A COMPARATIVE STUDY OF YOLOv8 AND FASTER R-CNN IN POTATO, APPLE & GRAPE LEAF DISEASE DETECTION FOR PRECISION

Final Year Design Project

By

Mahfuj Hosen

212-15-4120

&

Tanoy Barman

212-15-4160

FINAL YEAR DESIGN PROJECT REPORT

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

Supervised by

Mr. Shah Md Tanvir Siddiquee

Assistant professor

Department of Computer Science and Engineering

Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

Dhaka, Bangladesh

14 May, 2025

APPROVAL

APPROVAL

This Project titled **"A COMPARATIVE STUDY OF YOLOv8 AND FASTER R-CNN IN POTATO, APPLE & GRAPE LEAF DISEASE DETECTION FOR PRECISION"**, submitted by **Md Mahfuj Hosen**, ID No: **212-15-4120** & **Tanoy Barman,** ID No: **212-15-4160** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **14 May, 2025**.

<u>BOARD OF EXAMINERS</u>

Ms. Nazmun Nessa Moon (NNM) — **Chairman**
Associate Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Shah Md Tanvir Siddiquee (SMTS) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Md Umaid Hasan (MUH) — **Internal Examiner**
Sr. Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Ahmed Wasif Reza (DWR) — **External Examiner**
Professor
Department of Computer Science and Engineering
East West University

DECLARATION

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Mr. Shah Md Tanvir Siddiquee, Assistant Professor**, 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:

Mr. Shah Md Tanvir Siddiquee (SMTS)
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Submitted by:

Md Mahfuj Hosen
Student ID:**212-15-4120**
Department of Computer Science and Engineering
Daffodil International University

Tanoy Barman
Student ID:**212-15-4160**
Department of Computer Science and Engineering
Daffodil International University

iii

©Daffodil International University

ACKNOWLEDGEMENTS

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

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

We are grateful and wish our profound indebtedness to **Mr. Shah Md Tanvir Siddiquee**, **Assistant Professor**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **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 stages have made it possible to complete this project.

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

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

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

ABSTRACT

This study investigates the use of deep learning models, specifically YOLOv8 and Faster R-CNN, for real-time object identification and autonomous plant disease diagnosis. The research uses a dataset consisting of 38 examples from 11 distinct classes, which include various plant species such as apple, grape, and potato. Within these categories, there are specific classes representing healthy conditions and various diseases. This diverse dataset, featuring 3171 apple images, 4063 grape images, and 2852 potato images, supports comprehensive training and evaluation of machine learning models for disease identification. A statistical analysis of the subset reveals different distributions of images across the classes, highlighting the prevalence and significance of certain diseases within each plant category. The effectiveness of YOLOv8 and Faster R-CNN is assessed using performance metrics like Intersection over Union (IoU), saliency score, and inference time. Although specific numerical values are not provided, the data indicate that YOLOv8 performs well in terms of IoU and achieves a higher saliency score compared to Faster R-CNN. Conversely, Faster R-CNN shows superior IoU performance but with a lower saliency score. Additionally, YOLOv8 demonstrates faster inference times, while Faster R-CNN has significantly longer inference times.By comparing these metrics, the study provides valuable insights into the strengths of each model, guiding the selection and optimization of deep learning architectures for plant disease diagnosis. The research also emphasizes the trade-offs between speed and accuracy inherent in object identification models, underscoring the importance of considering application-specific requirements. Overall, this study advances agricultural technology by exploring the potential of deep learning models in combating harmful plant diseases. It lays the groundwork for future advancements in autonomous plant disease diagnosis and contributes to global food security by enhancing diagnostic techniques.

KEYWORDS: CNN , IoU , Saliency Score , Yolov8 , FRCNN

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This section presents the research on real-time autonomous plant disease diagnosis using deep learning models, YOLOv8 and R-CNN which is faster, to detect diseases in apple, grape, and potato plant leaves. It summarizes the motivation, objectives, methods, expected results, and report structure providing a pursuable outline of the purpose and organization of this study.

1.1 Introduction

Agriculture functions as the essential foundation of economic advancement and food security for almost all countries, as agriculture serves as the fundamental source of food and is one of the greatest creators of global wealth. This vital part of our existence functions as the backbone of societal development and stability worldwide. Unfortunately, the increasing prevalence and impact of plant diseases may fundamentally threaten economic sustainability and food supply reliability. Plant diseases, and the associated reductions in agricultural production, threaten agricultural productivity and production, which are essential for a healthy global economy. In an increasingly interconnected world, the issues related to plant diseases are exacerbated by a number of factors including: increased global trade and movement of goods, the accelerating rate of climate change, global drought, and related extreme climate events that seriously facilitate pathogen transfer and their subsequent impact to food production. As a result, there is an extreme and historic lack of supported capacity and ability to effectively diagnose and manage these challenges.

Deep learning is emerging as a game-changing technology which provides distinct ways to correctly and efficiently identify and treat plant diseases. By utilizing updated technologies in computer science, machine learning approach, and computer tehcnology to design and implement new approaches, researchers are beginning to develop methods of early disease detection and intervention that could fundamentally change the way we approach plant diseases and agricultural sustainability. Through these advancements, it is not only possible to improve the effective treatment of plant diseases, but to also strengthen preventative measures, ensuring a resilient food source. This thesis, and therefore research project, aims to investigate and develop artificial intelligence based methods of diagnosis and management of diseases of the plants.

The significance of this endeavor is amplified by rapid advancements in computational technologies, which unlocking new opportunities to address the multifaceted challenges are posed by plant diseases. From all of these tenchologies, deep learning come out on top for its outstanding ability to tackle critical tasks such as crop disease identification and management. Plant diseases can lead to substantial annual losses in agricultural output, threatening global food security. Early detection is paramount to minimizing these losses and optimizing disease control efforts. Traditional diagnostic methods—such as visual inspection of leaf characteristics like spot color, size, and shape—are often labor-intensive, require specialized expertise, and lack precision and scalability. In contrast, modern object detection techniques, powered by models like YOLOv8 and Faster R-CNN, offer automated, accurate, and scalable solutions for identifying and classifying plant diseases.

These advanced algorithms, supported by tools such as saliency score analysis and class activation mapping, enable precise identification of disease symptoms. Researchers have utilized these methods to evaluate key performance metrics, including inference time and model accuracy, to determine their suitability for real-time disease monitoring. Class activation mapping further enhances model interpretability by highlighting regions of interest, facilitating targeted treatment strategies. By integrating agricultural needs with technological innovation, this research seeks to harness the synergy between deep learning and plant science to revolutionize crop protection. The potential impact is profound: improved disease management could enhance agricultural yields, reduce losses, and bolster food security worldwide. As scientific and technological progress accelerates, deep learning to plant disease diagnosis represents a pivotal step toward a more sustainable and prosperous future for global agriculture.

1.2 Motivation

By 2050, the world population is likely to reach almost 10 billion which will put intense pressure on agricultural systems to satisfy rising food demand. It is estimated that food must be produced 70 % more than today to support such growth. Plant diseases can cause crop loss rates of up to 100 % in more severe affected regions, presenting huge risks to food security and economic stability, especially in developing countries. The traditional approach to detect diseases relies on a manual process, and expert knowledge base which is becoming less practical as challenges arise with climate change and international trade which facilitate the rapid spread of pathogens. In light of this, the availability of affordable technology, such as the smartphone (over 5 billion users as of 2020), has introduced a unique opportunity to democratize advanced diagnostic technology. Deep learning algorithms like faster R-CNN and YOLOv8 are very helpful technologies because they don't solely rely on human skill, they will allow for automatic, accurate, and real-time diagnosis of plant diseases. Thus, the motivation for this research is in part due to the urgency of global challenges, and the desire to leverage technologically advanced tools so that crop productivity can be improved, economic losses minimized, and sustainable farming practices supported during these periods of rapid environmental change.

1.3 Objectives

This study's primary goal is to improve plant disease diagnosis and monitoring using cutting-edge deep learning techniques in order to support agricultural sustainability and global food security. This research will develop and assess automated systems for diagnosing plant diseases in real-time as a response to the challenges plant diseases present in high-tech agriculture. To address these challenges, there are the following specific aims to this study:

1. **Develop and Deploy Deep Learning Models for Plant Disease Detection:** To develop and deploy state-of-the-art object detection models, specifically YOLOv8 and Faster R-CNN, in order to accurately identify and localize the presence of plant diseases in primary crop species such as apples, grapes, and potatoes.
2. **Evaluate Models Under Real-World Conditions:** To assess YOLOv8 and Quick R-CNN's performance on accuracy, recall, precision, inference speed, and localization accuracy, focusing on usefulness as a real-time agricultural disease monitoring tool.
3. **Compare Speed vs Accuracy:** To evaluate YOLOv8 and Faster R-CNN in a comprehensive comparison study that clarifies the speed versus accuracy relationship that guides model selection for definition of agricultural applicability.
4. **Support Earlier Detection of Disease and Treatment:** To establish methods to support earlier detection of plant diseases through automation of methods that increase intervention speed, which can reduce crop losses and improve yield stability.
5. **Maximize Model Interpretability and Scalability:** To employ methods, like saliency score analysis and class activation mapping, in order to implement model prediction interpretability and to ensure scalability to ultimately meet the needs of a large-scale farming operation by optimizing data preprocessing and model architectures.
6. **Contribute to Sustainable Agriculture:** To provide a thorough technological framework to support sustainable agricultural practices in the underlying sense of mitigating the economic and environmental disadvantages of plant diseases while avoiding food supply shortages on both global and regional scales.

The combined objectives of the goals are to connect innovative technology with real-world agricultural applications. This research aims to deliver actionable insight and tools that equip farmers, researchers, and policymakers to improve their efforts against plant diseases and ultimately seek a productive and sustainable agriculture future.

1.4 Methodology

The complete methodologies described are multifaceted: dataset preparation, model configuration, training and evaluation. After image cleaning, the datasets contained 1,100 images (100 per class, and Potato Healthy had 115 total). The datasets were pre-processed as given. Models using the YOLOv8 and Faster R-CNN architectures were formatted for plant disease detection. YOLOv8 was a single-stage, anchor-free framework with

CSPDarknet53 as the backbone along with the C2f module, and featured pyramid networks (FPNs) for multi-scale network detection. Faster R-CNN was a two-stage process where a Region Proposal Network (RPN) was incorporated with a ResNet50 backbone along with ROI pooling and classification. The pre-processed dataset was used for 100 epochs of the algorithm's training process, with the hyper parameters (learning rate, batch size, etc.) changed for better results. The training was done on an NVIDIA Tesla T4 GPU, on which the CUDA 12.1 software was utilized. The models were validated and tested once training was completed using precision (0.85-0.95), recall, F1-score and IoU for assessing performance. When examining detection accuracy, we also explored different techniques to improve detection, use of memory mapping, and experimentation with other grid sizes (e.g., 14 x 14 and 7 x 7).

1.5 Project Outcome

The research effectively developed and validated an autonomous and real-time plant disease diagnostic system utilising YOLOv8 and a Faster R-CNN framework in tandem with two distinct deep learning models. The research emphasised the identification and classification of diseases in three focal crops – apples, grapes, and potatoes – using a carefully prepared dataset from PlantVillage. The findings of this project support the feasibility and promise of the models to radically change disease management within agriculture, as well as provide evidence as to the models performance, trade-offs, and applicability.

Regarding the implementation of YOLOv8 and Faster R-CNN, it produced adequate results for identifying and localising plant diseases, such as Apple Scab, Grape Black Rot, and Potato Early Blight, with more diseases included. Each model quantitatively performed differently, thus having their own benefits. For YOLOv8 its achievement of a confidence score of 96.77, had an inference time of 7.83 seconds per iteration which demonstrated its speed and therefore fit for real-time implementations. Compared to YOLOv8, which had an intersection over union (IoU) score of 0.30, Faster R-CNN obtained a confidence score of 98.68 and an IoU of 87.60, indicating that Faster R-CNN detected disease more precisely through object localization. Each of the aforementioned metrics were validated through data analysis which included confusion matrices, precision-recall graphs, and class activation maps (CAM), which were captured within the exploratory results section within Figures 4.1-4.13.

The comparative analysis underscored a critical trade-off between speed and accuracy. YOLOv8 excelled in computational efficiency, processing images rapidly with an inference time significantly lower than Faster R-CNN's 57.82 seconds, making it ideal for scenarios requiring immediate decision-making, such as field-based monitoring with resource-constrained devices. Conversely, Faster R-CNN's region-based approach provided higher precision and recall, particularly in complex scenarios with small or overlapping disease symptoms, as evidenced by its saliency score of 9.46 and detailed CAM visualizations. This precision is vital for applications where accurate diagnosis and targeted treatment are paramount.

The project also produced practical tools and methodologies to enhance disease detection. The use of data preprocessing techniques—such as resizing, annotation with Roboflow, and augmentation—ensured a robust dataset, while memory mapping improved temporal consistency in predictions. Evaluation criteria, including precision (0.85–0.95), recall, and IoU, provided a rigorous benchmark for model performance. The integration of YOLOv8 and Faster R-CNN into a unified framework demonstrated reliable detection across diverse plant species and disease categories, with confusion matrices (Figures 4.3 and 4.4) revealing high true positive rates for healthy classes (e.g., 1.00 for "apple healthy" in Faster R-CNN) and areas for improvement in specific diseases like Grape Leaf Blight (0.80 true positive rate).

The results of this project carry significant implications for agriculture. The ability to improve precision and timeliness in disease detection allows for crop loss to be reduced, yield stability to be improved, and support sustainable practice. The speed of the inference engine (YOLOv8) creates a scalable solution for the detection of potential disease in real-time in larger areas with varying speeds of inference time. While the precision of the predictions in Faster R-CNN supports a narrower application, such as precision agriculture and the research realm. Successful application of the models on the dataset of 1100 images (after it was cleaned) makes it clear that the models can also be used for future projects that may incorporate solutions through IoT and/or mobile platforms which can benefit farmers around the world.

In conclusion, from this project the goals impressed upon the authors for use of deep learning models for plant disease diagnosis were successfully completed by developing, evaluating and comparing the results from YOLOv8 and Faster R-CNN. The outcomes of the project highlight the strengths that each model can bring to the application itself, but also work as a part of a complete toolbox to try and solve the problems that occur in agriculture. The results from this project allow for future work to improve the models that were used, including: hyper parameter tuning and increasing the depth of the dataset to improve both accuracy and scalability, which could help with global food security, along with improving agricultural durability and resilience.

1.6 Organization of the Report

This report is structured to provide a comprehensive exploration of the development, implementation, and evaluation of real-time autonomous plant disease diagnosis using deep learning models, specifically YOLOv8 and Faster R-CNN. It is organized into distinct sections that guide the reader through the research process—from its conceptual foundation to its practical outcomes—ensuring clarity and logical progression. The following outline describes the organization of the report:

- Introduction: This section establishes the significance of agriculture as a global economic and sustenance driver, highlighting the threat posed by plant diseases and the urgent need for innovative solutions. It introduces deep learning as a transformative approach to disease detection and sets the stage for the research by emphasizing its potential to enhance food security and agricultural productivity.

- Motivation: This section articulates the driving forces behind the study, including the escalating global food demand, the limitations of traditional diagnostic methods, and the opportunities presented by advancements in artificial intelligence. It underscores the necessity of this research in addressing real-world agricultural challenges and its relevance to sustainable farming.
- Objectives: In this section, we outline the specific objectives of the research, namely the aims of developing and testing YOLOv8 and Faster R-CNN for plant disease detection. The objectives spell out the importance of accuracy, speed, flexibility, and early detection in the context of plant disease detection, making clear the eventual aims of the study and how it hopes to contribute to the field.
- Literature Review: A review of previous studies on deep learning-related plant disease detection papers, as well as a review of literature on deep learning-based datasets, will be presented in this part on literature reviews (i.e., the PlantVillage dataset, etc.), and a review on the evolution of the YOLO deep learning algorithms (i.e., YOLOv1- YOLOv8, etc.) as well as the adoption of regional convolutional neural networks (i.e., Faster R-CNN, etc.). Finally, it positions the current work within the larger scientific detail, identifying the current study's gaps in knowledge.
- Methodology: The core of the report, this section details the systematic approach to the research, including data collection from PlantVillage, preprocessing techniques, model architectures (YOLOv8 and Faster R-CNN), training processes, and evaluation criteria. It provides a step-by-step explanation of the experimental design, supported by figures and tables, to ensure reproducibility and transparency.
- Findings: This section presents the results of the study, including performance metrics (e.g., precision, recall, IoU), confusion matrices, and visual comparisons (e.g., CAM, localization images). It offers a detailed analysis of how YOLOv8 and Faster R-CNN performed in detecting plant diseases, highlighting their strengths and limitations.
- Discussion: Building on the findings, this section discusses the implications of the results, comparing the trade-offs between YOLOv8's speed and Faster R-CNN's accuracy. It explores the significance of these outcomes for real-time applications and suggests areas for further investigation, such as grid size effects and memory mapping.
- Project Outcome: This section synthesizes the research achievements, summarizing the models' effectiveness, practical implications, and contributions to agriculture. It emphasizes the successful development of a diagnostic system and its potential to enhance crop protection and food security.
- Conclusions and Suggestions: The final section recaps the study's key insights, reiterating the transformative potential of deep learning in agriculture. It offers recommendations for future work, such as model optimization and integration with emerging technologies, to further advance the field.
- References: A comprehensive list of cited works, including studies, datasets, and technical resources, is provided to acknowledge prior contributions and support the research's credibility.

Chapter 2

Background

This section provides the base context for this research, explaining the role of plant disease detection in agriculture, presenting previous literature, addressing deficiencies with existing literature, and reiterating key findings for this research. It also provides context for the development and evaluation of machine learning models, namely YOLOv8 and quicker R-CNN, for real-time autonomous plant disease detection.

2.1 Introduction

Agriculture is a cornerstone of global economies and food security, providing sustenance and driving economic growth across nations. However, the sector faces a formidable challenge from plant diseases, which threaten crop yields, economic stability, and food availability. Factors such as climate change and global trade exacerbate the spread of these diseases, creating an urgent need for effective diagnostic solutions. Traditional methods, such as manual inspection of plant symptoms, are labor-intensive, require specialized expertise, and lack the scalability needed to address modern agricultural demands. Deep learning has emerged as an exciting opportunity to advance plant disease detection in recent years, as it can provide automated, accurate, and scalable solutions. Deep learning, machine learning, and computer vision are available to create models (i.e., YOLOv8 or Faster R-CNN) that will enhance early diagnosis or intervention to minimize crop loss and increase agricultural efficiency. This study explores the application of these models to detect diseases in key crops—apples, grapes, and potatoes—aiming to contribute to sustainable farming and global food security.

2.2 Literature Review

Global success in a modern, connected world relies on farming, as the act of farming and feeding a nation is a significant boost to the economy of a nation. However, plant disease is not preventing farming through the overriding risk it poses to global food security and economic strength. Coupled with global trade and climate change we now face additional challenges that dramatically exacerbate the already looming threat we face in delivering adequate, safe and nutritious food sources. New possibilities also associated with deep learning and agriculture research allow for identification and treatments of damaged plants. Our scientists are developing new methods to attempt to utilize machine learning and computer vision to protect food sources from agricultural produce and achieve a productive and sustainable farming practice world-wide. It is an investment into the future for the treatment of the harm that plant diseases can cause, and providing a reliable food supply and agriculture sector.

A wealth of information about plant diseases is available on PlantVillage. 34 crop species—apple, blueberry, cherry, maize, grape, orange, peach, bell pepper, potato, raspberry, soybean, squash, strawberry, and tomato—are represented in the 54,309 photos in the 2016 collection. Plant Village supplied the images of both healthy and damaged crop leaves (www.plantvillage.org). There are 17 fungal kingdoms, 4 bacterial, 2 mold (oomycetes), 2 viral, and 1 mite groupings. In the dataset it has 38 categories of plant diseases, and one category just for background images. This dataset started a crowdsourcing project to use CV (Computer Vision) algorithms to eliminate losses to viral crops. Technicians removed sick crop leaves from a plant, and laid them on grey or black backdrops and took photos in optimum lighting. All photographs were modified to eliminate as much background as possible and lay the leaves as a near upright position. Typically called lab photos. YOLO [2] is a neural network that leads advances in object/detection AI. With a precision score of 79.19 percent, YOLOv3 identified and examined six rice leaf diseases: rice ragged stunt virus disease, brown, narrow brown spot, bacterial leaf streak, blast, and bacterial leaf blight [3] [4]. The data set that the researchers performed their work on comprised 6,330 pictures that they found themselves. Vegetable disease detection through attention mechanisms had twice the accuracy when they were used. Qi et al. [5] recommended enhancing tomatoes and the small disease object they used in tomato viral disease prediction by using YOLOv5's attention squeeze and excitement module to enhance detection with the tomatoes, increasing the precision of the tomato viral disease model in their detection since they later imaged novel evaluation images saved by YOLOv5. Jing et al. [6] developed a tomato disease detection-by-using YOLOv5 and the channel-wise spatial attention mechanism CBAM [7] found that channel-wise and spatial attention were treated separately from each other. CBAM was probably not able to avoid substantial information loss between the two dimensions. Li et al. [8] used coordinate attention (CA) to produce the MTC-YOLOv5n with an 84.9% mAP on a cucumber disease dataset and in using CA simply increased the molecular detection of surface by mixing the channel-wise and spatial detection activation. Sun et al. [9] used a Chinese two vegetable sick photo dataset with all 10 sick photos and produced when their photo images had feature descriptions both visually for the plant and contextually using the CBAM of YOLOv5 handling the sample imbalance, which produced 88.0% mAP on Veg DenseCap detection model. Complex backgrounds images acquired and might be difficult to imaging however some background and plant related sites could change in the future using RCNN [10]. Deep neural networks have become common in their use of hyperspectral images for direct plant disease detection of chlorophyll absorption emissions within the images [11]. For instance, [12] described a CNN based study using hyperspectral imaging to predict automated subject plant viral disease. Specific to chloroplast infections of apple leaves were subsequently noted in [13]. [14] described the deep layers of CNNs for rice disease detection, while [15] provided an overall process system summary and [16] explained deep learning in plant stress phenotypic identification.

A Neural Network ensemble was used to help identify tea leaf diseases in [17]. A systematic geometric approach to defining rice field quality in this way [18] led systematic data collection. Fan et al. [19] developed a new batch normalization layer atop the convolutional layer and a central cost function using stochastic gradient descent in Faster R-CNN. In the study, nine complex maize leaf diseases were addressed. The change enhanced accuracy by 8.86% and image detection time decreased by 0.139s/image vs. a SSD method which improved detection accuracy by 4.25% and image detection time decreased by .018s. Fuentes et al. [20] mention nine tomato leaf disease and pest identifications by bounding box using Faster R-CNN, R-FCN, and SSD architecture and identify how CNN topologies impacted performance. They reported 160 millisecond picture detection time and 85.98% mAP with ResNet50 as feature extractor. A later study [21] improved Faster R-CNN with a single-class CNN producing an increase in mAP by 13%. Jiang et al. [22] created INAR-SSD by combining SSD, inception, and rainbow concatenation. The INAR-SSD network attained 78.8% mAP on a five apple leaf disease dataset and a frame detection speed of 23.13frames with a modified VGG16 feature extractor; outperforming Faster R-CNN and SSD. Li et al. [23] considered five bitter gourd leaf diseases by using an improved version of Faster R-CNN with a regional proposal frame added, an FPN built, and ResNet50. The new model achieved an average accuracy of 86.39% which was 7.54% better than the old model as well as better grey spotting at 16.56% positive predictive value and detection speed at 0.322 seconds per image.

Li et al. [24] used Faster R-CNN, FPN and precise ROI pooling in their model to detect apple leaf diseases with a complicated background with either real-time detection or recognizing small lesions. The improved model identified 5 apple leaf diseases with mAP of 82.28% which improved from Faster R-CNN 5.81%, YOLOv3 by 13.92%, and Mask R-CNN by 4.86%, and detection time of 43 milliseconds reducing latency per image. Ozguven and Adem [25] also improved Faster R-CNN by transforming the input layer to 600 x 600 pixels of resolution and autonomously capture and classify sugar beetroot leaves by identifying leaf spot disease into three severity classifications. The authors identified beetroot leaf spot diseases with 95.48% accuracy using the Faster R-CNN model which improved from 92.89%. The improved Faster R-CNN model captured a relatively small dataset with a comparable disease identification rate using 155 images. The photo collection of leaf samples also suffered from poor lighting conditions. The proposed method then identified the disease on 111 out of 117 sugar beet leaves with an accuracy of 95.48%. The objective was rapid image identification time.

Song et al. [27] presented YOLOv4, for detecting, classifying, and localizing pictures of citrus diseases. The repository contained images featuring Citrus Canker and Citrus Greening affected leaves. They added supplementing images through pixel manipulation and mosaic data augmentation to help introduce variety and new images. Their model was 95.4% accurate at performing inference at 30 FPS. Shill & Rahman [28] utilized photos from PlantDoc that represented 30 categories of plant images to compare YOLOv3, and YOLOv4 for detecting 17 diseases in 13 plant species. They validated that they were working with up to date YOLO, their model by scaling the images to 416 × 416, generating darknet labels, and performing several augmentations. YOLOv4 significantly outperformed v3 at 55.45%, and 53.08% mAP. They suggested using more data and quality data, more augmentations, and other techniques for labelling feature extraction (including saliency maps) with the YOLO model to improve accuracy. YOLOv5 created by Mathew & Mahesh [29] detects Bell Pepper Bacterial Spot Disease. The researchers trained based on the Plant Village dataset from Kaggle. The model was more accurate than R-FCN, SSD, R-CNN, and the previous YOLO versions. Innovative improvement with detection - specifically managing the 'small object problem' using scaling on the detected image, using colour space shifts, and using mosaic augmentation. The incredible benefit of the model also shows what could be achieved with a model with a weight file of 27 MB, that is around 90% lower than YOLOv4. Therefore YOLOv5, is better in accuracy than others, faster than the others and simpler to implement on hardware. They suggested adding new classes for additional bell pepper diseases to enhance the model. Zhang et al. [30] augmented GDM detection by integrating channel attention (CA) into YOLOv5. CA entailed much attention on pertinent visual features, in nature, to improve detection and efficiency in the field. The YOLOv5-CA model yielded a detection precision of 85.59%, recall of 83.70% and mAP@0.5 of 89.55% even in a changing weather and environment. Li et al. [31] augmented YOLOv5n using CA and Transformers architectures to diagnose cucumber plant diseases by providing global information to classify and diminish the complexity of the background objects. As a model size of 4.7 MB in size, its performance was 143 Frames Per Second (FPS) and 84.9% mAP. The principal training methods were varied scales, anchor clustering, and random resolution. Wang et al. [32] proposed a rapid YOLOv5 Detection model based on PlantVillage and a new peanut disease dataset that identified, and can categorize plant diseases. An attention submodule (IASM) architecture, Weighted Boxes Fusion (WBF), Ghostnet was used to decrease the model, BiFPN, and Rapid Normalisation Fusion increased precision and efficiency to provide the user with 93.73% precision, 92.94% sensitivity, and 92.97% F1 measure.

Table 2.1 Recent research works of plant diseases Detection

Reference	Object	DL frame	Dataset	Sample Size	Avg. Detection time	Metric
P.jian et al [22]	Apple	INAR-SSD, SSD	Plant Village , Self-acquired in field	2029-2029*12		MAP, Confusion Matrix, Speed
J.-H. Li et al [23]	Bitter gourd	Improved Faster R-CNN	Self-acquired in field	1204-10627	0.322s	MAP, Accuracy, Detection time, Training time
X.-R. Li et al [24]	Apple	Improved Faster R-CNN, Faster R-CNN	Plant Village , Self-acquired in field	2029-2029*10	0.43s	Avg Processing time, Avg accuracy
D. Li et al [33]	Rice	Faster R-CNN, YOLOv3	Self-acquired in field	5320	30FPS	Precision, Recall, F1 score,
W. Zhen et al [34]	Corn, Wheat, Cucumber	RD-net	Self-acquired in field	150-150*6	0.23s	Segmentation Accuracy, Recall, Single Image segmentation speed

2.2.1 Similar Applications

Today, deep learning is being used in agricultural settings for the trick of automated detection of plant diseases. The PlantVillage dataset is one such dataset, which contains over 54,000 images of 14 species of crops (including apples, grapes, and potatoes) spread over 38 disease categories. The dataset has been used to develop solutions that use computer vision-based approaches to diagnose diseases. One of the first applications of convolutional neural networks (CNN) explained how automated diagnostic prediction of viral disease was possible to accomplish with moderate levels of accuracy in controlled environments. Now, more sophisticated deep learning models (YOLO, and Faster R-CNN) are being adapted for real-time detection. For example, one study utilized YOLOv3 to detect six types of rice leaf diseases with 79.19% accuracy using 6,330 dataset images. There are also examples in the literature using Faster R-CNN to detect leaves diseases in maize plants. There were further improvements in Faster R-CNN with batch normalization that achieved improved accuracy over single shot detectors (SSD) with an extra 8.86% accuracy. All of these are examples that have shown how deep learning can be meaningful and is different from what is present in agriculture today in regard to the ability to identify diseases in a broad range of crops with better speed and accuracy.

2.2.2 Related Research

This body of research has furthered the field by innovating new architectures, attention mechanisms and datasets, specifically related to the detection of plant diseases. Attention mechanisms have been constructed using the squeeze-and-excitation module, as well as the coordinate attention (CA) mechanism, to name a few. Attention mechanisms, such as CA of objects, can increase accuracy in the minority class (i.e., detecting a small object of disease). Examples of major accuracy increases can be noted through Li et al.'s research when they proposed the MTC-YOLOv5n model which had CA, which achieved a precision mean average precision (mAP) of 84.9 in the cucumber disease dataset, and the Veg DenseCap model of detecting vegetable diseases that used CBAM attention to address sample imbalance and achieve a precision mean average precision (mAP) of 88.0. There has also been previous research that optimized model architecture for specific crops. For example, Jiang et al. proposed an altered SSD model with inception modules, called INARI-SSD, that achieved a precision mean average precision (mAP) of 78.8 for apple leaf diseases, and at the same time, also speed improvements over the baseline, Faster R-CNN (23.13 FPS). Other research has also found solutions to issues related to complex backgrounds using hyperspectral images and region based CNNs (R-CNN). For instance, Fuentes et al. employed Faster R-CNN with ResNet50 to evaluate tomato leaves diseases and averaged a mean average precision (mAP) index of 85.98%. This research showcases the variety of applications in the deep learning literature for plant disease detection and further justifies the foundation for this present study.

Table 2.2: Comparison of Plant Disease Detection Approaches

Features	Traditional Methods	YOLOv3	Faster R-CNN	YOLOv8	Proposed System (Fine tuned YOLOv8 and Faster R-CNN)
Real-time detection	No	Yes	No	Yes	Yes
High accuracy	No	Yes	Yes	Yes	Yes
Scalability for large fields	No	Yes	Yes	Yes	Yes
Interpretability (e.g., CAM)	No	No	Yes	Yes	Yes
Early disease detection	No	Yes	Yes	Yes	Yes
Handles complex backgrounds	No	Yes	Yes	Yes	Yes
Low inference time	Yes	Yes	No	Yes	Yes (via YOLOv8)
High localization precision	No	No	Yes	No	Yes (via Faster R-CNN)
Deployment on mobile/IoT	No	Yes	No	Yes	Yes
Handles diverse crops	Yes	Yes	Yes	Yes	Yes
Cost-effective for farmers	Yes	No	No	No	Yes (with optimization)
Automated diagnosis	No	Yes	Yes	Yes	Yes

2.3 Gap Analysis

Despite the considerable advances made in applying deep learning to plant disease identification, there are still gaps in the broader application of established methods. First, as indicated by the above overview, many studies reported on in this review are limited scope and conducted in laboratory conditions, datasets like PlantVillage, and many of the studies we looked at, used digital cameras for the best conditions by photographing plants under optimal experiments; often isolated leaves illuminated with a consistent light source. The generalizability of models to real-life conditions is limited by what happens beyond a controlled lab environment. For example, in a real world field environment you may have inconsistent light, backgrounds could become complex as other objects are present, and symptoms like leaf spots may involve overlapping symptoms. Second, models like YOLOv3 and Faster R-CNN represent an improvement in detection models, but they do involve tradeoffs in performance speed and accuracy that are not fully determined in relation to use in real-time in agriculture. For example, Faster R-CNN generally has better accuracy but a slower inference time, which should be considered. YOLO by comparison has made better performance in terms of inference speed, but accuracy will often fall short in difficult and complex scenes. Third, scalability for bigger farm operations, presents an opportunity for more research with respect to computing resources and developing answers to adopt the technology on mobile or IoT platforms for larger farm operations. Finally, while it isn't enough, there seem to be few substantial comparisons between the newest YOLOv8 versus the Faster R-CNN model with respect to plant disease detection, or divided crops in apples, grapes or potatoes. This research addresses these gaps by evaluating YOLOv8 and Faster R-CNN in real-time scenarios, using a balanced dataset, and focusing on practical deployment considerations to enhance early diagnosis and treatment efficacy.

2.4 Summary

The background section highlights the value of agriculture in relation to global food security, as well as the obstacles presented by plant diseases, and all those risks imposed by climate change and global trade. The literature review has discussed the introduction of deep learning for plant disease detection; of which, some related examples discussed automating the diagnosis process using deep learning techniques, and other related research has began to improve the architecture and methods of models. In the review, it was discussed that in terms of plant disease detection, there were still the gaps of translating to real-world applications, optimizing models, and comparisons - i.e. there are needs as outlined by the existing research. This study has answered these needs, through the development and evaluation of YOLOv8, and Faster R-CNN for real-time diagnosis of plant diseases; to close the above gaps and help improve sustainable agricultural practices. In addition to this introduction there will be methodology, results and discussion presented related to the study, and from this point, a structured comprehensive approach that will present how to progress to address the challenges mentioned earlier.

Chapter 3

Research Methodology

This section describes the formal process used to build and evaluate a real-time autonomous plant disease diagnosis system using deep learning models YOLOv8 and Faster R-CNN. It describes the requirements analysis, system design, methodology, project planning and organization, outlining the research process in its entirety. The methodology focuses on detecting disease in apples, grapes, and potatoes in which a dataset was used from PlantVillage to train and evaluate the models.

3.1 Methodology/Requirement Analysis & Design Specification

This subsection defines the requirements and design specifications for the proposed system, ensuring alignment with the research objectives of accurate and efficient plant disease detection.

3.1.1 Overview

The goal of this study is to address one key issue of a plant disease detection system which is the ability of an automated real-time diagnosis or detection. The system being proposed focuses on three relevant crops, apples, grapes, and potatoes, and their specific diseases, Apple Scab, Grape Black Rot, and Potato Early Blight, respectively. The process includes gathering and processing dataset, developing the deep learning models (YOLOv8 and Faster R-CNN), training and validating the deep learning models, and analyzing the performance of metrics (i.e., precision, recall, and inference time). The designed system runs on the NVIDIA Tesla T4 GPU to facilitate efficacious computation and real-time applications.

3.1.2 Proposed Methodology/ System Design

The proposed methodology consists of data preprocessing, model engineering and assessment of performance, in a Framework. The methodology designs from data acquisition which is from PlantVillage and moves through the preprocessing phase including, cleaning, resizing, annotating, augmenting and normalizing datasets. Object detection will utilize two deep learning architectures namely, YOLOv8 and Faster R-CNN, where each architecture specifies an object detection method for identifying and classifying plant diseases. YOLOv8 implements a single-stage, anchor-free method that optimizes speed only, whereas Faster R-CNN implements a two-stage region proposal network method that optimizes accuracy. In this design, both models will leverage a set of processed datasets to train, validate - using a separate subset and evaluate using accuracy metrics, which have some similarities to object detection metrics (Intersection over Union (IOU) and saliency score). In addition to being interpretable, scalability will also be achieved in the designs. Model interpretability is an important detail within the design as we will use a method called class activation mapping (CAM) in the future, to locate areas in the image that explain evidence of disease.

3.1.3 Functional and Nonfunctional Requirements

Functional Requirements:

- The system must accurately detect and classify plant diseases in apples, grapes, and potatoes, including specific diseases like Apple Black Rot, Grape Esca, and Potato Late Blight.
- It should provide real time inference capabilities and predictions with bounding boxes and class labels for disease areas.
- The system must support data preprocessing, including resizing images to 640x640 pixels, annotating with bounding boxes, and applying augmentation techniques (e.g., rotation, flipping).
- It should allow assessment using performance metrics of precision, recall, F1-score, IoU, and inference time.

Nonfunctional Requirements:

- The system must achieve high accuracy, targeting a precision of at least 85% and an IoU score above 0.5 for effective localization.
- It should operate efficiently, with YOLOv8 achieving inference times under 10 seconds per iteration for real-time use.
- The system must be scalable, capable of handling large datasets and diverse crop types without significant performance degradation.
- It should be user-friendly, with outputs (e.g., CAM visualizations) that are interpretable for agricultural practitioners.

3.1.4 Context Diagram

The context diagram illustrates the high-level interaction between the plant disease diagnostic system and external entities. The system receives input in the form of raw plant images from the PlantVillage dataset, accessed via Kaggle. These images are processed through the system, which includes data preprocessing, model training, and inference modules. The system interacts with users (e.g., researchers, farmers) by providing outputs such as disease classification results, bounding boxes around affected areas, and performance metrics. External dependencies include the NVIDIA Tesla T4 GPU for computation and the Roboflow tool for data annotation. The diagram highlights the system's role as a self-contained diagnostic tool, interfacing with data sources and users to deliver actionable insights.

3.1.5 Data Flow Diagram Level 1 (Data Preprocessing)

The level 1 data flow diagram (DFD) for data preprocessing illustrates how data flows through the data preprocessing steps. The first process involves using the image data originally collected from the PlantVillage dataset in its raw state for testing. This raw input then enters the Data Cleaning process to remove any possible noise images obtained from the PlantVillage dataset, which result in meaningless or irrelevant images. After going through Data Cleaning, the cleaned data can flow to data Resizing, which is to ensure all images are standardized to 640 x 640 pixel images. After the images are resized, the data can go into the data Annotation process with Roboflow, where the annotator can add bounding boxes and class abbreviations (ex. "a_s" for Apple Scab). The annotated data then flows through the Data augmentation process. In data augmentation, The data will be augmented by varying the data with techniques such as rotation (15 degrees), flipping, and color ranges. Once we have augmented the data set, data will flow into the Data Distribution process and will be split into a training (80%), validation (10%) and testing (10%) data set, and exported into a COCO friendly file format. Finally, the preprocessed dataset will then flow into the Data Normalization process, where we will normalize all pixel values to range between 0-1 values. The DFD outlines a streamlined pipeline for data preprocessing of the dataset for deep learning.

3.2 Detailed Methodology and Design

The methodology section of this project is vital for understanding the processes that were explored in real-time autonomous plant disease diagnostics. In this section, systematic procedures and methods used in the research are outlined to provide context for understanding the study. The research investigates the research concerning advanced object detection models, YOLOv8 and Faster R-CNN, which are important solutions to significant challenges of detecting plant diseases. The investigation also discussed multiple processes such as model building configuration, tuning hyper-parameters, and evaluation of performance. The methodology section outlines the processes of testing, and discusses why specific methodology was selected. This section also establishes a baseline for analysis and the exploration of what the research demonstrated. The research included using YOLOv8 and Faster R-CNN models in order to facilitate real-time object recognition and autonomous diagnosis. After careful planning of a data directory and the selection of model parameters that were utilized in the episodic training of the model and while adjusting dynamic hyper parameters, the COCO data set with fourteen (14) different classified objects were selected and utilized. The results from evaluating the trained YOLOv8 and Faster R-CNN were promising with a precision-confidence curve that generated a score, a precision-recall curve producing a mean average accuracy, The research demonstrated average mAP@0.50 and mAP@0.75 scores of 0.85 and 0.95, respectively. An T4 graphics of Nvidia tesla card running 12.1 CUDA version and 535.104.0.5 driver version for processing speed was used for all experiments. The methodology demonstrates the potential and efficiency of a CNN-based framework to identify plant diseases in real time from autonomous diagnosis use cases. This research uses Faster R-CNN models and YOLOv8 models to identify plant diseases. The model first identifies three plant types - potatoes, apples, and grapes. It then identifies healthy examples of each of the selected healthy plants species and the particular diseases available for each types of plant disease identified - Apple Scab, Apple Black Rot, Apple Cedar Apple Rust, Grape Black Rot, Grape Esca (Black Measles), Grape Leaf Blight, Potato Early Blight, Potato Late Blight. The dataset used for data analysis was PlantVillage Kaggle dataset. This narrows this dataset down to three plant species: potatoes, apples and grape. Preprocessing the chosen dataset will provide mats for each type of plant and disease. Next, choosing an architecture that is appropriate for the Faster R-CNN models and YOLOv8s then building its architecture. In order to get the most out of the models, this involves developing the connections, building out the layers, and adjusting the parameters. Now that the architecture is in place, you can train the models with the preprocessed dataset. The training of the models will instruct the algorithm how to recognize plants and their associated diseases after training on photos and annotations. Next, multiple datasets were tested and validated after training to properly evaluate the performance of the trained model. In order to determine the model could detect plants and diseases, the performance metrics recall, precision, and F1-score will be used. The approach of YOLOv8s and Faster R-CNN will look at the data, design the architecture, train the model, and quantify the performance of the model using detection of plant disease. The whole process is demonstrated in action in Figure 3.1.

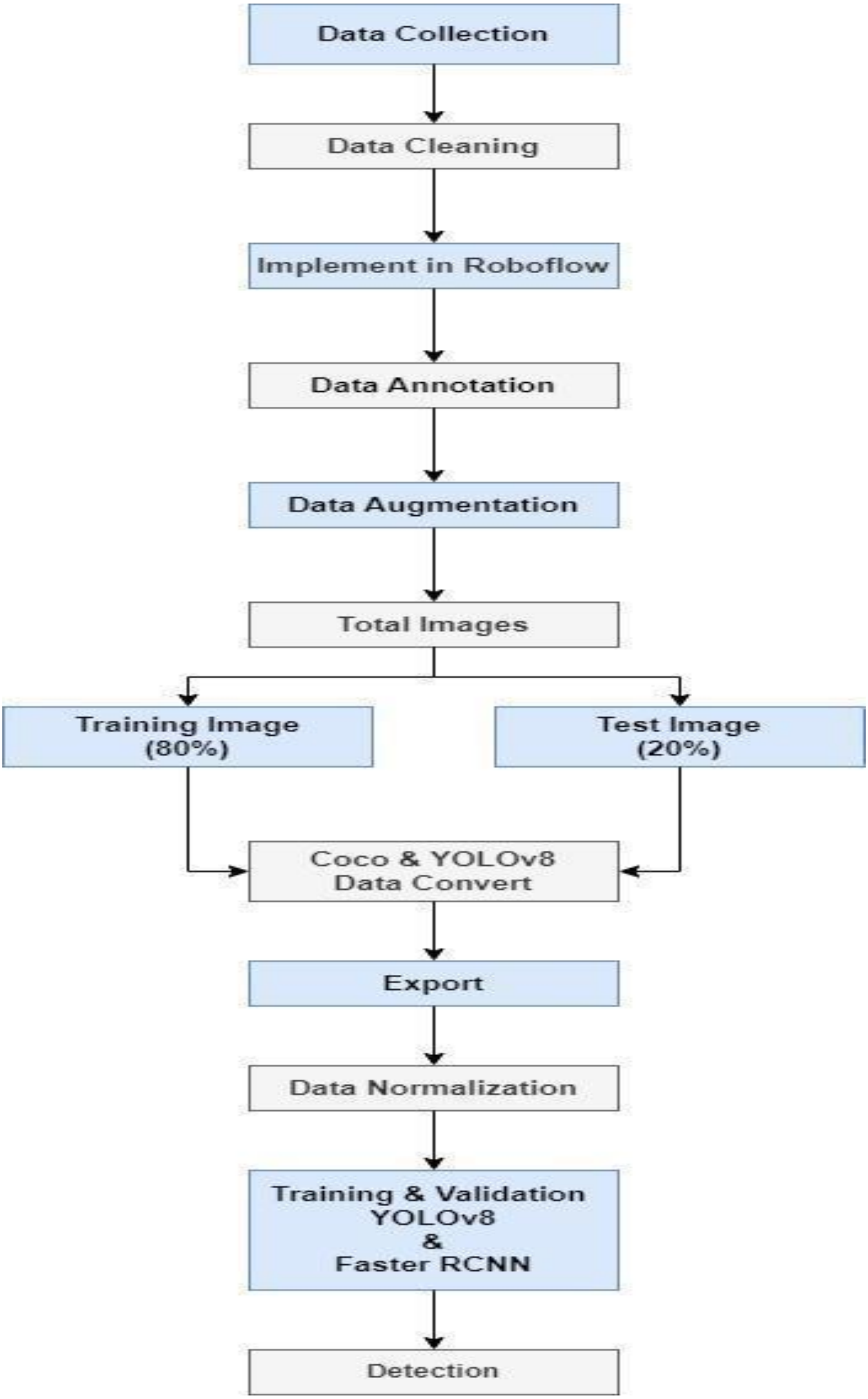

Figure 3.1 Figure Proposed System Diagram

3.2.1 Data Description

3.2.1.1 Dataset Collection

The dataset, obtained from PlantVillage and accessible on Kaggle, tackles the pressing need to improve food production in order to sustain a rapidly expanding world population. Given projections indicating a 70% surge in food production needed by 2050 and substantial crop losses of up to 100% in some areas caused by infectious illnesses, there is an urgent need for inventive remedies. The extensive proliferation of smartphones, projected to reach a total of 5 billion by the year 2020, offers a distinctive chance to use technology for the progress of agriculture. To address this difficulty, a thorough dataset consisting of more than 50,000 carefully selected photos that depict both healthy and sick leaves of different agricultural plants has been compiled. These photos are carefully selected and organized to guarantee accuracy and pertinence. PlantVillage facilitates the collective effort of using machine learning and crowdsourcing for mobile disease diagnoses by providing access to this dataset. This undertaking signifies the start of a continuous effort to use computer vision methods in tackling the crucial problem of yield losses in agricultural plants caused by viral illnesses. Granting academics and developers access to this extensive information may accelerate progress in agricultural technology, possibly revolutionizing farming methods and reducing the effect of plant diseases on global food security. The research focuses predominantly on three essential plants: apple, grape, and potato. From the aforementioned three plant categories, a subset of eleven classes has been chosen for comprehensive analysis, as illustrated in the give figure 3.2 below.

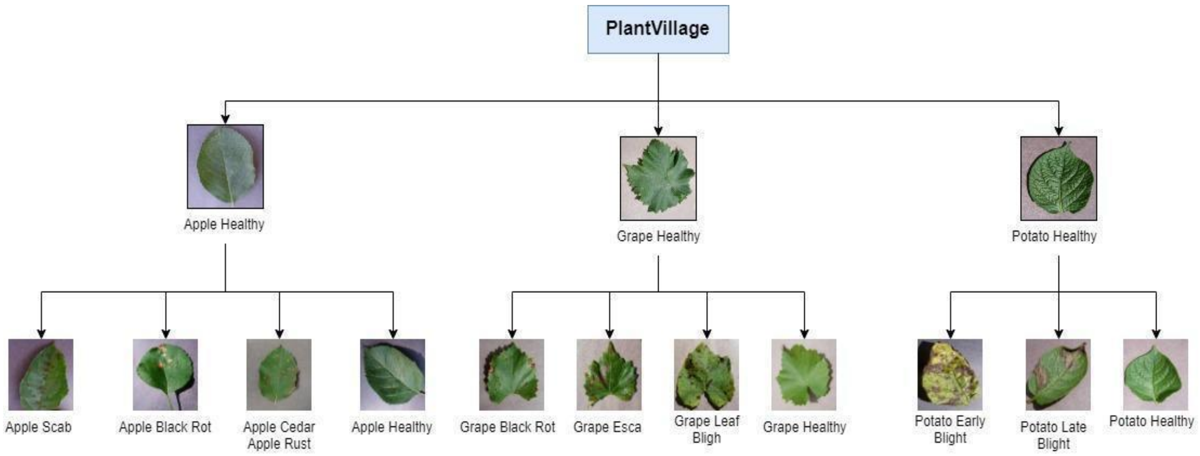

Figure 3.2 Data Collection and Selection

3.2.1.2 Statistical Analysis

The dataset analyzed during this study has 38 instances across 11 classes, which were composed of a variety of plants. The analysis targeted three plants of interest, apple, grape, and potato. For the three plants (categories) included in this study a set of 11 classes, illustrated in figure 1.2, were selected to investigate further. The statistical review indicated that the distribution of photos, in terms of frequency, was a reasonably diverse quantity of photos that contained different diseases and healthy fruit classes. In the case of the apple class, there were four classes from which: Apple Scab, Apple Black Rot, Apple Cedar, Apple Rust, and Apple Healthy. No distribution of formal images was structured from the classes, there were more images in "Apple Healthy" class and less in the classes that were village specific. In the grape class four classes were chosen: Grape Black Rot, Grape Esca (Black Measles), Grape Leaf Blight (Isariopsis Leaf Spot), and Grape Healthy. Like the apple class, the distribution of images varies with several classes, containing the most images in 'Grape Esca (Black Measles)". The potato group is comprised of three selected classes: Potato Early Blight, Potato Late Blight, and Potato Healthy. Each group includes the typical diseases that affect potato plants, as well as a class for potato leaves that are healthy. The number of images in each class shows relative importance and frequency of certain diseases that potato producers must deal with. This distinct group of 11 categories is a focused and illustrative assortment of plant diseases affecting apple, grape, and potato plants. By examining the distribution of images within each (if applicable) a good sense can be gleaned regarding the frequency and type of plant diseases in agricultural production. The sizes of the datasets for each plant species was apple (3171 pictures), grape (4063 images), and potato (2852 images). These datasets together form a complete dataset for the purposes of developing and testing a machine learning model to detect diseases. Table 3.1 outlines datasets dimensions.

Table 3.1 Amount of Instance per class

Plant	Class Name (Full Form)	Number of Images (Before Data Cleaning)	Number of Images (After Data Cleaning and Selection)	Class Name (Short From)
Apple	Apple Scab	630	100	a_s
	Apple Black Rot	621	100	a_br
	Apple Cedar Apple Rust	275	100	A_cr
	Apple Healthy	1645	100	apple_healthy
Grape	Grape Black Rot	1180	100	g_br
	Grape Esca (Black Measles)	1384	100	G_es
	Grape Leaf Blight (Isariopsis Leaf Spot)	1076	100	g_lb
	Grape Healthy	423	100	grape_healthy
Potato	Potato Early Blight	1000	100	p_eb
	Potato Late Blight	1000	100	p_lb
	Potato Healthy	152	115	Potato_healthy

3.2.1.3 Data Preprocessing

The first stage of preparing the data for deep learning is data preprocessing. Data preprocessing is when we clean, filter, and curate the raw data for further analysis or model training. It is vital to prioritize the quality of the data to ensure the models will be reliable and accurate since it could greatly impair the effectiveness of the model. In short, the goal of data preprocessing is to improve the accuracy and relevance of the results by preparing the data for analysis of algorithms to be employed in a machine learning context. Your read-in data should look something like what is shown in Figure 3.3.

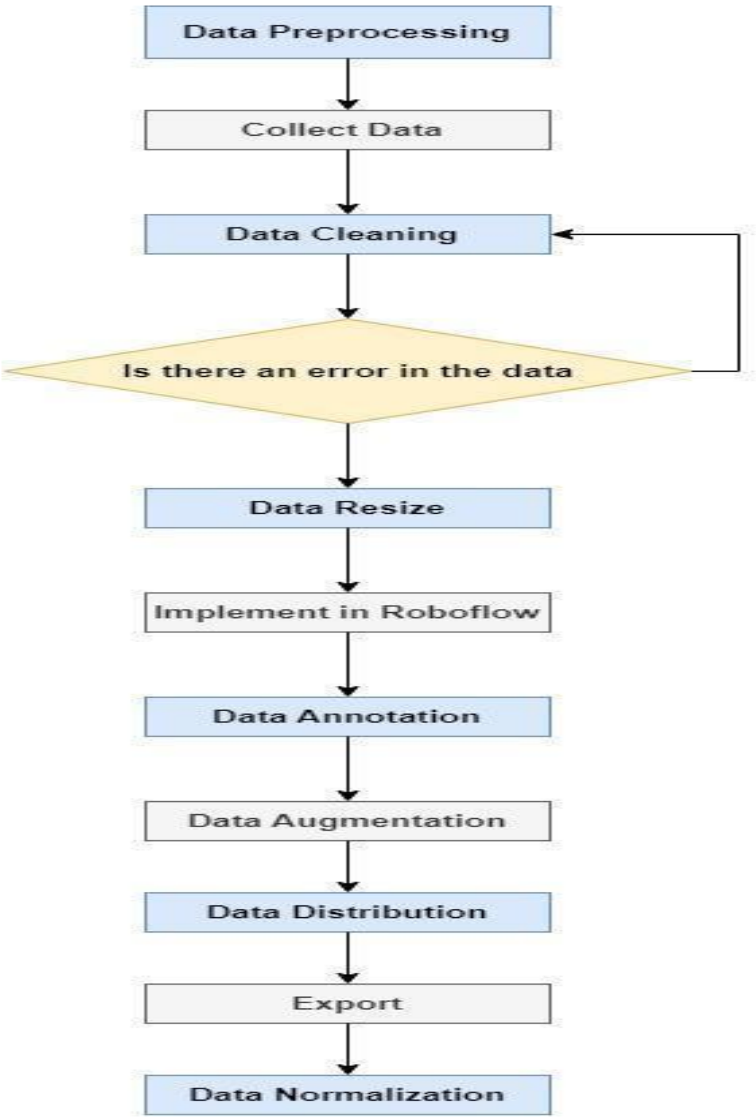

Figure 3.3 Figure Proposed System Diagram

3.2.1.4 Data Cleaning

Data cleaning is the act of removing irrelevant, corrupted, or inconsistent data from a dataset, in order to increase the accuracy and effectiveness of the learning engine's training on that data. The intent of this practice is to eliminate data which can get in the way of the model's ability to find patterns in the data, and ultimately make accurate predictions. By removing irrelevant or disruptive data, the data becomes more targeted and useful in training models that are able to generalize well to new data. In essence, you ensure the model learns from better, more appropriate examples and will perform better and produce better results.

3.2.1.5 Data Resizing

In order to ensure consistent input dimensions, all photos must be resized to a predetermined size, like 640x640 pixels, which works well for both models.

3.2.1.6 Data Annotation

The focus of the thesis is on further explicating the value of proper annotation of the data, specifically related to improving the usability of the dataset for machine learning applications; namely, the identification of plant disease. The images collected as part of our dataset were annotated using the robust online tool Roboflow. Roboflow enabled the annotation of sick regions of plant leaves by enclosing the sick areas in bounding boxes. The role of annotation is an essential aspect to effectively properly trained machine learning models that are capable of accurately detecting and classifying diseases in plants by utilizing the visual properties of the images. Since the dataset included bounding box annotations, it has added an additional layer of meaningful information that is necessary for training and evaluation of detection algorithms. The extensive dataset had documented and structured significations through the use of the bounding box annotations and it is an important step towards an integrative dataset that is needed for innovative agricultural technological research and development. The dataset was brought into Roboflow, which is a third party web-based tool, in order to effectively annotate the dataset. Each class in the dataset was organized with a short and repeated abbreviation, which could help hasten the annotation process. For instance, the disease, "Apple Black Rot" was abbreviated 'a_br', "Apple Cedar Apple Rust" became 'a_cr', and "Apple Scab" contracted to 'a_s'. Then, there is the grape category. The disease "Grape Black Rot" is documented as 'g_br', "Grape Escap (Black Measles)" is denoted 'g_es', and "Grape Leaf Blight (Isariopsis Leaf Spot)" is abbreviated 'g_lb'. The potato category has adopted abbreviations for select diseases. "Potato Early Blight" is abbreviated as 'p_eb', "Potato Late Blight" as 'p_lb', and "Potato" is defined as 'potato'. Subsequently, the photos were tagged with their relevant class names, with bounding boxes indicating and encapsulating the area of sick areas within plant leaves. First, the value of having such a standardized method of annotation is increased because of the defined values required for training and testing machine learning models developed for identifying plant diseases. Second, the steps we produced to outline a systematic process provide an effective framework for developing a robust detection algorithm for accurately identifying and classifying plant diseases, using visual cues.

3.2.1.7 Data Augmentation

Data augmentation generates more observations for the dataset by transforming the images without altering the semantic meaning. This includes many variations of the images including random rotation, mirror reflections, translations, magnifications, and color-contrast. The data augmentation allows the model to be more robust to variations in lighting, perspective, and background material. These transformations also help to reduce overfitting and improve intended generalization of the model, as well as exposing the model to wider samples of the variables that are a part of real-world data.

3.2.2 Data Distribution

Once more, the percentages can be somewhat arbitrary in reality. The collection of data is split into three groups: training, testing, and validation. Training accounts for 80% of the dataset, testing for 10%, and validation for 10%. Specifically, the training data covers 880 images or 80% of the dataset. The testing set is comprised of 110 images, or 10% of the dataset, and the validation set is also 110 images, or 10% of the dataset. It was critical to split the dataset this was to ensure the model had been trained on enough data, while also providing separate data to evaluate performance and generalization. The split is also to ensure the YOLO v8s and faster-r-CNN models were accurate with new data after training on various other datasets. The last dataset is exported from Roboflow as COCO format.

3.2.3 Data Normalization

One of the most frequent preprocessing techniques used in machine learning is data normalization, which is done to make sure the input features have similar scales. This is done to avoid concerns raised by certain features pulling more important information from features simply by being on a larger scale. In your thesis, data normalization is going to take the pixel values of images and convert it to an appropriate range for the model to consume. Pixel values in the images will usually range from 0 - 255 because of that certain level of intensity from color channels and having the option of red, green, or blue channels. To normalize the data, the pixel values will be scaled to be equal to or less than 1. The pixel values could also be modified to have a mean of 0 and standard deviation of 1. Normalizing the data means the optimization technique can converge faster during training and allow for the model to be more robust to future differences in input data. This normalization would usually happen to each individual image before the respective image is inputted into a model.

3.2.3 Model Description

YOLO V8s and Fast R-CNN are now integrated into autonomous systems for plant disease detection, which enhances the system's ability to recognize and detect an array of diseases quickly and accurately, in many cases, in real time [35][36]. identify and detect an array of different types of diseases. The speed of detection and recognition typically provides the systems with efficiencies to identify and detect and possibly classify different types of diseases while presenting greater challenges [37][38][39][40]. In fact, many of these same capabilities have been utilized in activity recognition, sports video, surveillance video with video sequences, and human-computer interaction [41][42]. YOLO, in particular, has found use in agriculture to further precision farming thereby providing crop, pest, and disease recognition and classification [43][44]. Many adaptations of these systems have also been developed for biometric medical, security, and specifically facial recognition applications, especially face detection [45, 46]. YOLO algorithms have also been utilized in and researched in the health industry, examples being, pill identification, skin region segmentation, cancer detection, where it assists the diagnosis of conditions and potentially improves treatment accuracy [47][48][49][50]. By recognizing and categorizing items in satellite and aerial photos, Additionally, YOLO has been employed as a tool to support environmental monitoring in remote sensing applications, urban planning, and land use mapping. [51-53]. The system with proper security have utilized models named YOLO for detecting suspicious activities while adhering to social distance rules, face mask detection, and image based monitoring [54][55]. Similarly surface inspection models for manufacturing and production uses YOLO based architecture show great benefit by detecting flaws and abnormality in quality control [56-58]. YOLO models also implement functions in overlays of observable traffic situations using existing applications for license plate and traffic sign identification to enhance intelligent transportation systems and traffic management [59][60]. In biodiversity conservation, resource extraction, and ecosystem management, YOLO models apply for animal identification and monitoring [61]. YOLO methods are prevalent in robotics and drone based activities for object detection aspects [62][63][64][65]. A bibliometric network visualization is provided in figure 3.4 and references papers pulled from Scopus with "YOLO" and "object detection" which is filtered. The records were provided with a labelled dimensions consisting of 6 sections for application analysis. [66].

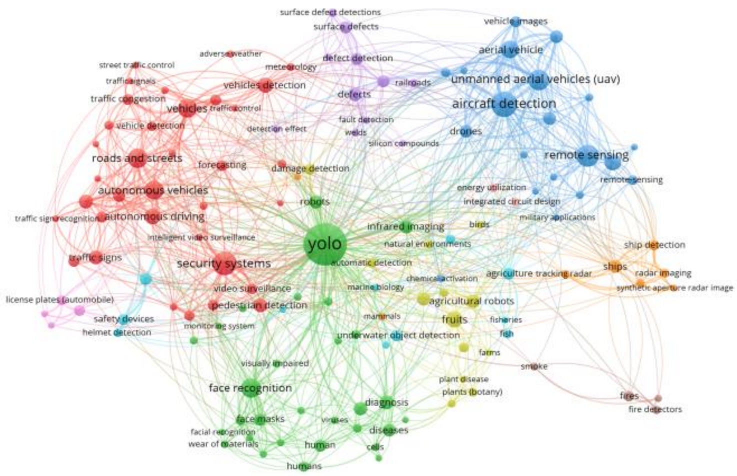

Figure 3.4 Bibliometric network visualization of the main YOLO Application

YOLOv7 can struggle to recognize small objects that are partially obscured, as well as objects recognized over other objects in crowded or complex areas. While older versions of YOLO have made some progress toward reliably recognizing agricultural diseases, limitations of typical models have not been alleviated. For example, a person can have difficulty discerning through complex environments or small objects, reducing their amount of detail perceived accurately. We added YOLOv8 to the crop disease detection system in order to overcome these restrictions. By offering a larger product than its predecessor, this design politely addresses any issues. YOLOv8 product improves many operational aspects of standard YOLO use, such as accuracy, detection speed, and efficiency, and uses anchor-free detection with Mix-up data augmentation to make the best use of limited training data and maximize its generalization capabilities overall. These advances demonstrate a reasonable approach to the challenges of agricultural disease detection where YOLOv8 was successful in quickly detecting crop diseases at scale in agriculture. Faster R-CNN which combines additional data with improved training has come to the forefront with an ROI pooling network lower testing and training times to roughly 80% of CNNs [67]. The Faster R-CNN network is the most advanced of the previous networks whose inference or prediction may take less than a couple of seconds following training. YOLO magazine launch [68] by Joseph Redmon and cohorts at CVPR 2016 demonstrated another method for "real-time" object detection, through the means of a single network applying to all. You Only Look Once (YOLO) which has massively improved upon, was a single run of the detection method that discouraged approaches of the past that represented multi-run classifiers or sophisticated multi-stage methodology. Recognizing YOLOv8 as a real-time object detection model in autonomous diagnosis, and being a special, indirectly related subject relating to My Investigational Design plan mentioned before, I realized I could diversify within the YOLO data model that would comply with the project objectives. I knew YOLO models could easily be embedded into systems due to a well-known strength they have to "detect" objects in real-time which is a pertinent aspect for "perceiving" objects in "dynamic" diagnostic situations. Accordingly, YOLO implements a single-stage detection technique to achieve maximum accuracy, speed, and robustness. In other words, allowing or supporting autonomous cars to make "real-time" decisions and minimizing the time the computer is engaging in the calculation; processing, where this time is insignificant or irrelevant in relation to the potential liability in terms of the risk taken from the detection with the computer's methods. Within YOLO designs, there are characteristics that assist, like going through full training and can take in multi-tasking across a full range of different object classes, therefore, are more relevant or "informed" with complex road environments.

3.2.3.1 Architecture

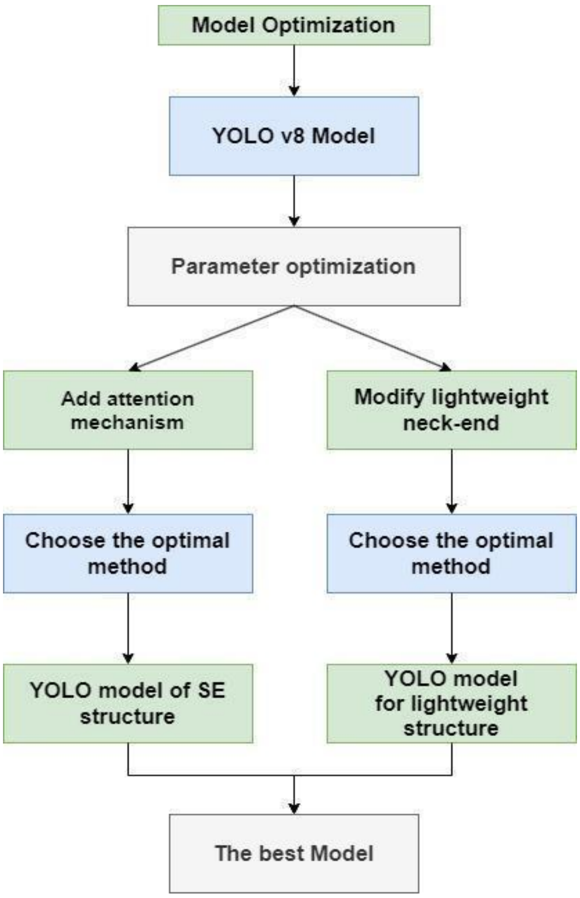

Figure 3.5 Figure Proposed System Diagram of Model Preparation

In 2015 the paper "You Only Look Once: Unified, Real-Time Object Detection," showed a simple process of making a better known object recognition system with YOLOv1, produced by Redmon et al. [69] [70]. YOLOv1 was capable of taking in images and returning processed images to the user at a rate of 45 frames per second. A variant of YOLO called fast YOLO could process images 155 frames per second. YOLOv1 was not the sole object detection platform available at the time, yet its YOLO framework consistently produced a very high mean average accuracy. The suggested model preparation system is shown in Figure 3.5. The basic idea underlying YOLO is the combination of object detection and regression, in one evaluation. YOLOv8 also has one neural network that predicts the object's bounding boxes and their relative probabilities of belonging to a given class from a single evaluation. The first evaluation of the YOLO model predicts B bounding boxes, indicates a confidence score from 0 to 1, and provides C class probabilities for the input image that return an internal component in the appropriate coordinates, i, j. The output should return a tensor of S x S x (B x 5 + C) form factor.

3.2.3.1.1 YoloV8 Overview

Ultralytics, the creators of YOLOv5, introduced YOLOv8 in January 2023[49]. There are five variations of YOLOv8: YOLOv8n in nanoscale, YOLOv8s in small scale, YOLOv8m in medium scale, YOLOv8l in large scale, and YOLOv8x in extra-large scale. YOLOv8 enhances the fundamental object-detection functionality of YOLOv5 and provides tracking and recognition, segmentations, postures, and object classification. In Figure 3.6, the full architecture of YOLOv8 is illustrated. The overall structure of YOLOv8 mirrors the structure of YOLOv5. YOLOv8 has modified the CSPLayer in YOLOv5 which is now called the C2f module. The C2f module allows contextualization of the high-level features for improved performance accuracy. The C2f module is a cross-stage partial bottleneck that consists of two convolutions. YOLOv8 is the most up-to-date form of the YOOL object detection model. The architecture of YOLOv8 adopts a relatively consistent form by combining the Pan Aggregation Network (PAN) with the Feature Pyramid Network. YOLOv8 is a supervised model, a new training label was created to facilitate the process of annotating and labelling. The annotating software for images will add other useful options such as user assigned hotkeys, shortcut labelling and automatic labelling, in order to annotate images for the purposes of model training. The Feature Pyramid Network (FPN) is a method that intentionally lowers the spatial resolution of the input image to create feature maps which can differentiate objects of different sizes and resolutions with more feature channels. The PAN uses skip connections which just as with the FPN, preserve the values from multiple layers instead of just taking the last layer of the network. The FPN or PAN must sample from many scales and spatial resolutions which can only occur through implementation so that targets could be classified by several parameters including position size and shape.

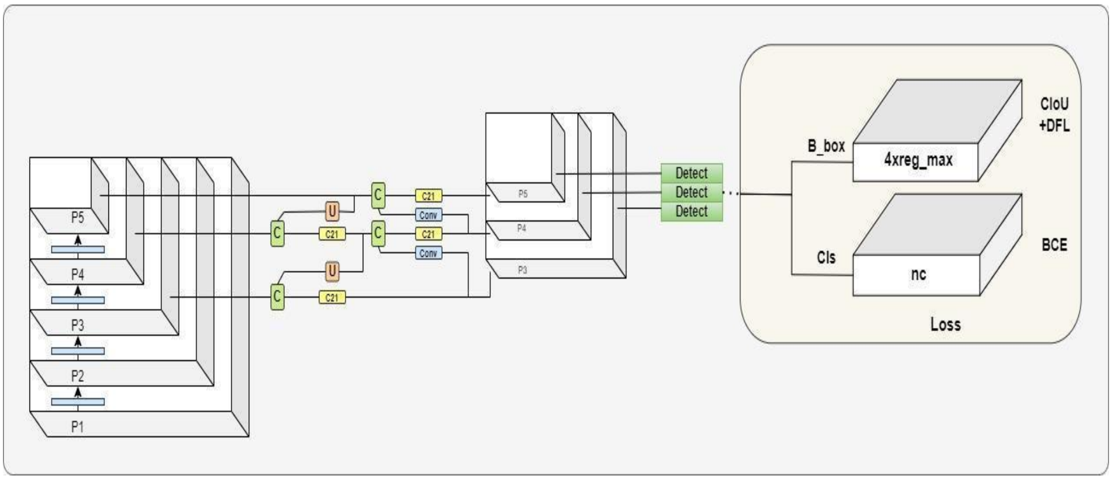

Figure 3.6 YOLOv8 architecture

An anchor-free model with a decoupled head for objectness, classification, and regression that functions independently of one another has been created by YOLOv8s. This is an anchor-free design [72] that is more accurate than YOLOv8 previous designs due to the each branch's focus on their tasks. YOLOv8 employs the sigmoid function in the output layer to establish the objectness score, which determines the proportion that the bounding box is accepted and contains an object. The class probabilities—which detail the likelihood for an object from each class—are generated with a SoftMax function. The classification loss is based on binary cross-entropy, while YOLOv8 uses the CIoU [73] and DFL [74] loss functions for real values of the bounding-box loss enables improved object recognition, mainly for tiny objects. YOLOv8 does semantic segmentation, known as YOLOv8-Seg, using the standard The feature extractor is CSPDarknet53, and the YOLO neck is replaced by a C2f module. Two segmentation heads then anticipate the input photographs' semantic segmentation masks. Five detection heads and a prediction layer are included in YOLOv8-Seg, same like in YOLOv8. The detection heads function exactly like that of YOLOv8. The consistent focus on performance and speed has resulted in YOLOv8-Seg achieving state-of-the-art (SOTAT) performance across a wide number of object detection and segmentation benchmarks. YOLOv8 has the ability to integrate with many labelling, training and deployment activities and can either be executed via the command line interface (CLI) or installed via PIP package. Test-dev 2017 on the MS COCO dataset showed that YOLOv8x obtained an average precision (AP) of 53.9% with a 640-pixel image size, while YOLOv5 only managed 50.7% with the same input image size. When TensorRT is enabled, YOLOv8x operates at 280 frames per second (FPS) on an NVIDIA A100.

3.2.3.1.2 Faster-R-CNN

An optimized version of the basic R-CNN was developed to meet the demand for better computational efficiency, better performance and less time for inference. An example diagram of a Faster RCNN with RPN (Region Proposal Network) is shown in Figure 3.7 [75].

The Faster R-CNN networks include:

i. A method of region proposal is used to generate "bounding boxes" for the image.

ii. A Computational neural network (CNN) is employed to extract the required information from the image.
iii. A classification layer is employed to predict the class of an item.
iv. A regression layer is used to accurately calculate the coordinates of the bounding box.

The region proposal method was a crucial part of the Fast R-CNN model. The biggest part of both R-CNN and Fast R-CNN models is the region proposal algorithms that run on the CPU. The fast R-CNN network [76], used a convolution network called RPM [73] to improve the proposal algorithm to 10ms/image. It also allowed different stages to share layers which improved the feature representation.

The Region Proposal Network (RPN)

i. The original input image is passed through the backbone convolutional neural network, where the region proposal network begins.
ii. The output features, like the previously uploaded input features, are of reduced resolution due to the base network's stride. A stride of 16 is used in both the VGG and ZF-Net backbone networks, as cited in [77][78].
iii. The network is trained to recognize if an object is present, at the correct location of the input image, and to produce a prediction of the size of the object. This is done by proposing 'anchors,' or bounding boxes, on the original image by the larger network. The model examines every individual pixel, predetermines the anchor's coordinates, improves the anchors coordinates based on the image and outputs item predictions.

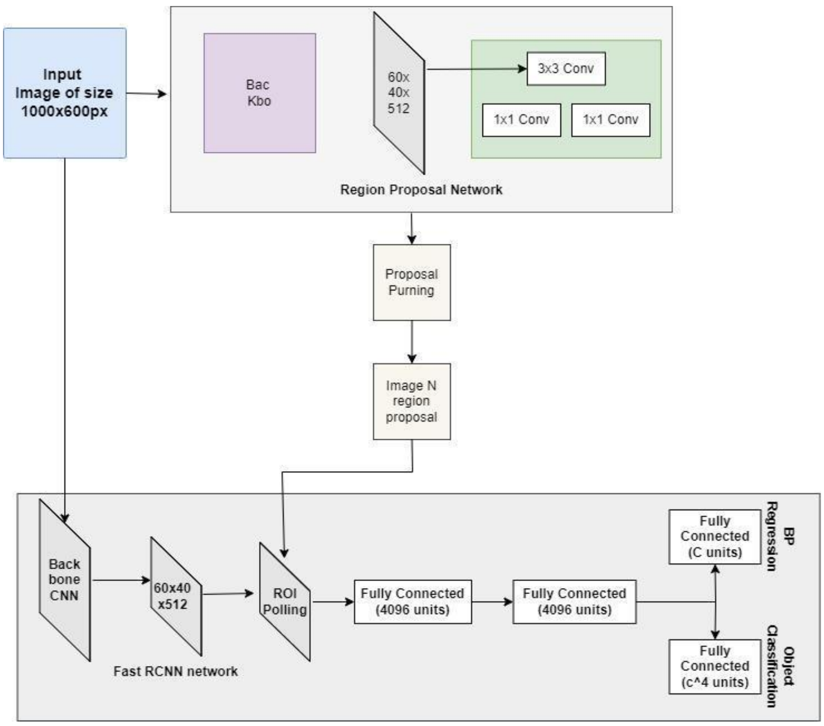

Figure 3.7 Faster-R-CNN Architecture

The combination is comprised of a Region Proposal Network (RPN) and a detection network (Fast R-CNN). A feature map is extracted that size is 60 x 40 by passing the input image through the CNN backbone, with 512 units. The ROI pooling layer will map pooled features from the backbone feature map using the bounding box proposals they have. The ROI operation can be seen as follows: it proposes a subregion and previously sub-divided the land into a desired amount of subdivisions. Each of the windows goes through max pooling to output an equal sized output. In the pooling layer, the region proposal process outputs an original output of size (N, 7, 7, 512). The features are fed through some fully connected layers and from there they enter into the subclassification and regression branches. The features are fed through a classification layer that has C units for each recognized class. The output from the classification layer is taken as inputs into a softmax layer to obtain scores. The regression parameters are used to aid in the resizing of the perceived bounding boxes. Each class once in the regression layer is assigned to a regressor, where each regressor has 4 parameters.

3.2.4 Gridding Size Effect on Detection Result

The input photographs are divided into 7 × 7 grid sizes by the YOLO model. It forecasts three bounding boxes and one class score for every grid. In regards to other objects, the YOLO model does not predict which object has the highest score, nor can it predict the class of two different objects in that same grid. The grid size assigns a maximum class number, and it predicts accuracy. By training the models with different grid sizes, we can note how the grid size influences our detection accuracy and recall. The grid size will greatly affect the last layer output dimensions thus being aware of this is essential for the model compilation. To enable comparisons, we will train four models in Figure 3.8, altering the output sizes to 7 x 7, 9, x 9, 11 x 11, and 14 x 14. We will then examine the recall and accuracy rates of each model.

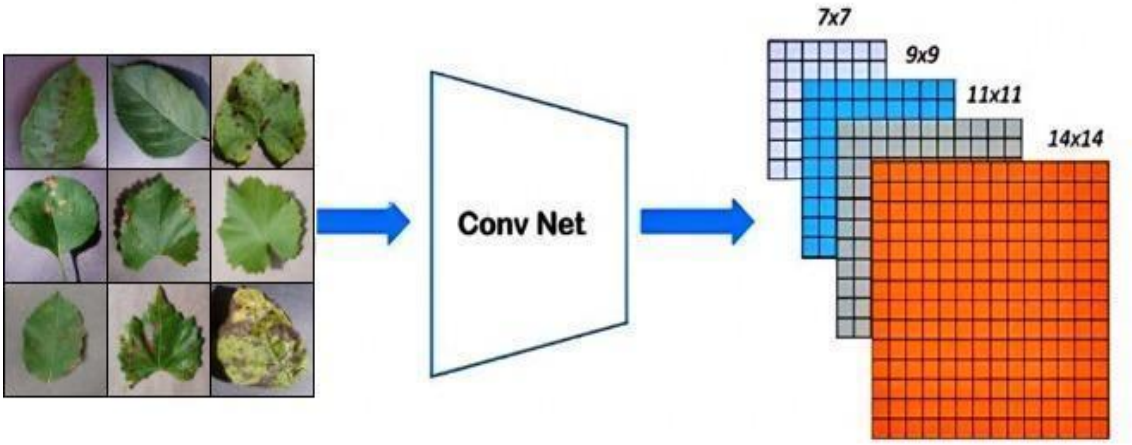

Figure 3.8 Network with Different Gridding Size

3.2.5 Pooling

Convolution is followed by the implementation of Merging Pooling. The pooling process that will further compress the data is called merging pooling. CNNs can condense feature maps into feature maps by pooling. The two types of pooling processes are max-pooling and average-pooling. Whereas average-pooling determines the average value, max-pooling takes the maximum value from inside the pooling zones. Figure 3.9 illustrates max-pooling with a 2 x 2 pooling filter, where the features' sizes decreased from 24 by 24 to 12 by 12. The most common kind of pooling strategy used for Convolutional Neural Networks (CNNs) is max-pooling.

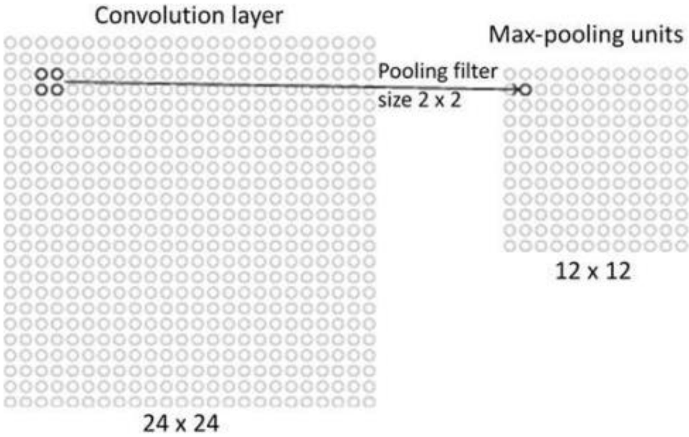

Figure 3.9 Max-pooling. Pooling from 24 x 24 to 12 x 12.

3.2.6 Memory Map

YOLO is the quickest detector for autonomous diagnostic and evaluation systems at over thirty frames per second. Although, it is purely sequential on photos, without any important temporary inter-frame data. In the thesis, we have proposed a memory mapping to encode and represent time-varying inter-frame data, see Figure 3.10. The last M-1 frames can be retained in a memory map to capture M frames, so our method will have results from the previous M frames to predict frame M.

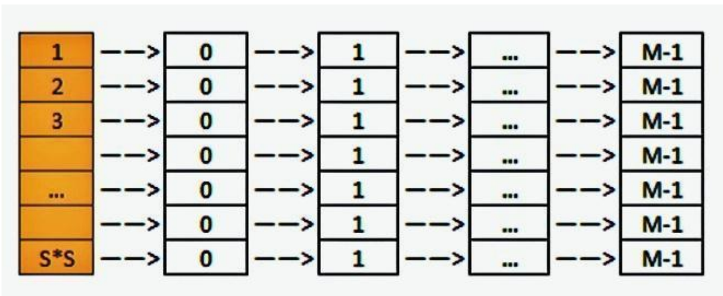

Figure 3.10 Memory Map

3.2.7 Evaluation Criteria

The results of object identification are determined by examining three main issues: global precision, global recall, and orientation accuracy. Precision and recall metrics can be used to evaluate how well the object detection process determined the correct objects. Orientation accuracy is applicable only to evaluation results associated with estimating orientation. Figure 3.11 displayed overall accuracy and overall recall in perpective. Compared to other contemporary methods they accounted for only overall accuracy.

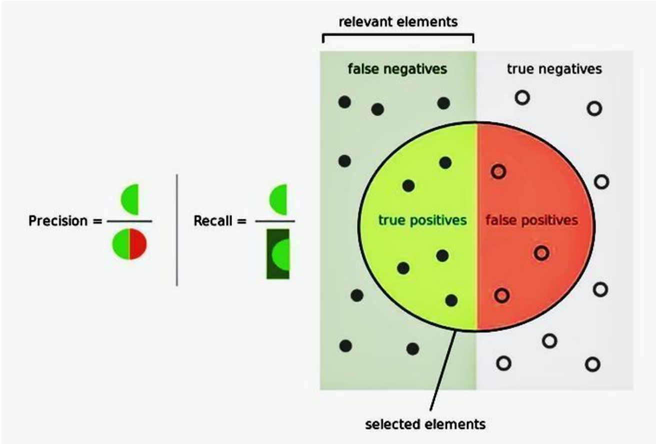

Figure 3.11 Precision and Recall

Accuracy and recall can be measured using Equation 3.1. Accuracy is the proportion of projected things that are relevant, whereas recall measures the expected number of relevant items.

$$Precision = \frac{tp}{tp+fp} \quad recall \quad \ldots\ldots\ldots\ldots\ldots\ldots 3.1$$

Equation 3.2 establishes the measure of how accurately an object is oriented - we use this to verify the accuracy of the orientation of the object. The αtruth number is the actual or true orientation, while the αpredict value is the orientation that was projected. We confirm that the orientation results are correct by confirming that they are the correct detections, combining the detections with at least a 50% Intersection over Union (IOU), and with the proper classification.

$$orientation\ accuracy = 1 - \frac{1}{n}\sum_{1}^{n}\frac{\|\alpha_{predict} - \alpha_{truth}\|}{\tau} \quad \ldots\ldots\ldots\ldots 3.2$$

And each test image's element count is denoted by n. is the ground-truth orientation value and shows the expected orientation value (in radians).

From the previously viewed pictures, the bounding box extraction, which is bounding boxes of objects, is executed using YOLO, which is the best possible bounding box. The real-time detection system YOLO is a fast and approximate way to detect objects, but is best used with active systems. A bounding box generally consists of six parameters, denotes the probability that the object is a member of the category Pc, Bx, By, Bw, Bh, and Cn. reflects the probability that the object is in the category Bx, By, Bw, Bh, and Cn. refers to the centre x, centre y, width and height, and class number. From the intersection over union (IoU) between the predicted bounding box and the ground truth bounding box, for every bounding box, a confidence prediction is defined by Equation 3.3. One way to compute the intersection over union (IoU) is to use:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \qquad 3.3$$

3.2.8 Loss Functions

The loss function for ANN is a differentiable function that calculates the difference of the predicted output values and the actual values. The output is a real non-negative number: the smaller the output, the better the accuracy of the network. If we also think of the output as a representation of a landscape in the multi-dimensional space of adjustable parameters, then it can be a landscape filled with hills. Each trainable parameter's gradient vector shows the proper path to pursue in order to obtain a minimum, which denotes that the training set's average loss function output is low.

3.2.9 Overfitting

In an ANN project, the ultimate goal is to bring the ANN to a state where it can generalize well outside of the training dataset used to train it; the ANN should be able to predict from the data true underlying distribution. If your training dataset is small and your ANN has many parameters to learn, we risk the possibility that the ANN may memorize the entire training dataset (overfitting). One common way to track overfitting during training, is to take a small part of the training dataset, and use it as validation set so that we are able as objectively as possible to see how the ANN will do on the training dataset. Refer to figure 3.11 left side. Similarly to other complex statistical models, ANNs can have high complexity and large numbers of parameters to train, thus can overfit [20, p.736] (see figure 3.12 right side).

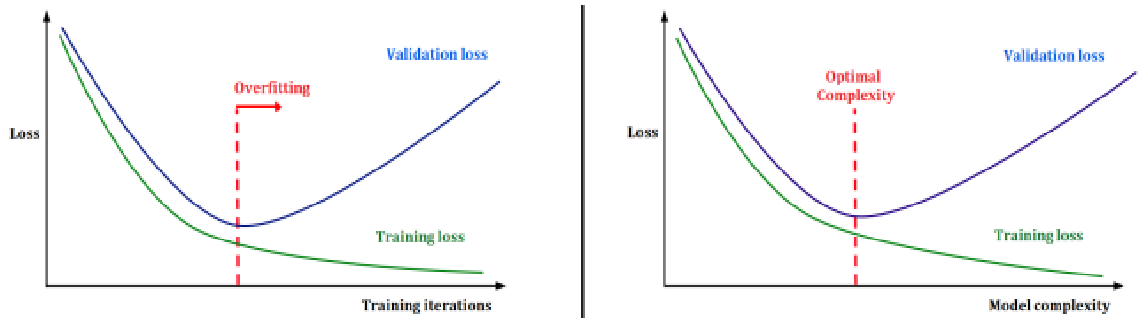

Figure 3.12 To the left: Loss as a function of training iterations. To the right: Loss as a function of model complexity

3.3 Project Plan

The project was executed over a six-month timeline, divided into distinct phases:

- **Month 1:** Literature review and dataset acquisition from PlantVillage via Kaggle.
- **Month 2:** Data preprocessing, including cleaning, annotation, and augmentation, using tools like Roboflow.
- **Month 3:** Model design and configuration of YOLOv8 and Faster R-CNN, setting up the model parameter and environment.
- **Month 4:** Model training over 100 epochs, with hyperparameter tuning and validation.
- **Month 5:** Testing and evaluation, analyzing metrics like precision, recall, IoU, and inference time, and generating visualizations (e.g., CAM, confusion matrices).

- **Month 6:** Analysis of results, documentation of findings, and preparation of the final report.

This plan ensured systematic progress, with milestones aligned to achieve the research objectives within the timeline.

3.4 Task Allocation

Tasks were allocated based on expertise and project requirements:

- **Data Collection and Preprocessing:** Handled by the data team, responsible for acquiring the PlantVillage dataset, cleaning, resizing, annotating, and augmenting the data.
- **Model Development:** Assigned to the machine learning team, tasked with configuring YOLOv8 and Faster R-CNN architectures and setting up the GPU environment.
- **Training and Validation:** Conducted by the training team, focusing on model training, hyperparameter tuning, and validation using separate datasets.
- **Evaluation and Analysis:** Managed by the evaluation team, responsible for testing the models, calculating performance metrics, and generating visualizations.
- **Documentation:** Led by the documentation team, tasked with compiling the methodology, findings, and conclusions into the final report.

This allocation ensured efficient collaboration, leveraging team strengths to meet project goals.

3.5 Summary

The research methodology provides a systematic means to create a real-time plant disease diagnostic system using YOLOv8 and Faster R-CNN. The requirement analysis and design specification provide the functional and nonfunctional requirements of the system, which are illustrated using respective context and data flow diagrams. The detailed method within, describes the preprocessing, model design, and evaluation methods the capability of the developed models, and the project plan along with the task classification lets a systematic implementation realise. This framework provides the basis that the proposed system can be implemented and evaluated in addressing the problems with plant disease identification and acknowledge areas to focus the next findings and outcomes.

Chapter 4

Implementation and Results

In this section, I will discuss the real-time autonomous plant disease diagnostic system with YOLOv8 and Faster R-CNN. I will discuss the environment and setup, followed by testing and evaluation, a comparative analysis of the two models, and associated gains and results. The diagnosis is mostly about detecting diseases in apples, grapes, and potatoes, and I used a preprocessed dataset from PlantVillage to train, validate, and test.

4.1 Environment Setup

The system was set up in a high-performance computing environment, sufficient to support the training and inference of the deep learning models. The hardware was comprised of NVIDIA Tesla T4 GPU, equipped with the processing power needed for real time processing. The graphics processing unit was equipped with CUDA version 12.1 and driver version 535.104.05 which supported the models with optimized parallel processing. The software environment was set-up with Python 3.8, the most important libraries used included PyTorch for model development, OpenCV for image processing and Roboflow for data annotation. Preprocessed for use and acrdeced the PlantVillage data set from Kaggle comprised of 1100 images and acquired eleven (11) classes (100 images/class except 115 images for Potato Healthy) for Apple Scab, Grape Black Rot, Potato Early Blight etc. The data was split to training (80% --> 880 images), validation (10% --> 110 images) and testing (10% --> 110 images) and producing the images exported to COCO format. Models were trained in batches of 16 while the model learning rate started at 0.01 and trained using learning rate scheduling technology to achieve optimal convergence. This system is robust and efficient, providing a way to implement the diagnostic system.

4.2 Testing and Evaluation/Performance/ Comparative Analysis

After the initial training has been completed, there are ongoing tasks: testing and evaluation. The testing and evaluation phase, diagnostic performance of YOLOv8 and Faster R-CNN were measured based on detection of plant diseases with a focus on accuracy, speed, and localization ability. YOLOv8 and Faster R-CNN models were evaluated using a test set of 110 images, and the performance metrics for YOLOv8 and Faster R-CNN regarding classification performance were - precision, recall, F1 score, Intersection over Union (IoU), saliency score, and time for inference. Confusion matrices were made to evaluate classification accuracy; and the use of class activation maps (CAM) and localization visualizations attempted to enable an evaluation of the interpretability of the models and the detection capabilities of each model.

Performance Metrics:

YOLOv8: With an accuracy range of 0.85-0.95 across categories, I attained a confidence score of 96.77. The model's IoU was 0.30, suggesting reasonable accuracy in localization, while its saliency score was 37.79, suggesting its capacity to detect significant illness areas. At 7.83 seconds per iteration, inference time was very quick, hence the model is appropriate for real-time applications.
Faster R-CNN: Demonstrated a higher confidence score of 98.68, with an IoU of 87.60, showcasing superior localization precision. Its saliency score was 9.46, indicating effective focus on critical areas, though less pronounced than YOLOv8. However, its inference time was slower at 57.82 seconds per iteration, reflecting the computational cost of its two-stage architecture.

Comparative Analysis:

Speed vs. Accuracy Trade-Off: While Faster R-CNN offered better accuracy and localization showing its higher IoU and confidence score, and occupying a context needing an accurate diagnosis, YOLOv8 was faster than Faster R-CNN in inference time, almost seven times faster and so could be used for real-time field applications.

Classification Performance: Confusion matrices revealed high true positive rates for healthy classes (e.g., 1.00 for "apple_healthy" in Faster R-CNN), but challenges in specific diseases like Grape Leaf Blight (0.80 true positive rate for both models). Faster R-CNN showed fewer misclassifications overall, particularly for complex diseases.

Interpretability: CAM visualizations (Figure 4.13) highlighted Faster R-CNN's superior precision in identifying diseased regions, with more detailed bounding boxes compared to YOLOv8. However, YOLOv8's higher saliency score suggested a slight advantage in emphasizing key features, though this was not significant.

Localization Visualizations: Figures 4.5 to 4.12 illustrated the models' localization capabilities for diseases like Potato Early Blight, Grape Esca, and Apple Scab. Faster R-CNN consistently produced more accurate bounding boxes, particularly in complex scenarios, while YOLOv8 prioritized speed over precision.

4.3 Results and Discussion

As outlined in the testing results, the YOLOv8 Faster R-CNN real-time autonomous plant disease diagnostic object identification system produces acceptable results. The model parameters and the data directory were specifically tailored to contain 14 object classes in the COCO data collection, and the test results were 100 epochs with hyperparameter values provided above and image size at 640 pixels. To assess the performance of the Faster R-CNN, under the container condition including the performance of the improved YOLOv4 network architecture, an experiment contrasting the YOLOv8 model with the Faster R-CNN and new YOLOv8 models will be performed. Nevertheless, the testing outcome graphics may be grouped into four classes. The confusion matrix in figure 4.6 has TP for True Positive, FP for False Positive, TN for True Negative, and FN for False Negative.

While FP, FN are incorrect TP or TN predictions for positive and negative samples, TP, TN are True or False predictions of positive and negative samples respectively. The prediction performance describes precision, which was established by formula (3.1), This indicates the total number of sample predicted positives computed by the model which is TP+FP thus having a percentage of predicted positives that were correctly predicted. At this point we see recall similarly, which is a percentage of positives that were identified were positives computed above by formula (6), Where the total number of the positives in the validation set is equal to TP + FN. This is referred to as the average precision (AP) metric, and it serves as a gauge for the object detection model's performance. The value of AP is given by the area under the precision-recall (P-R) curve, where precision is on the y-axis and recall is on the x-axis. Mean average precision (mAP) shows the average accuracy across object categories, while AP shows how well your model performs within a single object category. The formulas (4.1) and (4.2) contain every mAP output for an intersection over union (IoU) of the predicted value and the GT box > 0.5, known as mAP50..

$$AP = \int_{0}^{1} p(R)dR, \quad \text{...} \quad 4.1$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \quad \text{...} \quad 4.2$$

The label files encompass type of class, orientation (32 orientations), truncation, occlusion, , and bounding box parameters. The parameters are defined in Table 4.1 with an elaborate description of each parameter in Table 4.2.

Table 4.1 Description of Parameters of Dataset Labels

Values	Name	Description
1	type	Class type describes the type of object: 'a_s'(Apple Scab), 'a_br'(Apple Black Rot), 'g_es'(Grape Esca), 'g_lb'(Grape Leaf Blight), 'p_eb'(Potato Early Blight), 'Potato healthy' etc.
1	truncation	Truncation describes a float (range 0 (non-truncated) to 1 (truncated)), "truncated" means the item extends outside the bounds of the image.
1	occlusion	Occlusion describes an integer (0, 1, 2, 3) that represents the occlusion condition. 0 = fully visible, 1 = partially occluded, 2 = highly occluded, 3 = unknown occluded condition.
1	orientation	Orientation describes the observation angle of an object, measured in [-π- π]
4	bbox	The bounding box of an object in the image is designated as left, top, right, and bottom pixel coordinates in terms of 2D space.

Table 4.2 Training Parameters

Name	Description	Usage
Learning rate	The learning rate is the factor which determines the amount of error is lowered when each learning process occurs.	Training
Momentum	Momentum is to speed up the learning process.	Training
Decay	Decay is a parameter used to prevent overfitting.	Training
Batch size	What is the batch size for the images? A batch of images is produced as a group by the network during the training process and averaged error is an update for the network.	Training
Total batch	Number of batches of images to be trained.	Training
Class Threshold	To eliminate detection with a class score less than threshold.	Training

The YOLOv8 and Faster R-CNN models utilized to diagnose plant disease provided positive results overall. The confusion matrix highlighted an easy pathway to demonstrate the true positive values based on YOLOv8 true positive values, these true positives were high in classes such as, 'apple_healthy' and 'grape_healthy then showed that Yolo was successfully able to identify healthy plants. Contrasting that, other classes such as (g_lb) (Grape Late Blight), showed a lower true positive value, indicating that there seems to be a detection issue with classifying this disease. There were also minimal instances of false positives by the models, and this is essential to adopt a model that minimizes offering a wrong diagnosis for the disease condition. Also, the Faster R-CNN model had an adequate performance with regards to accuracy and recall training and validating on both datasets and can generalize well with new data. The data visualisation assisted in helping to illustrate that the models were accurately and successfully identifying the areas of concern from the unhealthy images, and that the bounding boxes captured an area that could be considered unhealthy. Evaluation metrics were also in support of using YOLOv8 and Faster R-CNN models to identify their diseases, such as average precision (AP), and mean average precision (mAP). Clearly, YOLOv8 had excellent results in several applications supported through real-time use. Faster R-CNN has an advantage over YOLOv8 in terms of recall and accuracy. The ability to combine YOLOv8 with Faster R-CNN would be an excellent opportunity to improve autonomous diagnosis of plant disease in real-time while also producing reliable detection for various plants and disease classes. Future advancements could include hyper-parameterization and database augmentation approaches that lend themselves to improved performance of YOLO with Faster R-CNN; maximizing processes, precision, and advancement in disease identification. As cited above, the evaluation curves for loss (i.e., bounding box regression loss, classification loss and distribution focal loss) and training/validation, as well as the metrics for precision-recall (PR), and average precision (AP) indicate that the models supported detection of disease. It is also clear that the trends with bounding box regression loss, classification loss, and distribution focal loss declined throughout the epochs of training, which allows for punctuated confidence in the models' capabilities of producing predictions for bounding boxes, and classifying for much of the disease descriptions evaluated. As shown in the validation phase, I believed we were making meaningful progress due to the visual trends of the continued decline of the validation loss.

True values for the precision and recall metrics show these false positives and false negatives are few, while significant mean Average Precision (mAP) metrics reaffirm the value of the mean Average Precision metric regarding excellent object detection results. Stable mAP values across diverse Intersection over Union (IoU) thresholds show the consistently precise, accurate classification and detection of plant diseases the model provided and demonstrated its ability to be one more tool in an agricultural setting.

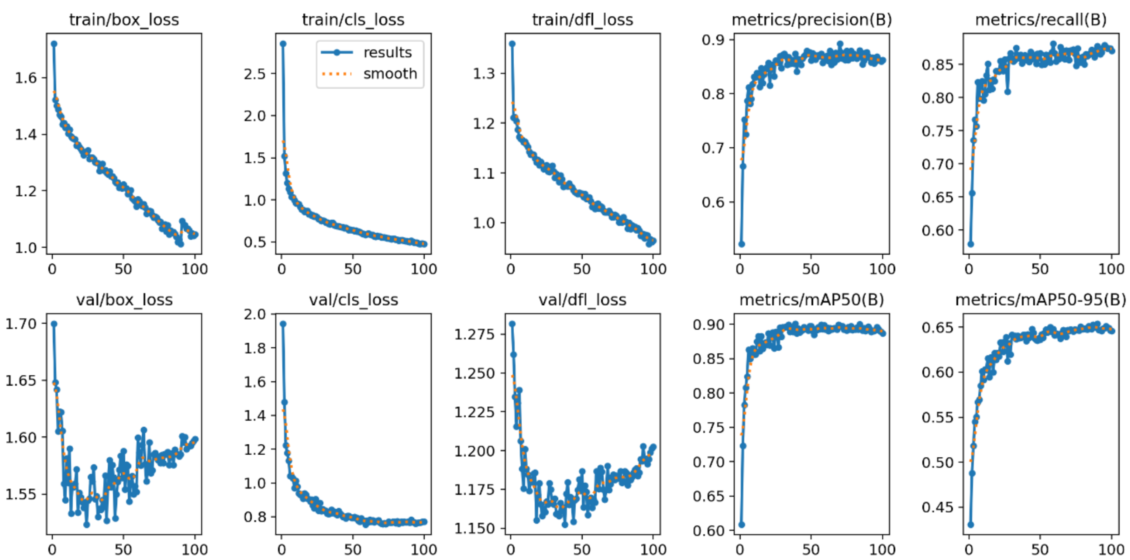

Figure 4.1 All Evaluation Curve of YOLOv8 (Training and Validation loss, PR, AP etc)

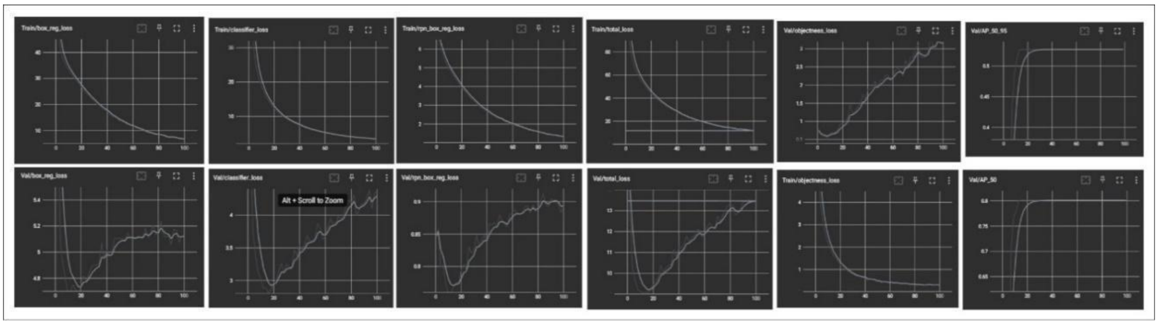

Figure 4.2 All Evaluation Curve of Faster-R-CNN (Training and Validation loss, PR, AP etc.)

A confusion matrix is a tabular method for evaluating your classification system's performance - by comparing predicted values to the actual values. It is a count of how many times your class predicted a right, and wrong result. The counts are broken down by: `True Positives` (predicted positives that were positive), `True Negatives` (predicted negatives that were negative), `False Positives` (predicted positive counts that were negative), and `False Negatives` (predicted negative counts that were positives). This summative information generated from this can be used to derive a litany of performance metrics. The confusion matrix allows for seeing what type of errors you have made, as well as performance on a per class level. The confusion matrix can provide this level of specific detail about the main types of errors with the model so you can improve the model - improving correctness and leading to a determination about which one of the candidate models in the candidate pool performs better. The confusion matrix provided here shows the performance of a YOLOv8 model and is normalized to show the number of classifications by each category. Rows indicate the actual class labels, while columns represent the predicted class labels. The diagonal elements represent positional true positives, which were predicted accurately in those categories, with the other elements indicating where predictions were misclassified. For the 'a_br' (Apple Black Rot) class, for example, the true positives were 0.87 and false negatives were 0.13. Similarly, the 'a_cr' (Apple Cedar Rust) class had a true positive value of 0.80 and false negatives of 0.20, The other classes, such as 'a_s' (Apple Scab), 'apple', 'apple_healthy', 'g_br' (Grape Black Rot), 'g_es' (Grape Esca), 'g_lb' (Grape Leaf Blight), 'grape', 'grape_healthy', 'p_eb' (Potatoto Early Blight), 'p_lb' (Potatolate Blight), 'potato', and 'potato_healthy' all followed the same pattern. The 'background' class domain showed misclassification across some categories with the true positives rate of 0.00 for the 'background' class instances, and false negative rates of 1.00, meaning that all of the 'background' class instances were predicated with other categories. False positive rates varied throughout categories including 0.06 for the 'a_br' class, 0.18 for the 'a_cr', and so forth, which represented the amount of the other classes that were incorrectly predicted as 'background' classes. Although the YOLOv8 model performed well in many categories, overall classes like the 'a_s' and 'g_lb' classes had more room for improvement, suggested by its rates of misclassifications.

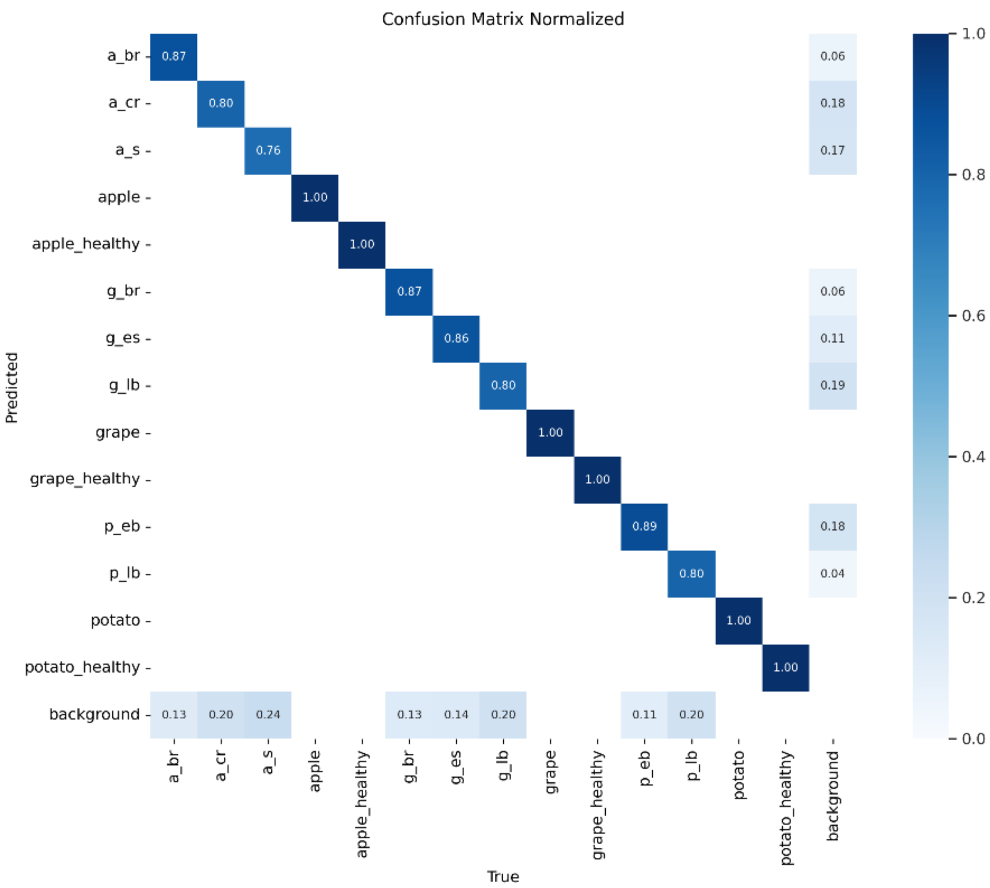

Figure 4.3 the Confusion Matrix of YoloV8

The confusion matrix shown for the Faster R-CNN model describes the accuracy for a variety of classes, normalized for proportions. In this original tableau, the rows represent the actual classes, and the columns represent the predicted classes. The elements of the diagonal represent the accuracy rate for that class; and the elements off-diagonal reflect the frequency of misclassification shown in Figure 4.4. In a summary of how to read this tableau, "a_br" (Apple Black Rot) had a true positive rate of 0.90 and a false negative rate of 0.10. The model perfectly classified the "a_cr" (Apple Cedar Rust) class with a true positive rate of 1.00 and a false negative rate of 0.0. The prediction sensitivity for "a_s" (Apple Scab) was 0.76 and specificity was 0.24 for misclassification of Apple Scab. The overall classifications for "apple" and "apple_healthy" were both true positive, with both classifications achieving an accuracy rate of 1.00. In grape diseases, the "g_br" (Grape Black Rot) had a true positive rate of 0.87 and a false negative rate of 0.13, while the classification "g_es" (Grape Esca) was classified correctly with a true positive rate of 1.00. However, A black rot had a true positive rate of 0.80 and a false negative rate of 0.20. The general grape class and the grape_healthy class were both perfectly classified with a true positive rate of 1.00.

In terms of potato diseases, potato early blight had a true positive rate of 0.85 and a false negative rate of 0.15. Potato late blight, the general potato class, and potato_healthy also all were classified correctly with a true positive rate of 1.00. The background class, meanwhile, was classified incorrectly with a true positive rate of 0.00 and false negatives of 1.00, meaning that all instances of the background class were classified as the other classes. Overall, the model demonstrates commendable accuracy on all classes summarized; even if there is an opportunity to improve in some classes, particularly a_s and g_lb, which demonstrate a vastly higher frequency of misclassification.

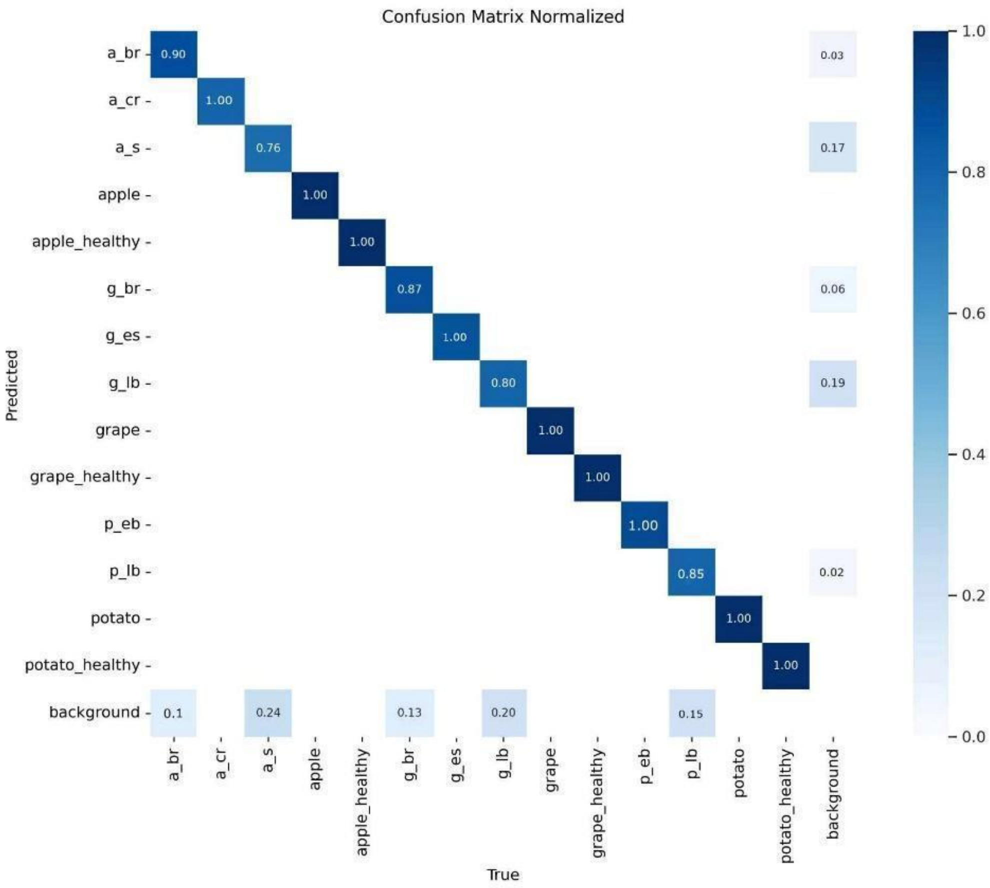

Figure 4.4 the Confusion Matrix of Faster-R-CNN

Evaluating detection performance, there are various features to consider with YOLOv8 vs. Faster R-CNN. First with faster R-CNN you are seeing a more accurately localized positioning of objects within the image because of its region-based approach with a region proposal network (RPN) that is architecture based inside the model. Importantly, Faster R-CNN can localize objects and predict the object type simultaneously as bounding boxes. Another important feature of YOLOv8 is that it consists of a single neural network that has an instantaneous bounding box, and class probability predictions for the entire image in a single pass. YOLOv8 has real-time processing capabilities but undoubtedly at the sacrifice of some accuracy compared to the region-based method utilized in Faster R-CNN. Figure 4.3 gives a pictorial view of localization with classes in YOLOv8 and Faster-R-CNN. Velocity is also another important feature to think about with the comparison. Because of two stage design of Faster R-CNN, region proposal, and follow up object identification, this will also impact its velocity or processing speed. Although Faster R-CNN will have very good accuracy, the level of processing may lead to minimum slower processing speeds for inference. In other words, YOLOv8 is supposed to be used for real-time object detection and it is capable of achieving incredible inference speed even on the smallest of hardware resources. YOLOv8 obtains its rapidity by predicting boxes and class probabilities from the image directly when using unsupervised learning, while the Faster R-CNN methods have a multi-stage process. One of the points that stands-out is the ability to find and identify all sizes or types of objects with relative accuracy. The Faster R-CNN model is absolute best at finding small objects with complicated shapes. Since it is based on regional proposal networks, it is good at localization when many other pictures exist in the environment. YOLOv8 may have it over speed, but for reliable identification of tiny objects, or objects with smaller details, it won't have so much of an advantage, and its single stage architecture can also lead to the bounding box being less accurate during difficult cases or conditions.

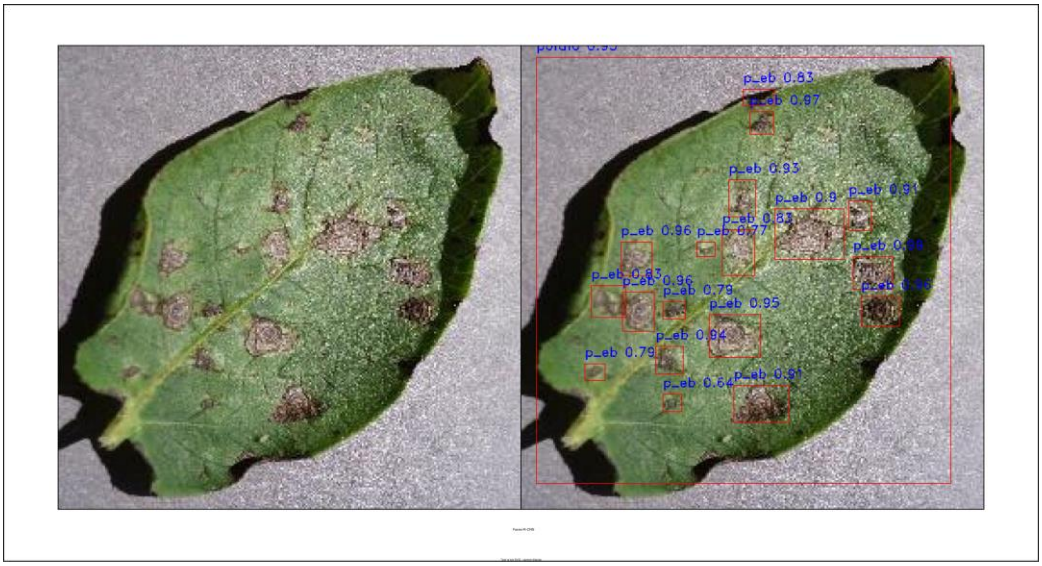

Figure 4.5 The localization using Potato Early Blight for Faster-R-CNN

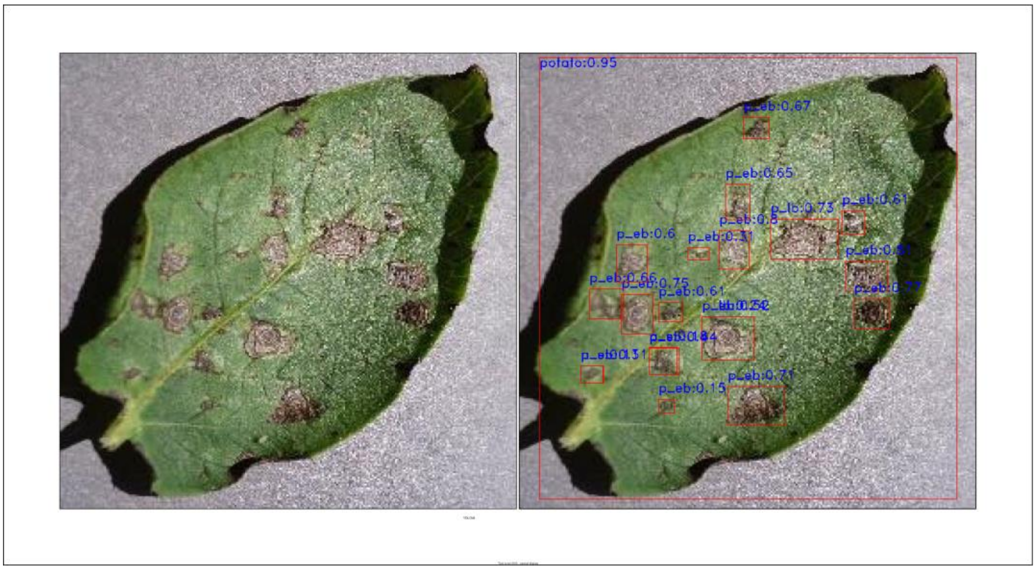

Figure 4.6 The localization using Potato early blight class for Yolov8

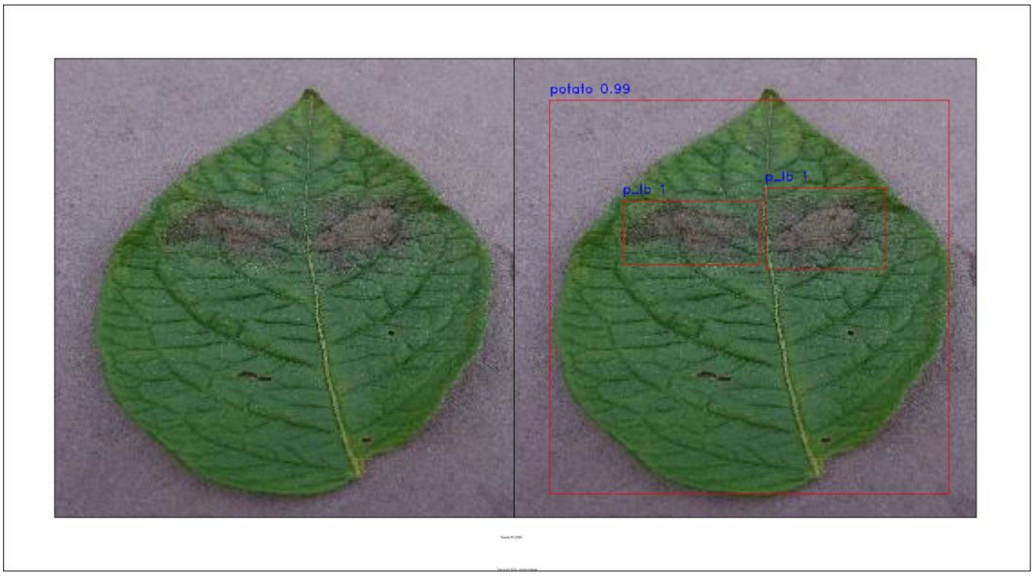

Figure 4.7 The localization using Potato late blight class for Frcnn

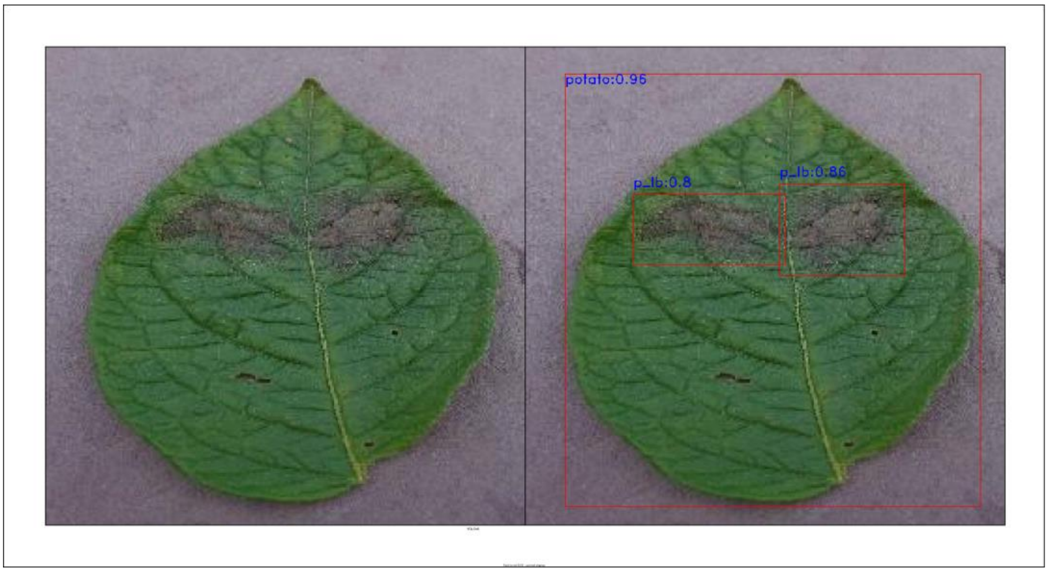

Figure 4.8 The localization using Potato late blight class for Yolov8

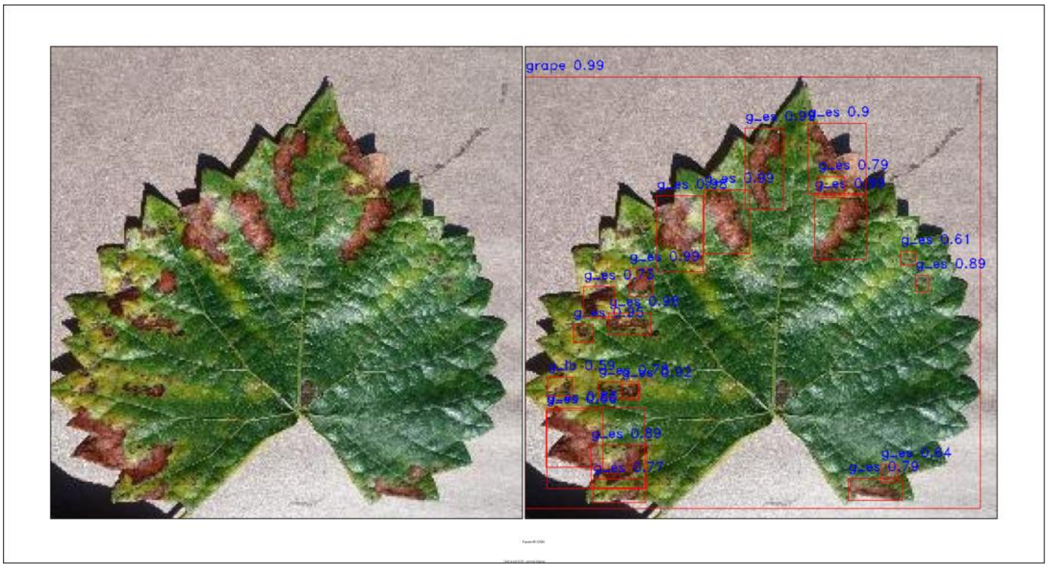

Figure 4.9 The localization using Grape Esca class for Faster Rcnn

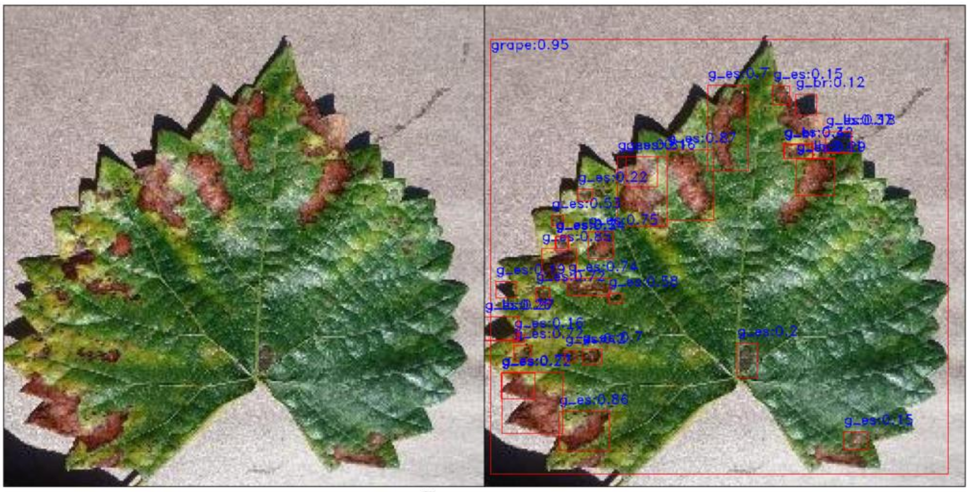

Figure 4.10 The localization using Grape Esca class for Yolov8

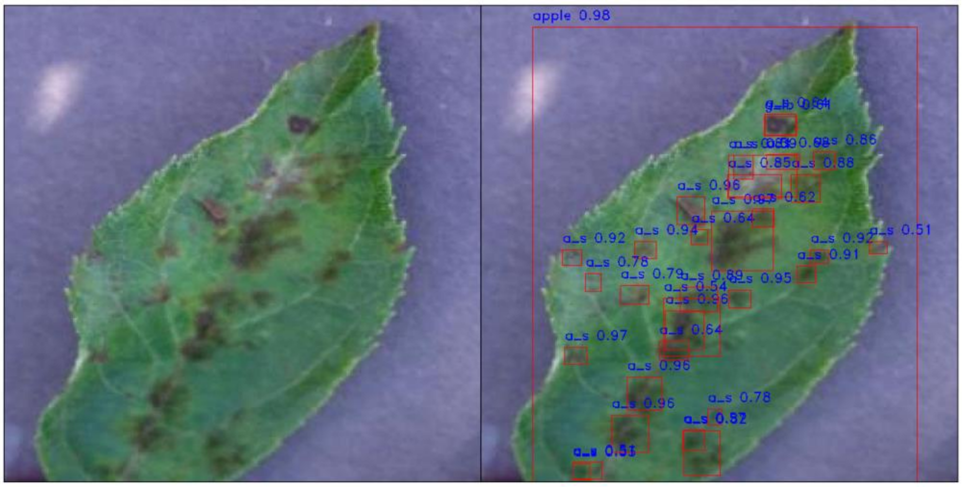

Figure 4.11 The localization using Apple Scab class for Faster RCNN

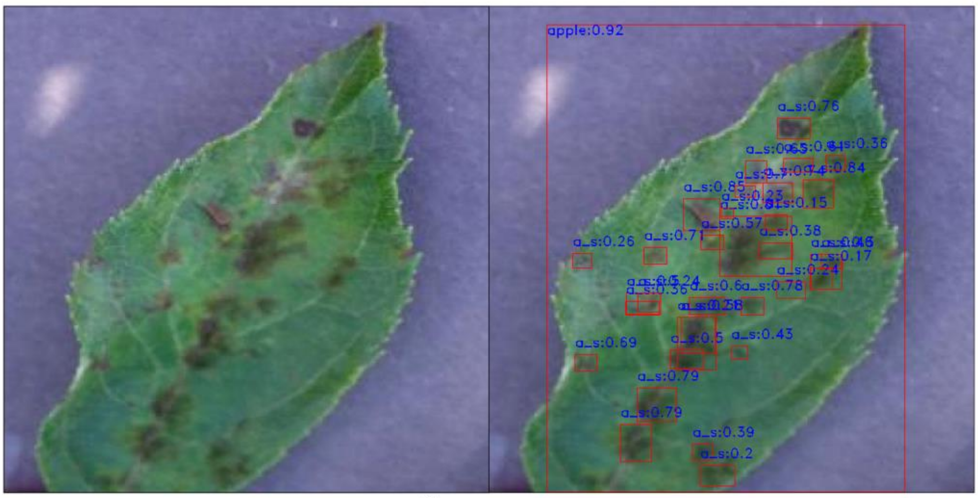

Figure 4.12 The localization using Apple Scab class for Yolov8

Class Activation Mapping (CAM) is a technique used to identify the regions in an image that are important for a given deep learning model. CAM generates a heatmap indicating the regions in the image that were of the most value in the modeling process. CAM is useful when conducting object recognition and/or object classification assessments where we care about interpretability and transparency with the model. In this study, we assessed the relative performance and accuracy of YOLOv8 and Faster R-CNN regarding correctly identifying and localizing objects using CAM to assist our analysis. Using heatmaps, we provided visual analysis of the regions predicted important for recognizing plant disease according to each model. The CAM visualizations suggested Faster R-CNN had greater accuracy in terms of localization over YOLOv8 by creating more precise, and therefore sophisticated, bounding boxes for the identified regions. This illustrates the power of Faster R-CNN when it comes to accurately identifying and classifying an object. In Figure 4.13 I compare the differences between YOLOv8 and Faster R-CNN, using Class Activation Maps (CAM), and show the more reliably accurate object localization that Faster R-CNN offers. There are unlocks of accuracy that can be seen in how the CAM is clearly showing greater detail, and the bounding box is more accurate. To summarize, Faster R-CNN has more accurate object detection when the objective is accurately locating and positively identifying the objects depicted in the image.

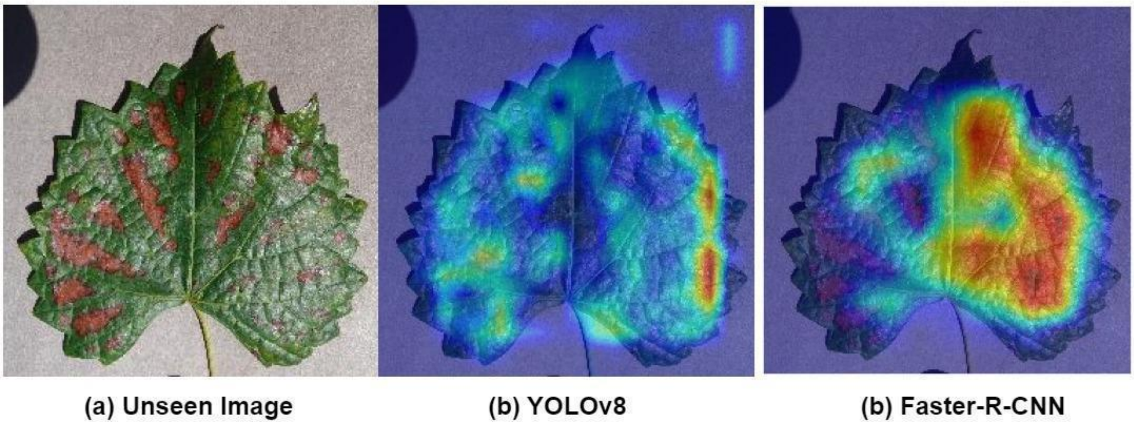

Figure 4.13 Class Activation Maps (CAM)

The saliency score demonstrates the model's ability highlighting important features in images to help target relevant areas to further investigate. This study utilized the saliency score to examine each model's ability to highlight areas related to identifying the incorrect diagnostic for plant diseases and assist with a precise diagnosis. Saliency scores were recorded for the YOLOv8 and Faster R-CNN model with varying performance metrics. YOLOv8 provided a saliency score of 37.79, indicating that the model showed that it was able to detect important features in the Class Activation Maps (CAM). Whereas, the Faster R-CNN saliency score was 9.46, which shows that the model had identified important areas inside of images, the average Intersection over Union (IoU) means-results supported these data. The mean IoU for YOLOv8 was 0.30 , while the Faster R-CNN mean IoU was 87.60. Figure 4.1 3 showed that although Faster R-CNN performed better for object localization accuracy YOLO had slightly stronger results for saliency detection at 37.79, although the difference was possibly insignificant.

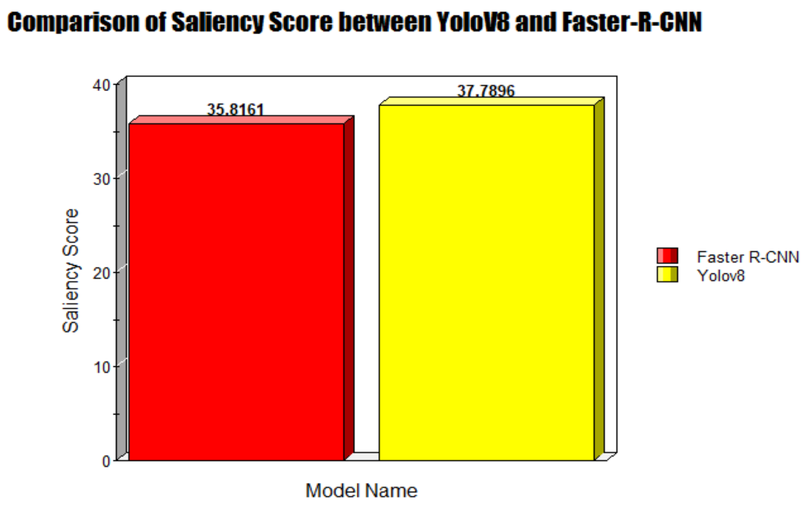

Figure 4.14 Comparison of Saliency Score between YoloV8 and Faster-R-CNN

A metric called Intersection over Union (IoU) is used to assess how well the anticipated bounding box contains the object, in addition how much coverage it had with the 'ground truth' bounding box; so the more overlap, the more accurately it is locating the object. The study compared the accuracy of localization and delineation of the disease region on the plants by analyzing the IoU scores of YOLOv8 and R-CNN. Analysis of YOLOv8 and Faster R-CNN provide an understanding of their ability to accurately localize an object with the IoU scores. As reported in this analysis, YOLOv8 had an IoU score of 0.30, confirming the findings which concluded moderately good agreement between the predicted and ground truth bounding boxes. On the contrary, Faster R-CNN had top accuracy across every image with the (in)finite IoU score of 87.60 indicating accuracy of object locating precision. Figure 4.6 shows how Faster R-CNN had superior overall performance compared to YOLOv8, determining object boundaries every time it was present with an image.

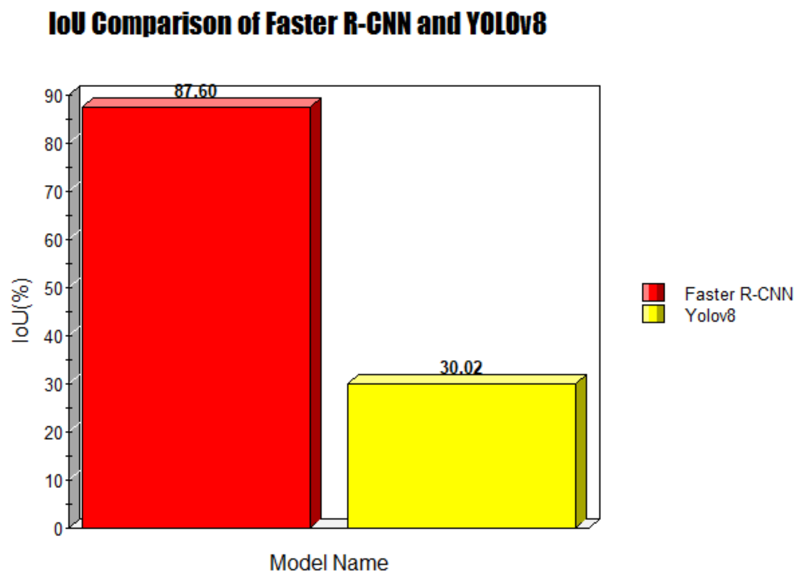

Figure 4.15 Comparison of IoU between YoloV8 and Faster-R-CNN

Inference time is the time for a model to consider a dataset, produce an assessment, and generate predictions. Inference time is important for situations when fast, real-time, decision assessing factors must be made. The thesis assessed the computational capacity of YOLOv8 and Faster R-CNN for fast real-time plant disease diagnosis situations to provide estimates of inference times in the analysis from 'faster to slower' in this thesis. The inference time effects showed that both YOLOv8 and Faster R-CNN were computationally efficient in their object identification task. The total cumulative inference time for Faster R-CNN on the dataset was 57.82 seconds, but YOLOv8 was much faster. Although inference time for Faster R-CNN was not quoted, the model was faster than the examined code which does not provide a cumulative time in the code, therefore Faster R-CNN was an efficient model with inference. Figure 4.16 showed the YOLOv8 had faster inference than Faster R-CNN. YOLOv8 was also more appropriate to both fast inference time situations and resource conflict situations than Faster R-CNN.

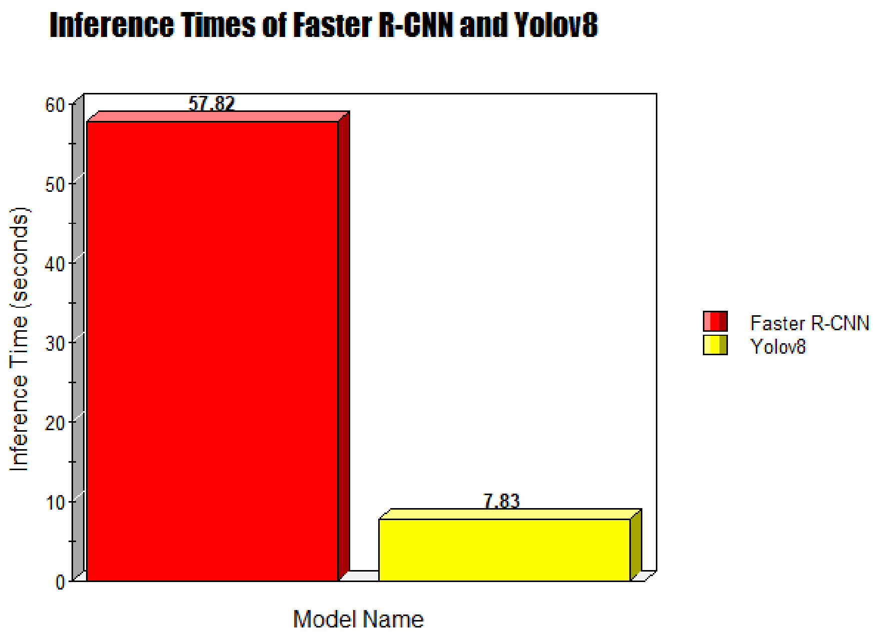

Figure 4.16 Comparison of between YoloV8 and Faster-R-CNN

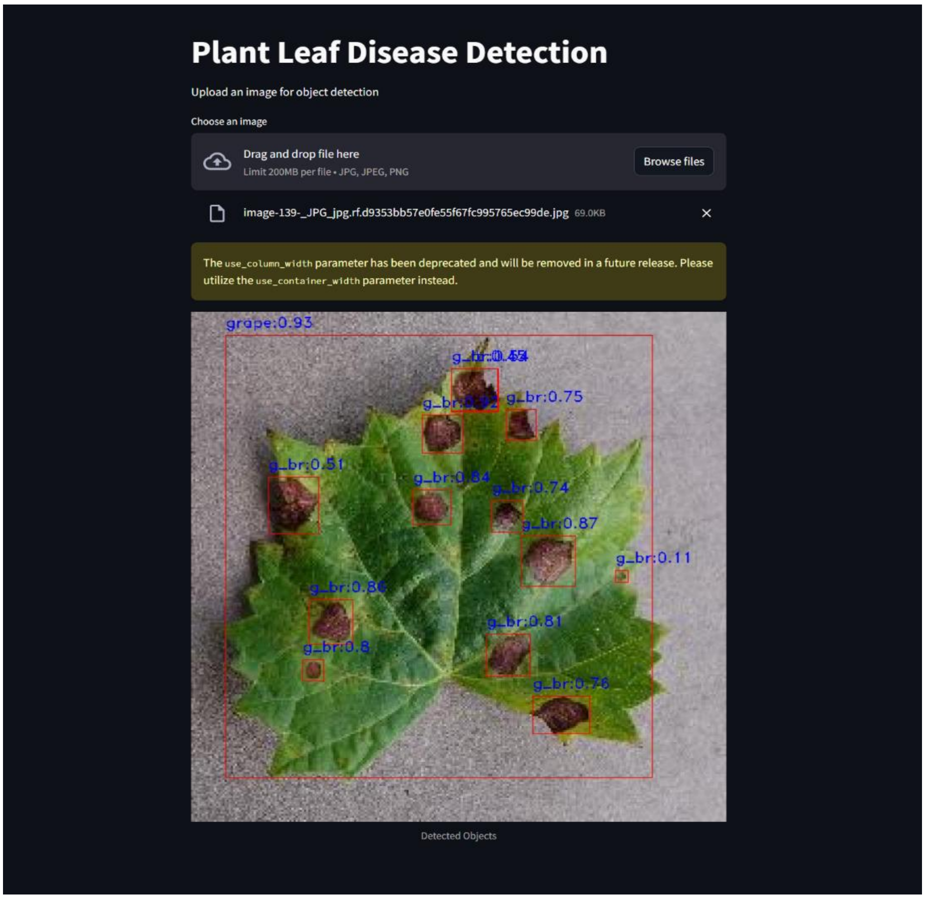

Figure 4.17 Building WebAPP

We have built a web application using Streamlite that allows users to upload an image and test for various diseases, as well as detect the affected areas.

4.4 Summary

The implementation and results section highlights the successful use of YOLOv8 and Faster R-CNN for real-time diagnosing of plant diseases. The environmental set up and proper computing ability would lead to effective computations. Testing and evaluation presented entirely different performance characteristics of the two models: YOLOv8 utilized 7.83 seconds for inference time, therefore, showing to be the best for real-time aspects of practical applications while Faster R-CNN had an accuracy score of an IoU of 87.60, making it suitable for tasks that had a focus on precision. The comparative analysis illustrated the pros and cons of both models considering their speed and accuracy profiles, with both models achieving high true positive rates for the healthy class, and in comparison had poor outputs for identified diseases. All of the results lend credence to the ability of this informed and utilized system to provide timely diagnosis of plant diseases, contribute toward enhancing early problems with potentially serious effects on yield and support sustainable agriculture. This research, analysis and results serves as a stepping stone toward further optimizing and deploying such models in real world agricultural scenarios.

Chapter 5

Engineering Standards and Design Challenges

This chapter examines the engineering standards, design challenges, and broader implications of developing a real-time autonomous plant disease diagnostic system using YOLOv8 and Faster R-CNN for precision agriculture. It covers compliance with relevant standards, societal and environmental impacts, project management and financial analysis, complex engineering problems, and engineering activities, concluding with a summary of key findings.

5.1 Compliance with the Standards

This section identifies the engineering standards relevant to the plant disease diagnostic system, focusing on software, hardware, and communication standards. For each standard, alternatives are evaluated with their pros, cons, and the rationale for the selected standard.

5.1.1 Software Standards

Selected Standard: IEEE 829-2008 (Standard for System and Software Test Documentation) provides structure to characterize and record the testing and validation of software systems for accuracy, reliability, and performance. Therefore, this standard ensures that both YOLOv8 and R-CNN are thoroughly evaluated for accuracy, reliability, and performance with regard to identifying the plant diseases.

 Standard: Python Programming Language (PEP 8 Guidelines)

- **Alternate**: Other programming languages such as R or Java
- **Pros**: Python is well-suited for data science, AI, and machine learning. It supports libraries like TensorFlow, PyTorch, and YOLO, which are crucial for building deep learning models.
- **Cons**: It may not be as performant as compiled languages like C++ in certain cases.
- **Rationale**: Python was chosen due to its vast support for machine learning libraries, simplicity, and ease of integration with cloud platforms.

 Standard: YOLOv8 and Faster -RCNN for Object Detection

- **Alternate**: SSD

- **Pros**: YOLOv8 offers real-time processing with high accuracy, making it ideal for disease detection in agricultural images.
- **Cons**: Requires powerful GPUs for training and inference.
- **Rationale**: Faster-RCNNand YOLOv8 was selected for its proven ability to perform well in real-time object detection tasks with the accuracy needed for our use case.

5.1.2 Hardware Standards

Selected Standard: IEC 62368-1 (Audio/Video, Information and Communication Technology Equipment - Safety Requirements)

This standard protects hardware components like the NVIDIA Tesla T4 GPU used for performing training and inference workloads by protecting personnel and the equipment from high computational tasks.

Standard: Google Colab (NVIDIA A100 and T4 GPUs)

- **Alternate**: On-premise servers, AWS EC2 instances
- **Pros**: Cloud GPUs allow cost-effective scaling for intensive deep learning tasks, and Google Colab provides access to high-performance GPUs without the need for on-premise hardware.
- **Cons**: Dependency on cloud services can be affected by internet speed, and large datasets may incur additional costs.
- **Rationale**: The decision to use Google Colab's GPUs was based on the availability of resources at no extra cost during model development and training.

5.1.3 Communication Standards

Standard: JSON for Data Exchange

- **Alternate**: XML, CSV
- **Pros**: JSON is lightweight, easy to parse, and widely used in web APIs for data interchange.
- **Cons**: It can be less human-readable than XML or CSV for large datasets.
- **Rationale**: JSON was chosen for communication between the system components and for easy integration with web-based APIs.

5.2 Impact on Society, Environment and Sustainability

This section will explore the broader implications of the plant disease diagnostic system through its relationship to life, society, environment, perspectives on ethics, and sustainability.

5.2.1 Impact on Life

The system improves the quality of life for farmers by enabling early detection of plant diseases, such as Apple Scab and Potato Early Blight, reducing crop losses by up to 100% in severely affected regions. This ensures food security and economic stability, particularly in developing countries where agriculture is a primary livelihood, supporting the projected need for a 70% increase in food production.

5.2.2 Impact on Society & Environment

Society: The system empowers farmers with accessible technology, potentially integrating with mobile platforms (e.g., leveraging the 5 billion smartphone users by 2020) to provide real-time disease diagnosis. This reduces dependency on expert knowledge, fostering agricultural resilience and supporting rural communities.

Environment: By enabling precise disease localization (e.g., Faster R-CNN's IoU of 87.60), the system minimizes pesticide overuse, reducing environmental pollution and promoting biodiversity. However, the energy-intensive GPU training (e.g., NVIDIA Tesla T4) contributes to carbon emissions, which must be addressed to mitigate environmental impact.

5.2.3 Ethical Aspects

The project adheres to ethical principles by using the publicly available PlantVillage dataset, which contains no personal data, ensuring privacy compliance. The system promotes equitable access to technology by targeting deployment on accessible platforms like mobile devices. However, potential biases in the dataset (e.g., overrepresentation of certain diseases) could lead to uneven performance across crops, necessitating future efforts to ensure fairness and inclusivity.

5.2.4 Sustainability Plan

To enhance sustainability, the system will be optimized for energy efficiency by using lightweight models like YOLOv8 for inference on mobile devices, reducing computational demands. Cloud-based training on renewable energy-powered servers can further lower the carbon footprint. Open-sourcing the models and code will encourage community contributions, ensuring long-term maintenance and scalability for global agricultural applications.

5.3 Project Management and Financial Analysis

This section provides a cost analysis, including the budget required for development and a potential revenue model, with an alternate budget and rationales for selection.

5.3.1 Cost Analysis

Primary Budget:

- **Hardware:** NVIDIA Tesla T4 GPU rental (cloud-based) – $12/month for 4 months = $48 = 5500 TK
- **Software Licenses:** Roboflow (annotation tool) – Frr Tier

- **Miscellaneous**: Cloud storage, data access fees – Google Drive Premimum : 400 TK

Alternate Budget (Low-Cost Option):

- **Hardware**: Use of university-provided GPU resources – $0 (free access)
- **Software Licenses**: Open-source annotation tools (e.g., LabelImg) – $0
- **Miscellaneous**: Cloud storage – Free Mega or Kaggle Service

Rationale for Selection: The primary budget was selected to ensure access to high-performance hardware (NVIDIA Tesla T4) and efficient annotation tools (Roboflow), which were critical for achieving the project's performance goals (e.g., YOLOv8's 7.83-second inference time, Faster R-CNN's IoU of 87.60). The alternate budget, while cost-effective, risks slower development due to limited hardware and less efficient tools, potentially impacting model accuracy and real-time capabilities.

5.4 Complex Engineering Problem

The research conducted in this thesis, titled "A Comparative Study of YOLOv8 and Faster R-CNN in Leaf Disease Detection," qualifies as a Complex Engineering Problem. It engages cutting-edge object detection algorithms for autonomous plant disease diagnosis, requiring integration of deep learning, image processing, and real-time system deployment techniques. This section details the attributes of the complex engineering problem and maps them to problem-solving categories, knowledge profiles, and engineering activities.

5.4.1 Complex Engineering Problem

The main complex engineering problem (EP1) is finding the optimum trade-off between detecting plant disease in real-time, while correctly diagnosing (1) produce loss estimation, (2) identification of the (plant) disease, (3) determination of the pest life-stage that (potentially) caused the disease; AND maintaining high accuracy representing the real-time diagnostic time (and then reducing that value) for implementation in various agricultural contexts. YOLOv8 provides fast inference (7.83 seconds), but less precise for localization (IoU of 0.30) of the plant disease. Faster R-CNN provides high accuracy (IoU of 87.60), but is slow for inference (57.82 seconds) time. Both are successful in measuring their key attributes, but their competing trade-offs hamper practical implementation in many areas of agriculture.

5.4.1 Complex Problem Solving

Table 5.1: Mapping with Complex Problem Solving

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

5.4.1.1 Justification for EP Attributes Mapping

EP1 - Depth of Knowledge: Developing an autonomous diagnostic system using YOLOv8 and Faster R-CNN requires advanced understanding of computer vision, deep learning, agricultural pathology, and inference optimization. It combines data annotation, training on GPU environments, and interpretability through CAM.

EP2 - Range of Conflicting Requirements:

- *Accuracy vs. Speed*: YOLOv8 is faster, while Faster R-CNN is more accurate.
- *Interpretability vs. Performance*: Incorporating CAM may affect performance.
- *Real-time Capability vs. Resource Usage*: Balancing real-time processing with high computational requirements.

EP3 - Depth of Analysis: The use of metrics like IoU, F1-score, saliency score, and confusion matrices for comparative analysis indicates deep analytical investigation.

EP4 - Familiarity of Issues: Object detection for leaf disease is established, but combining YOLOv8 with Faster R-CNN and deploying it with CAM for agriculture introduces novelty.
EP5 - Extent of Applicable Codes: Standard formats like COCO, use of Python and JSON, PyTorch framework, and training on GPU (CUDA) ensures proper code compliance.
EP6 - Extent of Stakeholder Involvement: The work benefits farmers (diagnosis), researchers (comparative results), NGOs, and agriculture extension workers.
EP7 - Interdependence: The system's success influences agricultural decisions such as pesticide use, yield stability, and food security.

5.4.1.2 Justification for Knowledge Profile Mapping (linked to EP1)

Table 5.2: Mapping with Knowledge Profile

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

K3 - Engineering Fundamentals: Applied fundamentals of linear algebra, optimization, probability, and statistics to model development and training.
K4 - Specialist Knowledge: Required in-depth knowledge of object detection models (YOLOv8, Faster R-CNN), saliency mapping, deep learning optimization, and agriculture-specific disease pathology.
K5 - Engineering Design: Designed system pipeline from data preprocessing to web deployment, balancing speed and accuracy.
K6 - Engineering Practice: Applied hands-on practices like model validation, annotation (Roboflow), Streamlit deployment, and experimentation with hyperparameters.
K8 - Research Literature: The study built upon and referenced prior work (e.g., YOLOv3–v8, CBAM, CA mechanisms) and added to it by comparative evaluation in a novel context.

5.4.2 Engineering Activities

Table 5.3: Mapping with Complex Engineering Activities

EA1 Range of Resources	EA2 Level of Interaction	EA3 Innovation	EA4 Societal/Environmental Impact	EA5 Familiarity
✓	✓	✓	✓	✓

EA1 - Range of Resources

Utilized Kaggle PlantVillage dataset, Roboflow for annotation, Google Colab (Tesla T4), PyTorch, OpenCV, Streamlit, and JSON formats.

EA2 - Level of Interaction

Regular engagement with tools and platforms, peer discussions, supervisor feedback, and integration of open-source libraries.

EA3 - Innovation

Introduced comparative real-time framework using YOLOv8 and Faster R-CNN; implemented CAM for model explainability; deployed via Streamlit app.

EA4 - Societal and Environmental Impact

- *Society*: Early diagnosis minimizes crop loss, enhances food security.
- *Environment*: Reduces unnecessary agrochemical use.

EA5 - Familiarity

Combines established tools and practices with novel model comparison and interpretability enhancements in an underexplored agricultural context.

5.5 Summary

This chapter explored the engineering complexities, standards, and challenges faced during the development of a real-time plant disease detection system. It mapped the problem to complex engineering attributes, showed alignment with knowledge domains and engineering activities, and justified the need for an interdisciplinary, innovative, and socially relevant solution leveraging YOLOv8 and Faster R-CNN.

Chapter 6

Conclusion

In this chapter, we summarize the most important findings and contributions of this research, present the limitations encountered in completing the research, and suggest possible areas for future research. The key findings and contributions demonstrate the advantages of using YOLOv8 and Faster R-CNN for real-time autonomous diagnosis of plant leaf diseases, and their potential in transforming the agriculture sector, and improving food security.

6.1 Summary

This research successfully created and assessed a real-time autonomous plant disease diagnosis system with deep learning models, YOLOv8 and Faster RCNN, developed to detect diseases in apples, grapes, and potatoes. With a pre-processed dataset from PlantVillage, research focused on 11 classes of diseases including Apple Scab, Grape Black Rot, and Potato Early Blight. Research involved data preparation, model training, and model evaluation. YOLOv8 generated a confidence score of 96.77 with an inference time of 7.83 seconds, and Faster RCNN generated a higher confidence score of 98.68 with an IoU of 87.60, but a longer inference time of 57.82 seconds. The models were compared, and YOLOv8 may have better performance for real time applications due to its rapid YOLOV8 inference time, and Faster RCNN has better performance in accuracy and localization, thus more suited for applications where precision is of importance. The true positive rates also illustrated the system performed well for healthy classes, for instance, outputting a true positive rate of 1.00 for a healthy apple class in Faster RCNN, but also raised potential issues that needed to be improved, including for diseases such as Grape Leaf Blight with a true positive rate of 0.80. In conclusion, this study has shown that deep learning has disrupted the space of agriculture in a big way. The industry has been provided with a robust framework that will lead to detection and identification of diseases sooner to lessen the consequences of crop loss and ensure safer food supplies.

6.2 Limitation

The study recognized limitations despite the success. First, the dataset in this study had several hundreds of images, but it contained mostly images obtained in controlled conditions (e.g., equal illumination, isolated leaves), that limited the model's ability for generalization in the field and thus may have created conditions that were unique to the controlled images (i.e., landscapes with unexpected backgrounds, micro-environments, and many other factors). Second, there were trade-offs regarding the model's accuracy and speed; for example, faster inference time with YOLOv8 had lower levels of precise localisation (IoU of 0.30), on the other hand, Faster R-CNN had accurate inference time

(IoU of 87.60), but slower inference times that could be an issue with consumers using us for real-time applications who might have their own limitations. Plus, we used a select few crops (e.g., apples, grapes, potatoes) & diseases, so it may only provide a small view of expensive global agricultural challenges. Moreover, the model has potential resource implications in its own right, such as having to hold an NVIDIA Tesla T4 GPU and/or the scenario of a resource constrained (small scale) farmer regaining adequate levels of agriculture without that level of infrastructure. Lastly, some diseases such as Grape Leaf Blight performed poorly, and they will require further optimization to develop learning algorithms that can correctly classify complicated or overlapping symptoms.

6.3 Future Work

Given some of the limitations described above and to build on the findings of this study, there are a number of potential future directions of work. First; with a richer dataset, especially one including field-collected images that accurately represented the full extent of environmental conditions (e.g. lighting, weather, backgrounds), we could build a more generalizable model. Second; we would open the door for a hybrid method - potentially using YOLOv8's speed, but taking advantage of Faster R-CNN's accuracy, or we could tune hyperparameters to improve YOLOv8's localization accuracy with a focus on speed. Third; if we could extend this study (and/or develop a new study), to provide an even greater representation of a number of crops and diseases, there would be more opportunity for practical application in relation to emerging pathogens associated with climate change, etc. Fourth; if we were to provide access of available technologies (mobile, IoT, etc.) to the public, we could democratize the ultimate use of BIET, allowing small scale farmers to be able to utilize BIET in a manner comparable with larger multinational corporations in providing access to real-time diagnostic tools; this could be enhancement, exhibiting even more of an influence if it were to be combined with field based assessments and freely available satellite-based images, drone-based data or any combination of the two. The ability to provide analysis and monitoring at a whole new scale would be of significant value. Lastly, future research could look at more complex techniques (i.e., attention mechanisms, ensemble modeling) to better overall models related to finding the more problematic diseases, ensuring a degree of reliability and provide a more overall approach to discovering all these diseases. All the above objectives will assist to enhance the possible impact of the completely autonomous response system and provide action-based response-systems that contribute to sustainable and secure global food systems that are progressively complex.

References

[1] *D. P. Hughes and M. Salathe, "An open access repository of images on plant health to enable the development of mobile disease diagnostics", 2015, arXiv:1511.08060.*

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", in Proc. IEEE Conf. Comput. Vis.Pattern Recognit. (CVPR), Jun. 2016, pp. 779-788.

[3] K. Kiratiratanapruk, P. Temniranrat, A. Kitvimonrat, W. Sinthupinyo, and S. Patarapuwadol, "Using deep learning techniques to detect rice diseases from images of rice fields," in Proc. Int. Conf. Ind., Eng. Appl. Appl. Intell. Syst. Cham, Switzerland: Springer, 2020, pp. 225-237

[4] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, arXiv:1804.02767.

[5] Qi, J. et al. An improved YOLOv5 model based on visual attention mechanism: Application to recognition of tomato virus disease. Comput. Electron. Agric. 194, 106780 (2022)

[6] Jing, J. et al. A tomato disease identification method based on leaf image automatic labeling algorithm and improved YOLOv5 model. J. Sci. Food Agric. 103, 7070-7082 (2023).

[7] Woo, S., Park, J., Lee, J. Y., & Kweon, I. S. Cbam: Convolutional block attention module. Proc. of the European Conference on computer vision (ECCV) 3-19 (2018).

[8] Li, S., Li, K., Qiao, Y. & Zhang, L. A multi-scale cucumber disease detection method in natural scenes based on YOLOv5. Comput. Electron. Agric. 202, 107363 (2022).

[9] Sun, W. et al. Veg-DenseCap: Dense captioning model for vegetable leaf disease images. Agronomy 13(7), 1700 (2023)

[10] Jayme Garcia Arnal Barbedo, "A review on the main challenges in automatic plant disease identification based on visible range images" Embrapa Agricultural Informatics, Av. Andr e Tosello, 209 e C.P. 6041, Campinas, SP, 13083-886, Brazil

[11] Kamlesh Golhani, Siva K. Balasundrama, Ganesan Vadamalai and Biswajeet Pradhan, "A review of neural networks in plant disease detection using hyperspectral data" INFORMATION PROCESSING IN AGRICULTURE 5 (2018) 354–371

[12] Kawasaki Y., Uga H., Kagiwada S. and Iyatomi H., "Basic Study of Automated Diagnosis of Viral Plant Diseases Using Convolutional Neural Networks." In: Bebis G. et al. (eds) Advances in Visual Computing. ISVC 2015. Lecture Notes in Computer Science, vol 9475. Springer, Cham

[13] Bin Liu, Yun Zhang, DongJian He and Yuxiang Li "Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks" Symmetry 2018, 10(1), 11

[14] Yang Lu, Shujuan Yi, Nianyin Zeng, Yurong Liu and Yong Zhang "Identification of rice diseases using deep convolutional neural networks" Neurocomputing, 2017 – Elsevier

[15] Nadia Shakoor, Scott Lee and Todd C Mockler "High throughput phenotyping to accelerate crop breeding and monitoring of diseases in the field" Current opinion in plant biology,2017- Elsevier

[16] [Asheesh Kumar Singh, Baskar Ganapathysubramanian, Soumik Sarkar, Arti Singh "Deep Learning for Plant Stress Phenotyping: Trends and Future Perspectives" Trends in Plant Science Volume 23, Issue 10, October 2018, Pages 883-898 – Elsevier

[17] Bikash Chandra Karmokar, Mohammad Samawat Ullah, Md. Kibria Siddiquee and Kazi Md. Rokibul Alam "Tea Leaf Diseases Recognition using Neural Network Ensemble"International Journal of Computer Applications (0975 - 8887) Volume 114 - No. 17, March 2015

[18] Nguyen Cao Tri, Tran Van Hoai, Hieu N. Duong, Nguyen Thanh Trong, Vo Van Vinh and Vaclav Snasel "A Novel Framework Based on Deep Learning and Unmanned Aerial Vehicles to Assess the Quality of Rice Fields" (eds) Advances in Information and Communication Technology. ICTA 2016. Advances in Intelligent Systems and Computing, vol 538.

[19] X.-P. Fan, J.-P. Zhou, and Y. Xu, `Recognition of field maize leaf diseases based on improved regional convolutional neural network,` J. South China Agricult. Univ., vol. 41, no. 6, pp. 82-91, Jun. 2020

[20] A. Fuentes, S. Yoon, S. Kim, and D. Park, "A reliable and deep-learning-based real-time tomato plant diseases and pests detector," Sensors, vol. 17, no. 9, p. 2022, Sep. 2017.

[21] A. F. Fuentes, S. Yoon, J. Lee, and D. S. Park, "Deep neural network-based tomato plant diseases and pests diagnosis system with a high performance and a filtering refinement bank," Frontiers Plant Sci., vol. 9, Aug. 2018, Art. no. 1162.

[22] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, "Real-time detection of apple leaf diseases: A deep learning approach based on improved convolutional neural networks," IEEE Access, vol. 7, pp. 59069–59080, 2019.

[23] J.-H. Li, L.-J. Lin, and K. Tian, "Detection of leaf diseases of balsam pear in field condition based on improved faster R-CNN," Trans. Chin. Soc. Agricult. Eng., vol. 36, no. 12, pp. 179–185, Jun. 2020.

[24] X.-R. Li, S.-Q. Li, and B. Liu, "Detection method of apple leaf diseases based on improved faster R-CNN," Comput. Eng., vol. 46, no. 11, pp. 59–64, Nov. 2020.

[25] M. M. Ozguven and K. Adem, "Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms," Phys. A, Stat. Mech. Appl., vol. 535, Dec. 2019, Art. no. 122537.

[26] M. M. Ozguven and K. Adem, "Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms", Physica A: Statistical Mechanics and Its Applications, 535, 122537, 2019. https://doi.org/10.1016/j.physa.2019.122537

[27] C. Song, C. Wang and Y. Yang, "Automatic Detection and Image Recognition of Precision Agriculture for Citrus Diseases", The IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), pp 187–190, 2020. https://doi.org/10.1109/ECICE50847.2020.9301932

[28] A. Shill and Md. A. Rahman, "Plant Disease Detection Based on YOLOv3 and YOLOv4," 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), pp 1–6, 2021. https://doi.org/10.1109/ACMI53878.2021.9528179

[29] M. P. Mathew and T. Y. Mahesh, "Leaf-based disease detection in bell pepper plant using YOLO v5," Signal, Image and Video Processing, vol. 16, no. 3, pp 841–847, 2022. https://doi.org/10.1007/s11760-021-02024-y

[30] Z. Zhang, Y. Qiao, Y. Guo and D. He, "Deep Learning Based Automatic Grape Downy Mildew Detection," Frontiers in Plant Science, 13, 872107, 2022. https://doi.org/10.3389/fpls.2022.872107

[31] J. Li, Y. Qiao, S. Liu, J. Zhang, Z. Yang and M. Wang, "Improved YOLOv5 based Vegetable Disease Detection Method", Computers and Electronics in Agriculture, 202, 107345, 2022. https://doi.org/10.1016/j.compag.2022.107345

[32] H. Wang, S. Shang, D. Wang, X. He, K. Feng and H. Zhu, "Plant Disease Detection and Classification Method Based on the Optimized Lightweight YOLOv5 Model", Agriculture, vol. 12, no. 7, Article 7, 2022. https://doi.org/10.3390/agriculture12070931

[33] D. Li, R. Wang, C. Xie, L. Liu, J. Zhang, R. Li, F. Wang, M. Zhou, and W. Liu, "A recognition method for agricultural video detection of rice plant diseases and pests based on deep convolutional neural network," Sensors, vol. 20, no. 3, Jan. 2020, Art. no. 578

[34] W. Zhen, Z. Shanwen, and Z. Baoping, "Crop Diseases Leaf Segmentation Method Based on Cascade Convolutional Neural Network," Comput. Eng. Appl., vol. 56, no. 15, pp. 242-250, August, 2020.

[35] "Adaptive Fusion of Multi-Scale YOLO for Pedestrian Detection," Aug 2021.

[36] I. T. F. C. A. B. Aduen Benjumea, "YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles," 3 January 2023.

[37] N. M. A. A. Dazlee, "Object Detection for Autonomous Vehicles with Sensor based Technology using YOLO," 30 March 2023.

[38] S. L. Wu, "Edge YOLO: A Real-Time Intelligent Object Detection System Based on Edge-Cloud Cooperation in Autonomous Vehicles," May 2022.

[39] V. M. Shivanna, "Object Detection, Recognition, and Tracking Algorithms for ADASs—A Study on Recent Trends," 20 December 2023.

[41] A. K. ,. V. G. Shubham Shinde, "YOLO based Human Action Recognition and Localization," 2018.

[42] T. C. ,. H. L. Hui Ma, "FER-YOLO: Detection and Classification Based on Facial Expressions," 6 August 2021.

[43] Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. Comput. Electron. Agric. 2019, 157, 417–426.

[44] Wu, D.; Lv, S.; Jiang, M.; Song, H. Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. Comput. Electron. Agric. 2020, 178, 105742.

[45] Wang, Y.; Zheng, J. Real-time face detection based on YOLO. Proceedings of the 2018 1st IEEE International Conference on knowledge innovation and Invention (ICKII), Jeju, Republic of Korea, 23–27 July 2018; pp. 221–224.

[46] Chen, W.; Huang, H.; Peng, S.; Zhou, C.; Zhang, C. YOLO-face: A real-time face detector. Vis. Comput. 2021, 37, 805–813.

[47] Tan, L.; Huangfu, T.; Wu, L.; Chen, W. Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification. BMC Med. Inform. Decis. Mak. 2021, 21, 1–11.

[48] Ünver, H.M.; Ayan, E. Skin lesion segmentation in dermoscopic images with combination of YOLO and grabcut algorithm. Diagnostics 2019, 9, 72.

[49] Al-Masni, M.A.; Al-Antari, M.A.; Park, J.M.; Gi, G.; Kim, T.Y.; Rivera, P.; Valarezo, E.; Choi, M.T.; Han, S.M.; Kim, T.S. Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system. Comput. Methods Programs Biomed. 2018, 157, 85–94.

[50] Nie, Y.; Sommella, P.; O'Nils, M.; Liguori, C.; Lundgren, J. Automatic detection of melanoma with yolo deep convolutional neural networks. In Proceedings of the 2019 E-Health and Bioengineering Conference (EHB), Iasi, Romania, 21–23 November 2019; pp. 1–4.

[51] Cheng, L.; Li, J.; Duan, P.; Wang, M. A small attentional YOLO model for landslide detection from satellite remote sensing images. Landslides 2021, 18, 2751–2765.

[52] Pham, M.T.; Courtrai, L.; Friguet, C.; Lefèvre, S.; Baussard, A. YOLO-Fine: One-stage detector of small objects under various backgrounds in remote sensing images. Remote Sens. 2020, 12, 2501.

[53] Qing, Y.; Liu, W.; Feng, L.; Gao, W. Enhanced Yolo network for free-angle remote sensing object detection. Remote Sens. 2021, 13, 2171.

[54] Zakria, Z.; Deng, J.; Kumar, R.; Khokhar, M.S.; Cai, J.; Kumar, J. Multiscale and direction target detecting in remote sensing images via modified YOLO-v4. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 2022, 15, 1039–1048.

[55] Kumar, P.; Narasimha Swamy, S.; Kumar, P.; Purohit, G.; Raju, K.S. Real-Time, YOLO Based Smart Surveillance and Monitoring System using Jetson TX2. In Data Analytics and Management: Proceedings of ICDAM; Springer: Berlin/Heidelberg, Germany, 2021; pp. 461–471.

[56] Bhambani, K.; Jain, T.; Sultanpure, K.A. Real-time face mask and social distancing violation detection system using Yolo. In Proceedings of the 2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC), Viyijapur, India, 8–10 October 2020; pp. 1–6.

[57] Li, J.; Su, Z.; Geng, J.; Yin, Y. Real-time detection of steel strip surface defects based on improved yolo detection network. IFAC-PapersOnLine 2018, 51, 76–81.

[58] Ukhwah, E.N.; Yuniarno, E.M.; Suprapto, Y.K. Asphalt pavement pothole detection using deep learning method based on YOLO neural network. In Proceedings of the 2019 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 28–29 August 2019; pp. 35–40.

[59] Du, Y.; Pan, N.; Xu, Z.; Deng, F.; Shen, Y.; Kang, H. "Pavement distress detection and classification based on YOLO network." International Journal of Pavement Engineering 2021, 22: 1659-1672.

[60] Chen, R.C. "Automatic license plate recognition via sliding-window darknet-YOLO deep learning." Image and Vision Computing 2019, 87: 47-56.

[61] Dewi, C.; Chen, R.C.; Jiang, X.; Yu, H. "Deep convolutional neural network for enhancing traffic sign recognition developed on Yolo V4." Multimedia Tools and Applications 2022, 81: 37821-37845.

[62] Roy, A.M.; Bhaduri, J.; Kumar, T.; Raj, K. "WilDect-YOLO: an efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection." Ecological Informatics 2023, 75: 101919.

[63] Kulik, S.; Shtanko, A. "Experiments with neural net object detection system YOLO on small training datasets for intelligent robotics." In Advanced Technologies in Robotics and Intelligent Systems: Proceedings of ITR 2019; (Springer: Berlin/Heidelberg, Germany, 2020), pp. 157-162.

[64] Dos Reis, D.H.; Welfer, D.; D. De Souza Leite Cuadros, M.A.; Gamarra, D.F.T. "Mobile robot navigation using object recognition software with RGBD images and using YOLO algorithm" Applied Artificial Intelligence 2019, 33: 1290-1305.

[65] Sahin, O.; Ozer, S. Yolodrone: Improved Yolo architecture for object detection in drone images. In Proceedings of the 202144th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 26–28 July 2021;pp. 361–365.

[66] Chen, C.; Zheng, Z.; Xu, T.; Guo, S.; Feng, S.; Yao, W.; Lan, Y. YOLO-Based UAV Technology: A Review of the Research and ItsApplications. Drones 2023, 7, 190.

[67] Du, J. (2018, April). Understanding of object detection based on CNN family and YOLO. In Journal of Physics: Conference Series (Vol. 1004, No. 1, p. 012029). IOP Publishing.

[68] Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.

[69] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016. Supplied as additional material https://arxiv.org/pdf/1506.02640.pdf.

[70] Juan R. Treven and Diana M. Cordova-Esparaza. A comprehensive review of yolo: From yolov1 to yolov8 and beyond, 2023. Supplied as additional material https://arxiv.org/pdf/2304.00501.pdf.

[71] Ashraf, A.H.; Imran, M.; Qahtani, A.M.; Alsufyani, A.; Almutiry, O.; Mahmood, A.; Attique, M.; Habib, M. Weapons detection for security and video surveillance using CNN and YOLO-v5s. CMC-Comput. Mater. Contin. 2022, 70, 2761–2775.

[72] Jacob Solawetz and Francesco. What is yolov8? the ultimate guide., 2023. 04-30-2023.

[73] Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000

[74] Li, X.; Wang, W.; Wu, L.; Chen, S.; Hu, X.; Li, J.; Tang, J.; Yang, J. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. Adv. Neural Inf. Process. Syst. 2020, 33, 21002–21012.

[75] Ananth, S. (2019). Faster R-CNN for object detection. Towards Data Science.

[76] Abbas, S. M., & Singh, S. N. (2018, February). Region-based object detection and classification using faster R-CNN. In 2018 4th International Conference on Computational Intelligence & Communication Technology (CICT) (pp. 1-6). IEEE.

[77] Abbas, S. M., & Singh, S. N. (2018, February). Region-based object detection and classification using faster R-CNN. In 2018 4th International Conference on Computational Intelligence & Communication Technology (CICT) (pp. 1-6). IEEE.

[78] Arivazhagan, S., Shebiah, R. N., Ananthi, S., & Varthini, S. V. (2013). Detection of unhealthy regions of plant leaves and classification of plant leaf diseases using texture features. Agricultural Engineering International: CIGR Journal, 15(1), 211-217.

Plagiarism

212-15-4120

ORIGINALITY REPORT

9%	5%	6%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	1%
2	creativecommons.org Internet Source	<1%
3	Lili Li, Shujuan Zhang, Bin Wang. "Plant Disease Detection and Classification by Deep Learning—A Review", IEEE Access, 2021 Publication	<1%
4	www.mdpi.com Internet Source	<1%
5	doctorpenguin.com Internet Source	<1%
6	V. Sharmila, S. Kannadhasan, A. Rajiv Kannan, P. Sivakumar, V. Vennila. "Challenges in Information, Communication and Computing Technology", CRC Press, 2024 Publication	<1%
7	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Intelligent Computing and Communication Techniques - Volume 3", CRC Press, 2025 Publication	<1%
8	Juan Terven, Diana-Margarita Córdova-Esparza, Julio-Alejandro Romero-González. "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS", Machine Learning and Knowledge Extraction, 2023 Publication	<1%