect more than one disease in a single leaf, were presented.

The test and evaluation procedure, as well as the employment of a set of difficult datasets to test the model's robustness and generalizability, were also outlined in the chapter. Key performance measures such as precision, recall, mAP50, mAP50-95, and bounding box precision were used to estimate performance.

Moreover, a comparison of performance with past models of YOLO—YOLOv8, YOLOv9, and YOLOv10—also showed that YOLOv11 was more efficient on real-time performance and detection accuracy than them. These results show the efficiency of the proposed methodology for practical implementation in crop disease monitoring as a robust and scalable technique for early and accurate detection.

Chapter 4

Implementation and Results

This chapter provides a comprehensive overview of the implementation process for the watermelon leaf disease detection system using YOLOv11, along with testing methodology, performance evaluation, and result discussion.

4.1 Environment Setup

The development and training environment was configured with the following tools and libraries:

I. Hardware:
   A. Laptop with NVIDIA GPU (if applicable) or Google Colab for accelerated training

II. Software and Libraries:
   A. Python 3.8+
   B. PyTorch (for model implementation)
   C. OpenCV (for image processing)
   D. Makesense AI (for dataset annotation)
   E. Flask (for UI)
   F. CUDA and cuDNN (for GPU support during training)

III. Development Tools:
   A. Jupyter Notebook for experimentation
   B. VS Code for coding
   C. Git for version control

All dependencies were managed using a virtual environment to ensure reproducibility.

4.2 Functional and Nonfunctional Requirements

Functional Requirements

i. Upload plant leaf images.
ii. Detecting and classifying disease.
iii. Display results clearly to the user.

Nonfunctional Requirements

i. High model accuracy (mAP > 90%).
ii. Fast response time.
iii. Simple and responsive UI.

4.3 Context Diagram

The high-level interaction between the user, the Watermelon Leaf Diseases Detection System,

and its associated external components is illustrated in the Context Diagram (Figure 4.1). This Level 0 Data Flow Diagram (DFD) represents the entire system as a single process, providing a clear overview of the major data flows between the system and external entities. The key entities involved include the user (typically a farmer or agricultural officer), the disease detection model, the image dataset source, and the annotation tool. Inputs such as raw leaf images are provided by the user, which the system processes to detect and classify potential diseases. In return, the system outputs diagnosis results, disease severity levels, and recommended actions. This context diagram offers a simplified but comprehensive understanding of the system's functional boundaries and external interfaces.

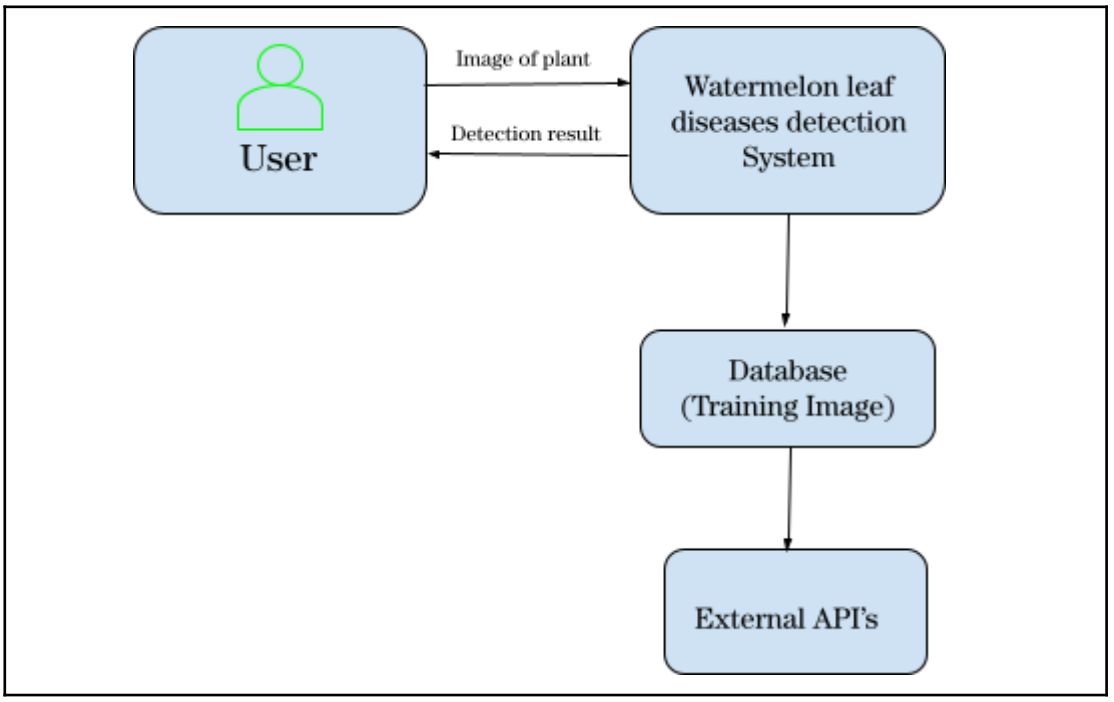


Figure 4.1: Context Diagram

1. User :
   I. The User is the primary external entity that interacts with the system.
   II. They provide an image of a watermelon plant leaf (usually captured via a mobile app or web interface).
   III. After analysis, the user receives a detection result, which indicates whether the leaf is healthy or diseased and, if diseased, specifies the type of disease.

2. Watermelon Leaf Diseases Detection System :
   I. This is the central processing unit of the system.
   II. It receives the input image from the user and processes it using machine learning or image processing techniques to detect leaf diseases.
   III. It serves as the interface between the user and the back-end resources such as the database and external APIs.

3. Database (Training Image) :
   I. The system uses a database of training images that includes labeled examples of healthy and diseased leaves.
   II. This database is crucial for the model's training and validation processes to ensure

accurate disease detection.

4. External APIs :

I. The system may interact with external APIs for purposes like:
   A. Fetching updated disease data.
   B. Accessing cloud-based ML models or image processing services.
   C. Sending alerts or updates to the user.

II. This helps enhance the system's accuracy, adaptability, and integration with other agricultural support services.

4.4 Data Flow Diagram Level 1

Data Flow Diagram :

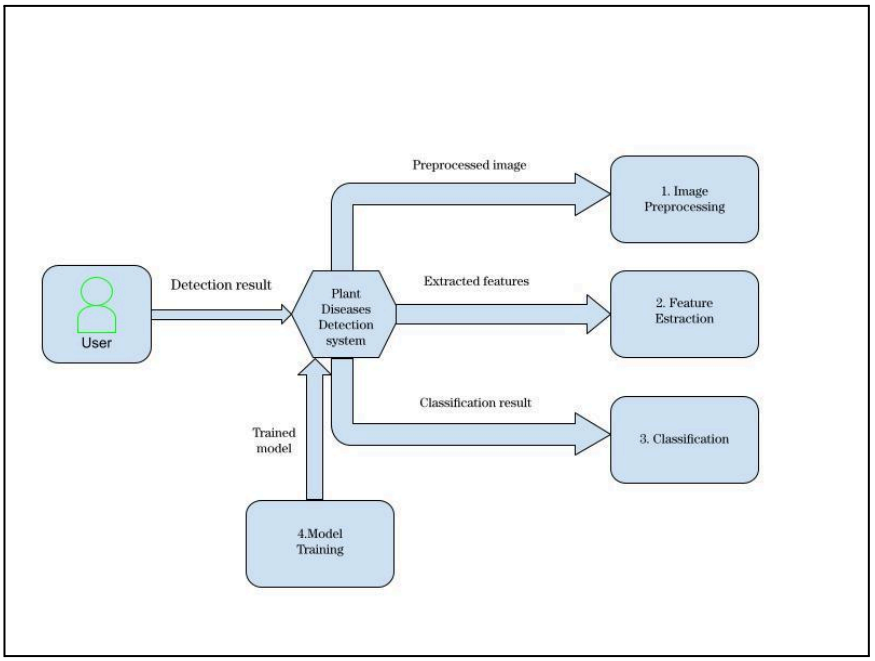


Figure 4.2: Data Flow Diagram

External Entity on Data Flow Diagram Figure (4.2)

User:

I. Inputs: Uploads an image of a watermelon plant leaf.
II. Receives: Disease detection result (diagnosis).

Processes:

1. Image Upload & Preprocessing

I. Receives image from the user.
II. Performs preprocessing tasks like resizing, normalization, and noise removal.
III. Data Flow:
   A. Input: Raw leaf image from User.
   B. Output: Preprocessed image to next process.

2. Disease Detection Engine

I. Core module that uses machine learning or deep learning models (e.g., CNN) to analyze the image.
II. Detects whether the leaf is healthy or diseased.
III. Data Flow:
    A. Input: Preprocessed image.
    B. Output: Detection result to the user and training logs to the database.

3. Database Management

I. Stores preprocessed images and results for future training/improvement.
II. Also manages the dataset of labeled leaf images for training.
III. Data Flow:
    A. Receives data from Process 2.0.
    B. Can be queried by a detection engine for retraining or validation.

4. External API Communication

I. Interfaces with third-party APIs for:
    A. Accessing pre-trained models.
    B. Fetching real-time agricultural updates or disease information.
II. Data Flow:
    III. Sends image/detection data.
    IV. Receives API responses.

Data Stores:

I. Training Image Database
    A. Contains labeled images used for model training and evaluation.
    B. Also logs past detections for retraining.

4.5 UI Design

The web application's user interface (UI) design is demonstrated by the following:

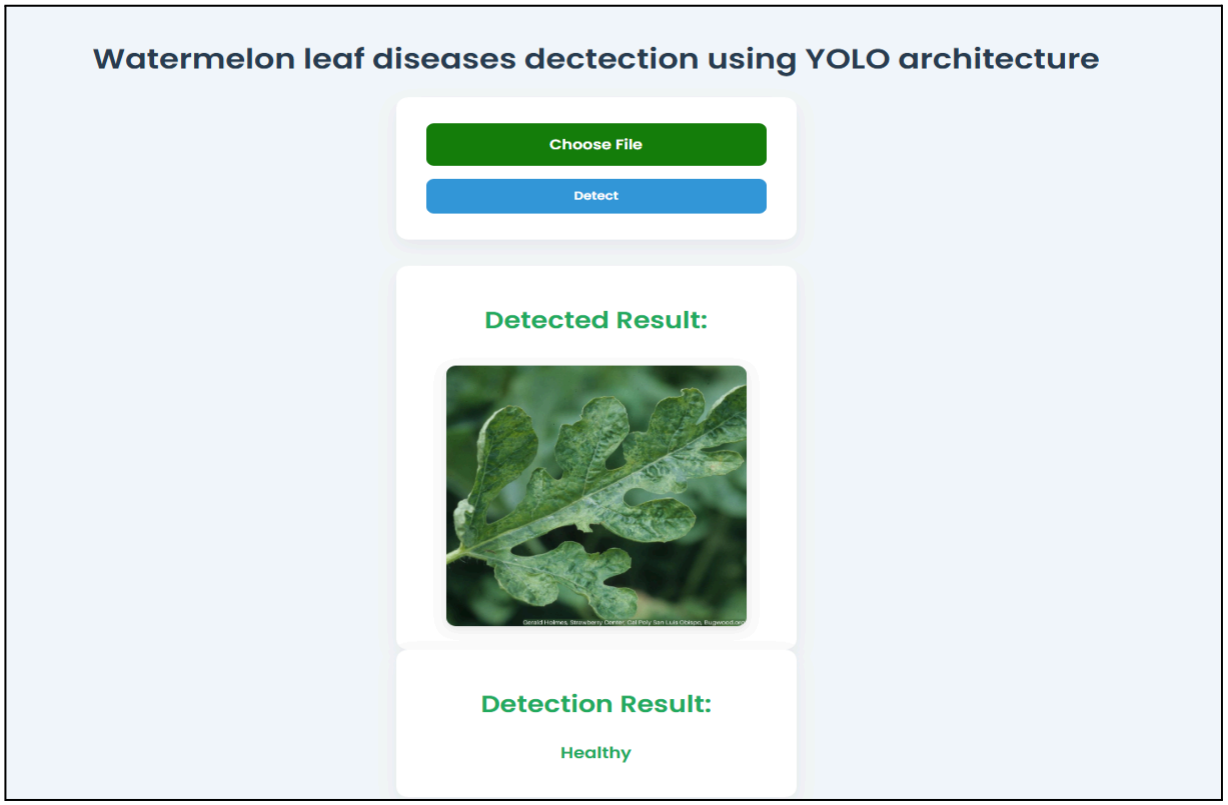


Figure 4.3: UI Design-1

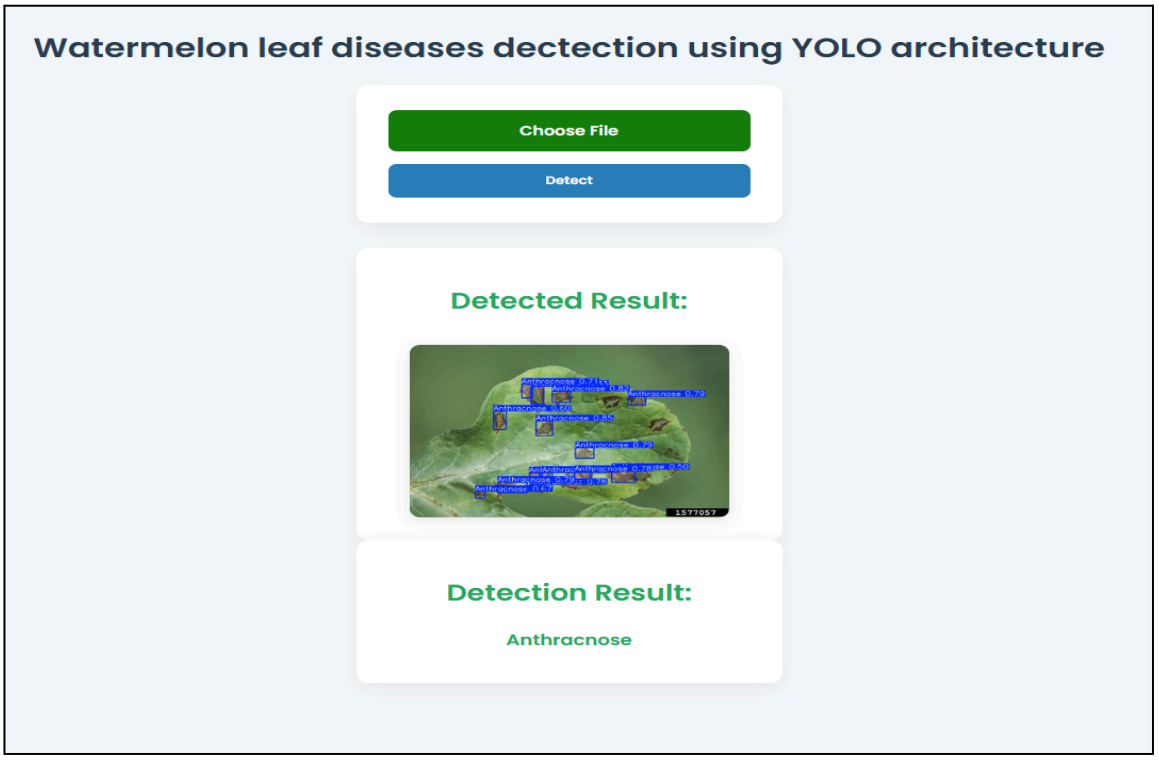


Figure 4.4: UI Design-2

The detection results are demonstrated in Figure 4.3 and Figure 4.4, highlighting the accuracy and responsiveness of the YOLOv11-based system. Figure 4.3 displays an image classified as *healthy*, with no signs of disease detected, while Figure 4.4 shows an image accurately

identified as affected by *Anthracnose*, a common fungal disease in watermelon leaves. The model not only successfully localized the diseased region but also labeled it with high confidence. The system achieved real-time performance with a very low inference time and high mAP@0.5 score, indicating both speed and precision in detecting and classifying watermelon leaf conditions.

The UI is designed for simplicity and usability:

I. Title Section: Clearly displays the project title: "Watermelon leaf diseases detection using YOLO architecture".
II. Upload and Detection Panel:
    A. Choose File: Green button to upload leaf images.
    B. Detect: Blue button triggers YOLOv11 model inference.
III. Output Section:
    A. Displays the detected image with bounding boxes and labels.
    B. Shows the predicted disease name (e.g., *Anthracnose*).

The design ensures that users can interact with the system intuitively and receive real-time visual feedback.

4.6 Deployment

I. The deployment phase involved building a user-friendly web interface using the Flask framework, which serves as the frontend for interacting with the YOLOv11-based detection model. The model itself was developed and integrated using the PyTorch backend, allowing for efficient loading, inference, and visualization.
II. When a user uploads an image of a watermelon leaf through the interface, the system performs real-time inference. The YOLOv11 model processes the image, identifies any disease regions, and draws bounding boxes with appropriate labels (e.g., "Healthy", "Anthracnose", etc.). These annotated results are then rendered back to the user directly on the image, providing immediate and clear feedback. This lightweight deployment ensures fast response times and enables usage even on low-resource devices, making the system accessible to farmers, agricultural officers, and researchers alike.

4.7 Project Plan

The project is organized over a 23-week timeline to ensure thorough development and implementation. The first three weeks focus on background research and literature review to understand existing solutions for plant disease detection and the capabilities of object detection models, especially the YOLO family. Weeks 4 to 7 are allocated for dataset collection, image cleaning, and manual annotation of watermelon leaf diseases. Model development, starting from architecture choosing, training and performance tuning using YOLOv11, is done during weeks 8 to 13. A variety of experiments run in between hyperparameter tuning is done to further improve detection and enhance accuracy. Weeks 14 to 17 are utilized developing and designing web-based UI followed by implementing trained YOLOv11 within the app. Weeks 18 to 20 are taken up in system testing, performance testing, and debugging for correctness and stability. Weeks 21 to 23, the final phase, are taken up with project documentation compilation, thesis report writing, and preparation for final presentation and deployment.

4.8 Task Allocation

I managed all facets of the project as an individual developer, I managed all facets of the project, from implementation to research. The activities were divided into clearly defined tasks to keep things in order and maintain steady progress throughout the 23-week period.

I. Research and Literature Review: Conducted a thorough review of plant disease detection methods and object detection models, with particular focus on the YOLO family of models.
II. Dataset Collection and Annotation: Retrieved images of watermelon leaves from public datasets and web sources. Annotated the images using tools likeMakesense AI to accurately label diseased regions.
III. Model Design and Training: Selected YOLOv11 as the fundamental architecture. Administered training environment deployment, model configuration, hyperparameter tuning, and evaluation metric with measurement using metrics like mAP, precision, and recall.
IV. UI Design and Creation: Designed and developed an interactive web-based UI for users to enter images and receive real-time feedback regarding detection of diseases. Deployed the interface using interfaces like Flask.
V. Integration and Testing: Integrated the trained YOLOv11 model into the web application with smooth interaction between frontend and backend. Tested for accuracy, usability, and performance.
VI. Documentation and Reporting: Completed the entire thesis report, maintained the project documentation, and prepared the material for final submission and presentation.

By independently managing all components of the project, I gained hands-on experience across the entire machine learning and web development pipeline, ensuring a cohesive and well-rounded final product.

Table 4.1: Tasks Allocation.

Tasks	**Weeks**																	
	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Data collection and preprocessing																		
Annotation																		
Model evaluation																		
Model train and test																		
Web development																		

Tasks	Weeks																	
	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Data collection and preprocessing																		
Annotation																		

Estimated Work Period	
Actual Work Period	

4.9 Summary

This research utilizes YOLOv11, a deep learning-based approach, to detect watermelon leaf diseases quickly and precisely. Users can input leaf images through an easy-to-use web interface, and they will obtain detection results with real-time bounding boxes and marked disease categories in an instant. With respect to earlier versions of YOLO, YOLOv11 was selected because it offers better accuracy and balance of real-time performance. The system is an asset for smart agriculture and disease monitoring of crops because it has a modularity and scalability architecture that allows for future extension to other plant diseases or crop types.

Chapter 5

Engineering Protocols and Design Difficulties

The engineering standards pertinent to the creation of the watermelon leaf disease detection system are described in this chapter. Along with giving a summary of the project's financial and project management components, it also draws attention to the ethical, environmental, and sustainability issues.

5.1 Implementation of Standard Protocols

Below is a curated list of engineering standards directly related to the project. For each standard, alternative options are discussed along with their pros and cons, and the rationale for final selection.

Software Standards

I. Selected: IEEE 830-1998 (Software Requirements Specification) – Offers clarity, consistency, and completeness in system requirements definition.
II. Alternative Considered: ISO/IEC/IEEE 29148:2018 – More detailed but more cumbersome for a single-developer academic project.
III. Justification: IEEE 830 provides a lightweight yet systematic approach, which is ideal for academic-sized ML projects.

Hardware Standards

I. Chosen: IEEE 802.3 (Ethernet), IEEE 802.11 (Wi-Fi) – For web application deployment via the internet.
II. Alternative Considered: IoT-specific standards (e.g., IEEE 1451) – Relevant to embedded applications, but not required for a web-based system.
III. Rationale: IEEE 802.x standards are ubiquitous in networking and accommodate deployment contexts like cloud infrastructure or on-premise servers.

Communication Standards

I. Selected: HTTP/HTTPS, RESTful API Design Principles – Enables smooth communication between frontend (UI) and backend (model).
II. Alternative Taken into Account: WebSocket Protocols – Too much for static image uploads but ideal for live streaming or high-frequency updates.
III. Rationale: HTTP-based REST APIs are light, stable, and well-suited for asynchronous image detection processes.

5.2 Impact on Society, Environment and Sustainability

Impact on Life

This project can positively benefit farmers and agricultural workers by enabling them to diagnose disease early, and therefore less loss of crops and improved food security.

Impact on Society & Environment

The system reduces the application of chemical treatments and manual inspection to identify disease through automation, promoting healthier ecosystems and reduced human exposure to pesticides.

Ethical Aspects

I. Data Privacy: Publicly available or personally collected datasets only were used. No personal data of users is collected.
II. Bias and Fairness: Precautions were taken to give balanced datasets for various kinds of diseases to avoid biased predictions.
III. Open Access: The system can be adopted in resource-deprived areas without any limitations, promoting equity.

Sustainability Plan

I. Environmental Sustainability: Early detection minimizes the use of pesticides, making agriculture sustainable.
II. Technical Sustainability: Modular construction can be applied to other crops with little retraining.
III. Long-Term Maintenance: Leverages open-source libraries and standard forms (PyTorch) to ensure long-term accessibility.

5.3 Project Execution and Cost Evaluation

Development and deployment resources are estimated here. Two models of budgets are presented..

Estimated Full Budget:

Table 5.1: Estimated Budget.

SN	Components	Estimated Cost(BDT)
1.	Data collection cost	7000
2.	Visiting data collection area	4000
3.	Software	3500
4.	GPU	35,000
5.	Cloud server	8000
	Total Estimated cost	57,500

Alternate Budget (Cost-saving model):

Table 5.2: Alternate Budget

SN	Components	Estimated Cost(BDT)
1.	Free datasets	0
2.	Free Google Colab tier	0
3.	Open-source tools	0
4.	Localhost testing	0
5.	Free google Cloud	0
	Total Estimated cost	0

Table 5.1 shows the Estimated Full Budget required to construct and deploy the watermelon leaf diseases detection system with specialized hardware and commercial software. It considers significant cost factors such as data collection, field visits, software expenses, GPU purchase for local training, and cloud server utilization. The total estimated budget is 57,500 BDT, and it is a practical estimation for a fully equipped setup.

In contrast, Table 5.2 presents an Alternate Budget, a cost-effective implementation plan that uses free and open-source tools. This includes employing publicly available datasets, the free version of Google Colab, and localhost test beds, which reduces the overall cost to zero BDT. This choice is viable for academic and low-budget projects, demonstrating how the project can be realized without spending money while still achieving functional results.

5.4 Advanced Engineering Challenge

Advanced Problem-Solving Techniques

This chapter identifies and maps the watermelon leaf disease detection system to complex engineering problem characteristics, outlines the associated knowledge profile, and evaluates the type of engineering activities involved in the project.

Table 5.3: Mapping with advanced problem solving.

EP1 Dept of Knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity with Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stakeholder Involvement	EP7 Interdependence
✓	✓	✓	✓	✓	✓	✓

Clarification of EP Attribute Mapping

EP1-Dept of Knowledge: The system requires understanding of deep learning (YOLOv11), image processing, and data annotation—drawing on multiple advanced concepts in computer vision, and software development.

EP2- Range of Conflicting Requirements: The model must balance accuracy, speed, and simplicity for real-time deployment—leading to trade-offs between detection performance and computational cost.

EP3- Depth of Analysis: Requires tuning hyperparameters, evaluating performance using metrics (mAP, precision, recall), and comparative analysis with other models.

EP4-Familiarity of Issues: Although object detection is a known domain, the specific challenge of detecting plant diseases in watermelon leaves remains niche with limited labeled datasets.

EP5-Extent of applicable codes and standards: The system complies with software engineering standards (IEEE 830) and web API standards (HTTP/REST), ensuring system reliability and interoperability.

EP6- Extent of Stakeholder Involvement: Target stakeholders include farmers, researchers, and agricultural support teams—each with different expectations (e.g., speed vs interpretability).

EP7- Interdependence: Data pipeline, model training, evaluation, and UI integration are highly interdependent. A change in one (e.g., model output format) affects others (e.g., frontend rendering).

Mapping to Knowledge Profile for Engineering Practice

This table 5.2 is designed to map the Engineering Practice to the Knowledge Profile.

Table 5.4: Mapping with knowledge Profile.

K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K8 Research Literature
✓	✓	✓	✓	✓

Reasoning Behind KP and EP1 Mapping

K3 (Engineering Fundamentals): Applied mathematics, programming logic, and basic computer vision principles are foundational to this project.

K4 (Specialist Knowledge): Requires advanced knowledge of deep learning models (YOLOv11), training workflows, and dataset handling.

K5 (Engineering Design): The system includes the design of the machine learning pipeline and UI, optimized for user experience and real-time performance.

K6 (Engineering Practice): Practical application of tools like PyTorch, LabelImg, and deployment frameworks to implement and test the system..

K8 (Research Literature): Literature review guided the choice of YOLOv11 over other models, helping justify its selection and relevance to agricultural use cases.

Engineering Activities

In this section, provide a mapping with engineering activities. For each mapping add subsections to put rationale (Use Table 5.4).

Table 5.5: Mapping Engineering Activities.

EA1 Rangeof re-sources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
✓	✓	✓	✓	✓

EA1 – Range of Resources
Used a diverse range of tools and resources: Python, PyTorch, open datasets, cloud compute, and visualization libraries.
EA2 – Level of Interaction
Involves integration between backend model inference and frontend web UI through REST APIs.
EA3 – Innovation
Adapted YOLOv11 to a novel problem—detecting diseases in watermelon leaves, where labeled data is scarce.
EA4 – Consequences for Society and Environment
Promotes precision agriculture, reduces pesticide use, and improves crop health—leading to better environmental and societal outcomes.
EA5 – Familiarity
While the general problem of object detection is familiar, its specific application in agricultural disease detection is less standardized.

5.5 Summary

This chapter proved that the suggested system is a complex engineering challenge that requires a great deal of analytical depth and multidisciplinary knowledge. The project's engineering rigor is validated by the mappings with problem-solving traits, knowledge profiles, and engineering activities. This project advances contemporary agricultural technology in a technological and morally responsible manner by means of meticulous design, creative deep learning application, and thoughtful assessment of sustainability and societal impact.

Chapter 6

Conclusion

This chapter summarizes the overall progress and outcomes of the project so far. It highlights the core achievements, identifies limitations encountered during the development, and outlines directions for future work.

6.1 Summary

The primary objective of this project was to develop an efficient and accurate system for detecting watermelon leaf diseases using advanced object detection and image processing techniques. Among the evaluated models, YOLOv11 demonstrated the best performance in terms of accuracy and inference speed, outperforming YOLOv8 through YOLOv10. A user-friendly web application was also developed, enabling users to upload images and receive real-time detection results. This system can serve as a valuable tool in agriculture, particularly for early disease detection and crop monitoring.

6.2 Limitation

I. The dataset used for training may not represent all variations of leaf diseases in different environments.
II. The web application currently supports only image uploads and result display; it lacks historical tracking or user profile integration.
III. Lighting, background noise, or partial leaf visibility may affect detection accuracy.

6.3 Future Work

Expand the dataset with diverse image sources to improve generalization.Add features like disease treatment suggestions and report generation. Integrate with mobile platforms for real-time field use. Enhance the system with multilingual support and offline detection capabilities.

References

[1] Wei, J., Ni, L., Luo, L., Chen, M., You, M., Sun, Y., & Hu, T. (2024). GFS-YOLO11: A Maturity Detection Model for Multi-Variety Tomato. Agronomy, 14(11), 2644.

[2] Zia, S., Khan, M. R., Shabbir, M. A., & Aadil, R. M. (2021). An update on functional, nutraceutical and industrial applications of watermelon by-products: A comprehensive review. Trends in Food Science & Technology, 114, 275-291.

[3] Banerjee, D., Kukreja, V., Gupta, A., Singh, V., & Brar, T. P. S. (2023, August). CNN and SVM-based Model for Effective Watermelon Disease Classification. In 2023 3rd Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-6). IEEE

[4] Wehner, T. C. (2008). Watermelon. In Vegetables I: asteraceae, brassicaceae, chenopodicaceae, and cucurbitaceae (pp. 381-418). New York, NY: Springer New York.

[5] HOSEN, M. M. (2016). PREVALENCE OF WATERMELON (Citrullus vulgaris) DISEASES IN PATUAKHALI DISTRICT (Doctoral dissertation, Dept. of Plant Pathology).

[6] Nakib, M. I., & Mridha, M. F. (2024). Comprehensive watermelon disease recognition dataset. Data in Brief, 53, 110182.

[7] https://worldpopulationreview.com/state-rankings/watermelon-production-by-state

[8] Lawal, O. M., Zhao, H., Zhu, S., Chuanli, L., & Cheng, K. (2024). Lightweight fruit detection algorithms for low‐power computing devices. IET Image Processing.

[9] Wei, J., Ni, L., Luo, L., Chen, M., You, M., Sun, Y., & Hu, T. (2024). GFS-YOLO11: A Maturity Detection Model for Multi-Variety Tomato. Agronomy, 14(11), 2644.

[10] Abdulridha, J., Ampatzidis, Y., Qureshi, J., & Roberts, P. (2022). Identification and classification of downy mildew severity stages in watermelon utilizing aerial and ground remote sensing and machine learning. Frontiers in Plant Science, 13, 791018.

[11] Adda, S. A., Nwaogwugwgu, I. B., & Wama, A. (2023). Detection of cucumber and watermelon diseases based on image processing techniques using K-means algorithm.

[12] Jiang, L., Jiang, H., Jing, X., Dang, H., Li, R., Chen, J., ... & Fu, L. (2024). UAV-based field watermelon detection and counting using YOLOv8s with image panorama stitching and overlap partitioning. Artificial Intelligence in Agriculture, 13, 117-127.

[13] Chen, G., Hou, Y., Cui, T., Li, H., Shangguan, F., & Cao, L. (2024). YOLOv8-CML: A lightweight target detection method for Color-changing melon ripening in intelligent agriculture. Scientific Reports, 14(1), 14400.

[14] Yao, C., Yang, Z., Li, P., Liang, Y., Fan, Y., Luo, J., ... & Mu, J. (2024). Two-Stage Detection Algorithm for Plum Leaf Disease and Severity Assessment Based on Deep Learning. Agronomy, 14(7), 1589.

[15] Soeb, M. J. A., Jubayer, M. F., Tarin, T. A., Al Mamun, M. R., Ruhad, F. M., Parven, A., ... & Meftaul, I. M. (2023). Tea leaf disease detection and identification based on YOLOv7 (YOLO-T). Scientific reports, 13(1), 6078

[16] Liu, Z., & Li, X. (2024). An improved YOLOv5-based apple leaf disease detection method. Scientific Reports, 14(1), 17508.

[17] Li, T., Zhang, L., & Lin, J. (2024). Precision agriculture with YOLO-Leaf: advanced methods for detecting apple leaf diseases. Frontiers in Plant Science, 15, 1452502.

[18] Banerjee, D., Kukreja, V., Gupta, A., Singh, V., & Brar, T. P. S. (2023, August). CNN and SVM-based Model for Effective Watermelon Disease Classification. In 2023 3rd Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-6). IEEE.

[19] Gomez, D., Selvaraj, M. G., Casas, J., Mathiyazhagan, K., Rodriguez, M., Assefa, T., ... & Espitia, E. (2024). Advancing common bean (Phaseolus vulgaris L.) disease detection with YOLO driven deep learning to enhance agricultural AI. Scientific Reports, 14(1), 15596.

[20] Suryavanshi, A., Mehta, S., Gupta, A., Aeri, M., & Jain, V. (2024, July). Agriculture Farming Evolution: Federated Learning CNNs in Combatting Watermelon Leaf Diseases. In 2024 Asia Pacific Conference on Innovation in Technology (APCIT) (pp. 1-6). IEEE.

[21] Yi, H., Song, K., & Song, X. (2024). Watermelon Detection and Localization Technology based on GTR-Net and Binocular Vision. IEEE Sensors Journa**l.**

211-15-4066

ORIGINALITY REPORT

17%	13%	9%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	4%
2	www.mdpi.com Internet Source	1%
3	www.frontiersin.org Internet Source	1%
4	dspace.daffodilvarsity.edu.bd:8080 Internet Source	1%
5	www.researchsquare.com Internet Source	1%
6	bpasjournals.com Internet Source	1%
7	Submitted to University of Finance – Marketing Student Paper	<1%
8	Submitted to Napier University Student Paper	<1%
9	Submitted to United International University Student Paper	<1%
10	researchspace.ukzn.ac.za Internet Source	<1%
11	Submitted to City University Student Paper	<1%
12	Piriyadharshini Singaravelu, Ezhilarasi Perumal. "Innovative solutions for plant disease identification: Leveraging DCoS-WOR and spiking neural networks", Expert Systems with Applications, 2025	<1%