Th_ECE: 19_24

A Lightweight YOLO-Based Model for UAV-Assisted Rice Leaf Disease Detection in Precision Farming

Asifur Rahman Mridul

Roll: 1909024

September, 2025

Department of Electronics and Communication Engineering

Khulna University of Engineering & Technology

Khulna- 9203, Bangladesh

A Lightweight YOLO-Based Model for UAV-Assisted Rice Leaf Disease Detection in Precision Farming

This thesis is submitted to the Department of Electronics and Communication Engineering, Khulna University of Engineering & Technology, in partial fulfillments for the requirements of the degree of

"Bachelor of Science in Electronics and Communication Engineering"

Supervised by

Dr. Md. Foisal Hossain
Professor,
Department of Electronics and Communication Engineering,
KUET, Khulna.

Submitted by

Asifur Rahman Mridul
Roll: 1909024
Department of Electronics and Communication Engineering,
KUET, Khulna.

September, 2025

Department of Electronics and Communication Engineering

Khulna University of Engineering & Technology

Khulna- 9203, Bangladesh

A Lightweight YOLO-Based Model for UAV-Assisted Rice Leaf Disease Detection in Precision Farming

Author

Asifur Rahman Mridul
Roll: 1909024
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

Supervisor

Dr. Md. Foisal Hossain
Professor
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

…….

Signature

External

Mr. Md. Khorshed Alom
Assistant Professor
Dept. of Electronics and Communication Engineering
Khulna University of Engineering & Technology

……

Signature

September, 2025

Department of Electronics and Communication Engineering

Khulna University of Engineering & Technology

Khulna- 9203, Bangladesh

Statement of Originality

I myself Asifur Rahman Mridul, student of the Department of Electronics and Communication Engineering of Khulna University of Engineering and Technology (KUET), Khulna, Bangladesh; I do hereby declare that this thesis has not been published anywhere in full, or in part. All items, or sub items that appear on the thesis are referenced, where needed. The thesis cannot be copied without the permission of the author. Also, this thesis has not submitted earlier as an undergraduate thesis.

The above declarations are true. Understanding these, this work has been submitted for evaluation of an undergraduate thesis.

Date: Author

Acknowledgement

Firstly, I would like to express my deepest gratitude to the Almighty Allah SWT for His infinite blessings, guidance, and mercy, which gave me the strength, patience, and perseverance to complete this thesis work.

I am profoundly grateful to my supervisor, **Dr. Md. Foisal Hossain, Professor, Department of ECE, KUET,** for his continuous guidance, invaluable suggestions, and constant encouragement to complete this work.

I would also like to extend my sincere appreciation to my external examiner, **Mr. Md. Khorshed Alom, Assistant Professor, Department of ECE, KUET,** for his valuable time, insightful comments, and constructive evaluation of this thesis.

My heartfelt thanks go to **my family** for their unconditional love, support, and encouragement, which have been the backbone of my academic journey.

I am also grateful to my respected teachers, lab assistants, and the staff of the Department of ECE, KUET, for their kind cooperation and support.

Abstract

Rice is the main food item in Bangladesh. However, rice cultivation is significantly affected by various diseases such as bacterial blight, rice blast, and brown spot. These result in a significant reduction in yield as well as quality. In order to address these challenges, early, automated, and efficient detection systems are needed. For this purpose, systems using Unmanned Aerial Vehicles (UAVs) combined with Artificial Intelligence (AI) are becoming crucial for prompt responses and sustainable crop management. Some of the existing works, such as UAV T-YOLO-Rice architecture, has shown good results in rice disease detection. However, it still suffers from high computational cost and limited detection accuracy, which limits its use for real-time UAV deployment. Therefore, there is a strong need for lightweight yet high-accuracy deep learning models that can operate efficiently on UAV platforms for practical field applications. In this work, we develop a lightweight YOLOv11n-based model by scaling depth, width, and maximum channels to reduce model parameters. The work aims to improve the computational efficiency and detection accuracy by using lightweight YOLOv11n models trained through the rice disease dataset which were used in UAV T-YOLO-Rice. The proposed YOLOv11n model having only 0.81M parameters achieve a higher testing mean average precision (mAP) as 94.7% which outperforms many other models from previous study.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

YOLO: You Only Look Once

UAV: Unmanned Aerial Vehicle

GDP: Gross Domestic Products

IoT: Internet of Things

CAGR: Compound Annual Growth Rate

GFLOPs: Giga Floating Point Operations Per Second

CNN: Convolutional Neural Network

CBAM: Convolutional Block Attention Module

SFCM: Spatial Feature Calibration Module

RFB: Receptive Field Block

PMA: Pooling by Multihead Attention

SVM: Support Vector Machine

SPP: Spatial Pyramid Pooling

EIoU: Efficient Intersection over Union

BiFPN: Bidirectional Feature Pyramid Network

C2f_MSEC: Cross Stage Partial Two-flow Multi-Scale Efficient Convolution

DCNv2: Deformable Convolutional Network v2

CSP: Cross Stage Partial

LSCSBD: Lightweight Shuffle Cross-Stage Bottleneck with Depthwise

RFCA-CSP: Receptive Field Cross-Attention Cross Stage Partial

SCSA: Spatial Channel Self-Attention

CARAFE: Content-Aware ReAssembly of FEatures

SPPF: Spatial Pyramid Pooling – Fast

C2PSA: Cross Stage Partial Spatial Attention

C3k2: Cross Stage Partial Bottleneck (3 layers, kernel size 2)

C3k: Cross Stage Partial Bottleneck (3 layers)

GPU: Graphics Processing Unit

mAP: mean Average Precision

IoU: Intersection over Union

CHAPTER 1

INTRODUCTION

1.1 Overview

Rice is a primary crop in most countries and it is essential to global food security. But its yield is badly affected by diseases such as bacterial leaf blight, blast, and brown spot. Traditional detection methods are often slow, and they require huge manpower. For this reason, the automated solutions are crucial. Fortunately, advancements in computer vision and deep learning, particularly lightweight YOLO models, enable fast and accurate disease detection using UAV devices. This study reduces the total parameters of YOLOv11 to get a computationally efficient model to deploy on a UAV-based system for detecting major rice diseases, and supporting precision agriculture and sustainable production.

1.2 Background and Motivation

In Bangladesh, approximately 72 % of the land area is devoted to agriculture [1]. Despite this extensive use, agriculture contributes only 11.16 % of GDP (2024) [2], while employing about 35-40 % of the total population [3]. These statistics indicates that although most of the lands and populations are engaged with agriculture, productivity gains are not keeping pace with the rise in area or investment.

This imbalance between huge amount of land use and limited productivity highlights the significance of smarter, technology-driven approaches in agricultural field in Bangladesh. One of the major bottlenecks in improving crop productivity is the timely and accurate detection of plant diseases. This result in reducing yield quality and quantity. Traditional methods are often inadequate to detect rice diseases in large-scale fields.

Three major diseases of rice leaves are:

***Bacterial Leaf Blight (BLB)*:** Bacterial Leaf Blight, shown in Figure 1.1, is one of the major rice diseases caused by a type of bacteria called Xanthomonas oryzae pv. Oryzae [4]. It usually starts as wet-looking streaks on the tips or edges of the leaves. These streaks turn yellow and then become straw-colored as the disease gets worse. In the early morning, a sticky substance may come out of the infected leaves. If the disease attacks young plants, they may dry out and die completely. This disease can reduce rice yield by 6% to 60% [5]. BLB lowers the number of healthy panicles, makes the grains lighter, and causes the grains empty or poorly filled. Which results in reducing both the amount and quality of rice produced.

Figure 1.1: Bacterial leaf blight disease [6]

Blast: Blast, shown in Figure 1.2, is a harmful rice disease. It is caused by a fungus called *Magnaporthe oryzae* [7]. This disease can attack rice plants at any stages. It affects the leaves, stems, and panicles. It starts as small grey-green spots that turn into spindle-shaped spots with white centers and brown edges. When it attacks the neck of the panicle (neck blast), the panicle dries out, leaving "white heads" that produce empty grains. Blast usually causes 10%-30% yield loss, but during severe outbreaks it can destroy up to 80% of the crop [8]. Because it spreads quickly and damages many parts of the plant, blast is a big challenge for rice farmers and a major cause of lost production.

Figure 1.2: Rice blast disease [6]

***Brown Spot*:** Brown spot, shown in Figure 1.3, is a deadly rice disease caused by a fungus called *Cochliobolus miyabeanus* [9]. This plant disease majorly damages the leaves of the rice plants but also can affect to stems and panicles. It starts off as small, round, moist-looking spots that eventually darken into brown spots with dark edges. The additional progression of the disease causes these lesions to increase in size creating larger patches of necrotic leaf tissue which could really harm the leaf tissue. The photosynthetic capacity is Reduced by brown spot. This results in stunted growth and less grain produced. It is known to cause up to 50% yield loss in severe cases making it a major constraint to rice production [10]. And in the worst

circumstance of hot dry climate, the problematic disease is a serious threat in warm wet soil condition, thus preventing rice from growing happily ever after.

Figure 1.3: Brown spot disease [6]

The smart agriculture market is booming globally as more farmers demand solution-oriented, technology-driven farming practices. Tools such as the Internet of Things (IoT), Unmanned Aerial Vehicles (UAVs), and AI-based analytics, will be critical for addressing issues of food security, resource utilization, and climate resilience. The topic of this thesis is thus the design, development and evaluation of lightweight YOLO-based models for rice disease detection using UAVs, within this context.

The smart agriculture market was valued at USD 25.36 billion in 2024 and will reach USD 54.71 billion by 2030, growing at a compound annual growth rate (CAGR) of 13.9% from 2025 to 2030 according to the grand view research [11].

Unmanned aerial vehicles (UAVs) have proved their capability in the field of agricultural monitoring since they can capture high-resolution imagery of large fields in a short time. Coupled with deep learning models, such systems mounted on UAVs can facilitate early warning and interventive precision targeting. For instance, when accomplish UAV deployment, the hardware of each UAV imposes hardware constraints related to onboard memory, processing capacity, and power consumption. Hence, the models being used should not only be precise but also light-weight and able to work with low computation power. Although there have been some deep learning-based rice disease detection studies, most of them use outdated architectures or limited scenarios. A comparative analysis of YOLO's latest versions addressing this trade-off of accuracy and efficiency in real-time applications remains to be explored. To bridge this gap, this study extensively investigates using and compares the

performance and computational efficiency of lightweight YOLOv8n and YOLOv11 for detecting three major rice diseases.

1.3 Objectives

- **To reduce computational complexity** by designing lightweight rice disease detection model that require fewer parameters and lower computational resources, making it suitable for deployment on UAV devices.
- **To improve mean average precision (mAP)** by enhancing the model's ability to accurately detect and classify rice diseases, even under challenging field conditions such as varying light, scale, and background.
- **To minimize inference time** so that the model can perform real-time detection, enabling quick decision-making for timely disease management in rice fields.

1.4 Research Methodology

This study followed five main steps. Including these five major steps, a flowchart of the research methodology is shown in Figure 1.4.

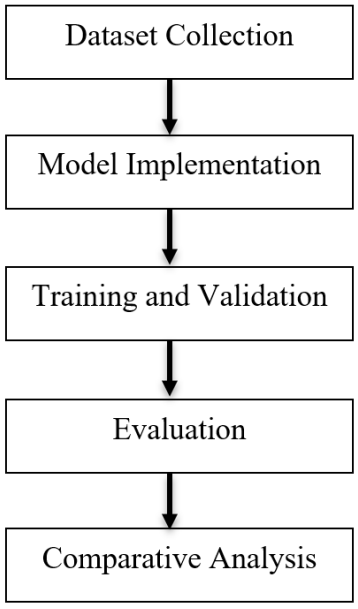

Figure 1.4: Flowchart of research methodology

1.4.1 Dataset Collection

A large dataset is required to create a deep learning model for object detection. Instead, this research used the same dataset that was used in training the UAV T-Yolo Rice network; this dataset has more than 15000 images and their corresponding labelled files.

1.4.2 Model Implementation

YOLOv8n and YOLOv11n were implemented with reduced parameters using Ultralytics framework. Both models were trained with decreased parameters to improve computational efficiency for UAV deployment.

1.4.3 Training and Validation

Models were trained with optimized hyperparameters e.g., batch size, image size, optimizer, initial learning rate, final learning rate and using GPU acceleration. Early stopping was applied to enhance the model performance.

1.4.4 Evaluation

The performance of YOLOv8n and YOLOv11n was evaluated based on both the evaluation metrics (e.g., precision, recall, f1 score, and mAP) and computational efficiency measures (total parameters, GFLOPs, inference time, and model size). Together, these metrics provided a comprehensive understanding of the trade-offs between detection accuracy and real-time ability to deploy in a UAV-based rice disease monitoring system.

1.4.5 Comparative Analysis

The evaluation results of differently scaled YOLOv8n and YOLOv11n models were compared to analyze the trade-off between detection accuracy and computational efficiency. This comparison enabled the choice of the most suitable model for UAV-based deployment.

1.5 Project Planning

The project planning was conducted using a Gantt chart. The project planning covers both the 1st and 2nd terms of the final year to complete this work.

1.5.1 Work Plan – Gantt Chart

Tables 1.1 shows the work plan through a Gantt chart. This work started on February, 2024 and finished on September 2025.

Table 1.1: Gantt chart of thesis work

Task Name	1st Term							2nd Term						
	Feb '24	Mar '24	Apr '24	May '24	Jun '24	Sep '24	Oct '24	Nov '24	Dec '24	Jan '25	Feb '25	Jul '25	Aug '25	Sep '25
Prepare Plan														
Research Literature														
Problem Finding														
Prepare Draft Budget														
Prepare and Submit Project Proposal														
Learn Python, Pandas														
Learn Basic ML, DL														
System Design														
Prepare and Present 1st Term Progress														
Study on Different YOLO Models														
Design Lightweight Model														
Training Models														
Testing and Evaluations														
Final Report														
Final Presentation														

1.6 Project Budget

The project budget was estimated at the beginning of the work. And it was revised at the end of the work. This section discusses the overall budget at the end of the work.

1.6.1 Overall Budget

The overall budget, shown in Table 1.2, is much less than that of estimated earlier. The final budget reduced because, free cloud-based GPU of Kaggle was used to train and evaluate the model.

Table 1.2: Overall budget of thesis work

SL NO	Item	Justification	Price (BDT)
01	Printing and Binding	Thesis Book and Report	**1500**
02	Software and Tools	Quillbot and Grammarly	**400**
Total (BDT)			**1900**
In Words: Nineteen Hundred Taka Only			

1.7 Organization of the Report

The thesis is organized as follows:

Chapter 2

The chapter includes a historical overview and past works related to this thesis.

Chapter 3

The chapter includes a theoretical overview of the model architecture.

Chapter 4

The chapter includes information regarding the implementation & necessary parameters that have been used for implementation of the model.

Chapter 5

Performance analysis and evaluation of various parameters have been described in this chapter.

Chapter 6

Impact of the model on societal, health, legal, cultural issues & impact on the environment and sustainability has been discussed in this chapter.

Chapter 7

In this chapter, complex engineering problems & activities has been addressed.

Chapter 8

The conclusion, limitation and future works are described in this chapter.

CHAPTER 2

LITERATURE REVIEW AND THEORETICAL BACKGROUND

2.1 Introduction

The evolution of both computer vision and deep learning technology has enabled researchers and practitioners to develop automated, accurate, and rapid methods for detecting plant diseases. Convolutional Neural Networks (CNNs) are a crucial technique in the field of deep learning and science, as various applications have demonstrated outstanding performance in agricultural settings [12]. Similarly, object detection models, especially YOLO (You Only Look Once) family models, reached the state-of-the-art performance [13], [14], [15]. Moreover, UAVs' image acquisition helps in monitoring fields at a broader scale with less human interference, making it an area of precision agriculture practices.

However, there are still a few obstacles to overcome. Disease detection becomes challenging due to the relatively small size of the objects captured using UAV images, which is compounded by varying illuminations, background clutter, and motion blurring. Additionally, the need for lightweight architectures with lower computational overhead and size, without sacrificing accuracy, is crucial in the deployment of deep learning models using UAV-based edge devices. As such, a wealth of research has emerged that aims to address these challenges, such as enhancing YOLO-based models with attention mechanisms (for example, CBAM, PMA) and sophisticated feature extraction modules (for example, SFCEM, RFB), to achieve better detection performance [13], [16], [17].

In this chapter, a broad literature review on rice disease detection in UAVs is provided. The review is organized to include the recent evolution in object detection models, new lightweight models within the YOLO family, attention mechanisms, and feature modules, as well as their fusion, making real-time UAV deployment possible. The aim is to point the key trends, summarize existing solutions and determine research gaps that justify development of the proposed lightweight YOLOv11 architecture.

2.2 Evolution of Computer Vision in Agronomy

Computer vision in agriculture has gone through different stages of development. In the early years, from the 1980s to the 2000s, researchers used basic image processing methods [18]. These include thresholding, edge detection, and color analysis. This kind of methods helped in checking the color and shape of leaves or counting crops from aerial photos.

However, these methods were not very reliable in real field conditions. They were sensitive to changes in light, background, and noise.

From the 2000s to around 2015, machine learning became popular. During this period, researchers used hand-designed features such as texture, color patterns, and shapes, along with classifiers like Support Vector Machines (SVM) or Random Forests [19], [20]. These methods improved tasks like weed detection, crop quality checking, and disease identification from leaf images. Still, they required a lot of manual feature design and did not work well across different crops or environments.

After this period, deep learning changed this field completely. Convolutional Neural Networks (CNNs) and transfer learning allowed models to learn features directly from images without manual design [21], [22]. This method made the crop disease detection, weed recognition, pest monitoring, and yield prediction much more accurate. UAVs and drones also became important, as they provided large-scale image data for monitoring crop growth and field health. The main challenge in this stage has been the high need for computational power and large labeled datasets.

In recent years, computer vision in agronomy has been moving toward more advanced directions. Lightweight models such as Tiny-YOLO and MobileNet are used for drones and edge devices to make real-time detection possible [13], [23], [24].

There are many deep learning models. Among them the YOLO (You Only Look Once) family has become very popular in agriculture. YOLO is fast and accurate, which makes it suitable for real-time applications in the field. Different versions of YOLO, such as YOLOv5, YOLOv8, and the most recent YOLOv11, are widely used for crop disease detection, weed identification, fruit counting, and pest monitoring [9], [25], [26]. YOLO models are compatible with drones and lightweight devices. And they are ideal for precision farming tasks that need quick decisions. YOLO has therefore become one of the most important tools for computer vision in agronomy today.

2.3 Requirements of Automated Rice Diseases Detection

For automated rice disease detection to become a reliable system, several crucial prerequisites must be met. The first requirement is large dataset. Images of rice collected at different aspects, lighting, and growth stages should be sufficiently clear. Image augmentation

with different noises such as, Gaussian noise, uniform noise, image filtering, and saturation and adjusting the brightness conditions like brightness reduction and enhancement can improve the robustness of the model [13], [27]. This type of image dataset enables the system to have a clear picture of real variation within the field. Reliable models also need to train on large datasets with a multitude of healthy and diseased samples.

Second, the system will require powerful algorithms for image processing and analysis. However, the advantages of deep learning models, specifically object detection models such as YOLOv11, are that they can detect and identify diseases in real time [15], [26]. They are used with caution to train models that can detect various rice diseases with high accuracy.

The system should be efficient and lightweight. It should be fast and memory-efficient because it may run on a drone or UAV in the field with constraints on power and memory as well. Relevant models must therefore strike a balance between fidelity and computational expense.

Ultimately, it should be an effective system in real-world settings. In rice fields, light, wind, water, and overlapping leaves are constantly evolving. The model should be able to deal with these variations and should not lose accuracy.

Last but not least, the installation needs to be simple enough for farmers to operate the system themselves. The outputs should be clear, specifying the disease detected and its location in the field, to enable farmers to take immediate management actions.

2.4 Classification of Rice Diseases Detection

Rice disease detection can be divided into different types based on how the system works. One common type is image classification. In this method, the whole image of a rice leaf is given to the model, and the model predicts whether the leaf is healthy or diseased [28]. If it is diseased, the model also predicts the type of disease. This method is simple but cannot show the exact location of the disease on the leaf.

Another type is object detection. In this method, the model not only identifies the disease but also draws a box around the affected area [29]. This helps in knowing where the disease is present. Models like YOLO are widely used for this type because they can detect diseases quickly and in real time [30].

There is also semantic segmentation. In this method, the model labels each pixel of the leaf image as healthy or diseased. This gives a very detailed view of the infected regions [31]. It is useful for measuring the diseased area. But this method needs more computation and larger datasets.

In general, image classification is used for simple detection, object detection is used for locating diseases in the field, and segmentation is used for detailed analysis. This work is focusing on object detection method to identify the diseases as well as the affected area which requires low computational cost using latest YOLO models.

2.5 Challenges in UAV-Based Rice Disease Detection

Rice disease detection itself is challenging to achieve using UAVs. One challenge is the quality of images. UAV captured images may be blurred or not clear due to wind or movement, or unstable flight. There are also shadows and weather conditions which affect light in our images.

The other is a confusing area field. The rice plants are grown at close distances, and the leaves sometimes overlap with each other. The sensor can be confused by distractions such as water, soil and weeds in the background. Due to this, healthy vs diseased areas are not clearly delineated.

The third challenge is the variance in disease. One single disease may present itself differently at different stages of growth. The symptoms also vary based on different lighting and environmental factors. And, collecting sufficient variation data is nearly impossible.

There are also technical challenges. Since UAVs have a limited battery life, storage, and computing power. The models have to be small enough to be able to run in real-time. Deep learning models need data and computation as well.

Finally, there are practical challenges. The advanced UAV systems can be complex for some farmers to use, if they even have access to them in the first place. In order for actual adoption to occur, the system needs to be inexpensive, easy to run and tailor simple output results to farmers.

2.6 Research gaps

After studying the theoretical background of the rice disease detection models, few research gaps are found that are summarized in a tabular form in Table 2.1.

Table 2.1: Research gap on recent studies

Year	Authors	Method(s)	Performance	Strengths	Research Gaps
2023	A. K. Sangaiah et al. [13]	Modified Tiny YOLOv4 with SPP, CBAM, SCFEM, Ghost module	mAP 86.97%	Small object detection, large dataset augmentation strategy	Not highest AP on all classes, Computational complexity is higher
2024	D. C. Trinh et al. [32]	YOLOv8 with proposed alpha-EIoU loss	mAP 89.9%	Improved loss function, good for small objects	Slightly lower AP on leaf folder and blast
2025	J. Wang et al. [14]	YOLOv8n with RepGhost, GhostConv	mAP 93.2%, Precision 88%, Recall 90.8%	High overall accuracy	Accuracy drops in large scale features
2024	L. Yang et al. [27]	YOLOv8n with C2f_MSEC, BiFPN	mAP 93.9%, Params 1.4M, Size 3.3MB	Small model size with high accuracy	Does not cover late-stage false smut; needs broader datasets
2024	Y. Lu et al. [25]	YOLOv8n with Triplet Attention, DCNv2, BiFPN	mAP 86.0%, Precision 82.3%, Recall 81.9%	Improved detection of rice leaf disease	Performance varies; some classes have lower recall/precision; needs larger datasets
2025	X. Jin et al. [33]	YOLOv8n with slim-neck, RFCA-CSP, lightweight shared head LSCSBD	mAP 86.0%, Precision 86.6%, Recall 81.4%, Params 1.93M	Good balance accuracy and efficiency; generalizes well; good for resource-limited scenarios	Slight precision and recall drop in generalization; still scope for fine-tuning
2025	K. Fang et al. [15]	YOLOv11n with SCSABlock, and CARAFE upsampling	mAP@0.5: 88.3%, Precision 91.6%, Recall 83.8%	Enhanced multi-scale semantic fusion, better small target detection, lightweight	Lower AP for Rice Blast due to limited samples and feature similarity

2.7 Summary

This chapter reviewed recent research on rice disease detection using UAV devices and YOLO-based object detection models. It highlighted key advancements in lightweight architectures, attention mechanisms, and feature extraction modules, along with challenges such as small-object detection and high computational cost. The identified gaps provide the motivation for designing the proposed lightweight YOLOv11 model.

CHAPTER 3

METYHODOLOGY

3.1 Introduction

This chapter covers the design of the proposed rice disease detection model. At first, it explains the preparation of the dataset. Dataset preparation includes class selection, data annotation, and augmentation techniques. The augmentation process provides a balanced and diverse dataset. Proper dataset preparation is important to achieve robust model.

The chapter then describes the design of the detection model. The YOLOv11 architecture is outlined at a high level. The chapter describes the backbone, neck, and head components of the model. Scaling strategies based on depth, width multiples, and maximum channel constraints are explained as well. This will highlight how a lightweight but accurate model is achieved.

3.2 Design Requirements

This section outlines the dataset requirements and model architecture considerations.

3.2.1 Rice Leaf Diseases Dataset

This work employed the rice leaf diseases dataset collected from an online repository (GitHub) [34]. The same dataset was used to train the UAV T-Yolo Rice model [13]. The dataset consists of images of rice leaves infected with three categories of diseases: Bacterial Leaf Blight, Rice Blast, and Brown Spot. The dataset contains the corresponding label files as well.

The original dataset was too small. It includes only:

- **Bacterial leaf blight**: 96 images
- **Rice blast**: 80 images
- **Brown spot**: 100 images

This work followed the data augmentation process of the UAV T-Yolo Rice network [13]. This process works nicely to improve the model's generalization by increasing the dataset's size and diversity. These techniques included:

- **Rotations**: 90°, 180°, and 270°, shown in Figure 3.1 (a).
- **Mirroring**: Horizontal and vertical flips, shown in Figure 3.1 (b).

- **Brightness adjustments**: Both enhancement and reduction, shown in Figure 3.1 (c).
- **Noise addition**: Gaussian, shown in Figure 3.1 (e), saturation, shown in Figure 3.1 (d), and uniform noise, shown in Figure 3.1 (g).
- **Image filtering**: To simulate low-quality images, shown in Figure 3.1 (f).

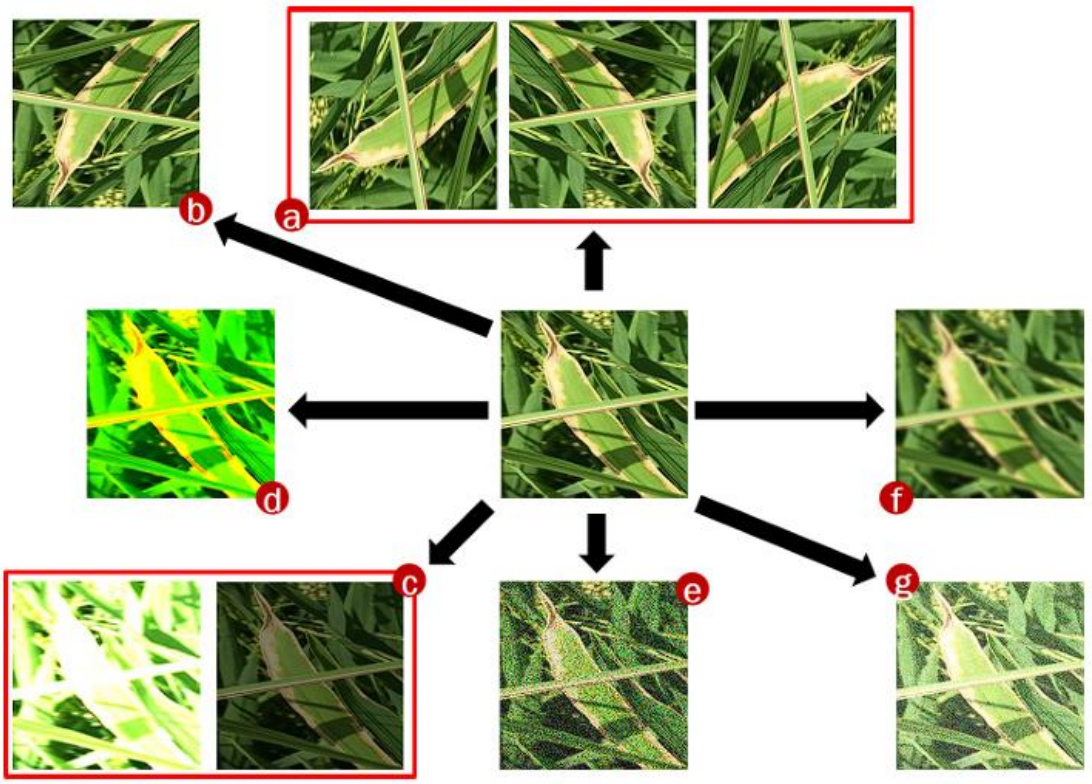

Figure 3.1: Image augmentations [13]

These augmentations significantly increased the dataset size and ensured better generalization of the model [13], [27]. Finally, a total of 15,456 images were obtained by expanding the current dataset.

Then the full dataset was split into three subsets:

- **Training set**: 70% of the samples
- **Validation set**: 20% of the samples
- **Testing set**: 10% of the samples

This dataset was crucial for training the lightweight YOLOv11 model, which was fine-tuned for detecting rice leaf diseases using the enhanced data.

3.2.2 YOLO (You Only Look Once)

YOLO is a state-of-the-art deep learning algorithm designed for real-time object detection. Traditional object detection models mainly apply a classifier to different regions of the image. But, YOLO treats object detection as a regression problem. YOLO does not perform separate steps for localization and classification. YOLO directly predicts bounding boxes and their associated class probabilities in a single forward pass of the network. This ability enables YOLO to detect objects in one step. Hence, the name is "You Only Look Once". This feature allows YOLO models to process images much faster than traditional methods [35].

YOLO divides an input image into a grid of cells. And each cell is responsible for predicting a fixed number of bounding boxes as well as their confidence scores. These confidence scores reflect how confident the model is that the predicted box contains an object, as well as the accuracy of the box's location. In addition to bounding boxes, each grid cell also predicts the class probabilities for the objects in the cell. By combining the outputs, YOLO is capable of detecting multiple objects in an image. The detected images may belong to different classes. All of the processes happen within a single step.

YOLO has become popular due to its exceptional speed and efficiency. These features make it suitable for real-time applications. Over the years, different versions of YOLO have been released. Each came with improved accuracy and speed. The latest version in the YOLO family is YOLOv11 [36].

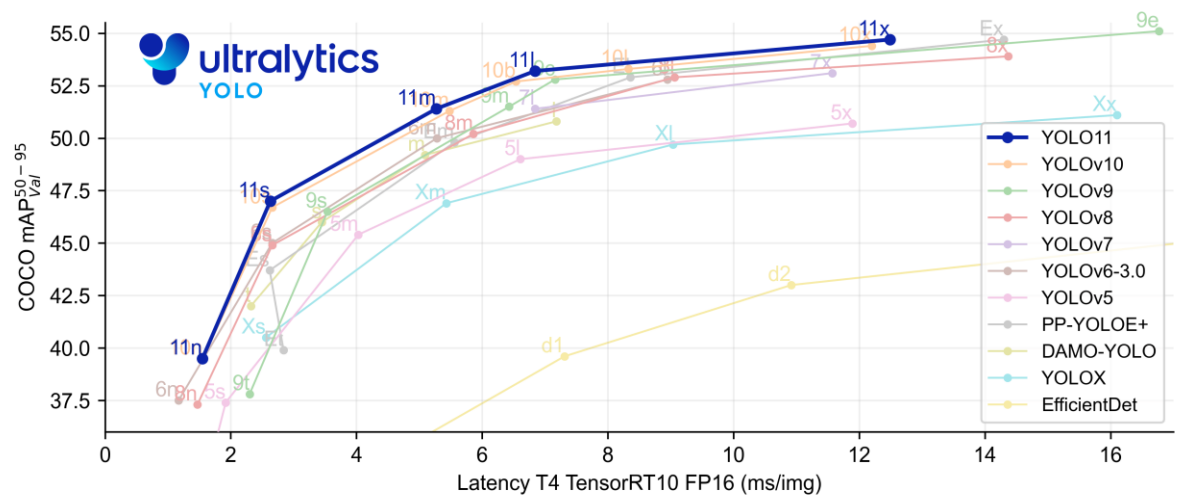

Figure 3.2: Accuracy-latency comparison between different Ultralytics YOLO models [36]

Figure 3.2 compares the accuracy (COCO mAP50-95) and latency (ms/img) of various object detection models running on T4 TensorRT10 GPUs with FP16 precision. It highlights the performance of various Ultralytics YOLO models. From the figure, it is clear that YOLOv11 achieves higher accuracy-latency trade-offs compared to the previous YOLO versions.

YOLO has some significant benefits, including:

- **Real-time object detection:** YOLO passed the whole image is in one go for processing. This approach makes it an excellent fit for applications that need instant feedback.
- **Good precision:** YOLO can predict the object's class as well as its location. Hence, it gives good precision.
- **End-to-end architecture:** YOLO simplifies the detection pipeline by localizing and classifying in one network.
- **Wide range:** It can recognize different objects in multiple environments, which makes it an adaptable model for many areas, such as agriculture, self-driving cars, and security.

Because of the efficiency and speed that YOLO offers, it is considered a suitable algorithm for UAV applications in detecting rice leaf disease in real-time, where rapid and accurate detection is crucial for monitoring large agricultural areas.

3.3 Model Architecture

The major blocks/layers of the YOLOv11 architecture, shown in Figure 3.4, are described in this section. YOLOv11 architecture is mainly divided into three major parts, i.e., backbone, neck and head.

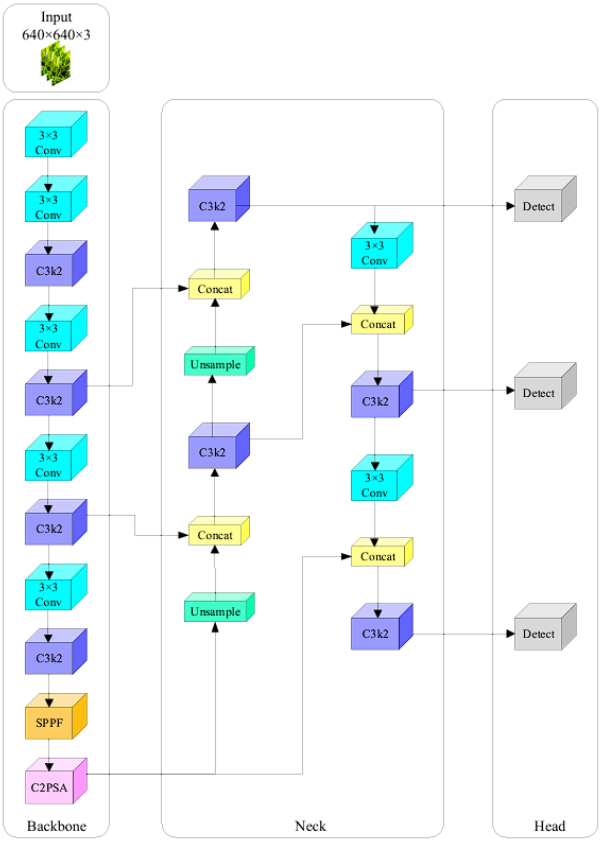

Figure 3.3: YOLOv11 model architecture [26]

3.3.1 Backbone

Backbone is the essential component of the YOLOv11 architecture. It is in charge of extracting features from input images at various scales. YOLOv11 utilizes a modified version of the Cross Stage Partial (CSP) architecture. One of the significant contributions to this backbone is the addition of the C3k2 block. This block replaces the C2f block used in previous YOLO's. Instead of one large convolution, the C3k2 block uses two smaller convolutions. It provides faster processing and maintains high performance. On top of this, YOLOv11 retains the previous take from YOLOv10; Spatial Pyramid Pooling - Fast (SPPF) block but introduces a new Cross Stage Partial with Spatial Attention (C2PSA) block This layer improves attention in its feature maps across space, allowing the model to focus more on salient areas in the image, and hence more accurately detecting objects of different sizes and locations. In a nutshell, the backbone of YOLOv11 is optimized for computational efficiency and accuracy. It enables the extraction of sophisticated features from images to support its efficient and precise object detection capability.

SPPF: It collects features from different receptive fields. Figure 3.4 depicts that a convolution layer extract features first. Then, three MaxPool2d layers process the data with

different pooling regions. Then, their outputs are concatenated. Another convolution blends the features. This lets the network understand both small and large objects in a single image.

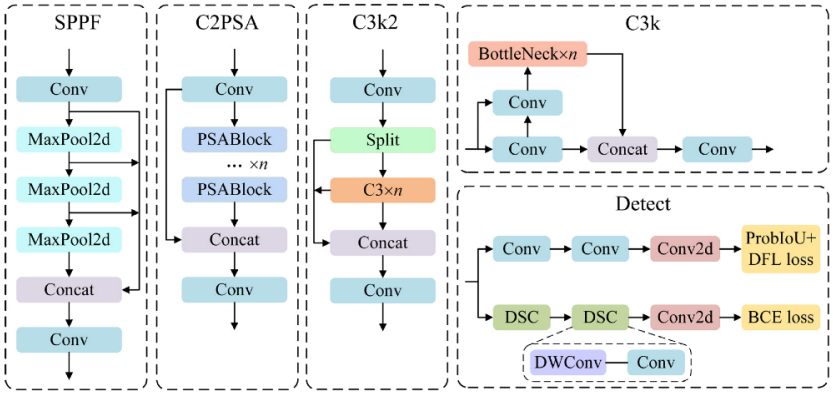

Figure 3.4: Architecture of SPPF, C2PSA, C3k2, and C3k modules [37]

C2PSA Module: At the end of the backbone, the C2PSA module, shown in Figure 3.4, is present. While the exact implementation can vary, C2PSA typically refers to a spatial attention or selective aggregation module. Its goal is to further refine and recalibrate the feature maps by emphasizing relevant spatial regions, preparing high-quality features for the Neck of the architecture.

C3k2: Mixes direct and processed features for better representation. Figure 3.4 shows that, a convolution layer starts the process. The output is split: one of the split parts skips direct to concatenation (shortcut), the other goes through several C3 blocks for feature processing. They are concatenated, and a final convolution layer combines and refines all information. This module creates both shallow and deep features for detection.

C3k: This block, shown in Figure 3.4, balances complex and simple features while learning. One branch feed through several Bottleneck blocks for complex transformations. And another uses fewer convolutions. Both branches are concatenated and passed through a convolution. This helps the module learn both detailed and basic patterns.

3.3.2 Neck

The neck of YOLOv11 plays an important role in combining and refining features from different layers of the network. It uses the efficient C3k2 block to merge multi-scale features coming from the backbone. After upsampling lower-resolution features to match the size of

higher-resolution ones, the neck concatenates these features to create rich, combined representations.

3.3.3 Head

The head module of YOLOv11 is responsible for producing the final object detection results. It contains multiple C3k2 blocks. These blocks further refine multi-scale features passed from the neck. Following these blocks, several Convolution-BatchNorm-SiLU (CBS) layers operate to stabilize, normalize, and extract relevant features for accurate classification and localization. The head ends with specialized convolutional detection layers that output the bounding box coordinates, class probabilities, and objectness scores. These layers use advanced losses like ProbIoU and Distribution Focal Loss (DFL) for bounding boxes and binary cross-entropy for objectness. Depthwise separable convolutions are also used in the head to reduce computation costs and maintain high accuracy. The design of the head ensures fast and precise predictions optimized for real-time object detection.

3.4 Depth and Width Multiples, and Max Channels

Scaling in YOLO models is a method used to create models of different sizes, such as nano (n), small (s), medium (m), large (l), and extra-large (x). An arbitrary scaling can also be done to modify the model size. The scaling enables the models to fit various computational needs and levels of accuracy. Scaling uses three important hyperparameters, including depth multiple, width multiple, and max channels. The depth and width of a model is indicated in Figure 3.5.

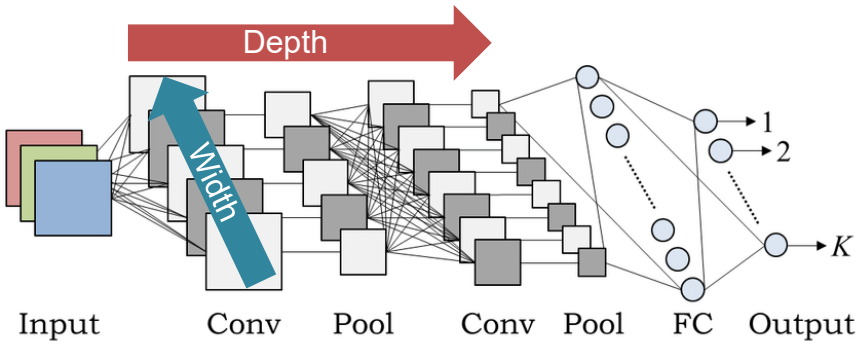

Figure 3.5: Depth and width of a model [38]

3.4.1 Depth multiple

The depth multiple is used to decide the layers of the model. It scales the number of repetitions of blocks or layers directly. If a block is repeated 3 times in the base model and the

depth multiple is 0.33, the scaled model will repeat that block just once, since 3 times 0.33 approximates to 1. The network is less deep with a smaller depth multiple. This approach speeds up the model at inference but weakens its ability to learn high-level features.

3.4.2 Width multiple

The width multiple rescales the number of channels in each convolutional layer. It increases or decreases the number of filters or feature maps by multiplying the value set as width multiple. For example, if width multiple is 0.25, then, a 256-channel base model layer will be downscaled to about 64 channels. Reducing the width a few times reduces the number of parameters, making the model lighter and faster, but may hamper its power of describing elaborate information.

3.4.3 Max channels

This parameter is a hard cap on the number of channels, independent of the scaling. For example, if scaling suggests 640 channels but the maximum number of channels is 512, then 512 channels will be used instead. It prevents very broad layers from reaching hardware memory limits and allows the model to be executed on different devices.

3.5 Design Methodology

Design methodology of the proposed model is shown in Figure 3.6. To design an optimum lightweight rice disease detection model, the dataset was collected at the very beginning of the work [34]. This work used the same primary dataset used in developing the UAV T-Yolo Rice network.

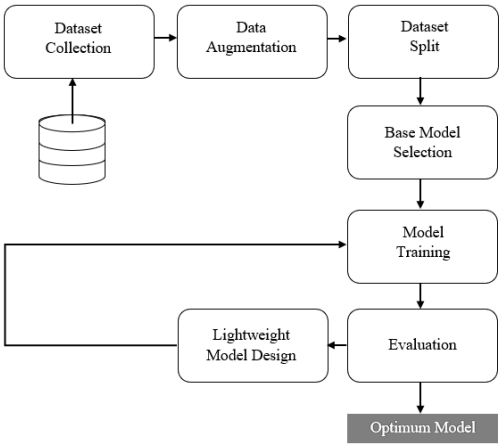

Figure 3.6: Flowchart of model design methodology

The primary dataset contains only 276 samples, but to train a deep learning model a large dataset is required. Hence, dataset is augmented in order to get a larger dataset. In this work the same augmentation procedure is followed as the UAV T-Yolo Rice performed [13]. Adding noise like Gaussian noise, uniform noise, saturation noise, image filtering helps to improve the robustness of the model, which results in a better generalization.

After augmentation the size of the dataset became 15,456. Then we split the dataset into train, validation, and test set which contain 70%, 20% and 10% of the samples respectively. The train set contains 10,819 samples, which is a good number for training a deep learning model.

Then, two state-of-the-art YOLO models, e.g., YOLOv8 and YOLOv11, were chosen as base models. These two models were trained and evaluated in an iterative manner.

After each evaluation both the computational cost and evaluation results were compared with existing state-of-the-art models for detecting rice disease detection. The iteration continued, with reduced total parameters and model size than the previous iteration, till we get a higher performance than the existing model.

The total parameters of the base models were reduced by scaling the width, depth and maximum channels of the models. Eventually, the optimum model was found, which is the proposed lightweight YOLOv11 model with just 0.81 M parameters.

3.6 Software Requirements

Ultralytics YOLO framework was used to implement the models. It provides a lightweight and modular toolbox for state-of-the-art object detection. The training and evaluation setup was done through the cloud-based GPU workstation, provided by Kaggle. It utilizes strong NVIDIA GPUs, large memory, and available computing resources. The large rice disease dataset, consisting of more than 15,000 labelled images, was processed quite efficiently by Kaggle.

3.7 YOLO Loss Functions

YOLO models are trained using a composite loss function. It consists of three types of loss:

Bounding Box Regression Loss: It evaluates how well the predicted bounding boxes fit over the ground-truth boxes. It utilizes advanced IoU-based metrics like Complete IoU that

take into account not only overlap but also the center distance and aspect ratio consistency, providing better localization results.

Objectness Loss: This component penalizes incorrect prediction of whether an object is contained in a bounding box or not. This loss function prevents false positives. It enables correct object detection. It helps the model to understand the difference between background and disease regions.

Classification Loss: It measures the cross-entropy loss between the actual disease classes and the predicted ones. Thus, it assists the model to accurately classify the samples.

The overall loss during training is a combination of these components, with a specific weight assigned to each, enabling a balanced optimization of localization and classification.

3.8 Optimizer

Stochastic Gradient Descent (SGD) with momentum is selected as the optimizer for model training. SGD is known for its effectiveness in converging to good minima. It generalizes well to unseen data. Momentum is set to accelerate convergence by smoothing updates and avoiding local minima.

3.9 Batch Size

A batch size of 64 is used during training. It balanced the computational efficiency and memory demands of the deep architecture on Kaggle's GPUs. Larger batch sizes generally help stabilize gradient estimates and speed up training.

3.10 Evaluation Metrics

To gain a comprehensive understanding of model performance, multiple metrics were employed:

Precision: The ratio of true positive to all positive detections. It reflects the correctness of predictions.

Recall: The ratio of true positive to total actual disease instances. It measures the completeness.

F1-Score: Harmonic mean of precision and recall. It provides a balanced accuracy indicator.

Mean Average Precision at IoU 0.5 (mAP50): Standard object detection metric that summarizes both localization and classification accuracy. It measures average precision over all disease classes at a 50% intersection-over-union threshold.

Inference Time: Measured in milliseconds per image. This metric quantifies the real-time readiness of the model on target hardware.

The combination of these metrics allows informed decisions on the trade-offs between accuracy and efficiency.

3.11 Computational Efficiency

Deep learning models usually run on high-performance platforms, but when deploying to resource-constrained platforms such as UAVs, computational efficiency is a major practical challenge. It determines if we can run inference in real-time, as well as affecting power usage, latencies, and performance as a whole. This performance is calculated and measured with some parameters:

3.11.1 GFLOPs (Giga Floating Point Operations)

GFLOPs is the number of floating-point operations taken by the model for processing a single input image, measured in billions. It is an approximation of the model computation and workload. The smaller the GFLOPs is, the better the computational efficiency.

3.11.2 Inference Time

It is the time taken by the model to predict a single input image and is usually measured in milliseconds (ms). This is an important metric for real-time applications such as UAV-based disease detection, where timely responses are required. Reducing inference time improves video stream or data set processing under stringent time constraints.

3.11.3 Total Parameters

This measure denotes the cumulative sum of trainable weights present in the system. So, a parameter-efficient model will be stored in memory with a smaller size, and the storage memory will be able to solve faster, which is helpful to deploy in an embedded system of a

small size. Note that if we use fewer parameters, we need to be very careful, as the model needs to have enough representational power to be accurate.

3.11.4 Model/Weight Size:

Model size refers to the amount of storage that the parameters or weights of the trained model occupy after they have been saved on the disk – it is usually measured in MB. A compact model can be easily transmitted, stored, and loaded into edge devices, such as UAVs. This is done to lower load times and memory usage, thus allowing deployment on devices with minimal storage resources, such as compact models.

Collectively, these measures of computational efficiency provide a holistic insight into the appropriateness of a model against realistic, hardware-constrained deployment scenarios typical of precision agriculture, whose ultimate purpose is to inform the design and optimization of lean yet effective detection architectures.

3.12 Summary

This chapter explained the process of dataset preparation, design methodology of the proposed lightweight rice disease detection model, technical aspects of the proposed lightweight YOLO models, covering Kaggle training environment, specific loss functions, optimizer settings, and hyperparameters including batch size and epochs. It brought to light evaluation metrics and computational efficiency challenges for real-time UAV deployment. This chapter provides the background necessary to proceed with the experimental results.

CHAPTER 4

INVESTIGATION, RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the experimental investigation, obtained results, and their comprehensive discussion. The main objective is to evaluate how variations in model scale, defined by different width and depth multipliers, affect the overall performance of the network. These experiments are designed to assess the trade-off between accuracy, inference speed, and model complexity in order to determine the most efficient model for practical deployment.

4.2 Parameter Settings

For training the model, this work uses the parameters shown in Table 4.1. Though the main dataset contains images having 300 x 300 size, for training and validation image size were chosen as 640 x 640. Because, when we train YOLOv8 or YOLOv11 models it is preferable to use high resolution images.

Table 4.1: The training parameters

Parameter Types	Parameter Settings
Input size	640 x 640
Batch size	64
Epoch	119
Optimizer	SGD
Initial Learning Rate	0.01
Final Learning Rate	0.0001

4.3 Experimental Results

Different scaled version of YOLOv8 and YOLOv11 models were trained with the same dataset and with the same parameters such as, input size, batch size, epoch, optimizer, and initial learning rate. From Table 4.2, it is clear that by scaling with different values of depth multiple, width multiple and maximum channel, total parameters of both the YOLOv8 and YOLOv11 were reduced to an optimum level. The optimum level was chosen by keeping the mean precision accuracy (mAP) higher than that of the state-of-the-art YOLO models. Table 4.2 shows that, as the number of parameters of the models decrease GFLOPs, inference time, and model size also decreases which results in an improvement in computational efficiency.

Table 4.2: Comparison of computational efficiency among trained models

Model	Depth Multiple	Width Multiple	Max Channels	Parameters	GFLOPs	Inference Time (ms)	Size (MB)
YOLOv8	0.33	0.25	1024	3,006,233	8.1	3.6	5.96
	0.25	0.25	1024	2,985,945	8.0	3.5	5.93
	0.167	0.167	1024	1,526,577	4.5	2.6	3.13
	0.25	0.125	1024	905,113	2.7	1.9	1.95
	0.125	0.125	1024	879,417	2.6	**1.8***	1.89
YOLOv11	0.5	0.25	1024	2,582,737	6.3	3.7	5.22
	0.25	0.25	768	1,969,457	5.8	3.6	4.05
	0.25	0.25	512	1,493,393	5.4	3.3	3.13
	0.25	0.167	768	1,086,944	3.6	2.9	2.36
	0.125	0.125	1024	**806,613***	**2.3***	2.0	**1.81***

If we compare Table 4.2 and Table 4.3, we can notice the performance metrics such as precision, recall, F1 score, and overall mAP50 decreases with the decrease in model parameters. Hence, if we want to reduce the computational efficiency, we must compromise the model performance a bit. For a scaling of depth multiple = 0.125, width multiple = 0.125 and max channels = 1024, we get the lightest model with only 0.81M parameters, 2.3 GFLOPs, 2.0 ms inference time and 1.81 MB model size. This model provides a descent performance with 94.90% precision, 88.90% recall, 91.80% F1 score and an overall 94.70% mAP50.

Table 4.3: Comparison of evaluation metrics among trained models

Model	Depth Multiple	Width Multiple	Max Channels	Precision (%)	Recall (%)	F1 (%)	mAP50 (%)
YOLOv8	0.33	0.25	1024	96.10	95.80	95.95	98.30
	0.25	0.25	1024	96.60	93.10	94.82	97.40
	0.167	0.167	1024	96.20	91.20	93.63	96.40
	0.25	0.125	1024	94.40	87.70	90.93	93.90
	0.125	0.125	1024	94.20	86.80	90.35	93.40
YOLOv11	0.5	0.25	1024	96.70	95.80	96.25	98.50
	0.25	0.25	768	96.90	92.30	94.54	97.40
	0.25	0.25	512	96.00	91.30	93.59	96.60
	0.25	0.167	768	95.10	90.80	92.90	96.00
	0.125	**0.125**	**1024**	**94.90**	**88.90**	**91.80**	**94.70**

Table 4.4: Evaluation results of optimum YOLOv11 (Scale = [0.125, 0.125, 1024])

Class	Images	Instances	Precision (%)	Recall (%)	mAP50 (%)
Rice Blast	448	1853	97.6	90.3	96.3
Bacterial Blight	538	565	94.0	97.9	98.9
Brown Spot	560	4375	93.3	78.4	88.9
All	1546	6793	94.9	88.9	94.7

Table 4.4 shows the evaluation results of the optimum lightweight YOLOv11 model. The model performs much better while detecting rice blast and bacterial blight. But, its performance slightly poor while detecting brown spot. The performance related to this class may fall due to the label noise. Sometimes, small lesions are harder to annotate accurately. Even small labeling errors can confuse the model. Another reason may be the class imbalance. Though the number of samples in the dataset are quite similar, the total instances in the brown spot class is much higher than that of the other two classes, shown in Figure 4.1.

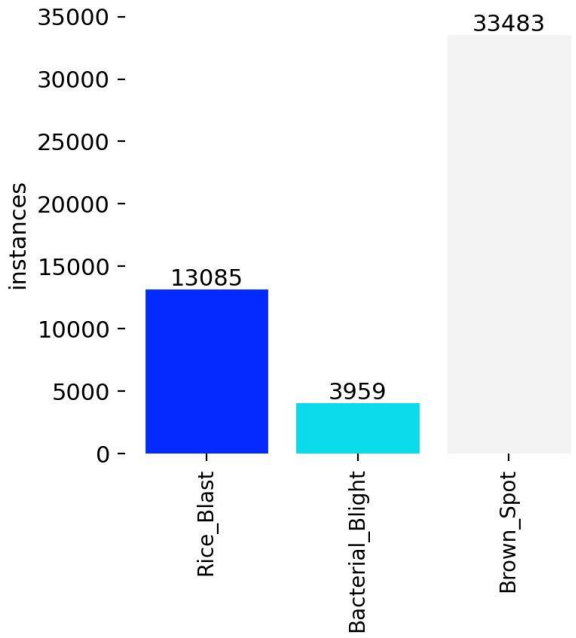

Figure 4.1: Total number of instances in different classes

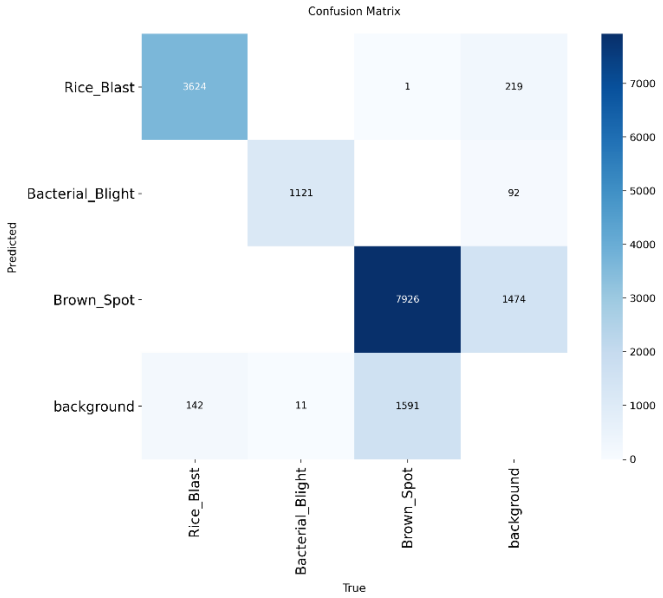

Figure 4.2: Confusion matrix of the optimum model

From the confusion matrix, shown in Figure 4.2, we can see the background is most often detected as brown spot. It may happen due to the label noise, as we stated earlier.

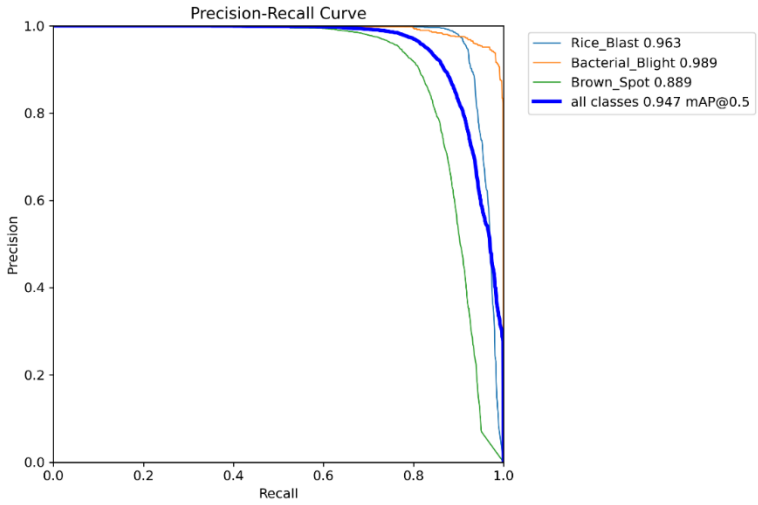

Figure 4.3: Precision-recall curve of the proposed model

The precision-recall curve, shown in Figure 4.3, depicts that the model exceptionally performs very well for rice blast and bacterial blight. These two classes achieve higher mAP, i.e., 96.3% and 98.9% respectively than that of the state-of-the-art RLDD-YOLOv11 model which achieves 69.9% and 98.2% mAP for rice blast and bacterial blight respectively [15]. Though this model achieves higher mAP for brown spot as 96.9%, Table 4.5 shows that, its overall mAP is 88.3% which is much less than the proposed optimum lightweight YOLOv11.

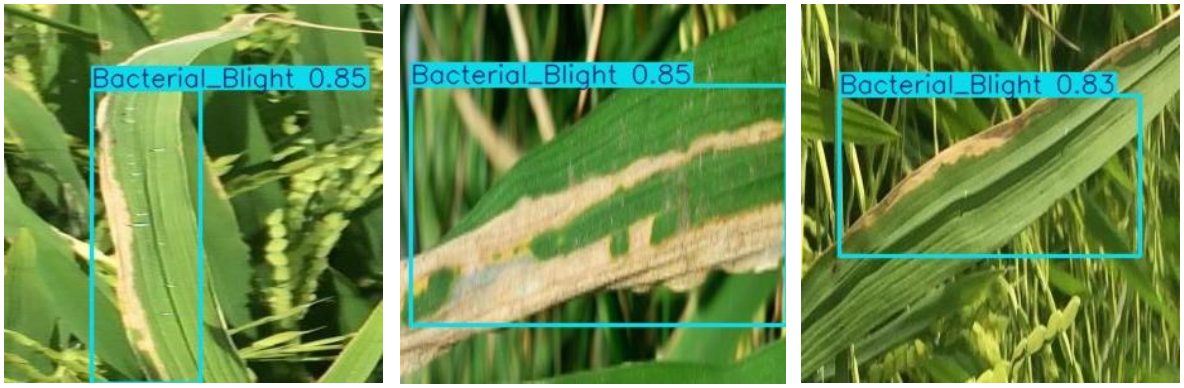

Figure 4.4: Prediction results (Bacterial Blight) using unseen and new dataset

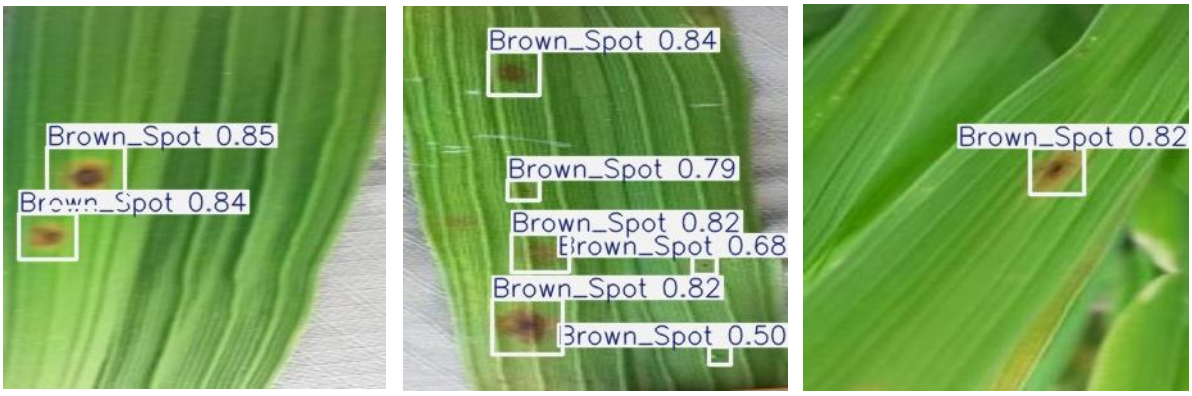

Figure 4.5: Prediction results (Brown Spot) using unseen and new dataset

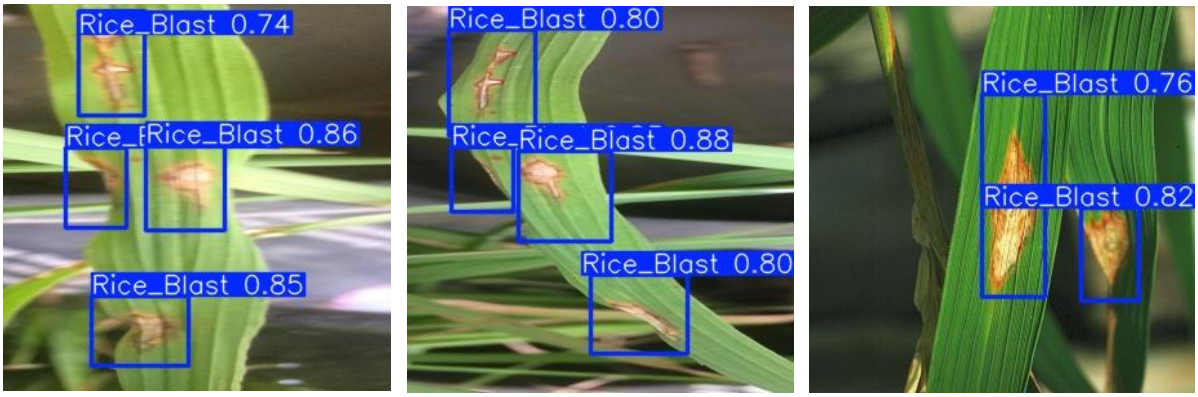

Figure 4.6: Prediction results (Rice blast) using unseen and new dataset

A random dataset from Kaggle was used to test the proposed optimum lightweight YOLOv11 model [6]. Figure 4.4, 4.5 and 4.6 show the result for different classes, which highlights that both the rice blast and bacterial blight classes are detected very well. But the for the brown spot class, the model is detecting few backgrounds as brown spot. Over all result of this test presents a testimony of the generalization capability of the proposed model.

4.4 Performance Comparison

Table 4.5: Performance comparison with state-of-the-art models

NO	Model	Precision (%)	Recall (%)	F1	mAP50 (%)	Parameters (M)	Size (MB)
1	UAV T-yolo Rice	-	-	0.810	86.97	-	-
3	RLDD- YOLOv11n	91.6	83.8	0.876	88.3	6.3	2.58
4	Proposed Model	**94.90**	**88.90**	**0.918**	**94.7**	**0.81**	**1.81**

Table 4.5 shows that, the proposed model of this study performs better with lower computational cost, which is our objective to achieve in order to deploy the model on UAV devices efficiently.

4.5 Summary

The work finds the optimum model for rice disease detection by doing a trade of between evaluation performance and computational cost, which can provide higher mean average precision (mAP) with a little computational cost.

CHAPTER 5

SOCIO-ECONOMIC IMPACT AND SUSTAINABILITY

5.1 Introduction

This chapter examines the socio-economic implications and sustainability aspects of the proposed rice disease detection project. It discusses its influence on societal well-being, public health, legal and policy compliance, cultural preservation, environmental protection, and long-term agricultural sustainability. The evaluation is framed within the context of national food policies and the United Nations Sustainable Development Goals (SDGs), particularly SDG 2: Zero Hunger and SDG 12: Responsible Consumption and Production (United Nations, 2015).

5.2 Impact of the Project on Societal, Health, Legal, and Cultural Issues

The impact of the project on societal, health, legal, and cultural issues are discussed in this section.

5.2.1 Societal Impact

Timely disease diagnosis reduces yield losses and makes rice production more stable, thereby boosting food security directly. This mitigates the risk of income volatility in farming communities brought on by disease outbreaks, enhancing rural livelihoods, which in turn contribute to poverty alleviation.

5.2.2 Health Impact

Consistent rice availability reduces the risk of malnutrition and food shortages, two major global health problems in developing countries. In addition, the use of check plants minimizes the overuse of pesticides, which is beneficial for both farmers and consumers, as it reduces chemical exposure and consequently positively affects public health.

5.2.3 Legal and Policy Impact

The use of smart agricultural practices and technologies can be a suitable option, considering the National Agricultural Policy of Bangladesh (NAP 2018), which encourages both sustainable and technological advances based on productivity improvements in the agricultural sector (Ministry of Agriculture, 2018). These initiatives also support the government in its digitalization of agriculture, which, in turn, enhances traceability, compliance, and decision-making throughout the agricultural value chain.

5.2.4 Cultural Impact

Rice is not just an economic commodity; it is a symbol of national identity in Bangladesh and other Asian countries. The project protects yields, which helps preserve traditional cuisines and food culture, ensuring that staple diets remain stable and accessible to future generations.

5.3 Impact of the Project on the Environment and Sustainability

The impact of the project on the environment and sustainability is discussed in this section.

5.3.1 Environmental Impact

The project can help foster sustainable agriculture through less over use and unplanned use of pesticides. Preventing degradation of soil and water, and killing good organisms, can be done by careful intervention, to keep the Agroecosystem sustainable.

5.3.2 Sustainability Impact

A pillar of sustainable agriculture is the efficiency of resource use. The system also allows for timely action by closely monitoring the disease and ensuring the judicious use of inputs like water, pesticides and fertilizers. As a result, greater output is generated per unit of input, which mitigates the environmental footprint of rice.

5.3.3 Long-Term Outlook

The project is a variation, but it fits closer toward precision ag, which combines technology and data analytics to make decisions. By embracing climate-smart agriculture, resilience to climate change is increased, contributing to sustainability, food security in the long-term, and delivering outcome 13 due to the impact that these can make against SDG 13: Climate Action (FAO, 2021).

5.4 Summary

The rice disease detection system offers a range of benefits, from on-farm applications to national and international sustainability targets. This contributes to food security, safeguards public health, upholds government agricultural policies, and preserves traditional rice-eating cultural practices. Promotes sustainable pesticide use, decreases pollution, and preserves ecosystem vitality. It encourages the implementation of precision agriculture, which not only

improves the resource efficiency of rice production systems but also ensures the long-term sustainability of the system. These contributions enhance the project's relevance in tackling local agricultural challenges and in addressing broader global goals like the UN SDGs.

CHAPTER 6

ADDRESSING COMPLEX ENGINEERING PROBLEMS AND ACTIVITIES

6.1 Introduction

This chapter highlights how the thesis work addresses complex engineering problems and complex engineering activities. The research involved solving real-world challenges in rice disease detection, requiring innovative solutions, and consideration of performance constraints.

6.2 Addressing Complex Engineering Problems

The project encountered several complex engineering problems, shown in Table 6.1, that demanded a deep understanding of computer vision, machine learning, and system design principles:

Table 6.1: Complex engineering problems of the project

Problem	Description	Mapped BAETE Attributes
High Intra-Class Similarity	Diseases such as Bacterial Leaf Blight, Blast, and Brown Spot share visual patterns, making classification difficult.	P2: In-depth engineering knowledge required to distinguish between visually similar classes.
Limited Dataset Size & Overfitting Risk	Small agricultural datasets lead to poor model generalization.	P3: Involves uncertainty and incomplete data; requires probabilistic and statistical approaches.
Balancing Accuracy vs. Computational Complexity	Larger models improve accuracy but increase inference latency and memory usage.	P4: Multiple conflicting constraints (speed vs. accuracy) to balance.
Real-Time UAV Deployment	Model must run with low latency on resource-constrained devices.	P6: Requires practical knowledge of hardware limitations and system-level integration.

6.3 Addressing Complex Engineering Activities

Solving the above challenges required carrying out complex engineering activities, shown in Table 6.2, involving multiple steps and interdisciplinary skills.

Table 6.2: Complex engineering activities of the project

Activity	Description	Mapped BAETE Attributes
Model Adaptation & Customization	Modified and trained YOLO architectures for rice disease detection.	A2: Involves diverse resources (datasets, compute, tools).
Trade-Off Analysis	Evaluated accuracy vs. latency for various YOLO variants.	A3: Requires creative design and technical judgment.
Multidisciplinary Integration	Combined image processing, machine learning, and performance evaluation.	A1: Involves diverse knowledge fields.
Use of Modern Tools	Implemented models in Darknet and compared with newer frameworks (Ultralytics YOLO).	A4: Use of advanced tools with appropriate calibration and validation.
Real-World Deployment Considerations	Ensured model runs efficiently on UAV hardware.	A6: Imposes significant practical constraints.
Ethical & Societal Considerations	Focused on improving food security and supporting farmers.	A7: Addresses societal and sustainability factors.

6.4 Summary

This chapter demonstrated that the research involved addressing complex, multidisciplinary challenges and performing activities that required integration of knowledge, tools, and engineering judgement. The combination of data augmentation, lightweight YOLOv11 model, and real-time considerations led to a practical and deployable solution. These approaches meet the criteria for addressing complex engineering problems and activities.

CHAPTER 7

CONCLUSIONS, LIMITATIONS AND FUTURE WORKS

7.1 Conclusion

This thesis presented the design, implementation, and evaluation of an optimum lightweight YOLO-based model for UAV-assisted rice leaf disease detection. The primary objective was to achieve a balance between high detection accuracy and computational efficiency, enabling deployment on UAV platforms for real-time field applications. Two state-of-the-art object detection models, YOLOv8 and YOLOv11, were trained and evaluated on a rice disease dataset containing three major diseases such as, Bacterial Leaf Blight, Blast, and Brown Spot. Experimental results demonstrated that:

- YOLOv11 model with depth multiple = 0.125, width multiple = 0.125 and max channels = 1024 scaling provides a lightweight model which requires about 69% less parameters than that of existing models, making it suitable for edge deployment.
- This optimum lightweight model provides about 6.4% higher mean average precision (mAP), i.e., 94.7% and outperforms several existing models reported in the literature review.
- This model achieves 2 ms/image inference speed which is very fast than any other existing models and this fast inference speed enables the model to deploy in UAV devices for real time detection.

The results ensure the testimony that the proposed model is highly suitable for resource-constrained UAV applications. Overall, the proposed models contribute to the development of practical, real-time, and scalable solutions for precision agriculture, supporting farmers in early disease detection and timely intervention.

7.2 Limitations

The research outcomes are promising, but several limitations must be mentioned:

- **Lack of Real-Time UAV Testing:** The model was evaluated in using software, but field deployment on UAV hardware was not performed.
- **Limited Disease Coverage:** The dataset contains only three rice diseases; thus, the models may not generalize well to other diseases or crops.
- **Hardware Constraints:** The research focused on model design and software implementation but did not extensively benchmark performance across diverse UAV hardware platforms.

7.3 Future Works

Several directions can be followed to extend and strengthen this research. Such as:

- **Dataset Expansion:** Include more rice diseases, different growth stages, and multi-crop datasets to improve generalization and robustness.
- **Field Deployment:** Integrate and test the model on UAV platforms in real paddy field conditions to validate real-time performance under varying environmental factors.
- **Advanced Architectures:** Investigate transformer-based object detection models (e.g., YOLOv12, DETR) and attention-enhanced hybrid networks for further performance gains.

References

[1] “Bangladesh - Agricultural Land (% Of Land Area) - 2025 Data 2026 Forecast 1961-2022 Historical.” Accessed: Sept. 12, 2025. [Online]. Available: https://tradingeconomics.com/bangladesh/agricultural-land-percent-of-land-area-wb-data.html

[2] “Bangladesh GDP share of agriculture - data, chart,” TheGlobalEconomy.com. Accessed: Sept. 12, 2025. [Online]. Available: https://www.theglobaleconomy.com/Bangladesh/share_of_agriculture/?utm_source=chatgpt.com

[3] 5, “Bangladesh - Agriculture Sectors.” Accessed: Sept. 12, 2025. [Online]. Available: https://www.trade.gov/country-commercial-guides/bangladesh-agriculture-sectors

[4] S. Shekhar, D. Sinha, and A. Kumari, “An Overview of Bacterial Leaf Blight Disease of Rice and Different Strategies for its Management,” *Int. J. Curr. Microbiol. Appl. Sci.*, vol. 9, no. 4, pp. 2250–2265, Apr. 2020, doi: 10.20546/ijcmas.2020.904.270.

[5] B. S. Teja *et al.*, “Biological control of bacterial leaf blight (BLB) in rice–A sustainable approach,” *Heliyon*, vol. 11, no. 2, p. e41769, Jan. 2025, doi: 10.1016/j.heliyon.2025.e41769.

[6] “RICE CROP DISEASES.” Accessed: Sept. 13, 2025. [Online]. Available: https://www.kaggle.com/datasets/thegoanpanda/rice-crop-diseases

[7] G. O. Agbowuro, M. S. Afolabi, E. F. Olamiriki, and S. O. Awoyemi, “Rice Blast Disease (Magnaporthe oryzae): A Menace to Rice Production and Humanity,” *Int. J. Pathog. Res.*, pp. 32–39, June 2020, doi: 10.9734/ijpr/2020/v4i330114.

[8] “Rice Blast - an overview | ScienceDirect Topics.” Accessed: Sept. 12, 2025. [Online]. Available: https://www.sciencedirect.com/topics/pharmacology-toxicology-and-pharmaceutical-science/rice-blast

[9] K. H. Kaboré, A. I. Kassankogno, and D. Tharreau, “Brown Spot of Rice: Worldwide Disease Impact, Phenotypic and Genetic Diversity of the Causal Pathogen *Bipolaris oryzae* , and Management of the Disease,” *Plant Pathol.*, vol. 74, no. 4, pp. 908–922, May 2025, doi: 10.1111/ppa.14075.

[10] “Crop Protection.” Accessed: Sept. 12, 2025. [Online]. Available: http://www.agritech.tnau.ac.in/expert_system/paddy/cpdisbrownspot.html

[11] “Smart Agriculture Market Size, Trends | Industry Report, 2030.” Accessed: Sept. 12, 2025. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/smart-agriculture-farming-market

[12] M. T. Ahad, Y. Li, B. Song, and T. Bhuiyan, “Comparison of CNN-based deep learning architectures for rice diseases classification,” *Artif. Intell. Agric.*, vol. 9, pp. 22–35, Sept. 2023, doi: 10.1016/j.aiia.2023.07.001.

[13] A. K. Sangaiah, F.-N. Yu, Y.-B. Lin, W.-C. Shen, and A. Sharma, “UAV T-YOLO-Rice: An Enhanced Tiny Yolo Networks for Rice Leaves Diseases Detection in Paddy Agronomy,” *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 6, pp. 5201–5216, Nov. 2024, doi: 10.1109/TNSE.2024.3350640.

[14] J. Wang *et al.*, “Improved Lightweight YOLOv8 Model for Rice Disease Detection in Multi-Scale Scenarios,” *Agronomy*, vol. 15, no. 2, p. 445, Feb. 2025, doi: 10.3390/agronomy15020445.

[15] K. Fang, R. Zhou, N. Deng, C. Li, and X. Zhu, “RLDD-YOLOv11n: Research on Rice Leaf Disease Detection Based on YOLOv11,” *Agronomy*, vol. 15, no. 6, p. 1266, May 2025, doi: 10.3390/agronomy15061266.

[16] W. Bao, Y. Ren, N. Wang, G. Hu, and X. Yang, "Detection of Abnormal Vibration Dampers on Transmission Lines in UAV Remote Sensing Images with PMA-YOLO," *Remote Sens.*, vol. 13, no. 20, p. 4134, Oct. 2021, doi: 10.3390/rs13204134.
[17] L. Zhang *et al.*, "A lightweight convolutional neural network model with receptive field block for C-shaped root canal detection in mandibular second molars," *Sci. Rep.*, vol. 12, no. 1, p. 17373, Oct. 2022, doi: 10.1038/s41598-022-20411-4.
[18] M. B. Ahmad and Tae-Sun Choi, "Local threshold and Boolean function based edge detection," *IEEE Trans. Consum. Electron.*, vol. 45, no. 3, pp. 674–679, Aug. 1999, doi: 10.1109/30.793567.
[19] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intell. Syst. Their Appl.*, vol. 13, no. 4, pp. 18–28, July 1998, doi: 10.1109/5254.708428.
[20] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.
[21] B. S. Bari *et al.*, "A real-time approach of diagnosing rice leaf disease using deep learning-based faster R-CNN framework," *PeerJ Comput. Sci.*, vol. 7, p. e432, Apr. 2021, doi: 10.7717/peerj-cs.432.
[22] J. Chen, D. Zhang, Y. A. Nanehkaran, and D. Li, "Detection of rice plant diseases based on deep transfer learning," *J. Sci. Food Agric.*, vol. 100, no. 7, pp. 3246–3256, May 2020, doi: 10.1002/jsfa.10365.
[23] F. Guo *et al.*, "Improved YOLOv7-Tiny for the Detection of Common Rice Leaf Diseases in Smart Agriculture," *Agronomy*, vol. 14, no. 12, p. 2796, Nov. 2024, doi: 10.3390/agronomy14122796.
[24] L. Jia *et al.*, "MobileNet-CA-YOLO: An Improved YOLOv7 Based on the MobileNetV3 and Attention Mechanism for Rice Pests and Diseases Detection," *Agriculture*, vol. 13, no. 7, p. 1285, June 2023, doi: 10.3390/agriculture13071285.
[25] Y. Lu, J. Yu, X. Zhu, B. Zhang, and Z. Sun, "YOLOv8-Rice: a rice leaf disease detection model based on YOLOv8," *Paddy Water Environ.*, vol. 22, no. 4, pp. 695–710, Oct. 2024, doi: 10.1007/s10333-024-00990-w.
[26] H. Teng, Y. Wang, W. Li, T. Chen, and Q. Liu, "Advancing Rice Disease Detection in Farmland with an Enhanced YOLOv11 Algorithm," *Sensors*, vol. 25, no. 10, p. 3056, May 2025, doi: 10.3390/s25103056.
[27] L. Yang, F. Guo, H. Zhang, Y. Cao, and S. Feng, "Research on Lightweight Rice False Smut Disease Identification Method Based on Improved YOLOv8n Model," *Agronomy*, vol. 14, no. 9, p. 1934, Aug. 2024, doi: 10.3390/agronomy14091934.
[28] S. Phadikar, J. Sil, and A. K. Das, "Rice diseases classification using feature selection and rule generation techniques," *Comput. Electron. Agric.*, vol. 90, pp. 76–85, Jan. 2013, doi: 10.1016/j.compag.2012.11.001.
[29] C. G. Simhadri, H. K. Kondaveeti, V. K. Vatsavayi, A. Mitra, and P. Ananthachari, "Deep learning for rice leaf disease detection: A systematic literature review on emerging trends, methodologies and techniques," *Inf. Process. Agric.*, vol. 12, no. 2, pp. 151–168, June 2025, doi: 10.1016/j.inpa.2024.04.006.
[30] P. Li, J. Zhou, H. Sun, and J. Zeng, "RDRM-YOLO: A High-Accuracy and Lightweight Rice Disease Detection Model for Complex Field Environments Based on Improved YOLOv5," *Agriculture*, vol. 15, no. 5, p. 479, Feb. 2025, doi: 10.3390/agriculture15050479.
[31] Z. Li *et al.*, "A Copy Paste and Semantic Segmentation-Based Approach for the Classification and Assessment of Significant Rice Diseases," *Plants*, vol. 11, no. 22, p. 3174, Nov. 2022, doi: 10.3390/plants11223174.

[32] D. C. Trinh, A. T. Mac, K. G. Dang, H. T. Nguyen, H. T. Nguyen, and T. D. Bui, "Alpha-EIOU-YOLOv8: An Improved Algorithm for Rice Leaf Disease Detection," *AgriEngineering*, vol. 6, no. 1, pp. 302–317, Feb. 2024, doi: 10.3390/agriengineering6010018.

[33] X. Jin, F. Yu, Y. Suo, X. Song, and R. Li, "SCL-YOLOv8n based rice disease lightweight detection method," *Meas. Sci. Technol.*, vol. 36, no. 5, p. 056006, May 2025, doi: 10.1088/1361-6501/add1fe.

[34] A. K. G. Francisco, *aldrin233/RiceDiseases-DataSet*. (July 22, 2025). Accessed: Sept. 13, 2025. [Online]. Available: https://github.com/aldrin233/RiceDiseases-DataSet

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[36] G. Jocher, J. Qiu, and A. Chaurasia, *Ultralytics YOLO*. (Jan. 2023). Python. Accessed: Sept. 13, 2025. [Online]. Available: https://github.com/ultralytics/ultralytics

[37] J. Huang, K. Wang, Y. Hou, and J. Wang, "LW-YOLO11: A Lightweight Arbitrary-Oriented Ship Detection Method Based on Improved YOLO11," *Sensors*, vol. 25, no. 1, p. 65, Dec. 2024, doi: 10.3390/s25010065.

[38] "Fig. 1: An example of CNN architecture.," ResearchGate. Accessed: Sept. 24, 2025. [Online]. Available: https://www.researchgate.net/figure/An-example-of-CNN-architecture_fig1_320748406