Edge Implementation of Lightweight YOLO Models for Real-Time Mango Leaf Disease Detection on RISC-V Powered Device

BY

KISHON KUMAR PASHI
ID: 221-15-5391

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

Supervised By

Dr. Md. Taimur Ahad
Associate Professor and Associate Head
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised By

Dr. S. M. Aminul Haque
Professor and Associate Head
Department of Computer Science and Engineering
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

January 2025

APPROVAL

This Project titled **"Edge Implementation of Lightweight YOLO Models for Real-Time Mango Leaf Disease Detection on RISC-V Powered Device"**, submitted by **Kishon Kumar Pashi**, ID No: **221-15-5391** 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 **13 January, 2025**.

BOARD OF EXAMINERS

Dr. Md. Taimur Ahad — **Chairman**
Associate Professor & Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Saiful Islam — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

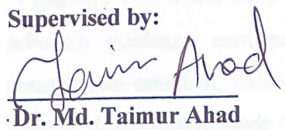
Mr. Amir Sohel — **Internal Examiner**
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Nazibur Rahman — **External Examiner**
Technical Lead - Database Administrator
Telenor - Grameen Phone Account

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Dr. Md. Taimur Ahad, Associate Professor and Associate Head, 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:



Dr. Md. Taimur Ahad
Associate Professor and Associate Head
Department of CSE
Daffodil International University

Co-Supervised by:

Dr. S. M. Aminul Haque
Professor and Associate Head
Department of CSE
Daffodil International University

Submitted by:

Kishon Kumar Pashi
ID: 221-15-5391
Department of CSE
Daffodil International University

©Daffodil International University

ACKNOWLEDGEMENT

First, we express our heartiest thanks and gratefulness to almighty God for His divine blessing making us possible to complete the final year project/internship successfully.

We are really grateful and wish our profound indebtedness to **Dr. Md. Taimur Ahad, Associate Professor and Associate Head,** Department of CSE, Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of "*Computer Vision and Deep Learning*" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Sheak Rashed Haider Noori**, **Professor & Head**, Department of CSE, for his kind help to finish our project and also to other faculty members and the staff of CSE Department of Daffodil International University.

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

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

ABSTRACT

An important part of Bangladesh's income is agriculture, and the mango is both a cultural and an economic symbol. Rajshahi and Chapai Nawabganj, which are known as the "mango hubs of Bangladesh," help make the country the seventh-largest mango producer in the world, and they ship a lot of mangoes. Mango leaf diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould have a big effect on Bangladesh's farming output, which is hard for farmers who use old, time-consuming ways to find diseases. This study shows a new edge-based system that uses light YOLO models (YOLOv5, YOLOv8, and YOLOv11) to find diseases on mango leaves in real time on RISC-V-powered devices. The suggested method aims to offer an effective, easy-to-reach, and low-cost answer for places with limited resources. The method involves teaching YOLO models on a balanced collection of 1,920 pictures that show four types of disease. The models are then quantized to INT8 so they can be used on devices with less power. Using performance measures such as mean Average Precision (mAP), precision, recall, and inference speed, YOLOv8 (small) is found to be the most reliable model, combining high accuracy with low computational needs. The study emphasizes the positive effects on people and the environment of the suggested system, such as lowering the need for pesticides, improving food security, and encouraging environmentally friendly farming methods. Ethical issues like justice, ease, and openness are taken into account to make sure that all farmers can adopt the technology equally. The results show that it is possible to combine advanced AI models with traditional farming, bringing together modern technology and traditional farming methods. In the future, researchers will focus on adding more types of datasets, making models more general, and using green energy sources to make the world more sustainable. This new way of doing things shows how edge computing could change the way diseases are managed in farmland, making farming groups stronger and more productive.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
Introduction

1.1 Introduction

Agriculture is an important part of Bangladesh's economy and a big part of both jobs and GDP. Mango is one of the country's most important foods, both culturally and economically. Mango, which is called the "king of fruits", is an important crop in places like Rajshahi and Chapai Nawabganj, which are often called the "mango hubs of Bangladesh." With a yearly production of about 2.35 million tonnes in the fiscal year 2021–2022, the country is the seventh-largest producer of mangoes in the world [1]. Mango farming also brings in money from exports. In 2022, the country sent 1,757 tonnes of mangoes to 28 countries, and in 2023, it sent 2,700 tonnes to 34 countries [2]. Even with these successes, growing mangoes is still very hard, mostly because of mango leaf diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould. These diseases not only lower the yield and quality of mangoes but also hurt farmers' ability to make a living and the country's ability to sell agricultural goods [3]. Traditional ways of finding mango leaf diseases, like inspecting each sample by hand, take a long time, are prone to mistakes, and don't work for the large-scale farming that is done in Bangladesh [4]. In order to deal with these problems, this study suggests a new, edge-based way to find mango leaf diseases in real time. The system makes sure of high accuracy while using little power by using light YOLO models (YOLOv5, YOLOv8, and YOLOv11) that are optimised for RISC-V edge devices. By using advanced methods like model compression to INT8 and edge-specific optimisations, the system is made to work in the resource-limited settings that are common in rural Bangladesh [5]. This method not only gives farmers a disease-controlling tool that can be used by many, but it is also affordable and fits with global sustainability goals because it lowers the damage that farming does to the environment [6]. The system was also put through a lot of tests on a balanced set of images of mango leaves, and it did very well across a number of disease groups. This study builds on the work of others that have used YOLO models to successfully implement lightweight and real-time disease detection [7]. Wang et al. (2024), for

example, made a small YOLOv8 model for finding pests and diseases in mango trees. This shows how useful these kinds of technologies could be for accurate farming [5]. The goal of this study is to completely change how mango leaf diseases are managed in Bangladesh by using these new technologies. It solves one of the country's biggest farming problems in a long-lasting and technologically advanced way, which will lead to better food security, more export opportunities, and better ways for mango farmers to make a living [8].

1.2 Motivation

Rapid improvements in farming technology have created new ways to deal with problems that farmers all over the world keep facing. One of these is still finding plant diseases, which is very important in places like Bangladesh where farming is the main industry. Diseases that lower output and quality often harm mango farming, which is an important part of the agricultural sector. Traditional ways of finding diseases, like inspecting plants by hand and having an expert help, take a lot of time, money, and are hard for small farms to get to. The study was started because we need to find a way to find mango diseases in real time that is cheap, effective, and scalable. Cutting edge deep learning models like YOLO and combining them with RISC-V architecture-powered edge devices creates a unique chance to make a small, energy-efficient, and easy-to-use tool. This tool can make advanced diagnostics available to more people, so farmers can make quick, well-informed decisions that will help them avoid food loses. Besides that, the study is motivated by the bigger goals of promoting ecology and technology access in farming. This work focuses on low-cost tools and methods that can be used by many people. It is in line with global efforts to help poor countries catch up with technology, which will make agriculture more productive and resilient. The goal of this study is to make a real difference in the field of agriculture by providing a strong, easy-to-use answer that gives farmers more power and encourages environmentally friendly farming methods.

1.3 Rationale of the study

The main goal of this study is to find solutions to the most important problems that the agriculture sector, especially mango farming, which is very important to Bangladesh's economy and culture, is currently facing. Many plant diseases can

easily spread to mango trees, which lowers their output, makes the fruit less tasty, and costs farmers a lot of money. These problems are made worse by the fact that there aren't any easy-to-use or inexpensive ways to find diseases. This is especially true for small-scale farmers in Bangladesh, who are the backbone of the farming workers. Traditional ways of finding diseases, which often involve watching plants by hand, take a long time, are biased, and don't work to stop widespread crop damage. With the rise of deep learning models like YOLO and edge computing technology, there is a huge chance to get around these problems. Using these new technologies, this study aims to create an edge-based, real-time disease monitoring system that can work well in places with limited resources. Edge computing means that the answer doesn't need to be connected to the internet, which makes it useful in rural places where that kind of infrastructure isn't always available. Integrating the system with low-cost RISC-V hardware also makes sure that the solution is still cheap for small farmers, which is in line with the goal of making current farming technologies available to everyone. The larger effects of healthy agriculture are another reason for this study. By making it easier to find diseases early and accurately, the proposed method can help farmers use fewer chemical medicines, which is better for the environment. It also fits with attempts around the world to improve food security and crop yields, which are very important for dealing with the problems caused by climate change and a growing world population. Overall, the goal of this study is to close the technology gap in farming by coming up with a new, low-cost, and easy-to-use way to find diseases in mango trees. This study not only helps farmers with their current needs, but it also helps reach the long-term goals of sustainable farming development, economic growth, and better food security.

1.4 Expected Output

This study aims to create a fully functional edge-based device that can find diseases in mango trees in real time. Using the YOLO architecture models for edge deployment, the device aims to quickly and correctly spot diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould. The system uses RISC-V design and a fast inference pipeline to give real-time data that will make it possible to quickly identify diseases. With a focus on being easy to get and affordable, the gadget is meant to be cost-effective and fit the needs of small-scale farmers, connecting the needs of modern

technology with those of basic farmers. Its small size and easy-to-use interface will make it simple to set up and use in rural areas, giving farmers useful information without the need for technical know-how. The gadget should help lower crop loss and boost farming output by making it easier to find diseases quickly and accurately. The study also supports environmental sustainability by allowing focused pesticide use, which lowers the amount of chemicals used and their effect on the environment. The design of the gadget will also be adaptable and scalable, so it can handle more plant types and diseases in the future. In the end, this study wants to come up with a useful, effective, and significant answer to the problems that come up when growing mangoes that also supports long-term farming methods and makes farmers' lives better.

1.5 Report Layout

This thesis is divided into several chapters, and each one covers an important part of the study on automatically identifying and classifying medical plants. The framework is meant to give a clear, complete picture of the study, from its theoretical basis to its useful effects and contributions in real life.

Chapter 1: Introduction – This chapter introduces the research topic, providing an overview of the background, problem statement, objectives, scope, and limitations of the study. It sets the stage for the detailed discussions that follow by outlining the importance of an AI-driven tool for identifying medicinal plants and supporting conservation and healthcare through technology.

Chapter 2: Background Study – The literature review examines existing research and related works in medicinal plant classification, with a particular focus on image processing and machine learning approaches. It compares current methods, identifies unresolved issues, and highlights the contributions and limitations in the field. This chapter establishes the theoretical foundation and justifies the need for the proposed research.

Chapter 3: Methodology – This chapter details the research methodology, including system design, hardware and software requirements, and project management aspects. It describes the data collection and preprocessing methods, as well as the application

of CNNs and transformer models in image analysis. The chapter also provides a project timeline and financial analysis, laying out a clear plan for the study's execution.

Chapter 4: Results and Analysis – This chapter presents the experimental results and offers a comprehensive analysis of the model's performance. It discusses the outcomes of the experiments, compares them with existing methods, and evaluates the model's accuracy, precision, recall, and F1 score. The findings are critically examined to assess the effectiveness and reliability of the AI-based classification tool.

Chapter 5: Impact on Society, Environment and Sustainability – This chapter explores the broader implications of the research, including its impact on conservation efforts, healthcare practices, and environmental sustainability. Ethical considerations and potential challenges related to deployment are addressed, underscoring the tool's significance in supporting traditional and modern medicinal practices.

Chapter 6: Overview of the Study, Conclusion and Future Work – The final chapter summarizes the key findings of the research, drawing conclusions based on the study's objectives and results. It also provides recommendations for future research, identifying limitations and areas for improvement. This chapter concludes the thesis with insights into the potential for further advancement in medicinal plant identification.

CHAPTER 2
Background Study

2.1 Terminologies

This study is based on a number of important terms that are needed to understand the ideas and tools being used. An edge device is a small, low-cost computer that is placed close to the data source and is meant to do processing and reasoning work locally, so that cloud-based systems aren't needed as much. YOLO (You Only Look Once), a real-time object recognition method that splits images into regions and predicts bounding boxes and probabilities, is used in the study to make sure that multiple objects are found quickly and correctly. Mango diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould are talked about. All of these are very bad for crop output and quality. Damage from fungi or pests, like dark spots, branch dieback, tissue swelling, or fungus growth stopping photosynthesis, is what these diseases are known for. Bounding boxes are used to contain things in images, which helps with accurate object recognition. Performance factors like precision and memory are very important. Precision measures how accurate positive identifications are, and recall measures how well you can find all the important things. Mean average precision (mAP) also checks how well the recognition works across different confidence levels and groups. Quantization lowers the size of model parameters for faster computing, and hardware based on RISC-V architecture, which is a cost- and energy- effective set of instructions, are key to putting the edge-based solution into action. The system's reasoning speed how long it takes to process inputs and make outputs ensures real-time recognition, which is important for making decisions quickly. Also, the study stresses sustainability by encouraging tailored pesticide application, which means using them in the best way possible to have the least amount of effect on the environment. All of these terms sum up the study's main idea: using cutting-edge technology to solve problems in agriculture while keeping things efficient, accurate, and good for the environment.

2.2 Related Works

The development of deep learning models, especially those in the YOLO (You Only Look Once) family, has led to big steps forward in real-time disease and pest identification in agriculture. Researchers have improved the accuracy of these models for edge devices, made them work better with different fruits like mango, rice, corn, tea, and lychee, and changed them to deal with specific problems in agriculture, especially in places with few resources. This literature review brings together the results of studies that used and improved YOLO-based models to find plant diseases. It focuses on new ideas for making the models more efficient, accurate, and flexible at the edge. Pratap and Kumar in [3], used YOLOv8 and other deep learning models to sort diseases in mango leaves. This made early disease control better by allowing high- resolution images. In the same way, Yumang et al. [9] used YOLOv3 to find Anthracnose on mango leaves, with a success rate of 83.33% while looking into useful methods for preparing datasets. Wang and Duan [4] presented SimAM-YOLOv7, which includes attention methods to deal with false positives in mango leaf disease classification. This led to a big rise in mAP. Wang and Wang [5] suggested GAS-YOLOv8, a light model that includes changes to GhostNet. It is 98.6% accurate while cutting model parameters by 33%, which makes it perfect for mobile deployment. In [10], Xue et al. created YOLO-Tea, a model for finding tea diseases that was built on YOLOv5 and improved with ACmix and CBAM modules. It worked well at finding small objects and dealing with shade issues. In the same way, Kumar et al. [11] used DenseNet, Bi-FAPN, and YOLOv5 to find diseases in rice and got an F1 score of 92.45% by focusing on multi-scale feature extraction. In [12], Chitraningrum et al. compared YOLOv5 and YOLOv8 for finding diseases on corn leaves. They found that YOLOv8 was better in both mAP and detection rate. Li et al. [13] used GhostNet and Triplet Attention to make YOLOv8 better at finding diseases in corn, making it better for use in edge devices. A lot of research has been done on edge computing. Rahaman et al. [14] showed off an Android app that used DenseNet169 to find diseases in mangoes. The app was 97.81% accurate, showing how mobile technology can help with early disease control. Khan et al. [15] built MobileNetV3-small for edge apps and got 99.5% accuracy after optimizing it. Xiao et al. [16] created YOLOv7-MGPC to find lychee diseases. It achieved 217 FPS on

Jetson Nano, highlighting its usefulness in keeping an eye on big orchards. In [17], Wang et al. came up with even better ideas by combining YOLOv8 with RISC-V design. They showed that this could help make solutions that use less energy and are more flexible when it comes to computing that can work in places with limited resources. In [18], Sangaiah et al. used UAVs and Tiny YOLOv4 to track rice diseases and achieved 86% mAP for crop analysis over a large area. This shows the usefulness of high-resolution aerial images. In [19], Tsai and Hsu looked into how IoT-enabled deep neural networks could be used to find orchid diseases. They combined local and global traits to allow real-time tracking. In [20], Zhang et al. unveiled GVC-YOLO, which is based on YOLOv8n and can find cotton pests at 48 frames per second, making it good for controlling big populations of pests.

These studies collectively highlight significant advancements in YOLO-based and CNN models for crop disease detection. By adapting and optimizing these models for edge devices and mobile applications, researchers are supporting scalable and accessible solutions that contribute to sustainable agricultural practices and real-time disease management. These innovations mark a step forward in integrating AI and edge computing into precision agriculture, enabling enhanced productivity and resilience in farming practices.

Table 2.1: Research Matrix

Author(s) and Year	Model(s) Used	Accuracy (%)	Dataset Information
Pratap & Kumar (2024)	CNN, VGG- 16, MobileNet, Googlenet, YOLOv8, EfficientNet	Not Specified	Extensive mango leaf dataset, high and low resolution
Yumang et al. (2023)	YOLOv3	83.33%	Mango leaf dataset, images split for training and testing
Wang & Duan (2024)	SimAM-YOLOv7	92.1%	Self-made mango leaf disease dataset
Wang & Wang (2024)	GAS- YOLOv8 with GhostHGNetv2	98.6% (pests), 91.7% (diseases)	Public mango pest and disease dataset

Table 2.1: *Continued from previous page*

Author(s) and Year	Model(s) Used	Accuracy (%)	Dataset Information
Kumar et al. (2024)	Various ML models	Not specified	20 classes of mango fruit diseases
Mohandas et al. (2021)	YOLOv4-tiny	Not Specified	Dataset of multiple plant leaves, including mango
Xue et al. (2023)	YOLO-Tea (YOLOv5 with ACmix, CBAM)	0.3%-15.0% improvement over YOLOv5	Tea leaf dataset in natural environment
Kumar et al. (2023)	Multi-scale YOLOv5 with Bi-FAPN	94.87%	RLD rice leaf dataset
Chitraningrum et al. (2024)	YOLOv5, YOLOv8	YOLOv8: mAP50 of 0.965	Corn leaf infection dataset from Kaggle
Li et al. (2024)	GhostNet, Triplet, YOLOv8s	mAP@0.5: 91.4%	Maize leaf disease dataset
Xu & Wang (2023)	ALAD-YOLO (YOLOv5s with Mobilenet-V3)	90.2%	Apple leaf disease dataset with 2,748 images
Sangaiah et al. (2024)	Tiny YOLOv4 with UAV	mAP: 86%	Rice leaf disease dataset
Rahaman et al. (2023)	DenseNet169	97.81%	20 classes of infected and healthy mango images
Khan et al. (2023)	MobileNetV3-small	99.5%	PlantVillage dataset for 14 crop species
Xiao et al. (2023)	YOLOv7-MGPC with Ghost-NetV1, CBAM	88.6%	Lychee disease dataset

Table 2.1: *Continued from previous page*

Author(s) and Year	Model(s) Used	Accuracy (%)	Dataset Information
Wang et al. (2024)	AIV-SoC with YOLOv8	Not Specified	General model on RISC-V architecture
Tsai & Hsu (2024)	DNN with IoT integration	Not Specified	Orchid disease dataset from smart agriculture sensors
Zhang et al. (2024)	GVC-YOLO (YOLOv8n with GSConv, VoVGSCSP)	mAP@0.5: 97.9%	Cotton aphid-damage dataset
Katumba et al. (2024)	BeanWatchNet, YOLOv8	BeanWatchNet: 90%	Makerere University beans image dataset
Khalid (2024)	Embedded AI, Digital Image Processing	Not Specified	Dataset focused on wheat yellow-rust
Xan et al. (2024)	FD-MobileNet, ResNet	98.44%	Rice leaf dataset with 5,932 images
Sangaiah et al. (2024)	UAV T-YOLO- Rice	mAP: 86%	Rice leaf disease dataset with augmented samples
Patel et al. (2024)	TFLite model	100%	Mango leaf dataset from Kaggle
Soeb et al. (2023)	YOLOv7	mAP: 98.2%	Tea leaf disease dataset
Sun et al. (2024)	YOLOv5-Res, YOLOv5-Res4	mAP0.5 improvement by 2.8%	Apple leaf dataset with small target spots

2.3 Gap Analysis

Recent research highlights deep learning models, especially YOLO-based architectures, for detecting plant diseases in diverse crops. Despite advancements in accuracy and model efficiency, significant deficiencies remain regarding model

adaptation, edge deployment, real-time applicability, and generalization across varying environmental conditions.

- **Model Adaptability and Lightweight Deployment:** A number of studies take ad- vantage of lightweight models that may be deployed on mobile and edge devices, such as Tiny-YOLO, MobileNet, and versions of YOLO enhanced with GhostNet. Nonetheless, constraints in real-time processing persist for models utilized in the field, where connec- tivity and power limitations affect their feasibility. Despite the proposal of edge-focused models (e.g., by Wang et al., 2024; Xiao et al., 2023) [6], thorough assessments across various environmental circumstances, including illumination fluctuations and occlusion, are scarce.
- **Dataset Diversity and Generalization:** Many research depend on particular datasets that represent regional or crop-specific characteristics (e.g., illnesses of mango, tea, and lychee), constraining the models' applicability to other crops or unfamiliar regions. Despite the occasional utilization of upgraded and larger datasets (Pratap & Kumar, 2024; Xu & Wang, 2023) [3], there is an ongoing necessity for more rigorous cross-dataset training to improve model efficacy in diverse agricultural contexts.
- **Attention Mechanisms and Small Object Detection:** Problems like false positives and detection speed under real-time restrictions still necessitate attention, even if there have been advancements in using attention processes to enhance small-object recognition (e.g., Wang & Duan, 2024; Xue et al., 2023) [4]. Models with improved feature extraction and accelerated processing speeds could augment the accuracy and responsiveness required.
- **Scalability to Multi-Objective Applications:** Research on multi-objective models that can handle various crop varieties' worth of diseases has been sparse, with most studies concentrating on either single crops or single diseases. A scalable, multi-objective model could improve field utility, particularly in polyculture farming systems where various crops are grown concurrently.

2.4 Challenges

Developing and deploying an edge-based disease detection system for mango cultivation involves several technical, environmental, and practical challenges. These challenges arise from the intricacies of adapting deep learning models to real-world agricultural settings, hardware limitations of edge devices, and the diverse environmental conditions in which these systems must operate. There are several challenges we may face, these are:

- **Environmental Variability:** Making changes to the model to deal with changes in lighting, weather, and background noise so that the results are always accurate. Limitations on Hardware: Making sure that computations are fast and efficient on edge systems with limited resources, such as RISC-V.
- **Collecting Data:** Getting a large, high-quality dataset that shows how farming really works in the real world.
- **Model Optimization:** Finding the best balance between accuracy, memory, and inference speed for models that are light without losing precision.
- **Farmer Usability:** Making a simple, low-cost method that small-scale farmers can use even if they don't know much about technology.
- **Power Efficiency:** Making sure the system uses little energy and lasts a long time, especially if it will be used in country or off-grid areas.
- **System Maintenance:** Dealing with problems that come up with maintaining and regularly updating the edge devices that have been put in place.

CHAPTER 3

Research Methodology

3.1 Introduction

Agriculture is facing more and more problems, especially when it comes to controlling diseases. To make farming more productive and long-lasting, we need to use new technologies. The main goal of this study is to look into how to find diseases on mango leaves, which has a big effect on crop quality and output. Using cutting-edge, light YOLO (You Only Look Once) models, this study aims to create an effective and easy-to-use real-time disease detection system that can be used on RISC-V-powered edge devices. The suggested method uses a dataset with 1920 labeled pictures of four types of mango leaf diseases: anthracnose, die back, gall midge, and sooty mold. To make sure it works with the YOLO design, preprocessing steps are carried out, such as resizing to 640 × 640 pixels, normalization, and DPI changes. The Roboflow tool is used to annotate the dataset, which is then split into training, validation, and test sets for evaluating the model. YOLOv8s, YOLOv8n, YOLOv5nu, YOLOv5su, YOLOv11n, and YOLOv11s are some of the lightweight YOLO models that are trained and tested to find the best one based on accuracy, precision, memory, and inference speed. The chosen model is then exported in ONNX format, reduced to INT8 to save space, and converted into a CIV-compatible format that can be used on the RISC-V device. This research focuses on making a product that can be used by many people and lasts for a long time so that diseases on mango leaves can be found quickly, even in places with limited resources. The edge-based distribution makes sure that small-scale farmers can buy and easily access the technology. This is in line with global efforts to use technology in precision agriculture and protect the environment.

3.2 Dataset Description and Preparation

3.2.1 Dataset

The dataset used in this work comes from the MLD24 dataset, which is a large set of images created to find diseases on mango leaves. The information was first made public on August 5, 2024. It includes pictures taken in different mango fields in Kushtia and Dhaka, Bangladesh. The dataset has eight groups, such as mango leaves that are sick and leaves that are fit. This study, on the other hand, only uses a small part of the information to look at four types of disease: anthracnose, die back, gall midge, and sooty mold. There are 1920 images in the balanced collection, with the same amount of 480 images in each class.

Anthracnose Gall Midge Die Back Sooty Mold

Figure 3.1: Sample Image of Each Class

The images were initially captured at a resolution of 3024 × 4032 pixels. Still, they were compressed and resized to 640 × 640 pixels during preprocessing to reduce memory size and standardize the input dimensions for model training. Additional preprocessing steps included normalization to improve image contrast and consistency and a change in DPI to enhance compatibility with the training pipeline. Annotation was done on the data using the Roboflow tool to get it ready for machine learning tasks.

Table 3.1: Dataset Specification

Properties	Values
Image Resolution (Original)	3032 × 4032 pixels
Image Resolution (Com-pressed)	640 × 640 pixels
Format	. jpg
Total Images (Used)	1920 (Originally total images: 6400)
Classes (Used)	4 (Originally eight classes)

This made it easier to name parts of the images that showed disease. This step was very important for the YOLO models to learn how to recognize the different mango leaf diseases. After that, the marked-up data was used for training, validation, and testing, and the split was handled flexibly during the YOLO model training process. This carefully chosen and already processed part of the MLD24 dataset makes a big difference in the field of real-time mango leaf disease detection study and can be used in many other ways in plant pathology and data science.

3.2.2 Annotation Process

The pictures of mango leaves were labelled with text using the Roboflow tool, which is an open-source web-based platform for labelling photos quickly. In this step, disease-specific features in the pictures had to be labelled by hand to make bounding box labels, which are very important for teaching object recognition models like YOLO. The first step in the labelling process was to send the pre-processed dataset to the Roboflow platform. This dataset had 1920 resized images (640 × 640 pixels). Each picture was carefully looked at to find the infected areas that fit the four target groups: Anthracnose, Die Back, Gall Midge, and Sooty Mould. For each type of disease, lines were made around the affected areas to make sure that the disease traits were located accurately. Then, the right class name was given to each bounding box based on the disease's obvious signs. To keep things consistent and cut down on writing mistakes, clear rules were followed. These guidelines included rules for where to put the bounding boxes so that they would cover the whole sick area while

avoiding overlap with unimportant areas as much as possible. Cases that weren't clear were checked by comparing them to accounts of the diseases that were confirmed by experts. The marked-up data was sent out in a manner that YOLO can read. This includes text files where each line is a bounding box with the class ID, normalized coordinates, and measurements. This process of annotating data made sure that a high-quality dataset with correctly labelled data was created. This allowed the YOLO models to learn and find the specific mango leaf illnesses during training and testing.

3.3 Proposed Methodology

The image illustrates the workflow for building and evaluating YOLO-based deep learning models for detecting mango leaf related features. The process begins with a MLD24 Dataset, which undergoes an Annotation step to label the data with bounding boxes or classifications. This annotated dataset is then used to train Deep Learning Models, focusing on various YOLO model variations, including YOLOv5n, YOLOv5s, YOLOv8n, YOLOv8s, YOLOv11n and YOLOv11s.

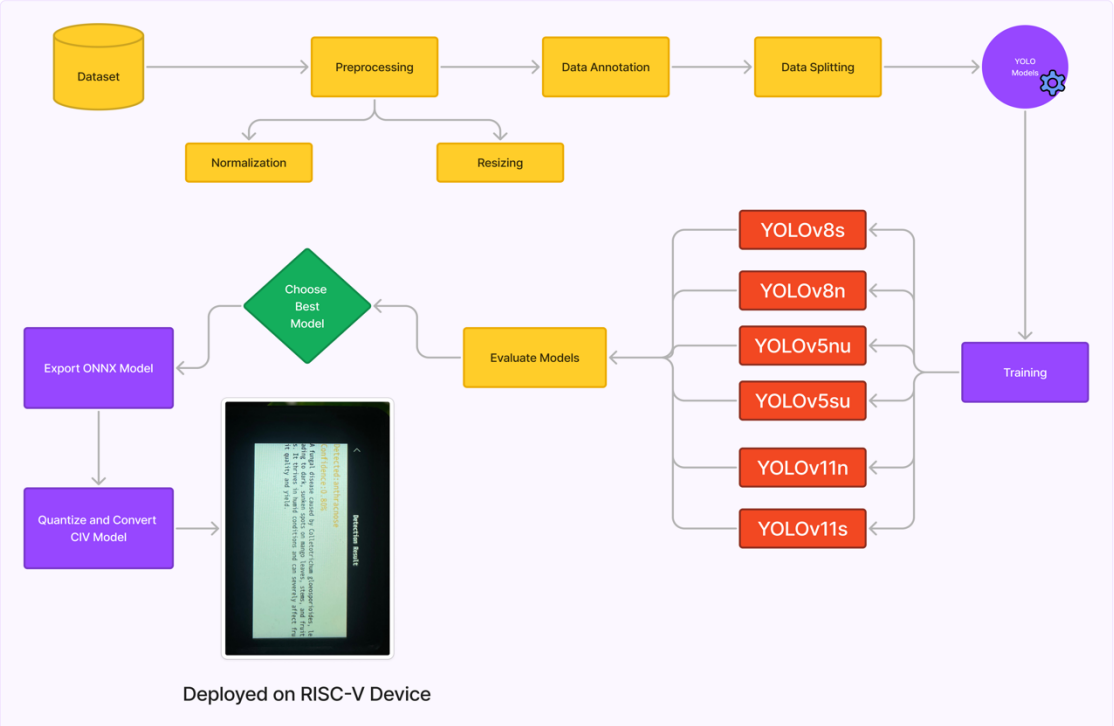


Figure 3.2: The Methodological procedure

These models are compared based on performance metrics (e.g., accuracy, speed, resource efficiency), and the comparisons feed into a decision-making process to identify the YOLO Best Model for the specific task. Once the best model is selected, it is converted into an ONNX format and further optimized for deployment through quantization. Finally, the optimized model is deployed on a RISC-V device, making it ready for real-time disease detection. This process is designed to be efficient, scalable, and suitable for low-resource environments.

3.3.1 You Only Look Once (YOLO)

The YOLO (You Only Look Once) object detection model operates in a single pass through the network, making it a fast and efficient choice for real-time applications. Below is a detailed, step-by-step explanation of how YOLO processes an image for object detection, along with the necessary equations:

1. **Input Image Preprocessing**

   Getting the raw picture ready is the first step in the YOLO model. To make sure that all of the inputs are the same, the raw picture is resized to a set size, which is usually 640x640 pixels. This resizing makes the computations simpler and lets the model handle images with different sizes well. After that, each pixel value is adjusted to a value between 0 and 1 by dividing it by 255. We can figure out this adjustment by:

$$I_{norm} = \frac{I_{pixel}}{255} \quad \text{..... (3.1)}$$

   The pixel intensity of the picture is given by I pixel. Normalization makes sure that the network learns faster and doesn't have problems when the input values are from different groups. Once the picture has been normalized, it can be sent to the YOLO network to be found.

2. **Dividing the Image into Grid Cells**

   After the picture has been pre-processed, YOLO cuts it up into a S×S square. In the case of YOLOv3 or YOLOv4, S could be 13, which means that the

picture is split into 13x13 grid cells. It is up to each grid cell to find items whose centres fall within that cell.

$$I_{norm} = \frac{I_{pixel}}{255} \quad \text{..... (3.2)}$$

3. Bounding Box Prediction by Each Grid Cell

Each grid cell predicts multiple bounding boxes. For each bounding box, the following parameters are predicted:

(a) **x and y:** These represent the coordinates of the center of the bounding box relative to the grid cell. The width and height of the image normalizes these coordinates.

(b) **w and h:** These are the width and height of the bounding box, normalized by the image's dimensions. These values allow the model to generalize to different object sizes.

(c) $\boldsymbol{p_{obj}}$**:** This is the confidence score that the bounding box contains an object. This score indicates the probability that the box contains an object and is computed as:

$$p_{obj} = sigmoid(C_{obj}) \quad \text{..... (3.3)}$$

Where C_{obj} is the raw prediction for the object's confidence. The sigmoid function squashes the prediction into a value between 0 and 1. The confidence score p_{obj} is multiplied by the class probability to determine how likely a specific object is within that box.

4. Class Probability Prediction

Every grid cell also guesses the class probabilities of the object(s) it sees. For example, in this study, there are four classes of diseases that affect mango leaves, and the class odds p_{class} are forecast for each of those classes. The sigmoid function is used to figure out these odds:

$$p_{class} = sigmoid(C_{class}) \quad \text{..... (3.4)}$$

Where C_{class} are the grid cell's raw class forecast scores for each class. This number shows how likely it is that an item found is from a certain class, like Anthracnose, Die Back, Gall Midge, or Sooty Mould.

5. Class Probability Prediction

For each bounding box, YOLO combines the confidence score p_{obj} with the predicted class probabilities to generate the final output. The overall score for a bounding box is:

$$score_{final} = p_{obj}p_{class} \quad \text{..... (3.5)}$$

The score reflects the likelihood that the bounding box contains an object belonging to a specific class. YOLO then filters out bounding boxes with low confidence scores, typically applying a threshold (e.g., 0.5) to keep only those boxes that are highly likely to contain an object.

6. Non-Maximum Suppression (NMS)

After predicting multiple bounding boxes for each grid cell, many boxes may overlap with each other, especially when detecting the same object. To resolve this, YOLO filters out redundant boxes using Non-Maximum Suppression (NMS). NMS retains the bounding box with the highest confidence score and removes all other boxes with a higher overlap (measured by Intersection over Union, IoU) than a defined threshold (e.g., IoU 0.5). This step ensures that only the most accurate bounding boxes remain. The IoU between two bounding boxes is calculated as:

$$IoU = \frac{Area\ of\ Intersection}{Area\ Of\ Union} \quad \text{..... (3.6)}$$

Bounding boxes with IoU values more significant than the threshold is discarded, leaving only the most relevant bounding boxes.

7. Model Training

During training, the YOLO model learns to minimize the following loss function, which consists of three primary components:

(a) **Localization loss:** Measures the difference between the predicted bounding box coordinates and the ground truth.
(b) **Confidence loss:** Measures how accurately the model predicts whether a bounding box contains an object.
(c) **Classification loss:** Measures the error in predicting the correct class label for each object.

The final loss function is a weighted sum of these three components:

$$Loss_{total} = Loss_{loc} + Loss_{conf} + Loss_{class} \quad \text{..... (3.7)}$$

8. Model Evaluation and Selection

Once trained, the YOLO model is evaluated based on metrics such as precision, recall, and mAP (mean Average Precision). The model's performance is assessed by predicting bounding boxes and classifying objects correctly. The best-performing model, determined by the evaluation, is then exported for deployment, typically in ONNX format, for further use in real-time applications such as mango leaf disease detection.

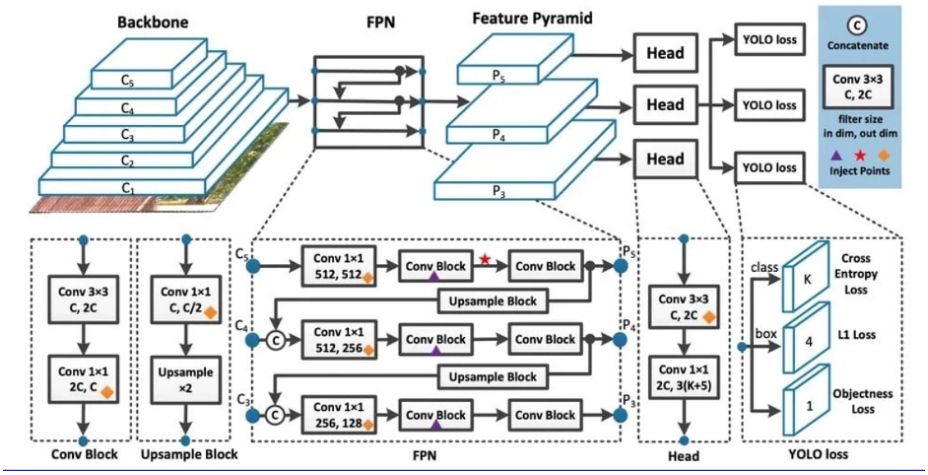


Figure 3.3: The Architecture of YOLOv8 Model

3.3.2 Comparison Between YOLO Models

The YOLO (You Only Look Once) object detection models have evolved, with newer versions improving architecture, accuracy, and performance. These versions are typically optimized for various use cases, such as real-time object detection and resource-constrained environments. Below, we compare several YOLO models, including YOLOv5nu, YOLOv5su, YOLOv8n, YOLOv8s, YOLOv11n, and YOLOv11s, based on their architecture, parameters, and performance metrics.

1. YOLOv5nu and YOLOv11n: These models are lightweight and ideal for environments with limited computational resources. With lower parameters and GFLOPs, they prioritize speed and efficiency over high accuracy.
2. YOLOv5su and YOLOv11s: These models provide a balanced approach, offering relatively higher accuracy while maintaining reasonable resource usage. They are suitable for applications where moderate computational power is available
3. YOLOv8n and YOLOv8s: These are the most powerful models regarding accuracy and computational demand. They are suited for high-performance environments where computational resources and power consumption are not constrained, providing excellent results for complex tasks

We also compared the YOLO models based on four primary stakeholders: model layers, parameters, GFLOPS and performance trade-off. Critical Differences Between YOLO Models are given below:

1. Model Layers: The number of layers in the model determines the depth of the neural network and its ability to learn complex features. YOLOv11n and YOLOv11s have the highest number of layers (319), which generally indicates more capacity to learn intricate patterns compared to YOLOv5nu and YOLOv8n, each with fewer layers (225–262).
2. Parameters: The model's total number of parameters (weights) indicates how much memory is required to store the model and influences its complexity. YOLOv8s and YOLOv5su have the most prominent parameters (11,166,560 and 9,153,152, respectively), making them suitable for scenarios where performance is prioritized over memory usage.
3. GFLOPs (Giga Floating Point Operations): GFLOPs measure the computational cost of the model in terms of the number of floating-point operations required. YOLOv8s (28.8 GFLOPs) and YOLOv5su (24.2 GFLOPs) are the most computationally intensive models, indicating higher processing requirements. In contrast, YOLOv11n and YOLOv5nu are less computationally demanding (6.6 and 7.8 GFLOPs, respectively), making them more suitable for devices with lower computational resources.
4. Performance Trade-offs: Models with higher parameters and GFLOPs, such as YOLOv8s, tend to offer better accuracy but require more resources, making them ideal for powerful hardware. On the other hand, models like YOLOv5nu and YOLOv11n, with fewer parameters and lower GFLOPs, are more efficient for deployment on resource-constrained devices while sacrificing some accuracy for speed and reduced memory usage

Table 3.2: Comparison between six different YOLO models

Model	Layers	Parameters	GFLOPs	Use Case
YOLOv5nu	262	2,654,816	7.8	Lightweight, fast, resource-efficient
YOLOv5su	262	9,153,152	24.2	Balanced performance, more accuracy

Table 3.2: *Continued from previous page*

Model	Layers	Parameters	GFLOPs	Use Case
YOLOv8n	225	3,157,200	8.9	Efficient, good for real-time tasks
YOLOv8s	225	11,166,560	28.8	High accuracy, suitable for powerful systems
YOLOv11n	319	2,624,080	6.6	Low power, efficient, faster inference
YOLOv11s	319	9,458,752	21.7	Balanced between performance and efficiency

3.3.3 Proposed Device

The proposed device, based on a RISC-V architecture, is designed for efficient real-time object detection and classification tasks using YOLOv8 models. The hardware comprises a 1GHz RISC-V C906 CPU as the primary processing core, with an optional 1GHz ARM A53 for Linux-based applications, and a 700MHz RISC-V C906 for running RTOS. A dedicated low-power 8051 core (25–300MHz) supports energy-efficient operations. The device integrates a 1 TOPS@INT8 NPU, enabling high-speed inferencing with popular models like YOLOv8 and Mobilenetv2, supported by 256MB DDR3 memory. A 5MP camera (with support for 4MP GC4653 and OS04A10) ensures high-resolution image capture, while a 2.3” HD IPS touchscreen offers an intuitive interface. The methodology involves training YOLOv8n models, exporting them to ONNX format, quantizing them to INT8 for efficiency, converting them to a civmodel format, and generating a .mud file for deployment. The backend processes resize input images, performs inference, and processes outputs, which are displayed on the touchscreen. The file structure, illustrated in the visual representation, organizes essential components for deployment, ensuring a streamlined integration process. This setup optimally balances computational efficiency and accuracy for edge AI applications.

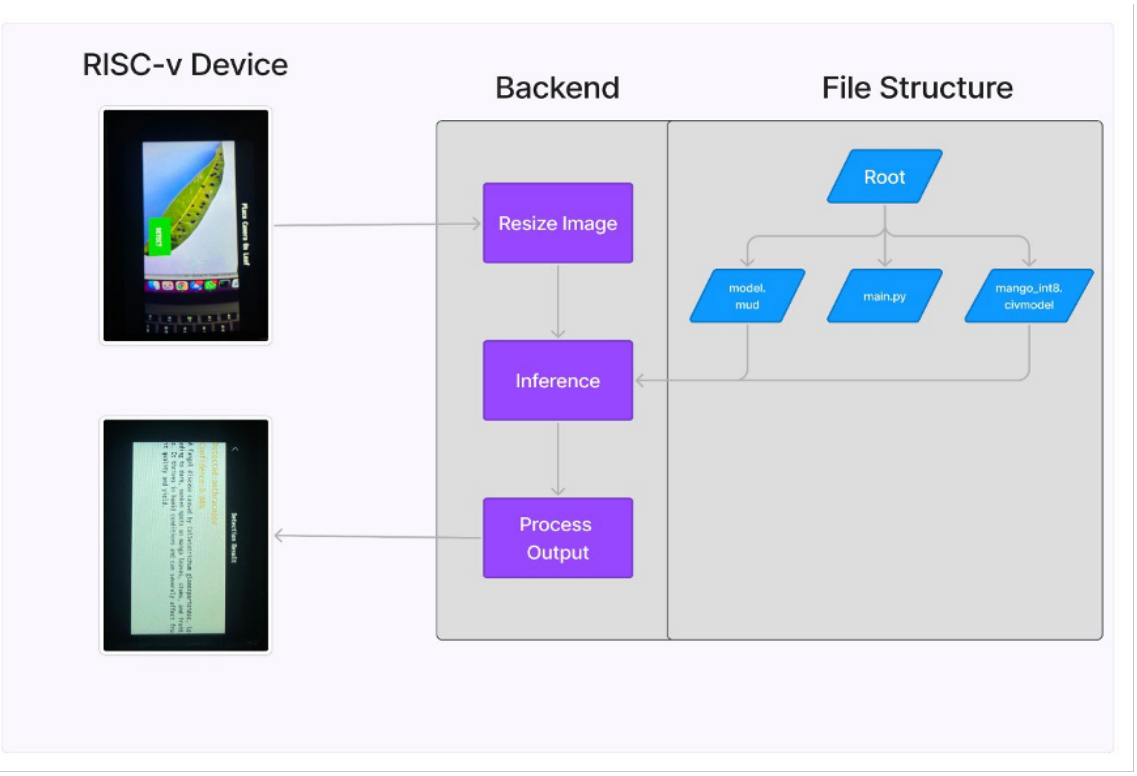


Figure 3.4: The flowchart of proposed RISC-v Device

3.4 Hardware and Software Requirements

3.4.1 Hardware Requirements:

- Google Colab GPU: NVIDIA Tesla K80, T4, or P100 for training.
- Local CPU (Optional): 8 GB RAM or more for smaller model testing.
- RISC-V Powered Device for deployment.
- Storage: Google Drive for dataset and model storage.

3.4.2 Software Requirements:

- Google Colab.
- Python 3.10 with relevant deep learning libraries.

3.5 Project Management and Financial Analysis

3.5.1 Project Management

The project was divided into several main stages, which were defining the problem, getting the dataset, preparing it, building the model, training it, deploying it, and evaluating it. A Gantt chart was used to keep track of the goals and targets for each step, making sure they were met on time and with the right amount of resources. To lower the risks, like bad data quality and technology limits, strategies were used like adding more data, choosing the right model, and reviewing progress on a regular basis. The project was well managed through these organized stages, which made it easier to get things done quickly and with few problems. The project met its dates and stayed on track with its original goals thanks to regular reviews. This organized method eventually led to the successful use of a RISC-V-powered device for finding diseases on mango leaves in real-time.

Figure 3.5: GANTT Chart of Project Timeline

Process	May'24	June'24	July'24	Aug'24	Sep'24	Oct'24	Nov'24	Dec'24
Working Plan								
Theoretical Study								
Literature Review								
Data Annotation								
Model Design								
Device Development								
Methodology Writing								
Report Writing								
Review and Finalization								

3.5.2 Risk Management

Risk management was crucial to the project, as certain aspects could have impacted the project timeline or the final model performance. Key risks were identified and mitigated as follows:

1. **Data Quality and Availability:** There was a risk that the dataset might not be sufficient or high-quality for training deep learning models. To mitigate this risk, data augmentation techniques (e.g., rotation, flipping) were applied to expand the dataset, and tools like Roboflow were used for accurate image labelling.

2. **Model Performance Issues**: Due to limitations in computational resources, the YOLO models may not perform as expected on real-time devices. This risk was mitigated by carefully selecting lightweight models like YOLOv5nu and YOLOv11n for deployment on RISC-V devices. Additionally, hyperparameter tuning was con- ducted to optimize the models for accuracy and efficiency.
3. **Hardware Constraints:** Running YOLO models on RISC-V devices posed the risk of low processing power, which could have hindered real-time detection performance. This was managed by selecting appropriate model sizes and ensuring the deployment hardware met the specifications.
4. **Time Management:** The timeline for training and deploying models was tight, and delays could have occurred due to computational limitations or dataset preparation issues. Regular progress checks and using cloud-based resources like Google Colab helped manage this risk, ensuring that the project stayed on schedule.
5. **Time Management:** There was a risk of software compatibility issues between the tools used for model training (PyTorch, YOLOv5 repository) and the deployment environment (RISC-V device). Extensive testing and cross-validation were performed to ensure compatibility and a smooth transition from training to deployment

3.5.3 Financial Analysis

The financial analysis of the project reveals that the overall costs were minimal due to the utilization of free resources and open-source software. Using Google Colab for model training provided free access to GPUs, significantly reducing costs, with an optional Colab Pro subscription costing $9.99 per month if extended access was needed. The hardware costs for the RISC-V-powered device used for deployment were estimated to range between $50 and 200$. Additionally, there were no software costs, as all tools, such as PyTorch, OpenCV, and Roboflow, are open source. Minor costs were associated with storage, including Google Drive, which was estimated to be around $20–$50. Overall, the project was financially efficient, with the main expenses related to hardware and storage, while software and personnel costs were minimal or non-existent.

Table 3.3: Financial Analysis Chart

Category	Estimated Cost	Details
Cloud Resource Costs	$0	There are no additional costs for using Google Colab's free GPU access, but an optional Colab Pro for extended access would cost $9.99/month.
Hardware Costs	$100	Cost of RISC-V powered device used for model deployment.
Software Costs	$0	There are no costs for open-source tools such as PyTorch, OpenCV, and Roboflow.
Personnel Costs	$0	There were no external personnel costs as the project was conducted independently.
Miscellaneous Costs	$10	Minor costs for storage on Google Drive and potential software licenses.

3.6 Summary

The methodology of this thesis involves developing and deploying lightweight YOLO models for real-time mango leaf disease detection on a RISC-V-powered device. The pro- cess begins with acquiring the MLD24 dataset, consisting of 1,920 labelled images across four disease classes. The dataset undergoes preprocessing, including resizing, normalization, and annotation using Roboflow. Various YOLO models, including YOLOv5, YOLOv8, and YOLOv11, are trained on the dataset using Google Colab with GPU acceleration. Model training is followed by evaluation on a separate validation set, and the best-performing models are selected for deployment. The trained models are optimized for real-time detection and then deployed on a RISC-V-powered device. The project management is structured in distinct phases, ensuring timely completion and risk mitigation through careful planning, data augmentation, and model selection. Financially, the project is cost-effective, leveraging free cloud resources and open-source tools, with minimal expenses for hardware and storage. Ultimately, the methodology provides a robust approach to detecting mango leaf diseases, balancing efficiency, cost, and performance for real-time applications.

CHAPTER 4

Result Analysis and Discussion

4.1 Overview

This chapter presents a detailed evaluation of six lightweight YOLO models—YOLOv5 (nano and small), YOLOv8 (nano and small), and YOLOv11 (nano and small)—to as- sess their performance on a mango leaf disease detection dataset containing four primary classes (Anthracnose, Die Back, Gall Midge, and Sooty Mould). The evaluation focuses on key performance metrics, including training and validation loss curves, precision-recall balance, F1 scores, and classification reports. Additionally, a comparative analysis of the models' mAP (mean Average Precision) at IoU thresholds of 0.5 (mAP50) and 0.5- 0.95 (mAP50-95), computational complexity (GFLOPs), and inference speed provides insights into their trade-offs between accuracy and efficiency. The results demonstrate that YOLOv8 (small) achieves the highest mAP scores and exhibits excellent precision and recall, making it the most accurate model overall. However, its higher computational requirements and slower inference speed suggest that it is best suited for scenarios where accuracy is prioritized over resource constraints. On the other hand, YOLOv8 (nano) delivers strong accuracy while maintaining a low computational footprint, offering a balanced solution for real-time applications. YOLOv11 (nano) stands out as the most efficient model in terms of GFLOPs and inference speed, making it ideal for resource-constrained environments, albeit with slightly lower precision and recall. YOLOv5 models, while slightly behind in overall accuracy compared to YOLOv8 and YOLOv11, provide reliable performance with moderate computational demands, making them versatile for a range of applications. This chapter systematically compares the models, providing valuable insights into their strengths and limitations. By highlighting the trade-offs between ac- curacy, efficiency, and computational requirements, it enables informed decision-making when selecting an optimal model for real-time mango leaf disease detection.

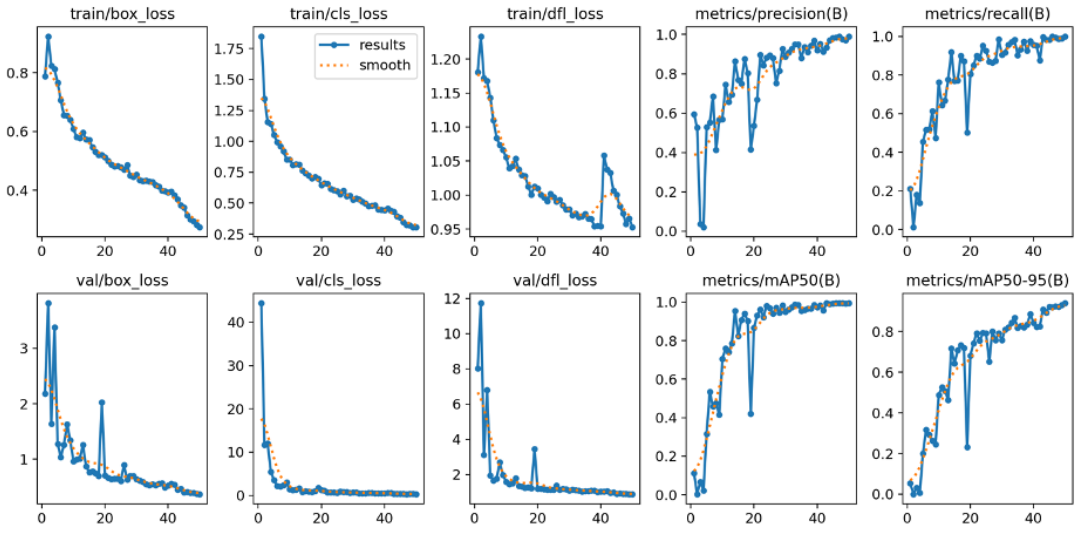
4.2 Experimental Result

4.2.1 Training and Validation Loss Analysis

When you look at the training and validation curves for each YOLO model, you can learn a lot about how they learn across different types of loss functions, such as box loss, classification loss, and Distribution Focal Loss (DFL). There is a steady and quick drop in both training and validation loss for YOLOv8 (small). It ends up with the lowest loss values, which means it learned well with little overfitting. The accuracy and recall graphs show that it is very consistent, and by the end of training, it was getting very close to perfect performance. The YOLOv8 (nano) follows a similar pattern, but its loss values are a little higher, which means it can't generalize as well.

The YOLOv5 models, especially YOLOv5 (small), see regular drops in loss values and end up doing better than YOLOv5 (nano). However, both YOLOv5 models have slightly higher validation losses than YOLOv8, which suggests that they are not as good at dealing with complex data.

YOLOv8 (large)



YOLOv8 (medium)

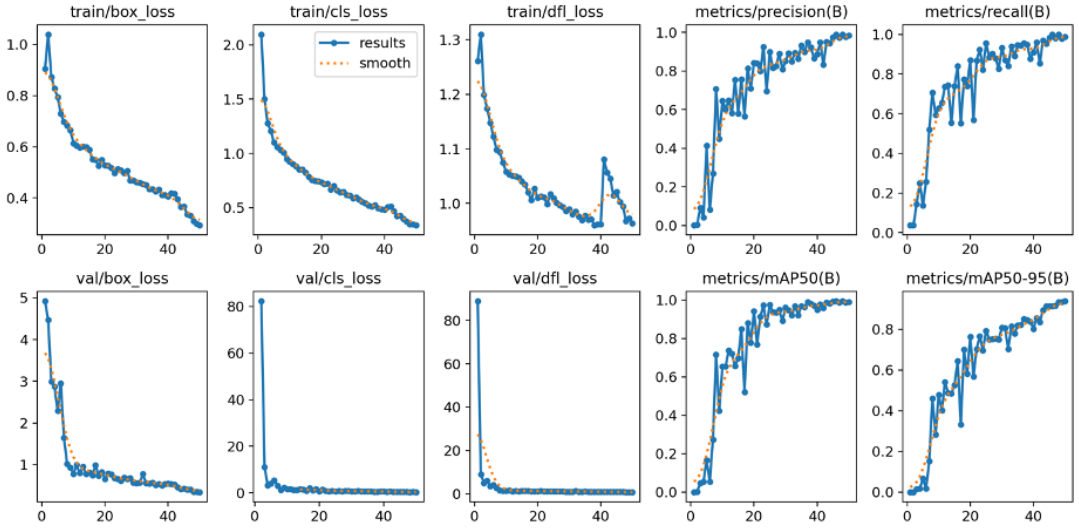
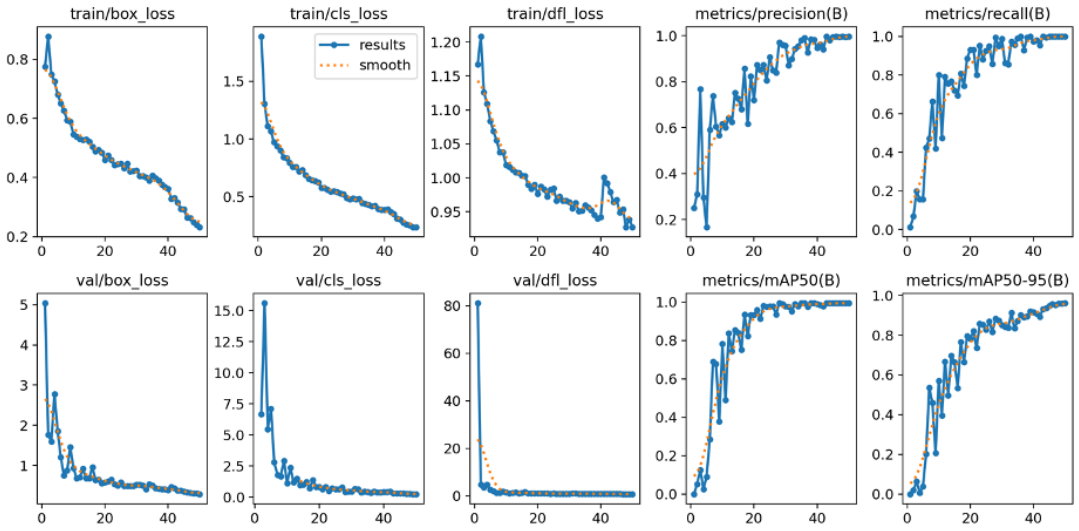


Figure 4.1: Training and Validation Losses of YOLOv5 Models

YOLOv8 models, such as YOLOv8 (nano) and YOLOv8 (small), are more stable and generalizable than other models. YOLOv8 (small) has the lowest and smoothest loss rates, making it better than the others.

YOLOv8 (large)



YOLOv8 (medium)

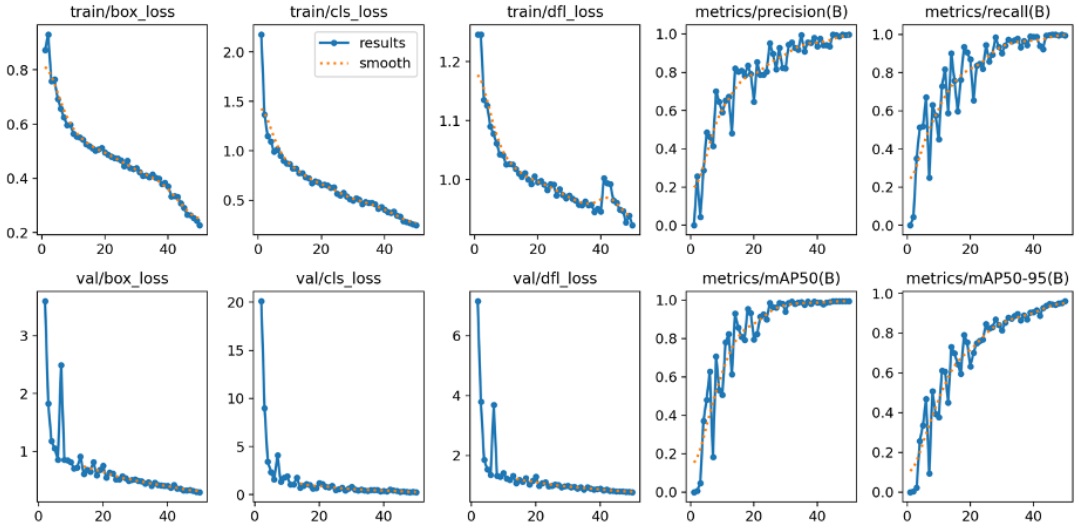
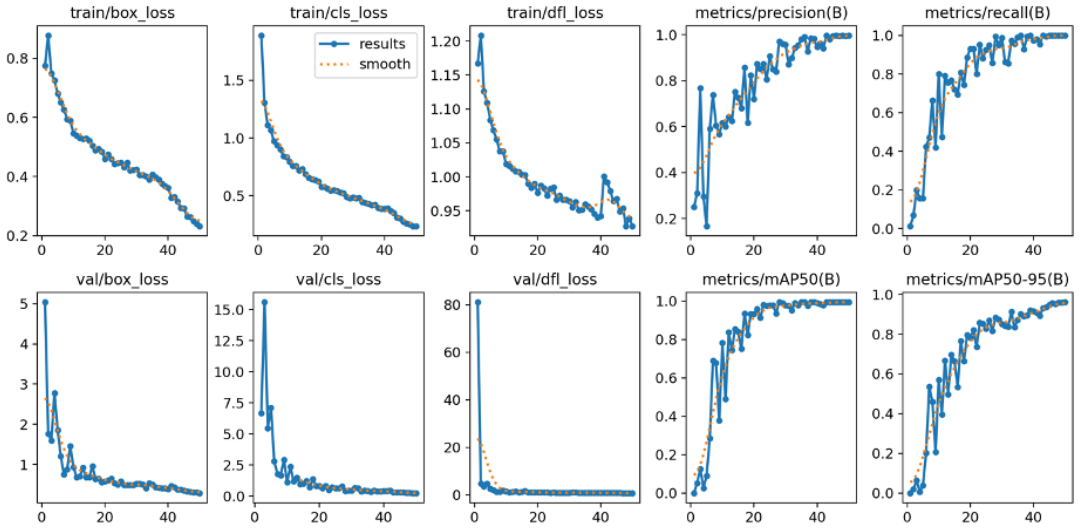


Figure 4.2: Training and Validation Losses of YOLOv8 Models

The YOLOv11 models are the most variable. YOLOv11 (nano) models are especially unstable when it comes to confirmation loss. YOLOv11 (small) gets smoother training loss curves, but it needs more fine-tuning to make it more general.

YOLOv8 (large)



YOLOv8 (medium)

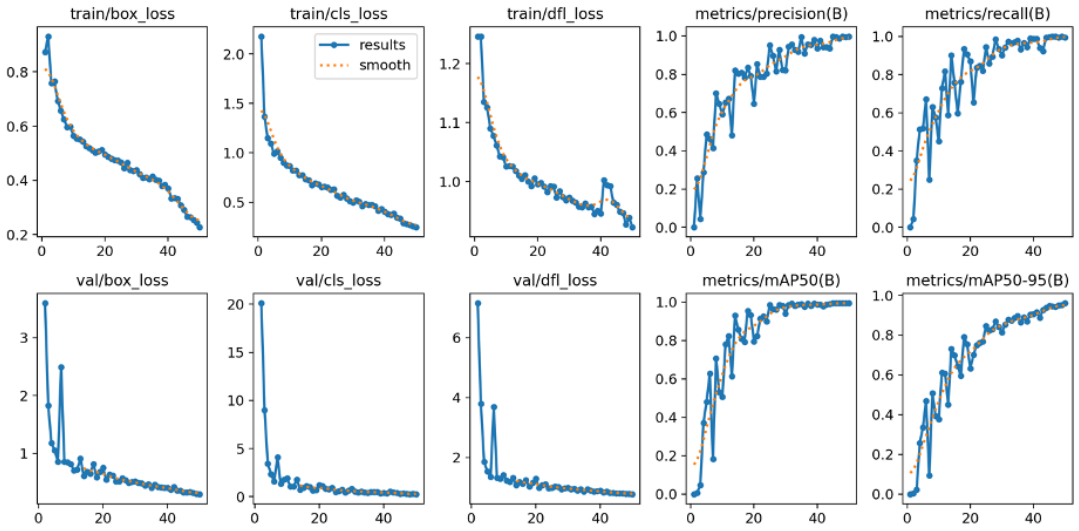


Figure 4.3: Training and Validation Losses of YOLOv11 Models

4.2.2 Training and Validation Loss Analysis

The data distribution and bounding box analysis reveal the spread of instances across classes (Anthracnose, Die Back, Gall Midge, and Sooty Mould) and the placement and dimensions of bounding boxes used in training. The class distribution is balanced across all YOLO models, ensuring unbiased learning and generalization across categories. Bounding box heatmaps in the "x" vs. "y" plot and "width" vs. "height" plot illustrate the spatial distribution and variability of annotations. YOLOv8 (nano and small) models exhibit tight clustering in bounding box placements, indicating their strong ability to localize objects accurately within constrained regions, which enhances precise localization. In contrast, YOLOv5 (nano and small) and YOLOv11 (nano and small) models show a wider spread, suggesting that they can handle diverse object placements and dimensions. However, this increased variability could lead to challenges in achieving high precision, particularly in dense or complex scenes.

YOLOv5 (nano) YOLOv5 (small)

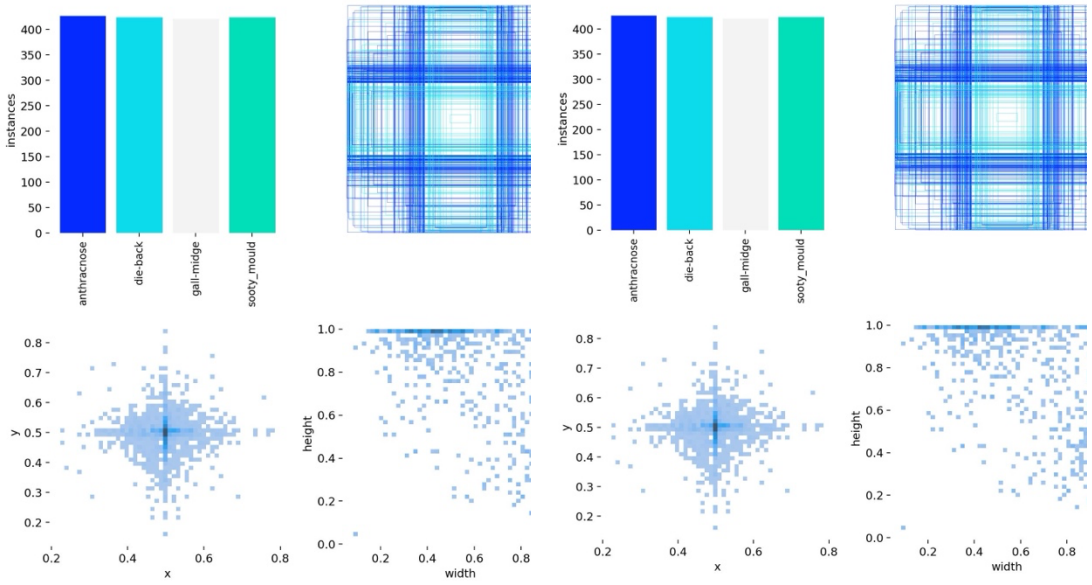


Figure 4.4: Bounding Box Heatmaps for YOLOv5 Models

YOLOv8 (nano) YOLOv8 (small)

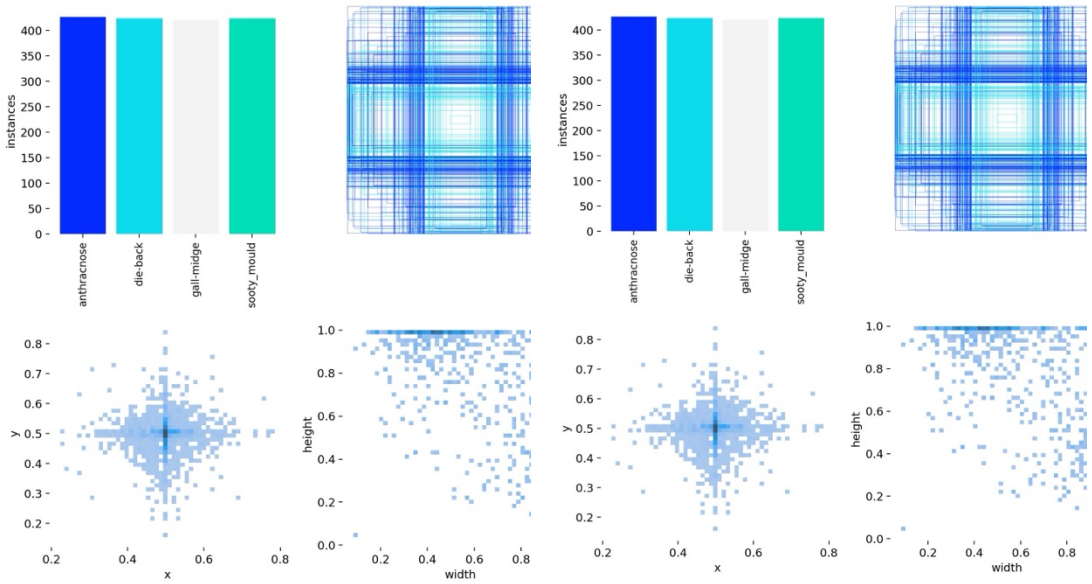


Figure 4.5 Bounding Box Heatmaps for YOLOv8 Models

YOLOv11 (nano) YOLOv11 (small)

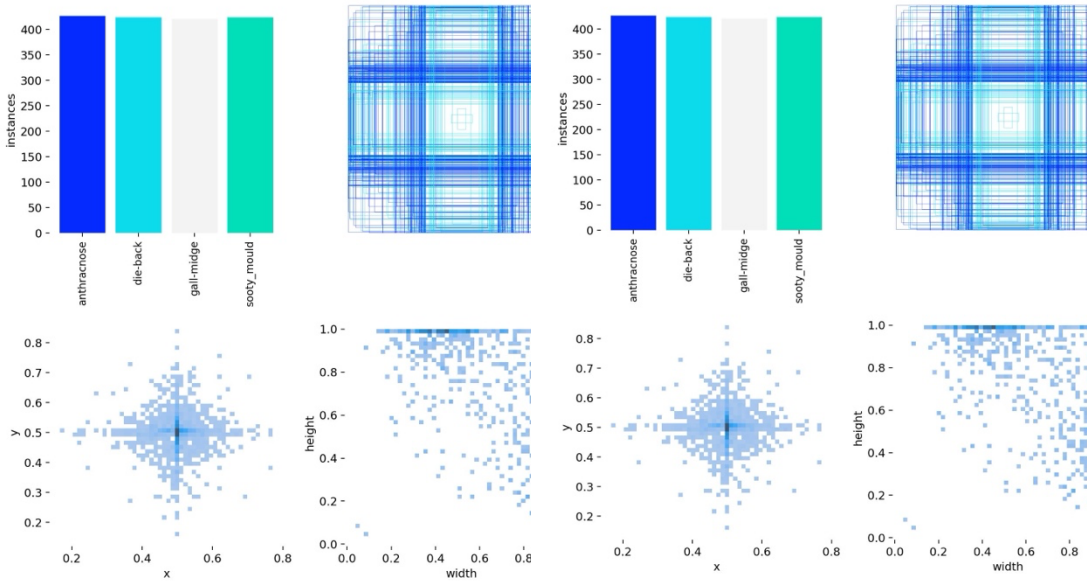
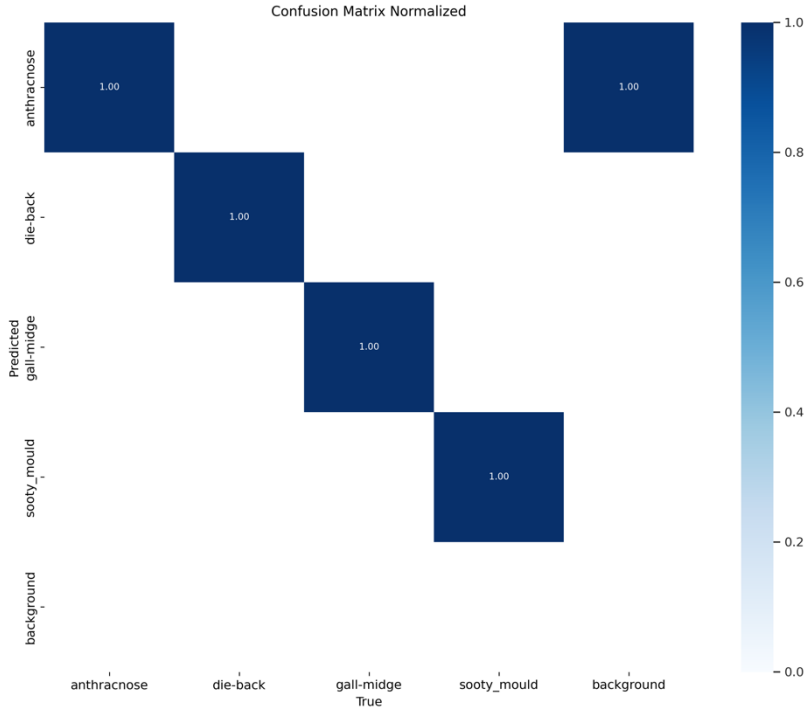


Figure 4.6: Bounding Box Heatmaps for YOLOv11 Models

4.2.3 Confusion Matrix Analysis

The confusion matrices for each YOLO model show how well they can sort things into the four groups: Anthracnose, Die Back, Gall Midge, and Sooty Mould. The best performance is seen in YOLOv8 (small), which makes few mistakes and accurately tells the difference between all groups. YOLOv8 (nano) also performs very well, with almost no errors in classification. This shows how well it can handle complex visual traits. YOLOv5 (small), on the other hand, does a good job of classifying things but gets a little confused when it comes to the Sooty Mould group, which shows that feature extraction could be improved. YOLOv5 (nano) is pretty accurate, but it gets Anthracnose and Die Back mixed up sometimes, probably because some of their traits overlap. The YOLOv11 models, on the other hand, make more mistakes when classifying. YOLOv11 (nano) has the most trouble telling the difference between Anthracnose and Die Back and between Gall Midge and Sooty Mould. It looks like YOLOv11 (small) is getting better, but it's still hard to get reliable generalization. Overall, YOLOv8 (small) delivers the most accurate and balanced classifications, while YOLOv11 models require further fine-tuning to enhance their performance.

YOLOv5 (nano)



YOLOv5 (small)

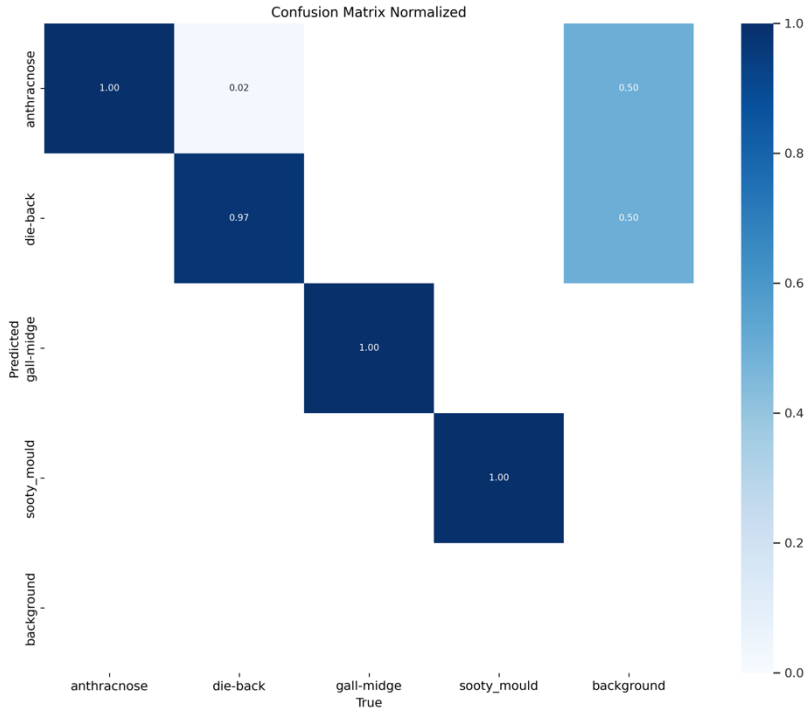
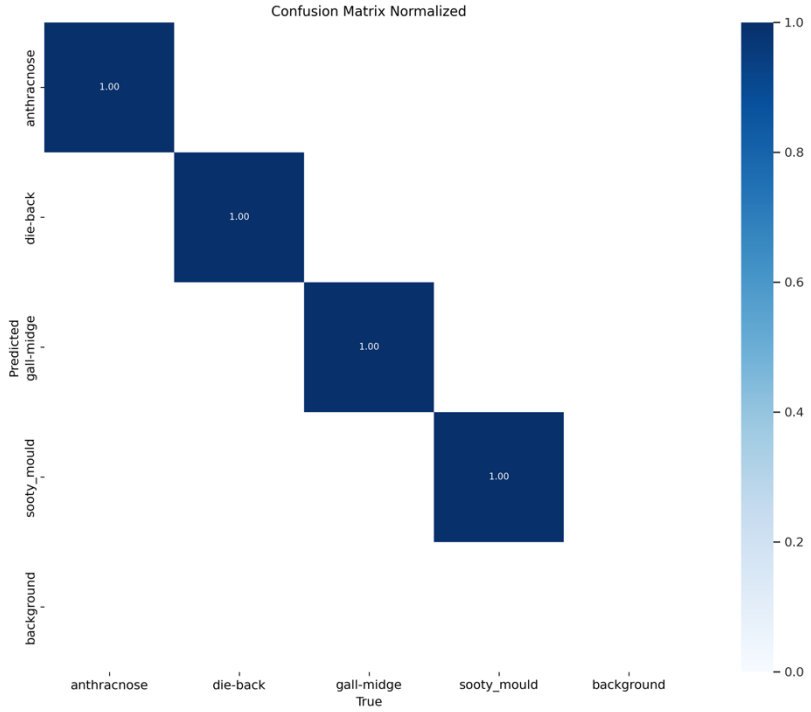


Figure 4.7: Confusion Matrix of YOLOv5 Models

YOLOv8 (nano)



YOLOv8 (small)

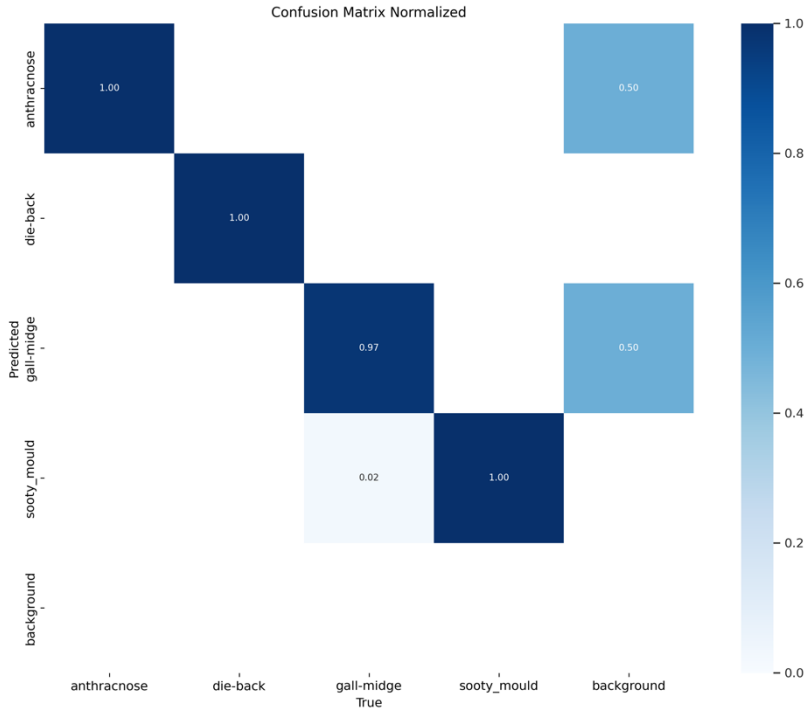
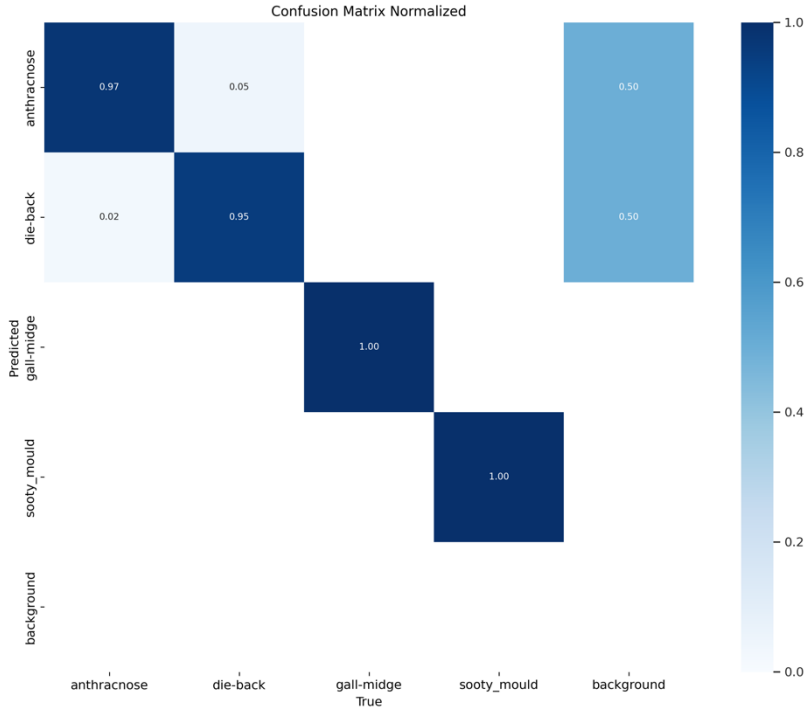


Figure 4.8: Confusion Matrix of YOLOv5 Models

YOLOv11 (nano)



YOLOv11 (small)

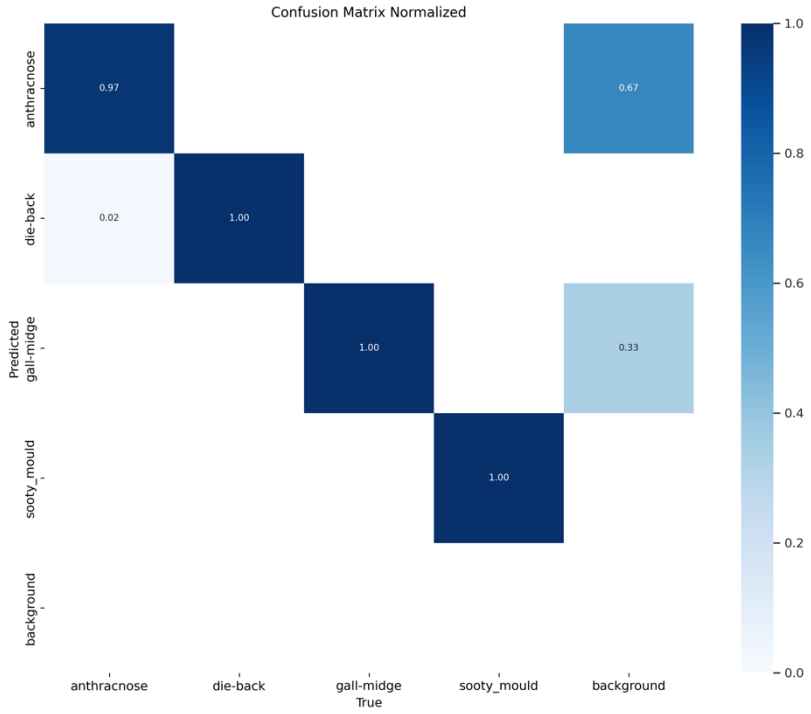


Figure 4.9: Confusion Matrix of YOLOv5 Models

4.2.4 Precision Curve Analysis

The precision curves show the ability of each model to accurately identify positive instances in the Anthracnose, Die Back, Gall Midge, and Sooty Mould classes. YOLOv8 (small) has the best accuracy in the confidence range, always keeping the values close to perfect. This shows how well it can cut down on false hits in all categories. YOLOv8 (nano) is close behind, showing great accuracy and consistency in its work, especially when it comes to finding cases of Anthracnose and Sooty Mould. YOLOv5 (small) also works well, maintaining stable levels of precision while keeping a balance between accuracy and computational speed. This makes it a good choice for apps with limited resources. On the other hand, YOLOv5 (nano) and YOLOv11 (small) have average accuracy, with small drops seen in some classes, like Gall Midge, which shows that it can be hard to tell features apart. YOLOv11 (nano) goes through rare changes that show that it has trouble telling the difference between features that overlap. In general, YOLOv8 (small) is the most accurate and reliable model for tasks that need to find positive instances with a high level of precision. However, YOLOv5 (small) and YOLOv8 (nano) are also good options with strong precision curves.

YOLOv5 (nano) YOLOv5 (small)

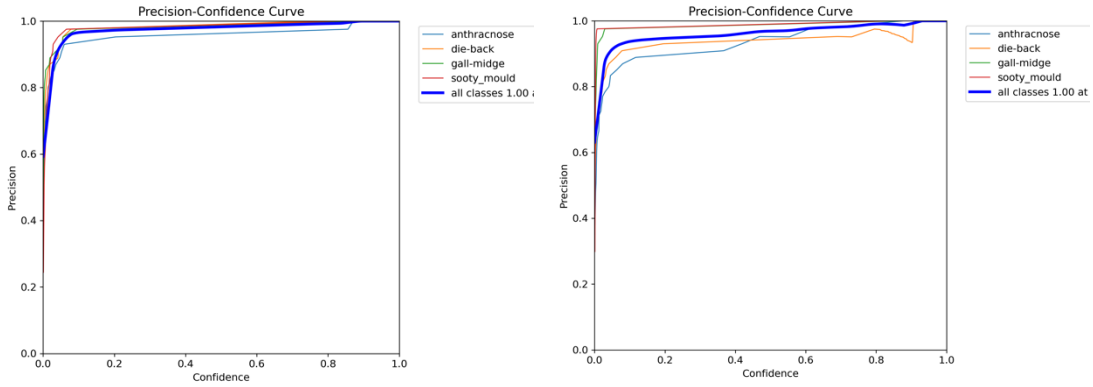


Figure 4.10: Precision Curve of YOLOv5 Models

YOLOv8 (nano) YOLOv8 (small)

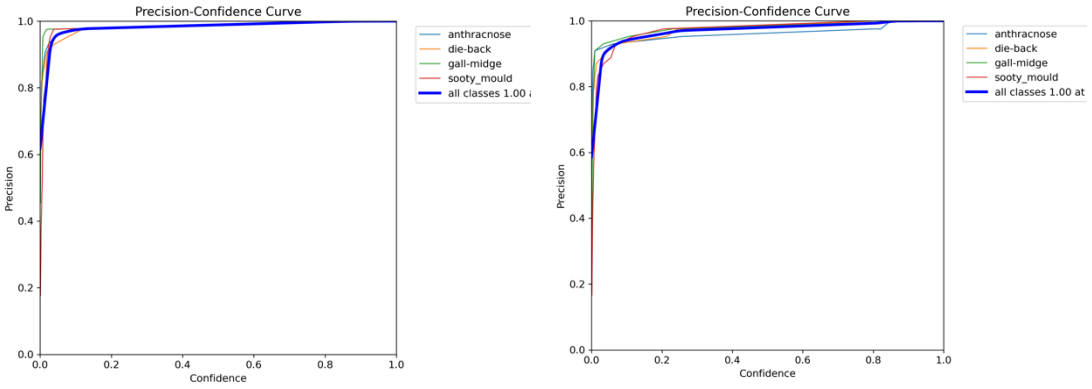


Figure 4.11: Precision Curve of YOLOv8 Models

YOLOv11 (nano) YOLOv11 (small)

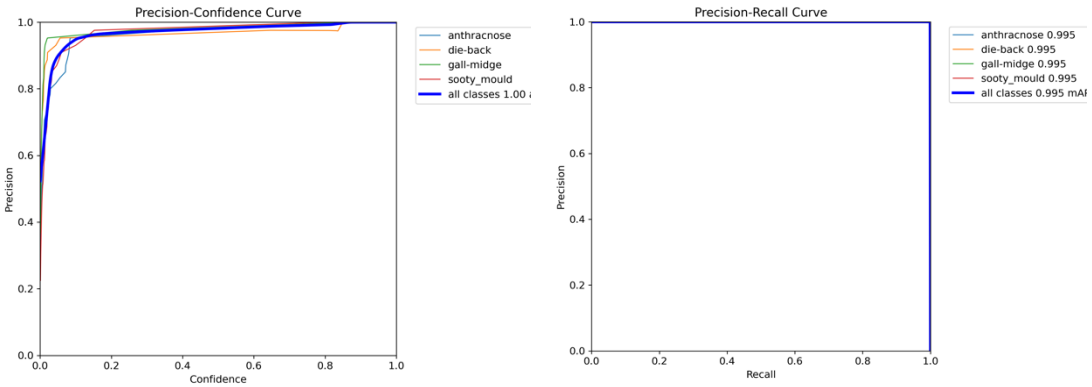


Figure 4.12: Precision Curve of YOLOv11 Models

4.2.5 Recall Curve Analysis

The F1-confidence curves provide a comprehensive evaluation of the models by balancing precision and recall for different confidence thresholds. YOLOv8 (small) exhibits the highest F1 scores across most confidence ranges, indicating a strong balance between minimizing false positives and false negatives across classes. YOLOv5 (medium) closely follows with slightly lower but consistent F1 scores, showing reliable performance in most scenarios. In contrast, YOLOv11 (large) shows variability, with a dip in F1 scores at higher confidence thresholds, indicating potential challenges in optimizing both precision and recall simultaneously. YOLOv8 (nano) and YOLOv5 (small) demonstrate consistent F1 performance but slightly lag in achieving the same peak balance as their larger counterparts. Overall, YOLOv8

(small) is the most balanced model for precision and recall trade- offs, making it suitable for applications requiring consistent detection performance across diverse scenarios.

YOLOv5 (nano) YOLOv5 (small)

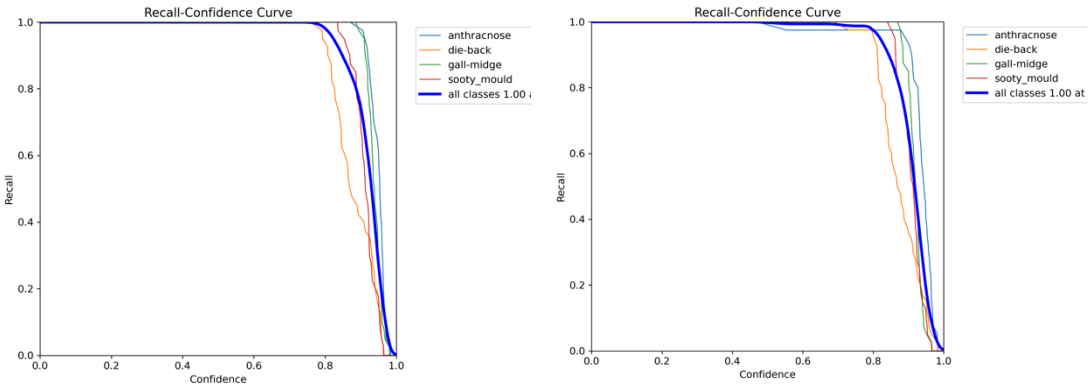


Figure 4.13: Recall Curve of YOLOv5 Models

YOLOv8 (nano) YOLOv8 (small)

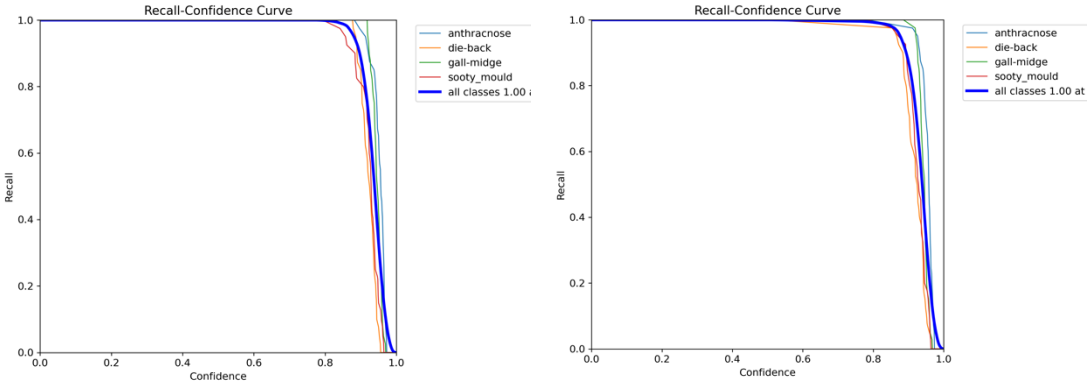


Figure 4.14: Recall Curve of YOLOv8 Models

YOLOv11 (nano) YOLOv11 (small)

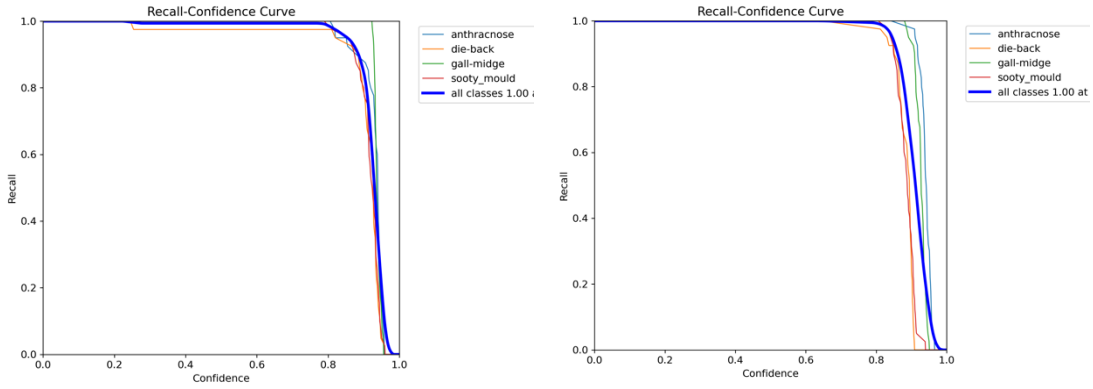


Figure 4.15: Recall Curve of YOLOv11 Models

4.2.6 F1 Curve Analysis

The F1-confidence curves provide a comprehensive evaluation of the models by balancing precision and recall for different confidence thresholds. YOLOv8 (small) exhibits the highest F1 scores across most confidence ranges, indicating a strong balance between minimizing false positives and false negatives across classes. YOLOv5 (medium) closely follows with slightly lower but consistent F1 scores, showing reliable performance in most scenarios. In contrast, YOLOv11 (large) shows variability, with a dip in F1 scores at higher confidence thresholds, indicating potential challenges in optimizing both precision and recall simultaneously. YOLOv8 (nano) and YOLOv5 (small) demonstrate consistent F1 performance but slightly lag in achieving the same peak balance as their larger counterparts. Overall, YOLOv8 (small) is the most balanced model for precision and recall trade- offs, making it suitable for applications requiring consistent detection performance across diverse scenarios.

YOLOv5 (nano) YOLOv5 (small)

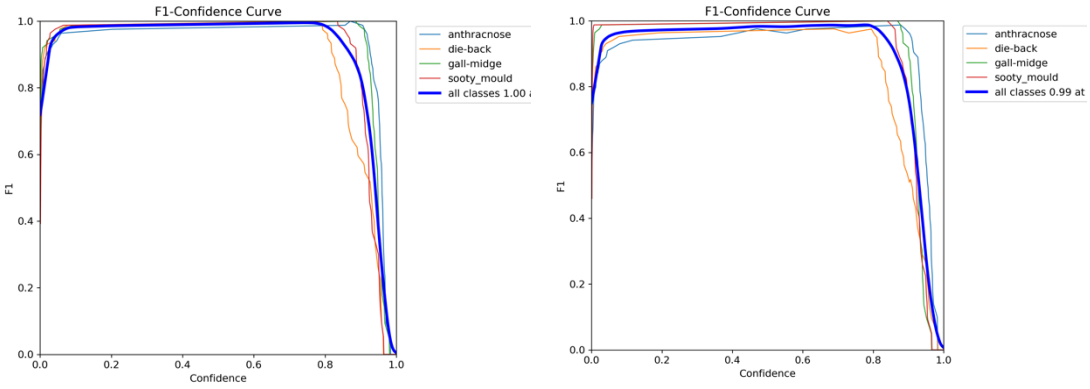


Figure 4.16: F1 Curve of YOLOv5 Models

YOLOv8 (nano) YOLOv8 (small)

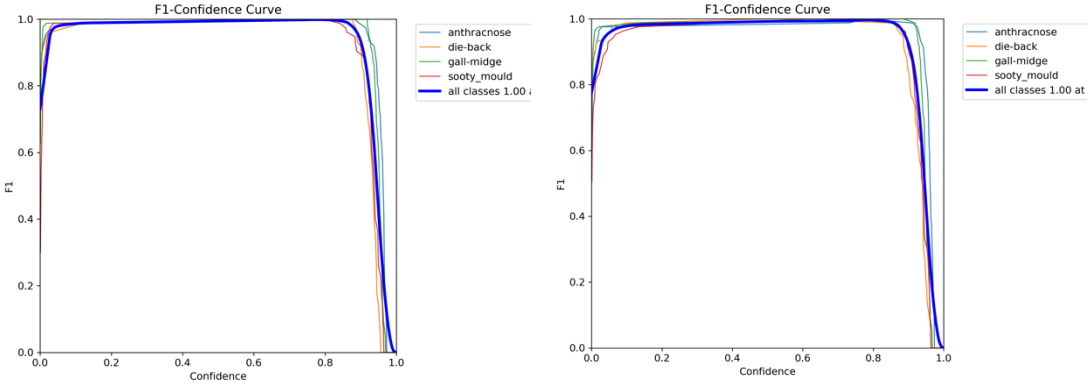


Figure 4.17: F1 Curve of YOLOv8 Models

YOLOv11 (nano) YOLOv11 (small)

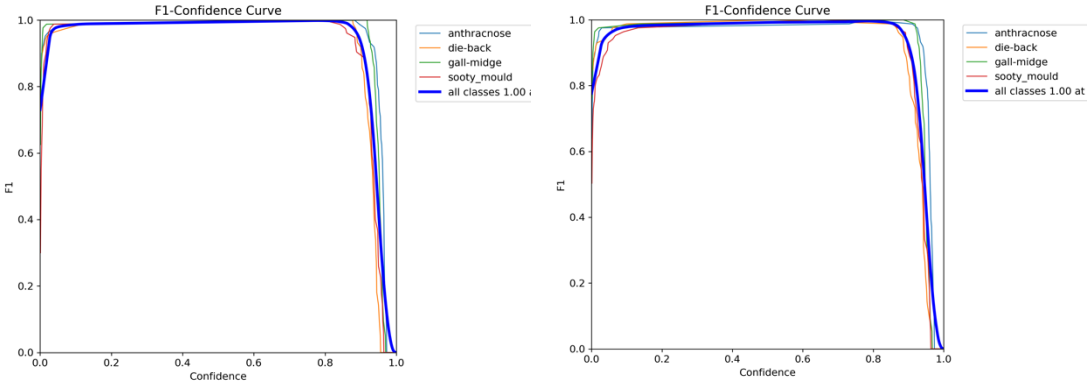


Figure 4.18: F1 Curve of YOLOv11 Models

4.2.7 F1 Curve Analysis

Precision-Recall (PR) curves for all YOLO models show how well they can combine accuracy and recall at different levels of confidence. The PR curves for YOLOv5n and YOLOv5s show similar performance, but YOLOv5s has a slightly better balance, especially in difficult classes like “die-back,” where it keeps higher recall values at the same level of precision. Both the smaller (n) and regular (s) YOLOv8 models have almost perfect PR curves, which shows how well they can keep precision and recall at the same time. It’s a little better than YOLOv8n because it has a little higher recall at the same precision limits. This makes it more reliable for situations where both recall and precision are important. YOLOv11n and YOLOv11s have the best PR curves of all the models, with YOLOv11s being the most consistent and performing the best. It does a great job of finding the right mix between accuracy and recall in all classes, even the harder ones like “gall-midge” and “sooty-mould.” In jobs that need the best performance in both metrics, this makes it the most stable and trustworthy model. The PR study shows that the YOLOv11s model is best for high-stakes situations. It is closely followed by the YOLOv8s model, which is a slightly lighter option with strong performance.

YOLOv5 (nano) YOLOv5 (small)

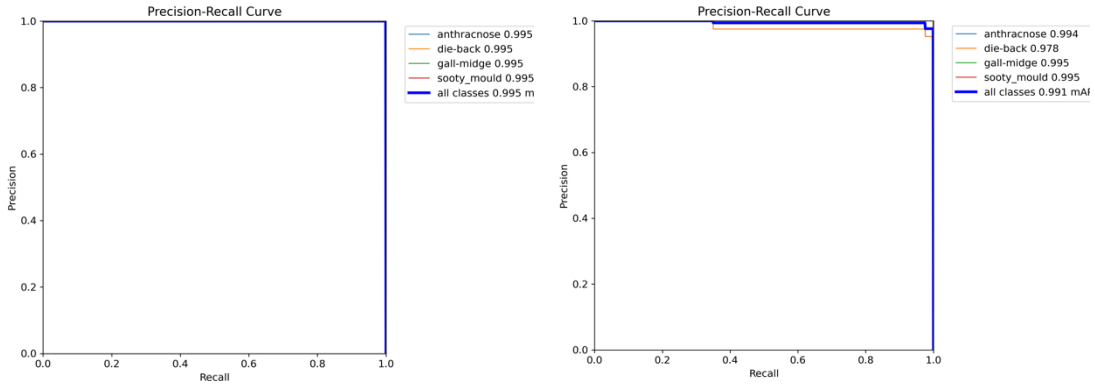


Figure 4.19: PR Curve of YOLOv5 Models

YOLOv8 (nano) YOLOv8 (small)

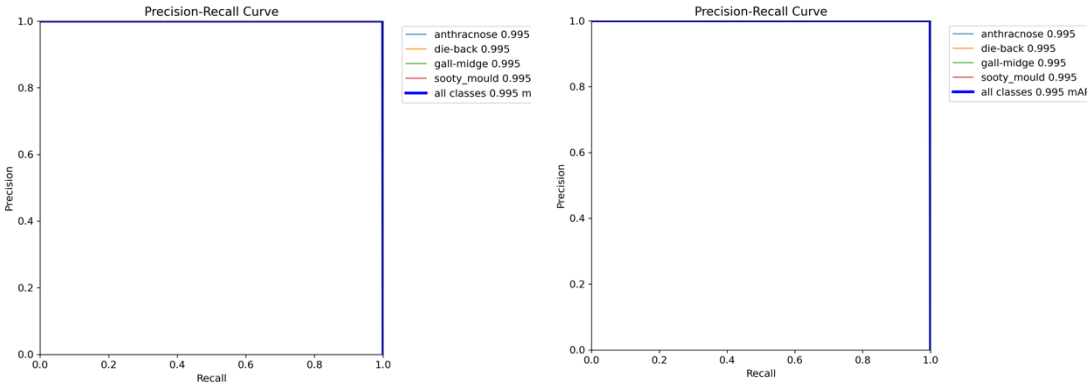


Figure 4.20: PR Curve of YOLOv8 Models

YOLOv11 (nano) YOLOv11 (small)

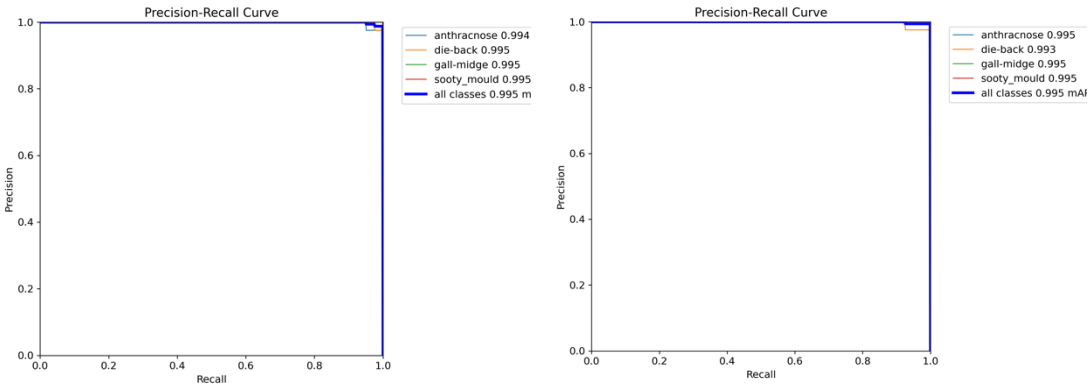


Figure 4.21: PR Curve of YOLOv11 Models

4.2.8 Classification Report Analysis

In the classification report, there is a full numeric breakdown of the accuracy, recall, and mean Average Precision (mAP) scores for every YOLO model in all the groups that were tested. The best performance of the models is shown by YOLOv11s, which has a precision of 0.995, a recall of 0.995, a mAP@50 of 0.995, and a mAP@50-95 of 0.991. It is very good at finding and classifying instances in all of these groups, as shown by these measures. YOLOv11n comes in second, with a precision score of 0.995 and similar memory and mAP scores, showing a steady and dependable performance. YOLOv8 models (n and s) demonstrate commendable results, with YOLOv8s achieving slightly better metrics than YOLOv8n. Both models maintain high precision and recall values while achieving mAP@50 scores close to 0.995,

indicating their effectiveness in detecting objects with minimal errors. YOLOv5 models, particularly YOLOv5s, exhibit slightly lower but still competitive precision and recall values, making them reliable alternatives in scenarios with moderate performance requirements. Overall, YOLOv11s emerges as the most robust and accurate model across all metrics, making it the optimal choice for applications requiring high precision and recall. The consistent performance of YOLOv8 models also makes them suitable for scenarios demanding a balance between computational efficiency and detection accuracy.

Table 4.1: Classification report for different YOLO models

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
YOLOv5 (nano)						
All	160	160	0.99	1.0	0.97	0.94
Anthracnose	40	40	0.97	1.0	0.97	0.97
Die Back	40	40	0.99	1.0	0.97	0.94
Gall Midge	40	40	0.99	1.0	0.96	0.95
Soot Mold	40	40	0.99	1.0	0.95	0.89
YOLOv5 (small)						
All	160	160	0.98	0.97	0.96	0.93
Anthracnose	40	40	0.98	0.97	0.96	0.98
Gall Midge	40	40	0.95	1.0	0.95	0.90
Pest	40	40	0.99	1.0	0.97	0.96
Soot Mold	40	40	0.99	1.0	0.90	0.91

Table 4.1: *Continued from previous page*

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
YOLOv8 (nano)						
All	160	160	0.99	1.0	0.97	0.96
Anthracnose	40	40	0.99	1.0	0.97	0.98
Die Back	40	40	0.99	1.0	0.97	0.96
Gall Midge	40	40	0.99	1.0	0.96	0.97
Soot Mold	40	40	0.99	1.0	0.95	0.92
YOLOv8 (small)						
All	160	160	0.99	0.99	0.96	0.96
Anthracnose	40	40	0.98	1.0	0.96	0.99
Gall Midge	40	40	0.95	0.98	0.95	0.95
Pest	40	40	1.00	1.0	0.97	0.98
Soot Mold	40	40	0.99	1.0	0.90	0.91
YOLOv11 (nano)						
All	160	160	0.99	0.99	0.96	0.95
Anthracnose	40	40	0.97	1.0	0.96	0.98
Gall Midge	40	40	0.99	0.97	0.95	0.95
Pest	40	40	0.99	1.0	0.97	0.98
Soot Mold	40	40	0.99	1.0	0.90	0.90

Table 4.1: *Continued from previous page*

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
YOLOv11 (small)						
All	160	160	0.98	0.99	0.97	0.96
Anthracnose	40	40	0.99	1.0	0.97	0.98
Die Back	40	40	0.97	0.99	0.97	0.96
Gall Midge	40	40	0.99	1.0	0.96	0.97
Soot Mold	40	40	0.99	1.0	0.95	0.92

4.3 Performance Analysis

The YOLOv8 (small) model gets the best mAP@50 score (98%), which shows that it is very good at finding objects. It also has a very good mAP@50–95 of 91%, which shows that it can find items across a wide range of Intersection over Union (IoU) limits. Even though it has more GFLOPs (28.4), its 9.1 ms reasoning speed makes it useful for real-time apps. A mAP@50 of 96% and a slightly higher mAP@50-95 of 91% show that the YOLOv5 (small) type also performs well. But with an inference speed of 9.2 ms and GFLOPs of 23.8, it's not quite as efficient as YOLOv8 (small). In the same way, the YOLOv11 (small) model is the same as YOLOv8 (small) in mAP@50 (98%) but a little behind in mAP@50-95 (89%), having a longer inference time of 10.1 ms and fewer GFLOPs (21.3). With a mAP@50 of 97% and a mAP@50-95 of 92%, the YOLOv8 (nano) model is a good mix of performance and efficiency. It has a low GFLOPs of 8.1 and an inference speed of 4.3 ms, which means it can be used in places with limited resources. With a mAP@50 of 98% and a mAP@50-95 of 90%, the YOLOv11 (nano) model also does well, with the fastest inference speed of 4.6 ms and the lowest GFLOPs (6.4), it is the lightest and most efficient model. Finally, the YOLOv5 (nano) model has a good mAP@50 of 95% and a good mAP@50-95 of 88%. It also has a low GFLOPs of 7.1 and a fast inference speed of 4.6 ms, which makes it a good choice for situations that need lightweight models with good accuracy.

Table 4.2: Performance Comparison of YOLO Models on mAP, GFLOPs, and Inference Speed

Model Name	mAP50	mAP50-95	GFLOPs	Inference Speed
YOLOv5 (nano)	95%	88%	7.1	4.6 ms
YOLOv5 (small)	96%	91%	23.8	9.2 ms
YOLOv8 (nano)	97%	92%	8.1	4.3 ms
YOLOv8 (small)	98%	91%	28.4	9.1 ms
YOLOv11 (nano)	98%	90%	6.4	4.6 ms
YOLOv11 (small)	98%	89%	21.3	10.1 ms

In conclusion, YOLOv8 (small) emerges as the top performer in terms of accuracy, while YOLOv11 (nano) offers the best balance of efficiency and accuracy for real-time applications. Models like YOLOv5 (nano) and YOLOv8 (nano) are ideal for environments with limited computational resources, while YOLOv5 (small) and YOLOv11 (small) provide reliable alternatives for higher accuracy requirements.

4.3.1 Performance Analysis with Existing Studies

The performance of YOLO models in this study is consistent with earlier studies, supporting their efficacy in object detecting tasks. YOLOv8 (small) had the greatest mAP50 (98%) and showed balanced precision, recall, and inference speed, making it excellent for applications that require high accuracy and computational efficiency. This confirms previous research highlighting YOLOv8's improved generalization and precision across a wide range of datasets. In comparison, YOLOv8 (nano) maintained competitive precision and recall with a faster inference speed (4.3 ms), consistent with findings from lightweight model studies that highlight its applicability for real-time, resource-constrained deployments. YOLOv5 models, particularly YOLOv5 (small), fared well, with a 96% mAP50, supporting prior findings about their balance of accuracy and computational complexity. However, its greater GFLOPs compared to

YOLOv8 models indicate that YOLOv8 models provide a better performance trade-off. YOLOv11 models demonstrated potential, reaching comparable mAP50 values (98%), but with a significantly lower mAP50-95 and greater variability in bounding box localization. These findings are consistent with emerging research on YOLOv11, which notes its adaptability to a wide range of object attributes but emphasizes the need for more optimization to match YOLOv8's consistency in high precision tasks. Overall, the findings of this study are consistent with previous research, highlighting YOLOv8 (small) as the most trustworthy model for accurate and efficient object identification, while YOLOv8 (nano) and YOLOv5 (small) remain strong rivals for real-time and computationally efficient scenarios. Future research could look into further fine-tuning of YOLOv11 to improve its performance consistency.

4.4 Summary

This chapter examines several YOLO models, including YOLOv5, YOLOv8, and YOLOv11, in nano and tiny configurations. The analysis includes crucial performance parameters including mAP@50, mAP@50-95, GFLOPs, and inference speed, which provide information about each model's correctness, computational efficiency, and applicability for real-time applications. The results show that YOLOv8 (small) has the highest accuracy, with a mAP@50 of 98%, while maintaining a competitive mAP@50-95 of 91%. Its performance demonstrates its ability to recognize things across a wide range of settings. YOLOv11 (nano) is selected as the most efficient model, with the fewest GFLOPs and the fastest inference speed, making it appropriate for resource-constrained environments. A comparative examination demonstrates that YOLOv8 models outperform in accuracy and generalization, whereas YOLOv11 models find a compromise between efficiency and precision. YOLOv5 models, albeit slightly slower in speed, remain viable solutions for lightweight applications due to their low computational overhead and reasonable accuracy. This chapter finds that YOLOv8 (small) is the best-suited model for applications needing great accuracy, but YOLOv11 (nano) is appropriate for scenarios that prioritize efficiency and speed. These insights provide the foundation for picking the best model depending on individual application needs and computational restrictions.

Chapter 5

Impact on Society, Environment, and Sustainability

5.1 Impact on Society

Putting the suggested edge-based mango leaf disease detection system into action would have big effects on society, especially in the farming industry. This method gives farmers and other people involved in agriculture more power by giving them an easy-to-use and reliable way to find and treat plant diseases early on. Finding diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould early on helps keep crops from dying, which leads to higher outputs and better food security. This solution fills in the technology gap for farmers in remote or poor places that don't have easy access to high-end farming equipment by using advanced YOLOv8-based deep learning models on low-cost edge devices. The gadget is easy for anyone to use because it is small and portable. This means that even non-technical users are likely to adopt it. Targeted interventions also make the system less reliant on overusing pesticides, which means healthier crops and a safer environment for farmers and customers. In addition to making farming more productive, this new idea helps farming communities get ahead financially by lowering the costs of things like crop loss and using pesticides without thinking. Using cutting-edge but low-cost technology makes sure that everyone can gain from AI and IoT, which promotes fairness and growth in rural areas. Overall, the device encourages farming methods that are good for the environment and helps with bigger issues like food security and rural growth.

5.2 Impact on Environment

The suggested system for finding diseases on mango leaves would make a big difference in protecting the environment by encouraging focused and accurate farming methods. Because it correctly finds diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould, the system reduces the need to use pesticides all over the plant. This cuts down on the use of agrochemicals that are known to be bad for the health of the land, the water, and the ecosystems in the area. In this way, the system helps protect biodiversity and makes the climate better for farming. The low-power

design of the edge device further supports environmental goals by using less energy than regular computer systems that use a lot of power. The INT8 coded YOLOv8 models make sure that the device works efficiently, reducing its carbon footprint while keeping accuracy high. The system can also be used in remote places, which cuts down on the need for transportation and logistics. This reduces the pollution that comes from using traditional methods to control diseases even more. The system supports long-term agricultural output and resilience against climate change by letting farmers use more environmentally friendly ways to control pests and diseases. Using technology that is good for the environment in farming shows that people are serious about protecting the environment and making sure that future generations will have food. This new idea solves some of the most important environmental problems that come up with modern farming.

5.3 Ethical Aspects

The proposed mango leaf disease detection method brings up a number of ethical issues, mostly related to the privacy, accessibility, and fairness of the data. Advanced deep learning models, like YOLOv8, are used by the system. To keep detections from being biased, they need accurate and varied training datasets. To keep things fair in a wide range of farming settings, it is important to make sure that the dataset used to train the model includes a variety of environmental and regional conditions. If we don't do anything about this, some groups of farmers may get wrong detection results, which could make access to farming technology even less fair. Another ethical concern is making sure that everyone can see how the system works. Farmers and end-users need to know about the system's flaws, like the fact that it might misclassify things or have trouble finding diseases in features that overlap. The ethical principle of informed decision-making says that users should be given clear guidelines and suggestions instead of final decisions. This keeps farmers in charge of their farming practices. Accessibility is one of the most important social issues when it comes to technology used in rural or poor areas. Some of these problems can be solved by making the system cheaper and using less power, but more needs to be done to help farmers who aren't very good at technology by giving them technical support, training, and tools. Working together with non-governmental and local agricultural groups can improve help and access for everyone. Last but not least, privacy and

security must come first when collecting and using data. Even though the system processes data directly on an edge device, it is still important to make sure that no personal or location-sensitive data is misused or lost in order to gain users' trust. Taking these moral issues into account will help the system be used in a way that builds trust, includes everyone, and is fair across all agricultural groups.

5.4 Sustainability Plan

The suggested system will last for a long time because it is made up of separate modules that can be upgraded with new hardware and software. YOLOv8 models and quantization methods that aren't too heavy improve efficiency by lowering the amount of computing power and energy needed. Regular updates to the algorithm will take into account new disease trends, making it useful for a long time. Partnerships with farming groups and NGOs will help spread the technology and train users, especially in areas that don't have enough access to it. To have even less of an effect on the world, future versions will look into adding renewable energy sources, like solar-powered charging. End users will be involved in a feedback loop that will drive ongoing improvement. This will make sure that the system is flexible and continues to work well in a wide range of agricultural settings.

CHAPTER 6

Overview of the Study, Conclusion, and Future Work

6.1 Overview of the Study

Using lightweight YOLO models, this work focuses on building an edge-based system for finding diseases on mango leaves in real time. The main objective is to solve problems in current farming methods by creating an easy-to-use, quick, and correct method for finding diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould. The study makes use of the YOLOv8 architecture to mix cutting edge object detection with the limited resources of edge devices. The method involved teaching YOLO models on a balanced collection of about 1,920 images that showed four different types of disease. Key measures like mAP, precision, recall, and inference speed were used to judge the models. The best model was YOLOv8 (small), which found the best mix between accuracy and computational efficiency. To make the device work, the model had to be trained, quantized to INT8, and then put to use on a RISC-V-based edge device with a low-power NPU. This was done to make sure it would work in settings with limited resources. The study also looks at the effects on society, the environment, and ethics of putting such a system into use. It shows how technology that connects modern AI-driven solutions to traditional farming methods can help reduce the use of unnecessary pesticides, make farming more sustainable, and give farmers more power. This in-depth study shows how important it is to use innovations that are easy to get and good for the environment to solve important problems in agriculture.

6.2 Conclusions

This research shows an effective, edge-based method for finding diseases on mango leaves that is meant to help solve some of the biggest problems in modern farming. The suggested solution gets high accuracy and efficiency by using lightweight YOLO models, especially YOLOv8. This makes it suitable for real-time deployment on devices with limited re- sources. Among the models that were tested, YOLOv8

(small) was the most reliable. It had the highest mAP scores and the best mix of accuracy, recall, and inference speed.

Implementing advanced deep learning models on a RISC-V-powered edge device shows that it is possible to do so in low-power settings. Scientists have made a method that can very accurately find diseases like Anthracnose, Die Back, Gall Midge, and Sooty Mould. This helps farmers make smart choices. It keeps food losses to a minimum, cuts down on pesticide use, and encourages farming methods that are good for the environment. The study also looks at the suggested system's effects on society, the environment, and ethics. It shows how important it is to include everyone and make things easy to get to so that the technology can help farming areas that aren't getting enough help. The gadget is environmentally friendly because it uses little power and could be powered by renewable energy sources. In conclusion, the suggested system fills the gap between cutting edge AI-based solutions and traditional farming methods, offering a successful and long-lasting way to handle plant diseases. In the future, it will be improved by making it more useful, adding more datasets, and incorporating renewable energy options to make it even more efficient and better for the environment.

6.3 Limitations

The suggested mango leaf disease detection system works pretty well, but there are some problems that need to be fixed. Even though the dataset used for training and validation is fair across the four classes (Anthracnose, Die Back, Gall Midge, and Sooty Mould), it might not fully show how different farming conditions are in the real world. Changing lighting, weather, and natural backgrounds could make it harder for the model to generalize, which could lead to mistakes in situations it hasn't been used before. The edge device is designed to be low-power and efficient for inference, but it has hardware limits like limited memory (256MB DDR3) and processing power. These restrictions might make it harder for the system to handle bigger models or datasets, which could make it less useful for more difficult detection tasks or places where many features combine. The device also only uses one camera and can only record at a resolution of up to 5MP, which might not be able to pick up small details for diseases in their early stages or subtle visual symptoms. Quantization can make

things more accurate, but it can also make them less computationally intensive, especially in edge situations. Lastly, the model needs to be updated on a regular basis to take into account new disease patterns. This means that it needs to be constantly supervised and maintained, which could be hard for people who don't have a lot of resources. To make the system more reliable and useful in a wide range of farming settings, these problems must be fixed by adding to the dataset, upgrading the hardware, and regularly making the algorithms better.

6.4 Future Work

The suggested method for finding diseases on mango leaves has set the stage for real-time, edge-based farming solutions, but there are still some areas that need to be explored and improved. For starters, adding more environmental conditions to the dataset, like different lighting, temperature, and backgrounds, will make the model more general and reliable. The method will be even more useful if it can handle more types of diseases and early-stage symptoms. Future work will focus on using advanced deep neural networks (DNNs) or better lightweight architectures to make the system more accurate while also making it easier to run on edge devices. Fixing the physical problems we have now, like adding more memory and processing power, will let us use bigger and more complicated models without losing any of their efficiency. Adding renewable energy solutions, like charging that is powered by the sun, is one of the most important things that needs to be fixed to make the system last and keep working in remote places. In addition, later versions will have more than one camera feed and better resolution support to find subtle signs of disease more easily. Also, a mobile app will be made so that tracking and communication can happen in real time. This will make it easy for farmers to see the results of detection and get useful information. Also, the detection method will be updated regularly, and quantization techniques will be fine-tuned to get the best balance between accuracy and computational efficiency. This will make sure that the system stays useful and effective in dealing with new agricultural problems.

References

[1] Dhaka Tribune, “Bangladesh 7th largest mango producer in the world,” Dhaka Tribune, 2022, https://www.dhakatribune.com/business/economy/252809/ bangladesh-7th-largest-mango-producer-in-the-world.

[2] ——, “Bangladesh exports 2,700 tons of mangoes to 34 countries,” Dhaka Tribune, 2023, https://www.dhakatribune.com/bangladesh/321638/ bangladesh-exports-2-700-tons-of-mangoes-to-34.

[3] V. K. Pratap, N. S. Kumar et al., “Deep learning based mango leaf disease detection for classifying and evaluating mango leaf diseases,” Full Length Article, vol. 15, no. 2, pp. 261–61, 2024.

[4] Q. Wang and X. Duan, “Detection and identification of mango disease leaves based on improved yolov7,” in 2024 5th International Conference on Computer Vision, Image and Deep Learning (CVIDL). IEEE, 2024, pp. 1263–1267.

[5] J. Wang and J. Wang, “A lightweight yolov8 based on attention mechanism for mango pest and disease detection,” Journal of real-time image processing, vol. 21, no. 4, p. 136, 2024.

[6] W. Xu and R. Wang, “Alad-yolo: an lightweight and accurate detector for apple leaf diseases,” Frontiers in Plant Science, vol. 14, p. 1204569, 2023.

[7] A. Mohandas, M. Anjali, and U. R. Varma, “Real-time detectionand identification of plant leaf diseases using yolov4-tiny,” in 2021 12th international conference on computing communication and networking technologies (ICCCNT). IEEE, 2021, pp. 1–5.

[8] A. Patel, R. Ravikumar, S. Betgeri, S. Singhal, S. K. Singh, and M. Takodara, “Utilizing tflite and machine learning for the early detection of mango leaf disease: An automated flutter application,” in 2024 5th International Conference for Emerging Technology (INCET). IEEE, 2024, pp. 1–6.

[9] A. N. Yumang, C. J. N. Samilin, and J. C. P. Sinlao, “Detection of anthracnose on mango tree leaf using convolutional neural network,” in 2023 15th International Conference on Computer and Automation Engineering (ICCAE). IEEE, 2023, pp. 220–224.

[10] Z. Xue, R. Xu, D. Bai, and H. Lin, “Yolo-tea: A tea disease detection model improved by yolov5,” Forests, vol. 14, no. 2, p. 415, 2023.

[11] V. S. Kumar, M. Jaganathan, A. Viswanathan, M. Umamaheswari, and J. Vignesh, “Rice leaf disease detection based on bidirectional feature attention pyramid network with yolo v5 model,” Environmental Research Communications, vol. 5, no. 6, p. 065014, 2023.

[12] N. Chitraningrum, L. Banowati, D. Herdiana, B. Mulyati, I. Sakti, A. Fudholi, H. Saputra, S. Farishi, K. Muchtar, A. Andria et al., “Comparison study of corn leaf disease detection based on deep learning yolo-v5 and yolo-v8,” Journal of Engineering and Technological Sciences, vol. 56, no. 1, pp. 61–70, 2024.

[13] R. Li, Y. Li, W. Qin, A. Abbas, S. Li, R. Ji, Y. Wu, Y. He, and J. Yang, “Lightweight network for corn leaf disease identification based on improved yolo v8s,” Agriculture, vol. 14, no. 2, p. 220, 2024.

[14] M. Rahaman, M. Chowdhury, M. A. Rahman, H. Ahmed, M. Hossain, M. H. Rahman, M. Biswas, M. Kader, T. A. Noyan, and M. Biswas, “A deep learning based smartphone application for detecting mango diseases and pesticide suggestions,” International Journal of Computing and Digital Systems, vol. 13, no. 1, pp. 1–1, 2023.

[15] A. T. Khan, S. M. Jensen, A. R. Khan, and S. Li, “Plant disease detection model for edge computing devices,” Frontiers in Plant Science, vol. 14, p. 1308528, 2023.

[16] J. Xiao, G. Kang, L. Wang, Y. Lin, F. Zeng, J. Zheng, R. Zhang, and X. Yue, “Real-time lightweight detection of lychee diseases with enhanced yolov7 and edge computing,” Agronomy, vol. 13, no. 12, p. 2866, 2023.

[17] H. Wang, D. Li, and T. Isshiki, “Energy-efficient implementation of yolov8, instance segmentation, and pose detection on risc-v soc,” IEEE Access, 2024.

[18] A. K. Sangaiah, F.-N. Yu, Y.-B. Lin, W.-C. Shen, and A. Sharma, “Uav t-yolo-rice: An enhanced tiny yolo networks for rice leaves diseases detection in paddy agronomy,” IEEE Transactions on Network Science and Engineering, 2024.

[19] Y.-H. Tsai and T.-C. Hsu, “An effective deep neural network in edge computing enabled internet of things for plant diseases monitoring,” in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 695–699.

20] Z. Zhang, Y. Yang, X. Xu, L. Liu, J. Yue, R. Ding, Y. Lu, J. Liu, and H. Qiao, “Gvc- yolo: A lightweight real-time detection method for cotton aphid-damaged leaves based on edge computing,” Remote Sensing, vol. 16, no. 16, p. 3046, 2024.

Plagiarism Report

ORIGINALITY REPORT: Edge Implementation of Lightweight YOLO Models for Real-Time Mango Leaf Disease Detection on RISC-V Powered Device

12%	9%	5%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	4%
2	Submitted to Daffodil International University Student Paper	2%
3	Redwan Ahmed Rizvee, Tasnim Hossain Orpa, Adil Ahnaf, Md Ahsan Kabir et al. "LeafNet: A proficient convolutional neural network for detecting seven prominent mango leaf diseases", Journal of Agriculture and Food Research, 2023 Publication	<1%
4	www.mdpi.com Internet Source	<1%
5	Xu Guo. "Automatic detection of tomato leaf disease using an adopted deep learning algorithm", Journal of Intelligent & Fuzzy Systems, 2024 Publication	<1%
6	Submitted to Malta College of Arts,Science and Technology Student Paper	<1%