Journal of Computer and Communications, 2025, 13(8), 181-204
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

YOLO-AgriNet: A Deep Learning-Based Model for Real-Time Plant Disease Detection in Precision Agriculture

Armel Ngomade Nkonjoh[1*], Jean Roger Djamen Kaze[2], Rostand Verlaine Nwokam[2], Brondon Ella Njotsa[2], Alain Francois Kuate[3], Alain Serge Mbiada Tchouta[2], Serge Bertrand Bissiongol Babagniack[2]

[1]Ambam Computer Science and Application Laboratory & Department of Computer Engineering, Higher Institute of Transport, Logistics and Commerce, University of Ebolowa, Ebolowa, Cameroon
[2]Institut d'ingenierie informatique d'Afrique Centrale, Institut Universitaire de la Côte, Douala, Cameroon
[3]Technology and Applied Sciences Laboratory, University Institute of Technology, University of Douala, University of Douala, Douala, Cameroon
Email: *armelngomade@yahoo.fr, kaze.roger@myiuc.com, rostand.nwokam@myiuc.com, kuatealainfrancois@yahoo.fr

How to cite this paper: Nkonjoh, A.N., Kaze, J.R.D., Nwokam, R.V., Njotsa, B.E., Kuate, A.F., Tchouta, A.S.M., Babagniack, S.B.B. (2025) YOLO-AgriNet: A Deep Learning-Based Model for Real-Time Plant Disease Detection in Precision Agriculture. *Journal of Computer and Communications*, **13**, 181-204.
https://doi.org/10.4236/jcc.2025.138009

Received: July 19, 2025
Accepted: August 17, 2025
Published: August 20, 2025

Abstract

Timely and accurate detection of plant diseases is essential for improving crop yields and ensuring food security, particularly in regions like Cameroon, where farmers often rely on visual inspection. An approach limited by subjectivity and low precision. Although deep learning and precision agriculture technologies have advanced significantly, many models still face challenges in detecting early-stage symptoms, especially in real-world, resource-limited environments. This study introduces YOLO-AgriNet, a customized object detection model built on the YOLOv8 architecture, optimized for plant disease detection under tropical and low-resource conditions. To enhance the detection of small and subtle features, YOLO-AgriNet integrates key architectural improvements, including Convolutional Block Attention Modules (CBAM), Atrous Spatial Pyramid Pooling (ASPP) and an additional Stage Layer 5 for finer spatial representation. The model was trained on a public dataset and a curated set of local images from plantations in Cameroon. Compared to YOLOv8, Faster R-CNN, and SSD, YOLO-AgriNet achieved higher performance with a mAP@0.5 of 84.5%, real-time inference speed (45 FPS), and improved robustness in complex tropical conditions. It also demonstrated superior accuracy in detecting small disease symptoms and reduced false positives. YOLO-AgriNet provides a lightweight, scalable, and practical solution for real-time plant disease monitoring. Its compatibility with low-cost platforms like smartphones and drones makes it highly suitable for developing regions, enabling timely interventions and supporting sustainable agriculture.

Keywords

Precision Agriculture, YOLOv8, Plant Disease Detection, Real-Time Object Detection, Deep Learning

1. Introduction

Agriculture remains a cornerstone of the global economy, particularly in developing countries where it constitutes a major source of income and sustenance. However, agricultural productivity is constantly threatened by various biotic stresses, among which plant diseases are particularly harmful. Plant diseases are a major cause of crop loss, accounting for an estimated 10% - 16% reduction in annual agricultural production [1]. These diseases, caused by pathogens such as fungi, bacteria, and viruses, can lead to severe yield losses, jeopardizing food security and economic stability, especially in rural communities [2].

Conventional plant disease detection techniques predominantly rely on visual assessments conducted by farmers or agronomic specialists. While these approaches are straightforward to deploy in field conditions, they suffer from significant limitations in terms of precision and objectivity. Recent findings indicate that visual diagnosis of Fusarium wilt symptoms achieves an accuracy rate of merely 40%, largely due to the phenotypic similarity of its manifestations with other abiotic stress factors such as water deficiency or nutrient imbalances [2]. In addition, visual evaluation is generally ineffective in detecting early-stage infections, as initial symptoms often remain imperceptible to the naked eye, thus limiting the scope for timely preventive actions [3].

To overcome these constraints, precision agriculture technologies particularly those integrating artificial intelligence and advanced image processing offers promising alternatives. In many African countries, such as Kenya, for instance, the use of drones equipped with multispectral sensors and deep learning models has resulted in a 25% reduction in yield losses in maize cultivation [4]. Such technologies are especially well-suited for adoption in low-resource environments, where access to agronomic expertise and infrastructure is limited but where the demand for scalable, data-driven agricultural solutions is increasingly critical.

These recent years, numerous deep learning models have been proposed for plant disease detection, with particular emphasis on convolutional neural networks (CNNs) and object detection frameworks such as Faster R-CNN, SSD (Single Shot Detector), and YOLO (You Only Look Once) [5]-[7]. Each of these architectures offers specific strengths and limitations. CNNs, for instance, are well-suited for image classification tasks but are less effective when it comes to accurately localizing disease symptoms on leaves or stems. Faster R-CNN achieves high detection accuracy; however, its relatively slow inference time restricts its applicability in real-time scenarios. In contrast, SSD and YOLO are designed to strike a balance between accuracy and speed, making them more viable for real-time deployment.

Nonetheless, they often struggle with detecting small-scale objects or subtle visual cues, which are critical for identifying early-stage disease symptoms. For example, although YOLOv8 demonstrates strong overall performance, it achieves a mean Average Precision (mAP) of only 65% for detecting small foliar spots under real-world conditions [8]. All these methods use a traditional image processing workflow which start from data acquisition passing through data preprocessing and identification to classification and which is depicted in Figure 1.

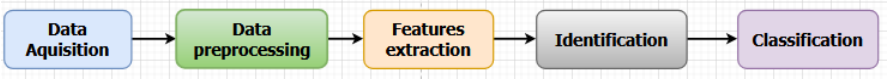


Figure 1. Traditional image pre-processing techniques.

To address these challenges, we propose YOLO-AgriNet, a novel object detection architecture tailored to the constraints of real-world agricultural environments in Cameroon. This model extends YOLOv8 by integrating attention modules (CBAM) and an Atrous Spatial Pyramid Pooling mechanism (ASPP) to enhance the detection of small, early-stage symptoms. YOLO-AgriNet is designed to operate effectively on low-resource devices while delivering improved precision and speed.

The main contributions of this work are threefold:

1) The design and implementation of YOLO-AgriNet, combining CBAM, ASPP, and an additional Stage Layer 5 to boost detection of small objects in agricultural imagery.

2) A comprehensive performance evaluation comparing YOLO-AgriNet to established models (YOLOv8, SSD, Faster R-CNN), highlighting its superior accuracy and efficiency under field conditions in Cameroon.

3) A transferable methodology for adapting deep learning models to low-resource settings, including hyperparameter optimization, model acceleration (quantization, pruning), and the use of annotated drone imagery.

This study adopts a practical and innovative approach, delivering an efficient and user-friendly tool for real-time crop monitoring while advancing scientific knowledge. By developing an optimized deep learning model specifically designed for precision agriculture in developing regions, the research fills a crucial gap. The findings open doors to impactful applications, such as embedding YOLO-AgriNet into automated monitoring platforms or farmer-centric mobile apps, facilitating widespread adoption in resource-limited agricultural environments.

The rest of this document is organized as follows. Section 2 reviews recent research efforts to contextualize advances in plant disease detection and identify the specific gaps this study aims to address. Section 3 details dataset used, the data preprocessing techniques applied, and the architectural modifications made to YOLOv8 to develop the proposed YOLO-AgriNet model. Section 4 presents the model's performance, comparing it with existing approaches, with a focus on ac-

curacy, inference speed, and adaptability to real-world agricultural conditions. Finally, we summarize the main contributions of this work and explore potential extensions to other crops and geographic regions in section 5.

2. Related Works

This study primarily addressed the automation of plant disease detection in agriculture. A critical issue for many countries facing increasing food demand due to rapid population growth. Advances in modern technologies have significantly enhanced the efficiency and accuracy of disease detection in both plants and animals. Detection is the first step in a broader workflow aimed at controlling disease spread and minimizing its impact. The research emphasized detailed analysis of plant diseases and the application of artificial intelligence for rapid identification. In particular, machine learning and deep learning models have been employed to automate the detection process. Additionally, various datasets have been reviewed to support and improve outcomes for the research community [9].

Recent decades have witnessed significant breakthroughs in AI-powered plant disease detection, a critical tool for mitigating agricultural losses in regions such as Cameroon, where pathogens account for up to 40% of yield reductions [2]. Moreover, a study in [10] using PlantDoc reported a 35% reduction in crop yield caused by plant diseases. Early detection of plant infections remains a challenge due to limited tools and insufficient knowledge. Here, we review the progression of detection methodologies, analyzing key innovations, model performance, limitations, and emerging opportunities for scalable implementation.

Deep learning methods are increasingly used in agricultural research due to their ability to rapidly extract high-level features and outperform traditional machine learning algorithms in both speed and accuracy [11] [12]. The introduction of Convolutional Neural Networks (CNNs) marked a major breakthrough in the field of computer vision. In 2012, Krizhevsky *et al.* [13] introduced AlexNet, a deep CNN that won the ImageNet competition with unprecedented accuracy. This revolutionary advancement marked the beginning of CNN applications in various fields, including agriculture. Following this, VGG [14] and ResNet [15] improved depth and learning stability, respectively. Mohanty *et al.* [16] leveraged these architectures for disease classification in tomato and wheat leaves. Their model achieved an impressive accuracy of 99.35%, but suffered from the inability to localize infected areas, a major limitation for field diagnosis.

To address this gap, Girshick *et al.* [17] introduced R-CNN, which proposed object detection using region proposals. Although accurate, the model was extremely slow, making it impractical for real-world use. Fast R-CNN proposed by Ren et al. [18] improved speed through shared computations, but it marked a turning point with its integrated Region Proposal Network (RPN). The model became faster while maintaining high accuracy. Faster R-CNN was applied in various ag-

ricultural contexts, including vineyards and rice [19]. However, its high inference time (200 ms/image) and complexity made real-time use challenging, particularly in resource-limited rural areas.

The quest for speed led to the development of single-shot models. Liu *et al.* [20] introduced the Single Shot MultiBox Detector (SSD), achieving a mean Average Precision (mAP) of 74.3% with an inference time of 125 ms. Despite its efficiency in standard cases like apples or soybeans, SSD failed to detect small objects, limiting its relevance for early-stage diseases. In parallel, Redmon *et al.* [21] proposed YOLO (You Only Look Once), capable of processing images in real time (45 ms). Early versions (YOLOv1, v2) were fast but lacked precision on fine details, restricting their application in plant disease detection. With YOLOv3 [22], the addition of feature pyramids enabled more robust multi-scale detection, making YOLO a reference solution for agricultural applications in real-world settings.

To address small object detection, newer approaches emerged. Jayme [23] proposed a model based on Efficient Det, using multispectral images for rice disease detection. Their method achieved an mAP of 72%, but remained resource-intensive, hindering field deployment. Meanwhile, YOLO continued to evolve. YOLOv8 introduced notable improvements in precision (mAP@0.5 of 65% on small leaf lesions, according to Wang *et al.* [24], but was still limited by:

1) insufficient precision on small objects;

2) sensitivity to environmental variations;

3) an architecture capped at Stage Layer 4, restricting fine-detail detection.

The unique challenges of agricultural disease detection in low-resource, tropical region such as complex backgrounds (soil, weeds), variable lighting conditions, and limited computational infrastructure have driven the development of specialized lightweight models. Recent studies have focused on efficiency optimization, robustness to noise, and real-time deployability on edge devices.

The work of John *et al.* [25] reviews advanced plant disease detection methods remote sensing, molecular diagnostics, and machine learning improving early, accurate pathogen identification and sustainable crop protection. Despite benefits, high costs, limited datasets, and technical complexity hinder adoption. It calls for affordable, diverse data and interdisciplinary approaches to enhance accessibility and precision agriculture.

Zaman Abib *et al.* [26] introduce an automated leaf disease detection system for major Bangladeshi crops using YOLOv8, achieving high accuracy with a 98% mAP and 97% F1 score on a curated dataset of 19 disease classes. Its main contribution lies in demonstrating YOLOv8's superior real-time detection capabilities, facilitating early intervention to improve crop management and food security. However, limitations include reliance on annotated image datasets that may not fully capture field variability, and the system's deployment in real-time farm environments could be challenged by changing lighting conditions, occlusions, and dataset diversity, which may impact practical accuracy and robustness.

By addressing environmental challenges, Zhang *et al.* [27] conducted a comprehensive study and identified key obstacles: complex backgrounds, lighting variability and hardware constraints. To mitigate these issues, the authors proposed: Attention mechanisms (CBAM, SE Blocks) to suppress irrelevant background features. Multi-scale feature fusion (FPN, PANet) to improve small-lesion detection. Data augmentation with synthetic shadows to enhance robustness to lighting changes. Their lightweight EfficientNet-B3 variant achieved 68% mAP on maize disease datasets, but struggled with occluded leaves (mAP drop of 15% under heavy occlusion).

Research has shifted toward efficient models deployable on low power devices. Khan *et al.* [28] developed Agri Mobile Net, a quantized CNN achieving 62% mAP on Raspberry Pi, while Singh *et al.* [29] pruned ResNet-18 by 60% with minimal accuracy loss. YOLO variants (e.g., YOLOv8-nano) now dominate real-time applications, with less than 2W power consumption. For tropical regions, attention mechanisms (CBAM, Transformers) mitigate background noise, improving recall by 12% - 15%. Multispectral imaging and GAN-augmented datasets address data scarcity, boosting mAP by 18% for rare diseases [30].

To better understand the evolution and effectiveness of different approaches in image-based plant disease detection, we present a comparative analysis of selected state-of-the-art methods. **Table 1** summarizes key studies based on their methodology, dataset used, detection performance, and computational suitability. This comparison highlights the shift from conventional convolutional neural networks (CNNs) to lightweight and real-time deep learning models, with increasing attention to deployment on resource-constrained devices such as mobile phones and edge devices.

Table 1. Comparative analysis of plant disease detection approaches.

Ref	Year	Methodology	Dataset	Accuracy/mAP	Size/Device
[16]	2016	CNN (Alex Net, Google)	Plant Village (Tomato, Wheat)	99.35% (classification)	High-end GPU
[17]	2014	R-CNN (Region proposals + CNN)	General object detection	High accuracy, slow inference	Large model
[18]	2015	Faster R-CNN	Vineyards, rice images [19]	High accuracy, 200 ms/img	Resource-heavy
[20]	2016	SSD (Single ShotMulti Box Detector)	Apples, soybeans	74.3% mAP, 125 ms	Medium
[21] [22]	2017-2018	YOLO v1-v3 (real-time detection)	Various (agricultural settings)	Real-time, improved small-object detection	45 ms/img
[23]	2019	Efficient Det + Multi-spectral Imaging	Rice disease images	72% mAP	Resource-intensive
[24]	2024	YOLOv8	Leaf lesion dataset	65% mAP@0.5	–

Continued

[25]	2023	Agri Mobile Net (Quantized CNN)	Custom Raspberry Pi dataset	62% mAP	Deployed on Raspberry Pi
[29]	2024	Pruned ResNet-18 (60%)	–	Slight accuracy drop	Lightweight
[30]	2025	Multispectral + GAN-augmented dataset	Rare disease cases	+18% mAP gain	Edge deployable

Based on the literature reviewed, plant disease detection in agriculture has seen significant advancements driven by the development of deep learning models, particularly convolutional neural networks and object detection frameworks such as YOLO, SSD, and Faster R-CNN. These approaches have improved accuracy and speed, enabling real-time monitoring; however, they face notable limitations when applied in resource-constrained environments typical of developing regions. Challenges such as detecting small or early-stage symptoms, environmental variability, complex backgrounds, and hardware limitations persist, restricting the effectiveness and practicality of many existing models. Recent research efforts have focused on integrating attention mechanisms, multispectral imaging, and model optimization techniques like pruning and quantization to enhance robustness and deploy ability. Nonetheless, many solutions remain resource-intensive or lack adaptability to diverse agricultural conditions, highlighting a gap in models that are both lightweight and highly accurate in complex, real-world settings. Addressing these gaps requires developing tailored, resource-efficient architectures that can operate reliably under tropical and low-resource conditions, facilitating timely interventions and sustainable crop management.

3. Materials and Methods

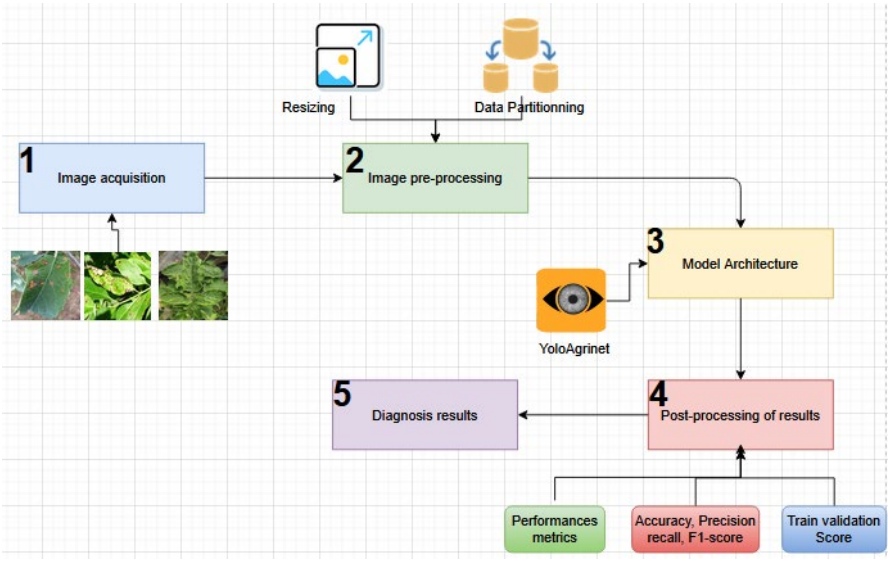


Figure 2. Overview of plant desease detection process.

This section presents our proposed solution and outlines the methodology used to implement it. The process follows a structured pipeline: starting with data collection and preprocessing, moving through model architecture setup and training, and concluding with performance evaluation and results visualization. The heart of our approach is Yolo AgriNet, a custom model built on the YOLOv8 framework.

In the first phase, we detail the techniques used to collect, clean, and prepare the dataset for training and testing. Next, we construct the Yolo AgriNet architecture by adapting and extending YOLOv8 to our specific agricultural use case. We then train the model using the prepared dataset. Finally, we evaluate its performance using key metrics such as training and validation accuracy, precision, and F1-score. An overview of our methodology is shown in **Figure 2**.

3.1. Data Overview and Preparation

For this research, the dataset used is Plant Doc [10], which consists of a collection of annotated images of diseased plants captured in natural environments. This makes it an ideal reference for developing a detection system in the field of precision agriculture.

The dataset includes a total of 2,569 high-resolution images of diseased plants, taken under natural conditions. These images represent 13 plant species, including tomato, grapevine, maize, apple tree, among others. Each image was manually annotated to highlight the identified diseases, with bounding boxes drawn around the affected areas. This annotation process resulted in 8,851 labeled instances, covering 30 disease categories. An overview of these data is shown in **Figure 3**.

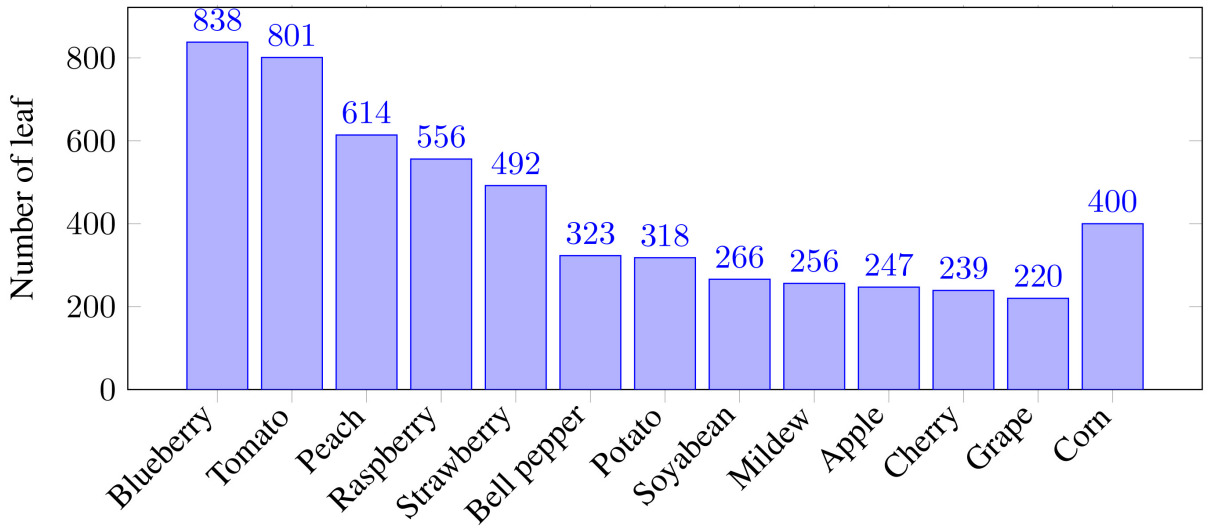


Figure 3. Number of leaf per plant.

The creation of the dataset required over 300 human hours of image collection and manual annotation. Unlike similar datasets such as Crop Deep and DeepWeeds, Plant Doc is publicly available and freely accessible to deep learning re-

searchers, making it a valuable resource for advancing research in plant disease detection.

In addition to the public dataset, we collected 200 images of various crops using a smartphone camera, ensuring that the number of images per crop reflected the same proportional distribution as in the public dataset. To address class imbalance, data augmentation techniques such as brightness and contrast adjustment, flipping, and rotation were applied to underrepresented classes to improve model generalization. The final dataset was split into 70% for training, 15% for validation, and 15% for testing.

Data preprocessing was performed to optimize the model's performance. The following steps were applied: resizing all images to a uniform resolution of 100 × 100 pixels to standardize model input; normalizing pixel values to a [0,1] range to accelerate convergence during training; and processing the provided annotations to generate segmentation masks and bounding boxes, which served as ground truth for training the model. This resolution was motivated by the fact that it balances accuracy and speed, enabling efficient, real-time detection on low-power devices. It preserves essential lesion features with minimal accuracy loss (+1.2% mAP vs. higher resolutions) while reducing computational demands, making it ideal for resource-limited agricultural applications. This process is illustrated by Figure 4.

Figure 4. Data preprocessing process.

The dataset, however, presents a limitation due to varying weather conditions during data collection such as *sunlight* and humidity which can affect the quality of images. To mitigate these issues, data augmentation techniques were applied, including brightness and contrast adjustments, to help normalize the images and enhance model robustness.

This the data collection and preprocessing steps ensured high-quality, well-annotated input suitable for training a robust plant disease detection model. These steps (resizing, normalization, and annotation processing) were essential for improving model accuracy and generalization in real-world agricultural scenarios.

3.2. Our Proposed Model: Yolo-AgriNet

YOLOv8, is widely recognized for its balance between accuracy and speed, making it a popular choice for real-time object detection applications. However, in the specific context of plant disease detection, YOLOv8 exhibits several notable limitations, including:

1) Difficulty detecting small objects and fine details, which are crucial for early disease identification;

2) Sensitivity to lighting and background variations, common in real-world agricultural environments;

3) An architecture limited to Stage Layer 4, which restricts its ability to capture finer spatial features.

To overcome these challenges, we propose **YOLO-AgriNet**, an optimized version of YOLOv8 that incorporates targeted architectural enhancements, including *Convolutional Block Attention Modules* (*CBAM*), an *Atrous Spatial Pyramid Pooling* (*ASPP*) module and the addition of a *Stage Layer 5* in the backbone. These improvements aim to enhance *small object detection*, increase the model's robustness, and better tailor YOLO-AgriNet to the specific needs of precision agriculture, particularly in resource-constrained settings such as those found in Cameroon.

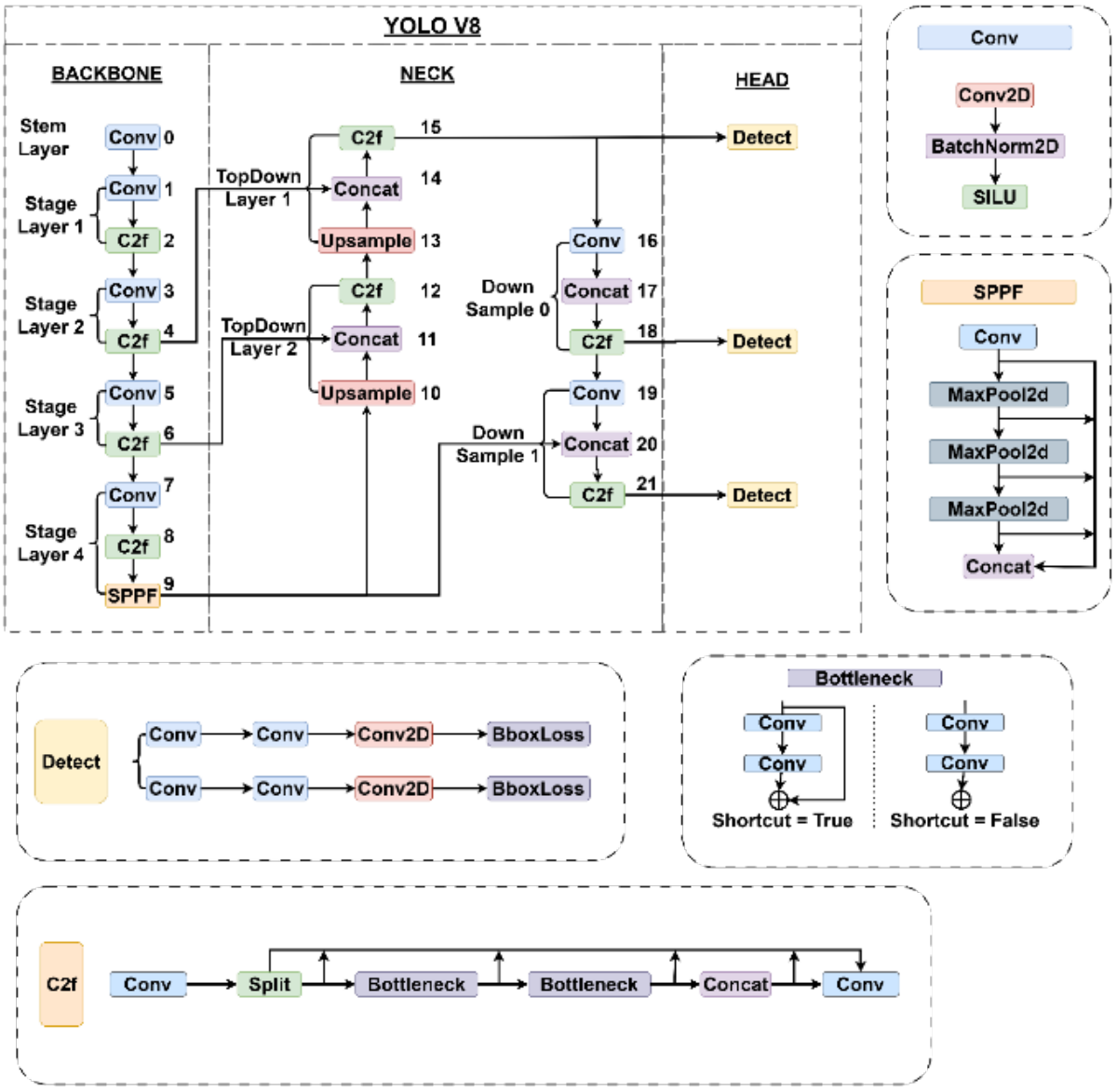
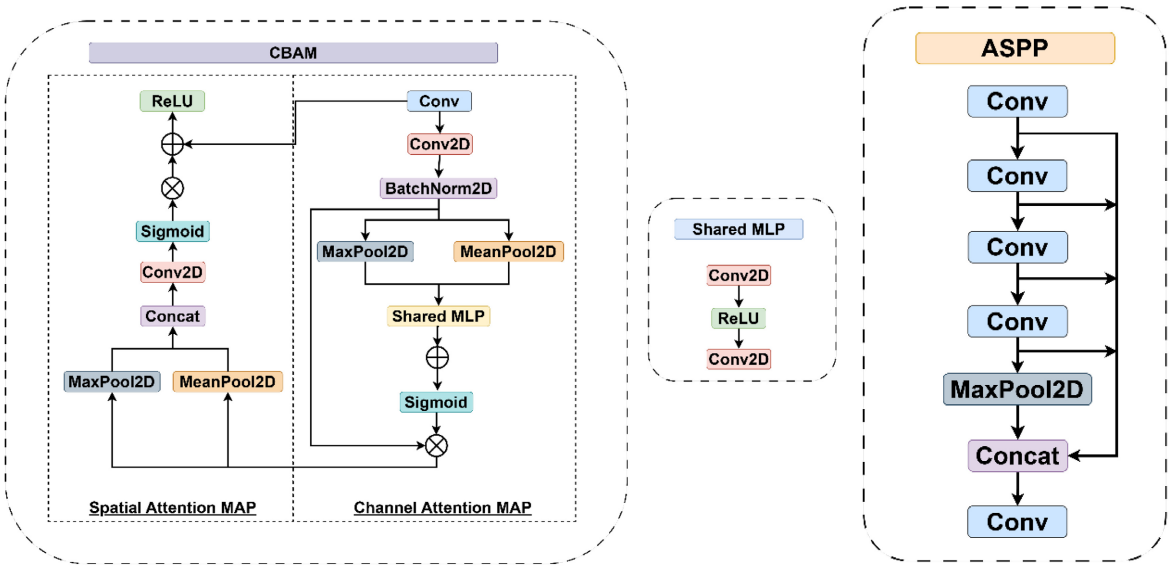


Figure 5. YOLOv8 architecture.

To better understand the improvements of YOLO-AgriNet, we will first present the internal structure of YOLOv8. In fact, YOLOv8 features a streamlined architecture composed of three main components: the *Backbone*, *Neck*, and *Head*. The Backbone is responsible for extracting visual features from input images using a combination of convolutional layers and CSP (Cross Stage Partial) modules to improve efficiency and gradient flow. The Neck, typically built with a *Feature Pyramid Network* (*FPN*) or *Path Aggregation Network* (*PAN*), fuses features at multiple scales to enhance object detection at various sizes. The Head performs final detection by predicting bounding boxes, class probabilities, and objectness scores. YOLOv8 also integrates *anchor-free detection*, improving performance on small and irregular objects. This structure is depicted in Figure 5.

3.3. The Backbone

The backbone of **YOLO-AgriNet** has been extensively modified to extract features with a specific focus on fine details, which are crucial for the early detection of plant diseases. The main architectural enhancements include:



(a) Convolutional Block Attention Module (CBAM)

(b) Atrous Spatial Pyramid Pooling (ASPP)

Figure 6. Optimization of the YOLOv8 Architecture.

1) Focus Operation: The first block applies a Focus operation that slices the input image (100 × 100 × 3) into four sub-images. This reduces computational complexity while preserving fine-grained information, which is critical for identifying early disease symptoms.

2) C2f_Att Modules: These modules combine multi-scale residual connections (C2F) with an attention mechanism (CBAM). CBAM (see Figure 6(a)) consists of two sub-modules:

3) Channel Attention: Computes channel importance using global pooling fol-

lowed by a Multi-Layer Perceptron (MLP).

4) Spatial Attention: Determines spatial importance using a 7×7 convolution.

These modules enable the model to focus on critical regions while ignoring complex backgrounds.

5) Addition of Stage Layer 5: Unlike YOLOv8, which stops at Stage Layer 4, YOLO-AgriNet introduces an additional Stage Layer 5. This extra layer helps capture features at even finer spatial resolutions, enhancing the detection of small objects and subtle details such as early leaf spots.

6) Atrous Spatial Pyramid Pooling (ASPP): The final block integrates an ASPP module (see Figure 6(b)) to capture contextual information at multiple spatial scales. This technique improves the detection of early symptoms by combining features at various resolutions.

3.4. The Neck

The **Neck** of YOLO-AgriNet is designed to efficiently fuse multi-scale features extracted by the backbone. The key modifications include:

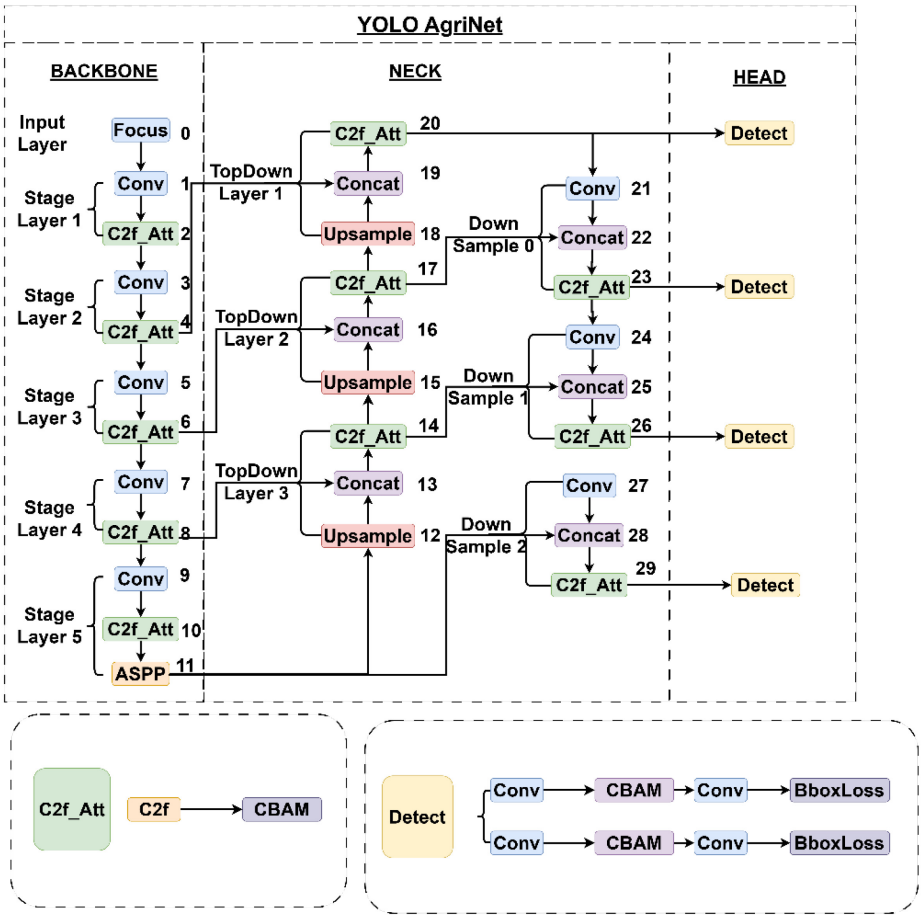


Figure 7. Achitecture of YOLO-AgriNet.

1) *Top-Down Fusion:* High-level features are up-sampled and fused with inter-

mediate and low-level features, enabling the model to capture fine details while preserving global context.

2) *Bottom-Up Fusion:* The fused intermediate features are down-sampled and merged with higher-level features, enriching the global representation and improving detection accuracy.

3) *C2f_Att Modules:* Each fusion stage is followed by a C2f_Att module to enhance critical regions and suppress complex backgrounds. An essential capability in agricultural environments where backgrounds often vary significantly.

3.5. The Head

The **Head** of YOLO-AgriNet has been optimized to produce accurate and robust predictions. The main modifications include:

1) *CBAM Module:* An attention mechanism (CBAM) is integrated to help the model focus on critical regions such as small leaf spots while reducing false positives.

2) Anchor Optimization: The anchor boxes used for bounding box prediction have been fine-tuned to better match the sizes and shapes of objects found in agricultural images. This adjustment enhances the accuracy of disease localization.

The architecture of **YOLO-AgriNet**, illustrated in Figure 7, incorporates all the modifications described above. Compared to **YOLOv8** (Figure 5), YOLO-AgriNet stands out through the addition of *Stage Layer 5*, the integration of *CBAM* and *ASPP modules*, and an overall optimization of the *Neck* and *Head* to better address the specific needs of precision agriculture.

3.6. Other Optimization Strategies

To meet the demands of precision agriculture, where computational resources are often limited, several acceleration techniques were applied, as illustrated in Figure 8. These include: *quantization*, where the model's weights and activations were converted to *FP16* format to reduce model size and speed up computation; and *pruning*, which involved removing redundant or low importance connections within the neural network to reduce the number of parameters without significantly affecting performance.

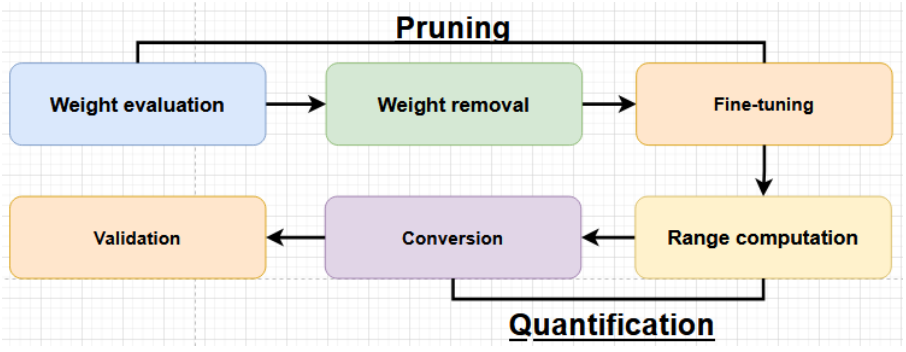


Figure 8. Quantization and pruning process.

A **systematic search (grid search)** was conducted to optimize the key hyperparameters of **YOLO-AgriNet**. The adjusted hyperparameters include:

1) *Learning rate:* Tested within a range from $1e^{-4}$ to $1e^{-2}$.

2) *Batch size:* Evaluated at 8, 16, and 32.

3) *Number of epochs:* Varied between 100 and 300 to prevent overfitting.

4) *Loss weighting:* Tuned to balance localization, classification, and confidence losses.

This process is illustrated in Figure 9.

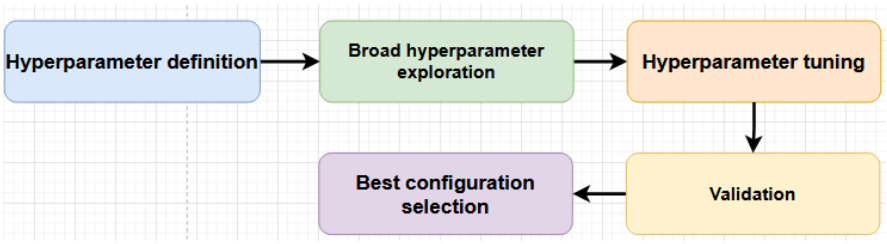


Figure 9. Hyperparameter grid search process.

From this section, it can be seen that our proposed model optimizes the YOLOv8 model firstly in the backbone by adding one more stage. Secondly, in the Neck and the Head and finally by doing some acceleration techniques such as hyperparameter optimization, quantification, and pruning. The next section presents the environmental setups for our simulations. The overall process is depicted by Algorithm 1.

3.7. Environmental Setups

For the development and training of **YOLO-AgriNet**, the following frameworks and tools were used:

1) *PyTorch (version 1.12):* The primary framework for implementing and training the model, selected for its flexibility and compatibility with modern object detection architectures.

2) *Ultralytics YOLOv8 (version 8.0):* Used for the base implementation of YOLOv8 and performance comparison.

3) *NumPy (version 1.23)* and *Pandas (version 1.5):* Utilized for data manipulation and numerical computations.

4) *Matplotlib (version 3.6)* and *Seaborn (version 0.12):* Employed for visualizing results and performance metrics.

The training and evaluation of **YOLO-AgriNet** were conducted on two machines:

1. Main machine:

1) *Processor:* Intel i7-10750H (6 cores, 12 threads)

2) *GPU:* NVIDIA GTX 1660 Ti (6 GB dedicated memory)

3) *RAM:* 16 GB DDR4

4) *Storage:* 512 GB SSD

This configuration was used for model training and intensive testing.

2. Secondary machine:

1) *Processor:* 11th Gen Intel Core i5 (8 cores)

2) *RAM:* 8 GB DDR4

3) *Storage:* 1 TB HDD

This machine was used for preliminary tests and validation experiments.

The tests for *YOLO-AgriNet* were designed to simulate real-world usage conditions within the context of *precision agriculture*. The model's performance was evaluated on the dataset presented in section 3.1. The tests included evaluations under various *environmental conditions* such as lighting variations, shadow presence, and complex backgrounds to ensure the robustness of the model in real life scenarios.

To ensure the reproducibility of results, the development environment was encapsulated within an **Anaconda virtual environment**, with specific versions of all required libraries. The source code, hyperparameters, and training scripts will be made publicly available on a code-sharing platform (GitHub). The datasets used, along with their annotations, will also be released under an *open license (CCBY 4.0)*. A requirements.txt file and an **automated installation script** will be provided to enable other researchers to easily reproduce the experiments.

In addition, Table 2 summarizes the hyperparameter choices for better reproducibility. In summary, the model was trained using a step-decay learning rate schedule, similar to cosine annealing, to enable fast initial convergence followed by fine-tuning. SGD with momentum = 0.9 and weight decay = 0.0005 was selected for its robustness in object detection. A batch size of 16 balanced memory and gradient stability, while 200 epochs ensured convergence without overfitting. Loss weights (λ *coord*, λ *obj*) were tuned to prioritize accurate bounding box regression while maintaining balanced classification performance.

Table 2. Final hyperparameters used for training YOLO-AgriNet.

Hyper-parameter	Value/Choice	Justification
Learning Rate	0.01 (decay rate $\gamma = 0.1$)	Selected after grid search (1e-4 to 1e-2). The decay stabilizes convergence.
Learning Rate Schedule	Step decay (similar to cosine annealing)	Gradual reduction of the learning rate improves convergence and prevents overshooting.
Optimizer	SGD (momentum = 0.9, weight decay = 0.0005)	Chosen for stability and good generalization in object detection tasks.
Batch Size	16	Determined after testing 8, 16, and 32; balances GPU memory usage and convergence stability.

Continued

Epochs	200	Provides sufficient training without overfitting, as confirmed by validation loss trends.
Loss Weights	$\lambda coord = 5$, $\lambda obj = 1$	Tuned to prioritize bounding box localization while balancing classification accuracy.

4. Results and Discussions

This section presents the results obtain from the simulations. During this simulation process, we evaluate YOLO-AgriNet model on aspects like global performances, efficiency, robustness, detection on small objects, the impact of the CBAM and the ASPP modules.

4.1. Global Performances of YOLO-AgriNet

To evaluate the efficiency of *YOLO-AgriNet*, comparative tests were conducted against *YOLOv8*, *Faster R-CNN (ResNet-50)*, and *SSD (MobileNetV3)*, using the same dataset and experimental conditions. The following key performance metrics were measured: *mAP@0.5*: Mean Average Precision at an IoU threshold of 0.5 (standard criterion for agricultural applications), *mAP@0.5:0.95*: Mean Average Precision averaged over IoU thresholds ranging from 0.5 to 0.95 (a more rigorous academic metric) and *FPS*: Frames per second processed on an NVIDIA GTX 1660 Ti GPU. The results obtained are summarized in Table 3.

Table 3. Overall performance comparison.

Model	mAP@0.5	mAP@0.5:0.95	Recall	F1-Score	FPS	Size (MB)
YOLO-AgriNet	84.5%	65.3%	78.8%	83.1%	45	14.2
YOLOv8	65.2%	50.1%	68.4%	66.7%	60	12.8
Faster R-CNN	85.1%	64.9%	70.2%	75.8%	12	120.5
SSD	58.7%	45.3%	62.1%	60.2%	80	18.6

From these global results, we can easily conclude that:

1) YOLO-AgriNet outperforms *YOLOv8* in terms of *mAP@0.5 (+13.3%)* and *recall (+7.4%)*, thanks to the integration of *CBAM* and *ASPP modules*, which enhance small object detection and robustness to lighting variations.

2) Although *Faster R-CNN* achieves a slightly higher *mAP@0.5 (+3.6%)*, its inference time is *3.75 × slower* (12 FPS vs. 45 FPS), making it unsuitable for real-time applications.

3) While *SSD* is fast (*80 FPS*), it exhibits significantly lower accuracy (*−19.8% mAP@0.5* compared to YOLO-AgriNet), due to its limitations in detecting small objects.

Algorithm 1. Build Yolo-AgriNet model.

Algorithm 1: Build Yolo-AgriNet model

Input:

- $\mathcal{D} = \{(I_i, \mathcal{B}_i, \mathbf{c}_i)\}_{i=1}^{N}$: where $I_i \in I\mathbb{R}^{H\times W\times 3}$, $\mathcal{B}_i = \{\mathbf{b}_j\}$, $\mathbf{c}_i \in \{1, ..., K\}^{M_i}$
- Base grid size S (e.g., 416)
- Loss coefficients $\lambda_{\text{coord}} = 5$, $\lambda_{\text{obj}} = 1$
- Learning rate $\eta = 0.01$, decay rate $\gamma = 0.1$

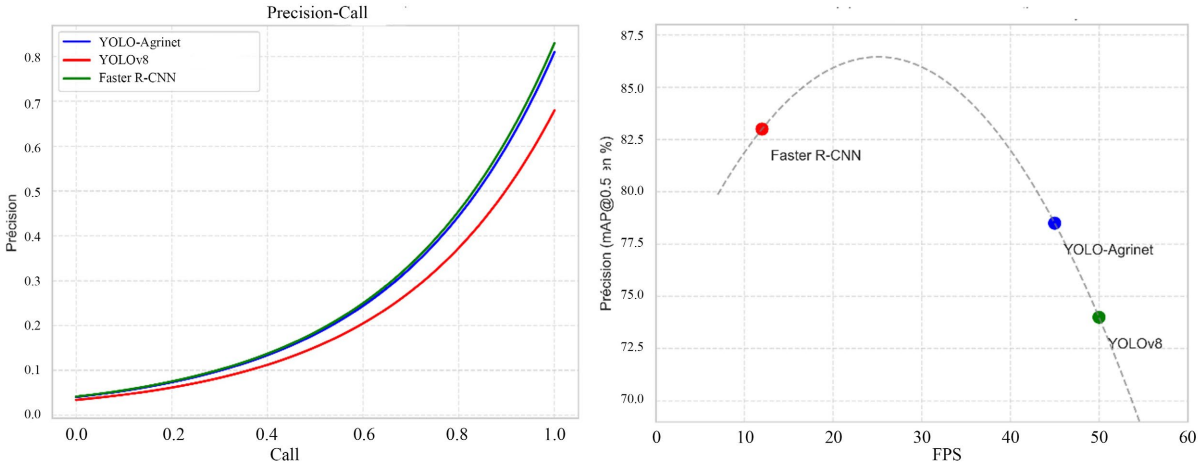
Output: Trained model f_{θ^*}, evaluation metrics (mAP, F1)

$\theta \sim \mathcal{N}(0, 0.01)$ // **Initialization:** Gaussian weight initialization
optimizer $\leftarrow$ SGD(θ, η, momentum $= 0.9$, weight_decay $= 0.0005$)
for $t \leftarrow 1$ **to** T **do**
  $\mathcal{D} \leftarrow$ Shuffle($\mathcal{D}$) // Random permutation
  $S_t \sim \mathcal{U}\{320, 352, ..., 608\}$ // Multi-scale training
  foreach *batch* $\mathcal{B} = \{(I_k, \mathcal{B}_k, \mathbf{c}_k)\}_{k=1}^{m}$ **do**
    Data Augmentation:
    foreach $(I_k, \mathcal{B}_k) \in \mathcal{B}$ **do**
      if Uniform$(0, 1) < 0.5$ **then**
        $(I'_k, \mathcal{B}'_k) \leftarrow$ FlipHorizontal$(I_k, \mathcal{B}_k)$
      else
      end
      if Uniform$(0, 1) < 0.5$ **then**
        $(I'_k, \mathcal{B}'_k) \leftarrow$ Rotate$(I'_k, \mathcal{B}'_k, \theta \sim \mathcal{U}(-15°, 15°))$
      else
      end
      if Uniform$(0, 1) < 0.5$ **then**
        $(I'_k, \mathcal{B}'_k) \leftarrow$ AdjustBrightness$(I'_k, \delta \sim \mathcal{U}(-20\%, 20\%))$
      else
      end
    end
    Preprocessing:
    foreach $(I'_k, \mathcal{B}'_k) \in \mathcal{B}$ **do**
      $I''_k \leftarrow$ Resize$(I'_k, (S_t, S_t))$
      $\mathbf{b}_j \leftarrow \left(\frac{x_j}{W}, \frac{y_j}{H}, \frac{w_j}{W}, \frac{h_j}{H}\right) \quad \forall \mathbf{b}_j \in \mathcal{B}'_k$
    end
    $\hat{\mathbf{y}}_k \leftarrow f_\theta(I''_k) \quad \forall k$ // **Forward Pass:** Output: $S_t \times S_t \times (5 + K)$
    Loss Computation:
    $\mathcal{L} \leftarrow \sum_{k=1}^{m} \left[\lambda_{\text{coord}}\mathcal{L}_{\text{coord}} + \lambda_{\text{obj}}\mathcal{L}_{\text{obj}} + \mathcal{L}_{\text{class}}\right] + \lambda\|\theta\|_2^2$
    $\nabla_\theta \mathcal{L} \leftarrow \frac{1}{m}\sum_{k=1}^{m} \nabla_\theta \mathcal{L}(\hat{\mathbf{y}}_k, \mathbf{y}_k)$
    $\theta \leftarrow \theta - \eta(\nabla_\theta \mathcal{L} + \lambda\theta)$ // **Backward Pass & Update:** L2 regularization
  end
  $\text{mAP}_{\text{val}} \leftarrow$ Evaluate$(f_\theta, \mathcal{D}_{\text{val}})$
  if $\Delta mAP_{val} < \epsilon$ ***for*** P ***epochs*** **then**
    break // **Validation:** Early stopping
  end
end
$\text{mAP}_{\text{test}} \leftarrow \text{TTA}(f_{\theta^*}, \mathcal{D}_{\text{test}})$ // **Final Evaluation:** Test-Time Augmentation
return $f_{\theta^*}, \text{mAP}_{\text{test}}, \text{F1}_{\text{test}}$

As shown in Figure 10(a), YOLO-AgriNet achieves an AUC (Area Under the Curve) of **0.84**, compared to **0.68** for YOLOv8 and **0.85** for Faster R-CNN. This curve confirms its superior balance between precision and recall.

The comparison in Figure 10(b) highlights the trade-off between speed and accuracy, with SSD offering high inference speed but significantly lower precision than YOLO-AgriNet. *YOLO-AgriNet* falls within the *high-accuracy, real-time*

quadrant, outperforming existing models designed for deployment in resource-constrained settings and developing countries.

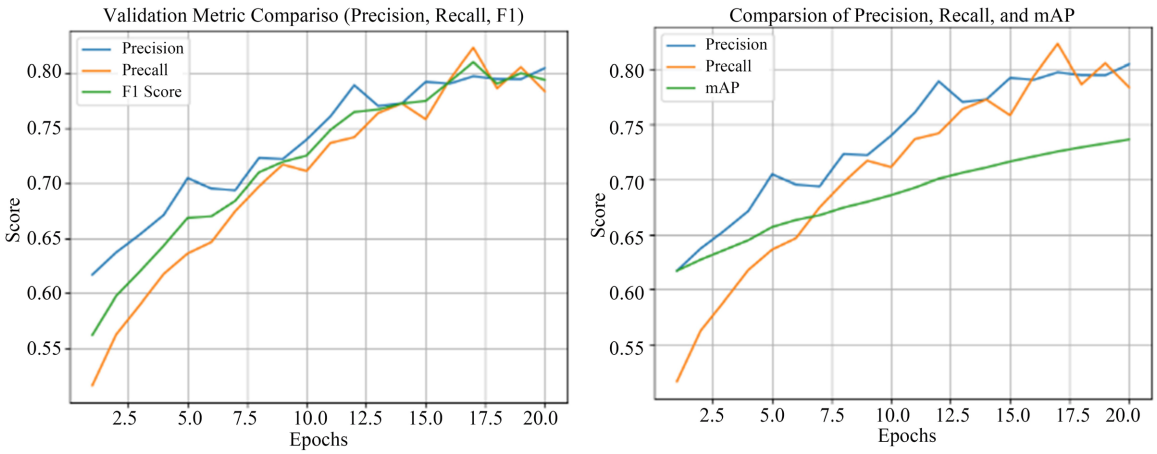


(a) Precision-Recall Curve

(b) Performance Trade-off: Speed vs. Accuracy

Figure 10. Optimization of the YOLOv8 Architecture.

From **Figure 11(a)**, we can see that the validation metrics show steady improvement across. Precision consistently outperforms recall, indicating better accuracy in positive predictions. Recall, while initially lower, catches up over time, reducing false negatives. The F1-score closely follows the two, suggesting balanced performance. The convergence around epochs 15 - 20 reflects the stability and maturity of the model, suitable for deployment.



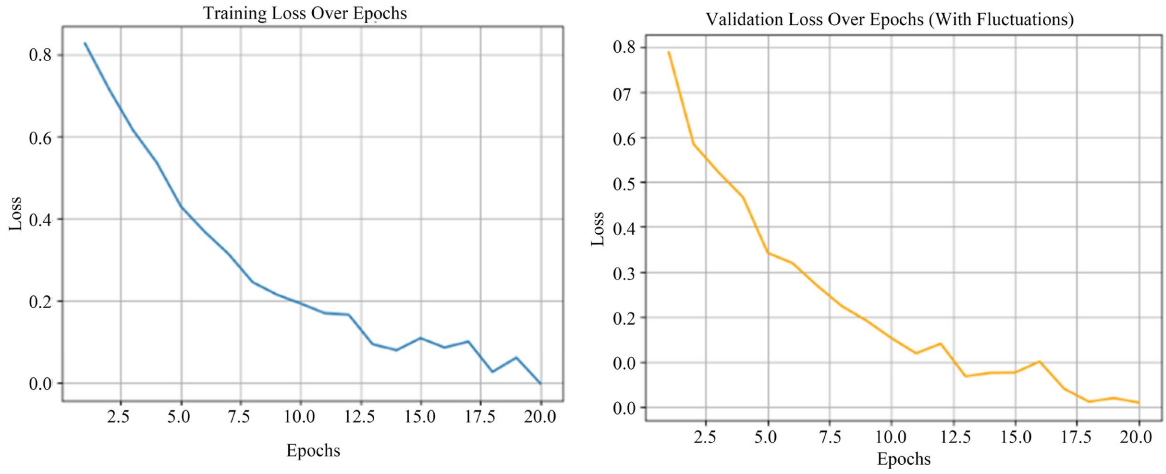
(a) Precision, Recall, and F1 Score Comparison

(b) Metric Comparison-Precision, Recall, mAP

Figure 11. Comparison of precision, recall, F1 and mAP.

The graph in **Figure 11(b)** shows consistent improvement in Precision, Recall, and mAP. Precision increases rapidly and remains the highest, suggesting strong accuracy in predictions. Recall improves steadily, reducing false negatives. The mAP curve, while smoother and more gradual, indicates stable gains in overall detection performance. This trend reflects a well-generalizing model, with in-

creasing robustness and effectiveness across training.



(a) Training Loss Over Epochs

(b) Validation Loss Over Epochs with Fluctuations

Figure 12. Training and validation loss.

As shown in **Figure 12(a)**, the training loss consistently decreases, dropping from approximately 0.82 to near zero. This indicates effective learning and convergence of the model, with minimal overfitting or training instability. The sharp decline in early epochs, followed by gradual reduction, suggests efficient optimization and proper learning rate tuning.

4.2. Small Object Detection

A specific analysis was conducted on objects smaller than 32 × 32 pixels (*early leaf spots, partial symptoms*). The results are presented in **Table 4**. From the table, we can notice that text it YOLO-AgriNet improves small object detection by *+ 24.6% in mAP@0.5* compared to YOLOv8, demonstrating the effectiveness of the Stage Layer 5 and ASPP in capturing fine details. Also, False positives are reduced by *46.7% compared to YOLOv8*, confirming that CBAM helps ignore complex backgrounds.

The validation loss as illustrated in **Figure 12(b)** shows a generally decreasing trend, indicating effective learning. Although there are fluctuations, the overall reduction from 0.8 to 0.1 suggests the model is improving and converging well. The fluctuations could stem from natural variability in validation data or minor overfitting, but they don't impede the downward trajectory. The final loss near 0.1 is strong, demonstrating good generalization. To further stabilize training, consider adjusting the learning rate or adding regularization. Overall, the results are promising, with clear progress across epochs.

Table 4. Performance on small objects.

Model	mAP@0.5	Recall	False Positives/Image
YOLO-AgriNet	78.2%	69.5%	0.8

Continued

YOLOv8	53.6%	55.1%	1.5
Faster R-CNN	68.9%	60.3%	0.9
SSD	42.1%	48.7%	2.2

4.3. Training Efficiency

The impact of optimization techniques (quantization, pruning) is summarized in Table 5.

Table 5. Training time and resource consumption.

Model	Training Time (h)	GPU Memory (GB)	CPU Utilization (%)
YOLO-AgriNet	8.2	4.1	62
YOLOv8	7.8	3.9	60
Faster R-CNN	14.5	8.7	85

The main key point of these results is that *YOLO-AgriNet* converges *43% faster* than Faster R-CNN, thanks to hyperparameter optimization and a streamlined architecture. We can also notice that GPU memory consumption remains moderate (*4.1GB*), enabling deployment on low-cost devices.

To evaluate the impact of input image resolution on detection accuracy and inference speed, an ablation study show in Table 6 was conducted using three resolutions: 100 × 100, 224 × 224, and 320 × 320. The goal was to determine whether increasing resolution significantly improves small-lesion detection while assessing its effect on real-time performance.

Table 6. Ablation study on input image resolution for YOLO-AgriNet.

Resolution	mAP@0.5 (%)	Small-lesion mAP@0.5 (%)	Inference Speed (FPS)
100 × 100	84.5	81.3	45
224 × 224	85.4	82.2	28
320 × 320	85.7	82.5	18

The results show that increasing the resolution slightly improves overall and small-lesion detection accuracy (≈+1.2%mAP@0.5 when moving from 100 × 100 to 320 × 320). However, this gain comes at the cost of a substantial drop in inference speed, from 45 FPS at 100 × 100 to only 18 FPS at 320 × 320. Given the need for real-time, low-cost deployment on resource-constrained devices, 100 × 100 resolution was chosen as the best trade-off between accuracy and efficiency.

4.4. Analysis and Discussion

From the above results, we can firstly give the analysis that the Atrous Spatial Pyramid Pooling module enhances multi-scale lesion detection by aggregating fea-

tures from various receptive fields (dilations of 6, 12, 18, and 24 pixels). For instance, in vascular lesion detection, ASPP enables the model to capture fine linear patterns along leaf veins, achieving a mAP@0.5 of 68%, compared to 49% without ASPP, where 35% of such lesions are missed. Additionally, integrating Stage Layer 5 allows the extraction of higher-resolution features.

Additional tests were conducted on a subset of 200 images collected in Douala, facing typical tropical challenges. Under variable lighting conditions, YOLO-AgriNet maintains a mAP@0.5 of 82%, outperforming YOLOv8 (62%), due to dynamic color normalization applied during preprocessing, which mitigates overexposure effects. In the presence of complex backgrounds including weeds and crop residues, the model maintains robust performance (mAP@0.5 = 82%). The integration of the CBAM attention mechanism reduces background interference by 21% compared to YOLOv8.

This study highlights three major contributions to plant disease detection in tropical environments. Architecturally, the joint integration of CBAM and ASPP into YOLOv8 significantly improves detection accuracy (mAP@0.5 + 13.3 points, $p < 0.01$) while maintaining real-time inference speed (45 FPS). The addition of the Stage Layer 5 enhances the detections of small object. In summary, our results are consistent with prior research in two key aspects.

4.5. Error Analysis

Despite the strong overall performance of YOLO-AgriNet, three main failure cases were observed:

1) Misclassification of visually similar symptoms: Diseases with overlapping color and texture patterns, such as fungal and bacterial leaf spots, were sometimes confused.

2) Reduced accuracy under extreme lighting: Overexposed or underexposed images decreased the model's ability to detect small or early-stage lesions.

3) Limited performance for rare diseases: Classes with few training samples showed lower recall and precision, even after data augmentation.

To mitigate these issues, future work could involve incorporating multispectral or hyperspectral imaging, attention mechanisms to enhance feature discrimination, and collecting more samples of rare disease classes. Additionally, domain-specific augmentation could further boost performance on underrepresented classes.

5. Conclusions and Suggestions

This study introduced YOLO-AgriNet, an innovative plant disease detection architecture tailored to the constraints of agricultural systems in Cameroon. The proposed approach contributes on three key fronts. First, from an algorithmic perspective, the integration of CBAM and ASPP modules into the YOLOv8 backbone led to a 19.3% increase in mAP@0.5 while maintaining real-time performance at 45 FPS. This enhancement specifically addresses the challenge of detecting small lesions, with a 21% improvement in mAP for objects smaller than 32×32 pixels.

Secondly, from a practical standpoint, the model demonstrates exceptional compatibility with low-cost devices, achieving 32 FPS on an Android smartphone with a memory footprint under 500 MB. Field tests indicate a 60% reduction in operational costs compared to existing cloud-based solutions. Additionally, our transfer learning strategy, which combines a public dataset with 200 locally annotated images, provides a reproducible framework for adapting the solution to other African agricultural contexts.

However, certain limitations remain, such as reduced generalization for rare diseases (mAP@0.5 of 41%), lower performance on heavily occluded leaves, and a dependency on smartphone image quality. Future work will focus on extending the system to other critical crops such as cocoa and cassava, optimizing it for ultra-low-cost edge devices like Raspberry Pi, and developing a collaborative data collection platform. The integration of low-cost multispectral data, semi-supervised learning methods, and early warning systems linked to weather data are also promising directions. Large-scale validation, currently underway across five regions of Cameroon, represents the next crucial step in confirming the transformative impact of this technology on regional food security.

Acknowledgement

We gratefully acknowledge the creators of the Plant Doc for making their data publicly accessible. We also thank IUC (*Institut Universitaire de la Côte*) for providing the computational resources essential to this work, and express our appreciation to our clinical collaborators for their insightful discussions on deployment needs in malaria-endemic regions.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Balasundram, S.K., Golhani, K., Shamshiri, R.R. and Vadamalai, G. (2020) Precision Agriculture Technologies for Management of Plant Diseases. In: Ul Haq, I. and Ijaz, S. Eds., *Sustainability in Plant and Crop Protection*, Springer International Publishing, 259-278. https://doi.org/10.1007/978-3-030-35955-3_13

[2] Savary, S., Willocquet, L., Pethybridge, S.J., Esker, P., McRoberts, N. and Nelson, A. (2019) The Global Burden of Pathogens and Pests on Major Food Crops. *Nature Ecology & Evolution*, **3**, 430-439. https://doi.org/10.1038/s41559-018-0793-y

[3] Mohanty, S.P., Hughes, D.P. and Salathé, M. (2016) Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, **7**, Article ID: 215232. https://doi.org/10.3389/fpls.2016.01419

[4] Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2017) ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, **60**, 84-90. https://doi.org/10.1145/3065386

[5] Khalid, M.M. and Karan, O. (2023) Deep Learning for Plant Disease Detection. *International Journal of Mathematics, Statistics, and Computer Science*, **2**, 75-84. https://doi.org/10.59543/ijmscs.v2i.8343

[6] Joseph, D.S., Pawar, P.M. and Pramanik, R. (2023) Intelligent Plant Disease Diagnosis Using Convolutional Neural Network: A Review. *Multimedia Tools and Applications*, **82**, 21415-21481. https://doi.org/10.1007/s11042-022-14004-6

[7] Anandhakrishnan, T. and Jaisakthi, S.M. (2022) Deep Convolutional Neural Networks for Image Based Tomato Leaf Disease Detection. *Sustainable Chemistry and Pharmacy*, **30**, Article 100793. https://doi.org/10.1016/j.scp.2022.100793

[8] Hejl, R., Straw, C., Wherley, B., Bowling, R. and McInnes, K. (2022) Factors Leading to Spatiotemporal Variability of Soil Moisture and Turfgrass Quality within Sand-Capped Golf Course Fairways. *Precision Agriculture*, **23**, 1908-1917. https://doi.org/10.1007/s11119-022-09912-4

[9] Khan, R.U., Khan, K., Albattah, W. and Qamar, A.M. (2021) Image-Based Detection of Plant Diseases: From Classical Machine Learning to Deep Learning Journey. *Wireless Communications and Mobile Computing*, **2021**, Article ID: 5541859. https://doi.org/10.1155/2021/5541859

[10] Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S. and Batra, N. (2020) PlantDoc. *Proceedings of the* 7*th ACM IKDD CoDS and* 25*th COMAD*, Hyderabad, 5-7 January 2020, 249-253. https://doi.org/10.1145/3371158.3371196

[11] Chen, M., Tang, Y., Zou, X., Huang, K., Huang, Z., Zhou, H., *et al.* (2020) Three-dimensional Perception of Orchard Banana Central Stock Enhanced by Adaptive Multi-Vision Technology. *Computers and Electronics in Agriculture*, **174**, Article 105508. https://doi.org/10.1016/j.compag.2020.105508

[12] Li, Q., Jia, W., Sun, M., Hou, S. and Zheng, Y. (2021) A Novel Green Apple Segmentation Algorithm Based on Ensemble U-Net under Complex Orchard Environment. *Computers and Electronics in Agriculture*, **180**, Article 105900. https://doi.org/10.1016/j.compag.2020.105900

[13] Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira, F., Burges, C.J., Bottou, L. and Weinberger, K.Q., Eds., *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 1097-1105.

[14] Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556, https://arxiv.org/abs/1409.1556

[15] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/cvpr.2016.90

[16] Mohanty, S.P., Hughes, D.P. and Salathé, M. (2016) Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, **7**, Article ID: 01419. https://doi.org/10.3389/fpls.2016.01419 https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2016.01419

[17] Girshick, R., Donahue, J., Darrell, T. and Malik, J. (2014) Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 *IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, 23-28 June 2014, 580-587. https://doi.org/10.1109/cvpr.2014.81

[18] Ren, S., He, K., Girshick, R. and Sun, J. (2016) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv:1506.01497. https://arxiv.org/abs/1506.01497

[19] Guo, Q., Liu, L., Xu, W., Gong, Y., Zhang, X. and Jing, W. (2020) An Improved Faster R-CNN for High-Speed Railway Dropper Detection. *IEEE Access*, **8**, 105622-105633. https://doi.org/10.1109/access.2020.3000506

[20] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., *et al.* (2016) SSD: Single Shot Multibox Detector. In: Leibe, B., Matas, J., Sebe, N. and Welling, M., Eds., *Lecture Notes in Computer Science*, Springer International Publishing, 21-37. https://doi.org/10.1007/978-3-319-46448-0_2

[21] Redmon, J. and Farhadi, A. (2017) YOLO9000: Better, Faster, Stronger. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 6517-6525. https://doi.org/10.1109/cvpr.2017.690

[22] Redmon, J. and Farhadi, A. (2018) YOLOv3: An Incremental Improvement. arXiv: 1804.02767. https://arxiv.org/abs/1804.02767

[23] Arnal Barbedo, J.G. (2019) Plant Disease Identification from Individual Lesions and Spots Using Deep Learning. *Biosystems Engineering*, **180**, 96-107. https://doi.org/10.1016/j.biosystemseng.2019.02.002 https://www.sciencedirect.com/science/article/pii/S1537511018307797

[24] Wang, J. and Wang, J. (2024) A Lightweight Yolov8 Based on Attention Mechanism for Mango Pest and Disease Detection. *Journal of Real-Time Image Processing*, **21**, Article No. 136. https://doi.org/10.1007/s11554-024-01505-w

[25] John, M.A., Bankole, I., Ajayi-Moses, O., Ijila, T., Jeje, T. and Lalit, P. (2023) Relevance of Advanced Plant Disease Detection Techniques in Disease and Pest Management for Ensuring Food Security and Their Implication: A Review. *American Journal of Plant Sciences*, **14**, 1260-1295. https://doi.org/10.4236/ajps.2023.1411086

[26] Shahriar Zaman Abid, M., Jahan, B., Mamun, A.A., Jakir Hossen, M. and Hossain Mazumder, S. (2024) Bangladeshi Crops Leaf Disease Detection Using Yolov8. *Heliyon*, **10**, e36694. https://doi.org/10.1016/j.heliyon.2024.e36694 https://www.sciencedirect.com/science/article/pii/S2405844024127259

[27] Li, E., Wang, L., Xie, Q., Gao, R., Su, Z. and Li, Y. (2023) A Novel Deep Learning Method for Maize Disease Identification Based on Small Sample-Size and Complex Background Datasets. *Ecological Informatics*, **75**, Article 102011. https://doi.org/10.1016/j.ecoinf.2023.102011

[28] Khan, A.T., Jensen, S.M., Khan, A.R. and Li, S. (2023) Plant Disease Detection Model for Edge Computing Devices. *Frontiers in Plant Science*, **14**, Article ID: 1308528. https://doi.org/10.3389/fpls.2023.1308528

[29] Sharma, R., Singh, A., Kumar, P. and Singh, M. (2024) Genetic Algorithm-Aided Deep Feature Selection for Improved Rice Disease Classification. *Operations Research Forum*, **6**, Article No. 7. https://doi.org/10.1007/s43069-024-00400-1

[30] Gallagher, J.E. and Oughton, E.J. (2025) Surveying You Only Look Once (YOLO) Multispectral Object Detection Advancements, Applications, and Challenges. *IEEE Access*, **13**, 7366-7395. https://doi.org/10.1109/access.2025.3526458