Sensors and Materials, Vol. 37, No. 7 (2025) 3005–3023
MYU Tokyo

S & M 4099

Constructing Multiclass Object Detection Model for Precision Rice Crop Agriculture

Wen-Tsai Sung,[1] Indra Griha Tofik Isa,[2,3] and Sung-Jung Hsiao[4*]

[1]Department of Electrical Engineering, National Chin-Yi University of Technology, Zhongshan Rd, Section 2, No. 57, Taichung City 411030, Taiwan
[2]Graduate Institute, Prospective Technology of Electrical Engineering and Computer Science, National Chin-Yi University of Technology, Zhongshan Rd, Section 2, No. 57, Taichung City 411030, Taiwan
[3]Department of Informatics Management, Politeknik Negeri Sriwijaya, Palembang City 30139, Indonesia
[4]Department of Information Technology, Takming University of Science and Technology, Taipei City 11451, Taiwan

(Received April 30, 2025; accepted June 17, 2025)

Keywords: multiclass object detection, rice plant, precision agriculture, lightweight object detection

Precision agriculture (PA) is a technology that integrates AI into the agricultural field. One of the uses of PA is to increase the productivity of rice plants, which is a staple food in high demand. In this study, PA technology based on vision computing will be developed, where the multiclass intelligent object detection model will be embedded into edge computing devices. The proposed model, which is called YOLO-Rice, is an improved version of You Only Look Once version 8 (YOLOv8), in which attention modules including GhostNet and the convolutional block attention module (CBAM) are combined to enhance the detection performance and model efficiency. There are 12 classes divided into three categories: rice grain, rice leaves, and insect pests. The stages of model development are carried out through dataset development, model construction, model evaluation and validation, and model deployment. The comparative experiment incorporates YOLOv5, YOLOv8, YOLOv8+CBAM, YOLOv8+GhostNet, and YOLOv9. The mean average precision value (mAP$_{50}$) of YOLO-Rice was the most optimal compared with other deep learning algorithms, attaining an accuracy percentage of 91.95%. Overall, based on the experimental results, YOLO-Rice has outstanding results in terms of detection accuracy and model efficiency.

1. Introduction

Agricultural production is one of the essential activities in Taiwan that support domestic and export needs. Exports increased in 2022 and 2023, especially for processed vegetables, soybean, and tea products.(1) In contrast, domestic production, especially rice crops, which are the staple food of the Taiwanese people, was recorded to have decreased from 2019 to 2022.(2) Constraints related to the decline in rice crop production include a decrease the number of supporting labor and an increase the population of aging farmers, irrigation systems, limited agricultural areas, and less-than-optimal growth monitoring processes.(3) In this case, a breakthrough solution that

*Corresponding author: e-mail: sungjung@gs.takming.edu.tw
https://doi.org/10.18494/SAM5714

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

can increase rice crop production, even with limited resources, is needed. The implementation of technology related to Industrial Revolution 4.0 has given rise to a new terminology, called “precision agriculture (PA)", which is a solution to increasing the rice crop production in terms of quality and quantity efficiently, quickly, cheaply, and sustainably.(4) On the other hand, the edge computing mechanism has the advantage of more efficient and faster computational processing, because the computational process is carried out at the networking node or the edge of the crop area. Therefore, data speed and security can be increased, latency and the amount of data sent can be reduced, and real-time calculation is possible.

Several studies related to the implementation of object detection models in rice crop fields have been conducted. Du *et al.* constructed the You Only Look Once-progressive spatial feature pyramid (YOLOv7-PSAFP), in which PSAFP implemented in YOLOv7 to perform rice-corn pest detection.(5) In addition to calculating the object loss, the authors utilized the two loss functions: varifocal loss and loss rank mining. The experimental stage indicates that the proposed model has an mean average precision (mAP) of 2.9% and 2.1% higher than that of YOLOv7.(5) Jain *et al.* incorporated YOLOv3-Tiny and YOLOv4-Tiny as assistance frameworks in a smartphone application called “E-crop doctor” to detect three common paddy leaf diseases: hispa, leaf blast, and brown spot.(6) Deng *et al.* developed YOLOX to detect tiny objects such as weeds in the area of rice crops. The model improves the backbone network, and at the experimental stage, has a better detection task performance than those of several existing models, such as YOLOv3, YOLOv4-tiny, YOLOv5s, and single shot detector (SSD).(7) Unmanned aerial vehicle (UAV) T-YOLO-Rice has been developed to detect rice leaf disease, which is originally the enhancement of YOLOv4-Tiny. Several improvements are integrated into the original model, including a convolutional block attention module, spatial pyramid pooling, sand clock feature extraction module, and ghost modules.(8) Meanwhile, Li *et al.* constructed TLI-YOLO, that is, transfer layer iRMB-YOLOv8.(9) The purpose of this model is to perform rice disease identification using the modified structure of YOLOv8. The model uses an iRMB attention module to integrate the inverted residual block and transformer. The experimental result shows the TLI-YOLO model having 7.6% higher mAP_{50} than the original model.(9)

In this study, we will integrate PA and computer vision through an embedded lightweight object detection model, called YOLO-Rice. In addition, an edge computing system will be incorporated to perform object detection tasks in practical applications. The proposed model was developed by modifying a state-of-the-art YOLOv8 through the incorporation of GhostNet architecture in the backbone section, replacing the original backbone CSPDarknet53 architecture, to create a lightweight network and generate redundant feature maps with the aim of accelerating data processing and reducing memory consumption while still maximizing the feature extraction process. The attention module will also be implemented in the “neck area” of YOLOv8, which employs the convolutional block attention module (CBAM), an integration of the channel attention module and spatial attention module. The purpose of CBAM is to improve the feature extraction performance of the model to produce more accurate and detailed object detection performance in detecting tiny objects. The GhostNet architecture and CBAM implemented in YOLOv8 make up the YOLO-Rice model, which, in this study, will be employed to detect complex multiple classes such as rice crop pests, conditions indicating whether the rice

leaves are normal or have a disease, and conditions indicating whether the rice grains are optimal, deflated, or damaged. Therefore, the contributions of this study include the following.
(1) Development of PA in which an edge computing system and computer vision are integrated in a rice field
(2) Development of a multiclass dataset for the rice crop detection model, which consists of 12 classes and is defined by three main categories, rice grain, rice leaves, and insect pests
(3) Enhancement of the model detection performance and model efficiency by integrating YOLOv8 and the attention modules GhostNet and CBAM

2. Materials and Methods

2.1 Edge computing technology

In a modern control system, efficiency, speed, and data connectivity are the main factors in data processing, so the emerging technology is edge computing. Edge computing brings the computation process close to the location where the data are captured. This allows for a fast computational process, efficient data distribution with a decentralized architecture, less memory consumption, and reduced latency. With the development of AI, edge computing is not limited to only data processing and representing information in real time.(10) However, machine learning and deep learning technology can be adopted to provide comprehensive data. In general, there are three layers within the edge computing architecture: physical layer, edge layer, and cloud layer.(11) The improved architecture depicted in Fig. 1 is adopted in this study. The data analysis model is embedded to provide accurate data, an intelligent system, problem-solving complexity, and a robust and integrated control system. The challenge is to create a model that has outstanding performance even with limited resources such as memory capacity. Thus, re-engineering and the improvement of data analysis models are also necessary to produce lightweight models and robust intelligent edge computing.

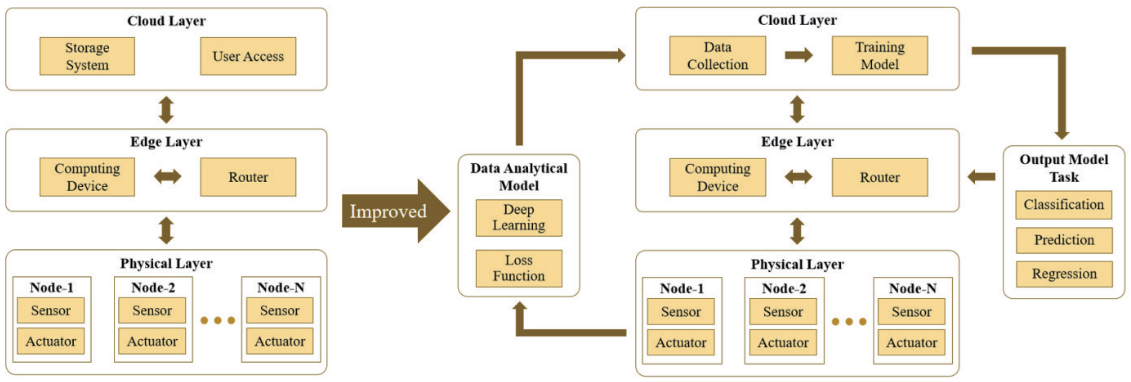

Fig. 1. (Color online) Improvement of edge computing.

2.2 Overview of YOLOv8 model

Since the YOLO-series object detection models were introduced and especially with the emergence of the state-of-the-art YOLOv4 architecture, which is defined as having a backbone, neck, and head, the YOLO series has developed significantly, attaining increasingly better accuracy, optimal detection performance for tiny objects, and stability.(12) YOLOv8 was introduced in 2023 by Ultralytics, who also invented YOLOv5, and is based on the Pytorch framework and has faster computing, making it ideal for multi-object detection and tracking; it is user-friendly, has better image classification performance, and excellent prediction performance.(13) YOLOv8 is highly suitable for deployment in edge computing in terms of the memory allocation consumption and high-inference speed, for which all variants of the model have outstanding performance results.(14) There are three improvements in YOLOv8 compared with the previous version, YOLOv5. The first is in the backbone, as represented by the exchange of the C3 module in YOLOv5 to the C2f module so that YOLOv8 is more lightweight. The idea of C2f is based on the ELAN module in YOLOv7, which obtains more gradient flow information while ensuring a light weight. The details of the C2f module can be seen in Fig. 2.

Meanwhile, the spatial pyramid pooling (SPP) module in YOLOv5 is replaced by spatial pyramid pooling–fast (SPPF) in YOLOv8. The SPPF module employs a less computationally intensive variable-sized pooling window.(15) The improvement can enhance the accuracy of the model, as depicted in Fig. 3.

The second improvement is in neck structure, which comprises the path aggregation network (PAN) and the feature pyramid network (FPN). FPN is the feature extraction architecture for the

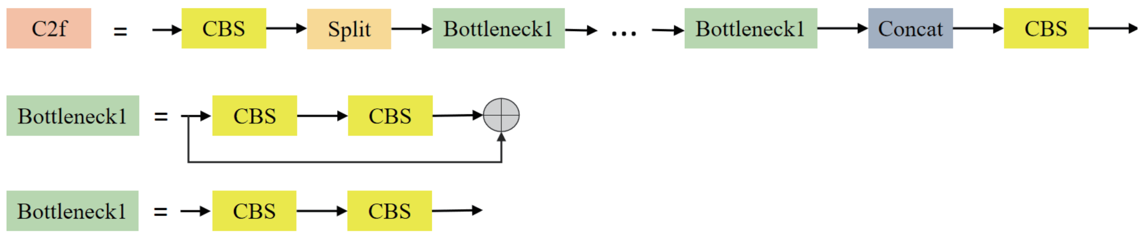

Fig. 2. (Color online) Detailed architecture of C2f module.

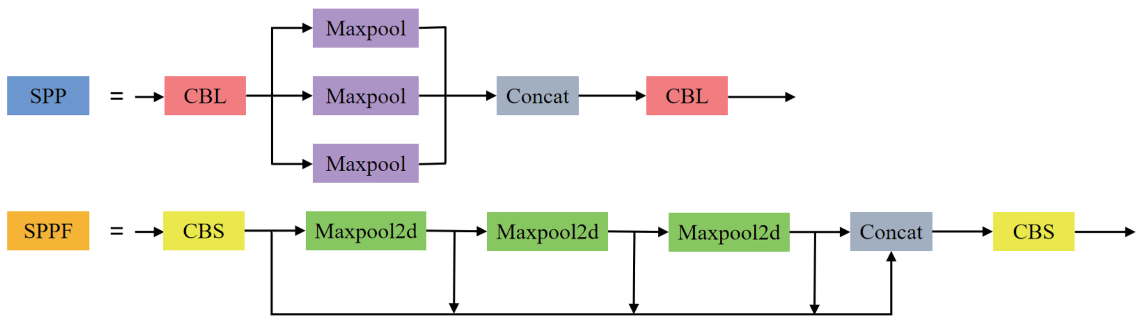

Fig. 3. (Color online) SPP and SPPF modules.

object detection task. In the improved YOLOv8, the convolutional layer was deleted before the upsampling module, and at the same time, the C3 module was replaced by the C2f module. The third is to optimize the head structure by introducing anchorless detection, which predicts the center of the object directly, avoiding misclassification, reducing multiple bounding box generation, and speeding up the process of nonmaximum suppression (NMS). YOLOv8 also employs the varifocal loss (VFL) or zooming loss as the classification loss; meanwhile, the regression loss of YOLOv8 is the integration of CIoU and distributed focal loss (DFL). In addition, VFL can dynamically scale sample categories of varying difficulty.[16] The weight improves the model's ability to detect difficult-to-classify samples while reducing the difficulty of easy-to-classify samples, hence increasing training efficiency and overall model accuracy. A new asymmetric loss function that weighs both positive and negative samples is incorporated in the VFL loss function. The following formulation represents the VFL loss function.

$$VFL(x,y)=\begin{cases}-x\big(y\log(x)+(1-y)\log(1-x)\big) & y>0\\ -\alpha x^{\gamma}\log(1-x) & y=0\end{cases} \tag{1}$$

The complete intersection over union (CIoU) ensures precise bounding boxes compared with conventional methods and is well aligned, whereas DFL focuses on improving regression in complex problems. The goal is to enhance the accuracy and efficiency of bounding box predictions. YOLOv8 balances accuracy and efficiency by integrating CIoU with DFL, where CIoU is expressed as below.

$$Loss_{CIoU} = 1 - IoU + \frac{\varsigma^{2}\left(\mathscr{b}, \mathscr{b}^{gt}\right)}{\mathbb{C}^{2}} \tag{2}$$

$$\alpha = \frac{\mathscr{v}}{(1 - IoU) + \mathscr{v}} \tag{3}$$

$$\mathscr{v} = \frac{4}{\pi^{2}}\left(\arctan\frac{\omega^{gt}}{\psi^{gt}} - \arctan\frac{\omega}{\psi}\right)^{2} \tag{4}$$

α is used to balance the proportion value, and $\mathscr{v}$ is used to maintain the balance ratio between the predicted frame and the actual frame. $\mathscr{b}$ and $\mathscr{b}^{gt}$ are the center points of the predicted and actual frame, respectively. ς indicates the measurement of Euclidean distance between the two center points. The diagonal distance of the smallest rectangle containing both the predicted and true boxes is represented as $\mathbb{C}$. The fundamental idea behind DFL is that instead of detecting a fixed bounding box, DFL predicts the probability distribution of bounding box coordinates to account for the uncertainty of the location. DFL employs the cross-entropy function for optimization. The probabilities of left and right placements near label y are based on the network's distribution in the neighborhood of the label value. This enables the model to quickly identify values near the label, as represented by Eq. (5), where ρ represents the actual frame on the left side and ρ_{i+1} represents the predicted frame on the right side.

$$DFL\left(\mathcal{S}_i, \mathcal{S}_{i+1}\right) = -\left(\left(\rho_{i+1} - \rho\right)\log\left(\mathcal{S}_i\right) + \left(\rho - \rho_i\right)\log\left(\mathcal{S}_{i+1}\right)\right) \tag{5}$$

2.2 Proposed YOLO-Rice model

Our original YOLO-Rice is a novel YOLOv8 with improvements in the backbone and neck sections aimed at achieving efficient computational processes, less memory consumption, and high performance in detecting objects, as depicted in Fig. 4. In the backbone structure, the original architecture model will be replaced and changed to the GhostNet architecture for the purpose of generating redundant feature maps and replacing the convolution with the ghost module. In the neck structure, there is an additional attention module of CBAM to enhance the performance of feature extraction and improve the accuracy of object detection.

The ghost module employs a 1 × 1 convolution technique to reduce the channel number and generate a part of the actual feature layer. To create the ghost feature map, a depth-separable convolution is applied to each channel of the actual feature layer.(17) The final output feature layer is generated by merging the real and ghost feature layers. A ghost bottleneck (G-block) is a bottleneck structure created by the ghost module and can be separated into two parts. The first part is a bottleneck with a step size of one. As depicted in Fig. 5(a), the width and height of the input layer are not compressed and are used to deepen the network's depth. To extract features from the input layer, the backbone uses two ghost modules. Finally, the output from the backbone section and the remaining edge are combined. The second half is a bottleneck construction with a step size of two. In Fig. 5(b), the backbone extracts features using a ghost module and compresses the input layer width and height using a depth-separable convolution with a step size

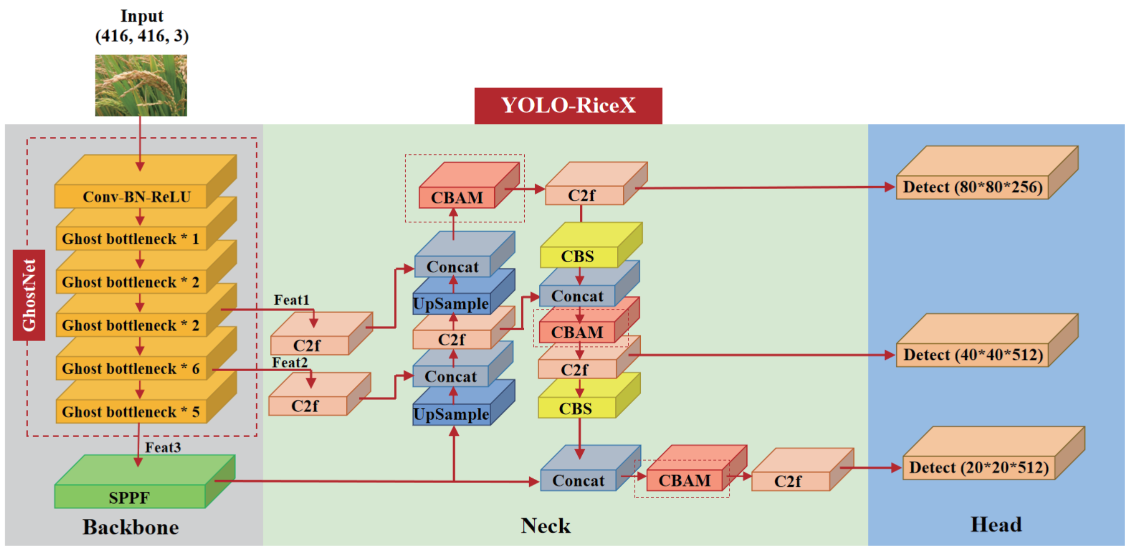

Fig. 4. (Color online) Proposed YOLO-Rice model.

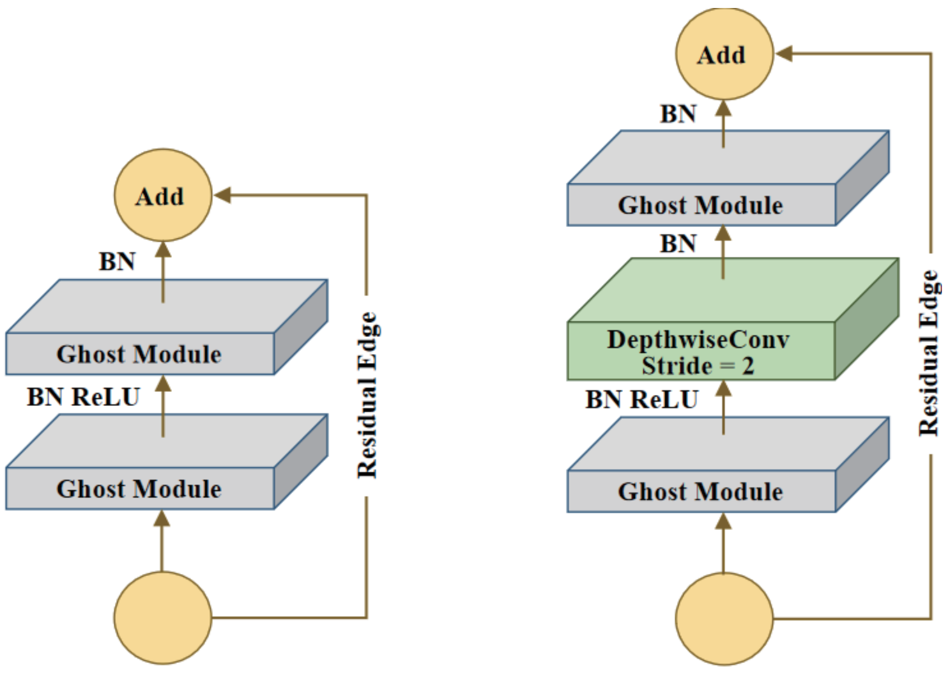

Fig. 5. (Color online) GhostNet bottleneck structure. (a) Step size of 1. (b) Step size of 2.

of 2. The ghost module is re-employed to extract features, and the results are summed with residual edges.

GhostNet is made of several ghost bottleneck structures that generate the three units called Feat1, Feat2, and Feat3, which have the feature map scales of 52 × 52 × 40, 26 × 26 × 112, and 13 × 13 × 160, respectively. Table 1 shows the network parameters of GhostNet.

CBAM was first introduced by Woo *et al.*(18) as an attention module that focuses on network learning of the important feature in both the channel dimension and the spatial dimension. CBAM has a small dimension but is effective for complicated data because it employs two attention modules: channel attention module (CAM) and spatial attention module (SAM). Figure 6 depicts the CBAM structure with CAM and SAM. The formulations of CAM and SAM are represented by Eqs. (6) and (7), respectively.

$$F_{CAM} = A_c\left(F\right) \otimes F \tag{6}$$

$$F_{SAM} = A_s\left(F_{CAM}\right) \otimes F_{CAM} \tag{7}$$

CAM employs elementwise summation to combine input characteristics and calculated output generated by CAM based on the identification of the most relevant semantic information channel, followed by interchannel interaction to increase the number of discriminative features.(19) To reconstruct image characteristics into multilayer perceptrons (MLPs) via weight (ω), two forms of pooling are used: average pool $\omega\left(F_{avg}^{c}\right)$ and max pool $\omega\left(F_{max}^{c}\right)$, as denoted in Eq. (8). Additionally, a sigmoid function (σ) is used to improve speed.

Table 1
Detailed parameters of GhostNet.

Structure	Operation	Step	Kernel	Input	Output
Feat1	Conv-BN-ReLU	2	(3, 3)	(416, 416, 3)	(208, 208, 16)
	Ghost-bottleneck 1	1	(3, 3)	(208, 208, 16)	(208, 208, 16)
	Ghost-bottleneck 2	2	(3, 3)	(208, 208, 16)	(104, 104, 24)
		1	(3, 3)	(104, 104, 24)	(104, 104, 24)
	Ghost-bottleneck3	2	(5, 5)	(104, 104, 24)	(52, 52, 40)
		1	(5, 5)	(52, 52, 40)	(52, 52, 40)
Feat2	Ghost-bottleneck4	2	(3, 3)	(52, 52, 40)	(26, 26, 80)
		2	(3, 3)	(26, 26, 80)	(26, 26, 80)
		1	(3, 3)	(26, 26, 80)	(26, 26, 80)
		1	(3, 3)	(26, 26, 80)	(26, 26, 80)
		1	(3, 3)	(26, 26, 80)	(26, 26, 112)
		1	(3, 3)	(26, 26, 112)	(26, 26, 112)
Feat3	Ghost-bottleneck5	2	(5, 5)	(26, 26, 112)	(13, 13, 160)
		1	(5, 5)	(13, 13, 160)	(13, 13, 160)
		1	(5, 5)	(13, 13, 160)	(13, 13, 160)
		1	(5, 5)	(13, 13, 160)	(13, 13, 160)
		1	(5, 5)	(13, 13, 160)	(13, 13, 160)

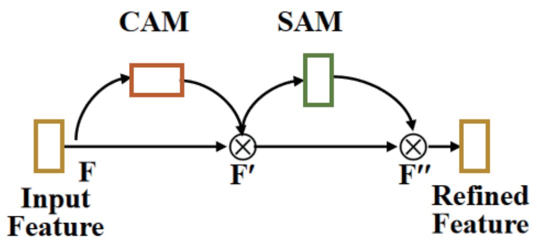

Fig. 6. (Color online) Structure of CBAM.

$$A_c(F) = \sigma\left(\omega_1\left(\omega_0\left(F_{avg}^c\right)\right) + \omega_1\left(\omega_0\left(F_{max}^c\right)\right)\right) \tag{8}$$

Simultaneously, SAM processes the feature's spatial information to determine the contribution of each pixel region in one channel to the rebuilt image.(20) The average and max pools are used to construct two 2D maps: $F_{avg}^s \in R^{1\times H\times W}$ and $F_{max}^s \in R^{1\times H\times W}$. A sigmoid function ($\sigma$) and a 7 × 7 convolutional method are used as denoted below.

$$A_s(F) = \sigma\left(f_{7\times 7}\left(\left[F_{avg}^s; F_{max}^s\right]\right)\right) \tag{9}$$

2.3 Construction of dataset

The rice image dataset used in the proposed project was obtained from the rice field industry in Wufeng District, Taichung City, Taiwan. The data were obtained for the six growth stages: tillering, stem elongation, panicle initiation, flowering, milk, and mature, as depicted in Fig. 7.

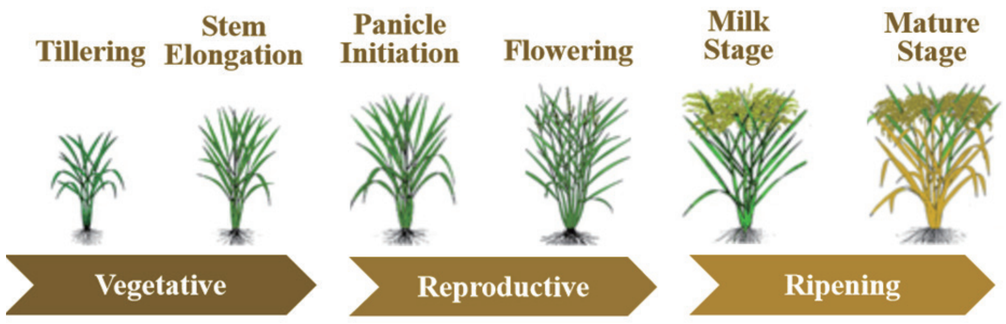

Fig. 7. (Color online) Six growth stages of rice crop.

This comprehensive data collection was carried out to ensure that the model can be used to monitor the condition of the rice plant during the vegetative phase, reproductive phase, and ripening phase. Therefore, the model can provide information on anomalies that occur during the plant growth process, the quality of the rice grains, whether the rice leaves grow optimally, and whether or not there are pests. To capture the images used for data acquisition, a Nikon Coolpix full HD 1080p digital camera was used at different angles from 45° to 90° and under different light conditions, such as nighttime and daytime.

For the model to detect the anomaly of insect pests on the rice, images of insect pests are collected to define eight types of common insect pest: (1) black bug, (2) zigzag leafhopper, (3) rice skipper, (4) rice thrip, (5) rice whorl maggot, (6) mealy bug, (7) mole cricket, and (8) ant. Figure 8 shows sample images of the insect pest dataset.

To increase the number of images to improve the performance of the model in recognizing objects, image augmentation through affine transformation techniques, including rotation, shear, and scaling, is needed.(21) Furthermore, to process the dataset in the modeling stage, the data is divided into training and validation data, and testing data at the percentages of 80 and 20%, respectively. The 12 classes of images are labeled or annotated using LabelImg software. Annotations make it easier for the model to recognize the context of the target object. In this study, annotation is conducted by superposing the bounding box on the target object. The format used is the YOLO file format, which will generate the .TXT file that consists of several elements such as class id, x-center, y-center, width of the bounding box, and height of the bounding box. Table 2 shows the detailed class image dataset.

The experimental computer environment comprised Intel Core i5-13500 13th Gen 2.50 GHz and 32 GB of RAM. The GPU is NVIDIA GeForce RTX 4070 with 12 GB of RAM. The dataset consists of the 12 class objects shown in Table 2. The total of 9040 images were divided into (1) training and validation data and (2) testing data at a ratio of 8:2, that is, training and validation data were obtained from 7200 images and testing data from 1840 images. The detailed dataset is shown in Table 3.

The initial parameters employed in the experiment include the learning rate of 0.001, 150 training epochs, and 64 batches as the training size in one iteration. In the experiment, several parameters were employed, including precision, recall, mean average precision (mAP), giga floating point operations per second (GFlops), and number of parameters (Params). Precision and recall are used to generate the AP and mAP metrics.(22) Meanwhile, GFlops is employed to

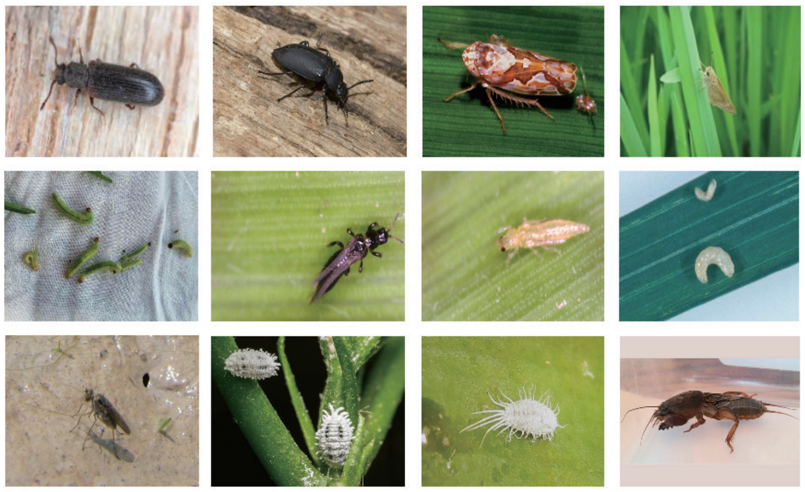

Fig. 8. (Color online) Insect pest image samples. Row 1 (Left to right: black bug, ant, zigzag leafhopper, rice skipper). Row 2 (Left to right: rice whorl maggot, mole cricket, rice thrip, and rice whorl maggot). Row 3 (Left to right: rice thrip, mealy bug, mealy bug, mole cricket).

Table 2
Class image of the dataset.

Classification object	Class name	Class ID	Image size
Rice grain	optimal_grain	Class 1	2100 × 1800
	not_optimal_grain	Class 2	2100 × 1800
Rice leaves	optimal_leaf	Class 3	1970 × 1520
	not_optimal_leaf	Class 4	1970 × 1520
Insect pest	black_bug	Class 5	742 × 669
	zigzag_leafhopper	Class 6	742 × 669
	rice_skipper	Class 7	742 × 669
	rice_thrip	Class 8	742 × 669
	rice_whorl_maggot	Class 9	742 × 669
	mealy_bug	Class 10	742 × 669
	mole_cricket	Class 11	742 × 669
	ant	Class 12	742 × 669

Table 3
Training, validation, and testing data of YOLO-Rice dataset.

Class name	Training and validation data	Testing data
optimal_grain	800	160
not_optimal_grain	800	160
optimal_leaf	800	160
not_optimal_leaf	800	160
black_bug	500	150
zigzag_leafhopper	500	150
rice_skipper	500	150
rice_thrip	500	150
rice_whorl_maggot	500	150
mealy_bug	500	150
mole_cricket	500	150
ant	500	150
Total	7200	1840

calculate the computational volume of the model, and the measurement of the model complexity is indicated by Params. The detailed metrics expressions are

$$Precision = {TP}\big/{\left(TP+FP\right)}, \tag{10}$$

$$Recall = {TP}\big/{\left(TP+FN\right)}, \tag{11}$$

$$mAP = \sum_{j=1}^{N} \frac{AP_j}{N}, \tag{12}$$

$$GFlops = C_o \left(\sum_{j=1}^{n} C_k \times C_{h\,j-1}^{2} \times C_i + \sum_{j=1}^{n} S^2 \times C_{h_i} \right), \text{ and} \tag{13}$$

$$Params = C_o \left(\sum_{j=1}^{n} S_j^2 \times C_{k_j}^2 \times C_{h_i-1} \times C_{h_i} \right), \tag{14}$$

where *TP*, *FP*, and *FN* represent true positive, false positive, and false negative, respectively. N indicates the number of values. Precision and recall indicate precision–recall curves. C_o is a constant order and C_k is the size of the convolution kernel. Channel number and input image size are represented by C_h and S, respectively.

3. Results and Discussion

In this section, we discuss the experiment, including the model performance on the dataset, model experiments with different deep learning algorithms, and the implementation of an intelligent model on the edge device. In addition, the findings are analyzed in depth to broaden the experimental insight.

3.1 Model performance on dataset

The proposed model training was conducted and the results are presented in Table 4: three indicator metrics (precision, recall, and AP_{50}) for 12 classes. For precision, all classes obtained percentage values higher than 80%. Three classes have values higher than 90%: 92.61, 91.98, and 90.56% for "not_optimal_grain", "mole_cricket", and "ant", respectively. For recall, the "zigzag_leafhopper" class has a percentage lower than 80%; otherwise, the other classes have percentages ranging between 80 and 85%. The parameter AP_{50} has percentages higher than 89%, with the lowest being 89.07% for "ant".

Figure 9 depicts AP_{50} for each class, where the top three percentages are 97.64, 95.15, and 94.27% for "black_bug", "not_optimal_leaf", and "optimal_leaf", respectively. The classes with AP_{50} percentages under 90% are "optimal_grain", "rice_thrips", "rice_whorl_maggot", "mealy_

Table 4
Model performance on dataset.

Class name	Precision (%)	Recall (%)	AP50 (%)
optimal_grain	89.23	82.43	89.65
not_optimal_grain	92.61	84.45	93.03
optimal_leaf	87.14	82.93	94.27
not_optimal_leaf	87.89	80.36	95.15
black_bug	88.74	83.35	97.64
zigzag_leafhopper	81.28	76.30	91.95
rice_skipper	89.98	84.31	91.55
rice_thrip	86.89	81.24	89.66
rice_whorl_maggot	85.98	82.36	89.23
mealy_bug	89.09	83.62	89.63
mole_cricket	91.98	84.07	92.52
ant	90.56	82.25	89.07

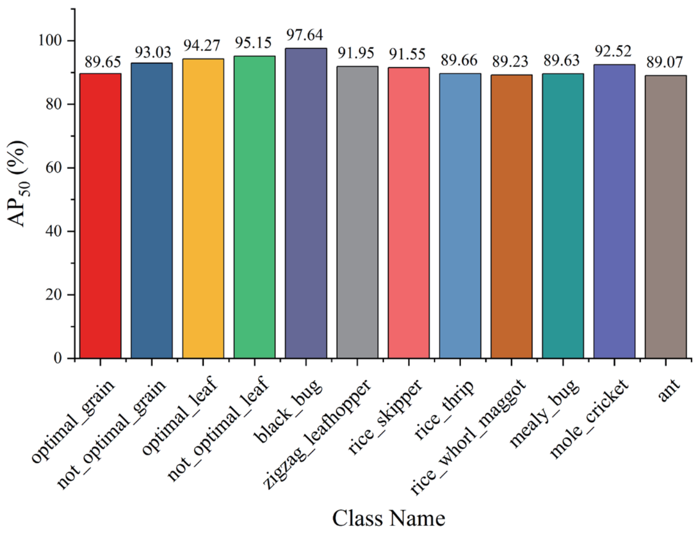

Fig. 9. (Color online) Results for the parameter AP50.

bug", and "ant". To validate and gain in-depth findings, comparative experiments and implementation in practical application should be conducted to evaluate the model objectively.

3.2 Ablation study

An ablation study was conducted to determine how the model compares with the original model, the combination of the single attention module and original model, and the proposed module where there are four experiments. The experimental configuration environment was Intel Core i5-13500 13th Gen 2.50 GHz and 32 GB of RAM. The GPU is NVIDIA GeForce RTX 4070 with 12 GB of RAM GDDR6X. The initial parameter consists of a learning rate of 0.001, 150 training epoch, and 64 batches as the training size in one iteration. The metrics mAP_{50}, GFlops, and Params were evaluated. Table 5 shows the results of the ablation study where four experiments were conducted. The first experiment employs the original model, namely, YOLOv8 with which $mAP_{0.5}$ was 80.11%, while GFlops and Params had the values of 82.8G and

Table 5
Ablation study experimental results.

Experiment	Original model	+ GhostNet	+ CBAM	mAP50 (%)	GFlops/G	Params/M
1	✓			80.11	82.8	64.1
2	✓	✓		87.32	7.2	11.0
3	✓		✓	85.41	83.7	65.8
4	✓	✓	✓	91.95	7.3	12.1

64.1M, respectively. However, GFlops and Params in the second experiment were approximately 91.30 and 82.84% lower than those of the original model.

In the third experiment, the accuracy with mAP_{50} of 85.41% was attained using the original model integrated with CBAM. The increase in mAP_{50} in this experiment is 6.69% compared with that in the first experiment, which was conducted with only the original model. In the third experiment, GFlops and Params were increased by approximately 1.09 and 2.65% or 83.7G and 64.1M in number of parameters, respectively. On the basis of the results of this third experiment, mAP_{50} increased, followed by the increase in GFlops and the number of parameters produced by the original model integrated with CBAM. In the fourth experiment, where the proposed model was used, the highest mAP_{50} of 91.95% among all experimental models was achieved. This is in line with the decrease in GFlops and the number of parameters, which were 7.3G and 12.1M, respectively. In this fourth experiment, GFlops and Params were reduced significantly, which had an impact on memory efficiency, but mAP_{50} remained optimal.

Figure 10 shows the mAP_{50} results of the ablation experiments. The y-axis represents the mAP_{50} percentage value and x-axis represents the experiment number that has been represented in Table 5. Experiment 1, which consists of the original model (YOLOv8), achieved the lowest mAP_{50} percentage compared with other experiments, with the percentage value of 80.11%. The combinations of single module, as represented in experiment 2 and 3, obtained the mAP_{50} percentages of 87.32% and 85.41%, respectively. Meanwhile, the proposed model (original model+GhostNet+CBAM) which is represented in experiment 4, has the highest mAP_{50} percentage, with the percentage value of 91.95%.

3.3 Comparative experiments with different deep learning algorithms

Comparative experiments were conducted to validate the proposed model performance in terms of both detection performance and model efficiency. The results are shown in Table 6. Eight models were used in the comparative experiment: YOLOv5, YOLOv8, YOLOv8+CBAM, YOLOv8+GhostNet, YOLOv9, YOLOv9+CBAM, YOLOv10, and YOLO-Rice, the proposed model. The model detection performance metrics are precision, recall, mAP_{50}, and $mAP_{50:95}$. Model efficiency is represented by GFlops/G and Params/M. YOLOv5, YOLOv8, and YOLOv8+CBAM have precision values under 80%: 68.34, 75.66, and 78.21%, respectively. YOLOv8+GhostNet, YOLOv9, YOLOv9+CBAM, YOLOv10, and YOLO-Rice achieved values greater than 80%: 80.04, 86.32, 86.47, 85.17, and 88.45%, respectively. The highest recall value of 84.26% was obtained with YOLOv10, whereas the YOLO-Rice model had a recall value of 82.31%. The combinations of the attention module and YOLOv8, such as YOLOv8+CBAM and YOLOv8+GhostNet, had recall values of 80.06 and 79.28%, respectively. Figure 11 depicts the

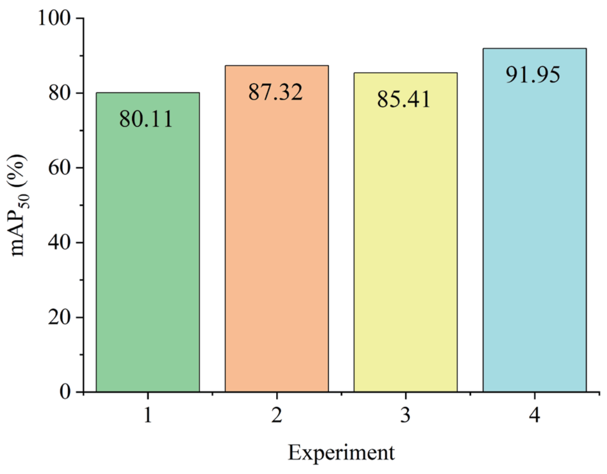

Fig. 10. (Color online) mAP50 results of ablation experiments.

Table 6
Comparative experiment results.

Deep learning algorithm	Precision (%)	Recall (%)	mAP$_{50}$ (%)	mAP$_{50:95}$ (%)	GFlops/G	Params/M
YOLOv5	68.34	70.24	75.35	55.24	84.1	70.2
YOLOv8	75.66	74.64	80.11	60.72	82.8	64.1
YOLOv8+CBAM	78.21	80.06	85.41	64.34	83.7	65.8
YOLOv8+GhostNet	80.04	79.28	87.32	70.04	7.2	11.0
YOLOv9	86.32	83.15	89.12	71.15	56.4	32.7
YOLOv9+CBAM	86.47	83.96	90.02	72.33	72.33	57.8
YOLOv10	85.17	84.26	89.44	70.24	70.24	52.7
YOLO-Rice	88.45	82.31	91.95	77.53	7.3	12.1

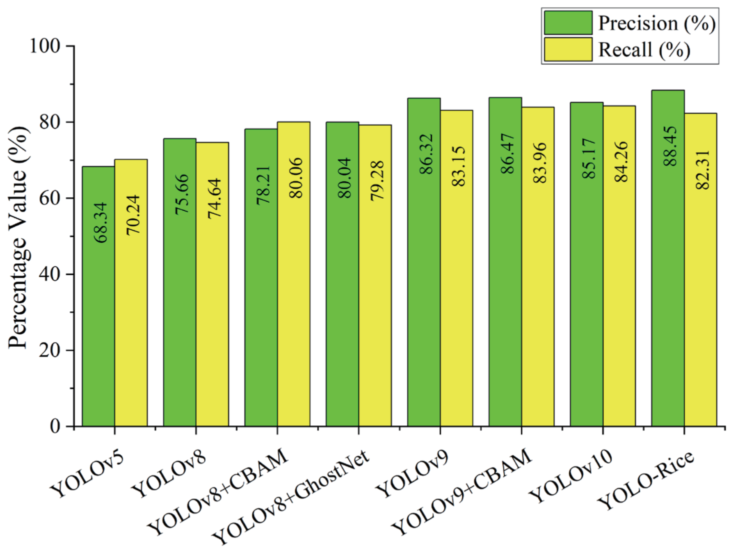

Fig. 11. (Color online) Precision and recall values with each different deep learning algorithm.

overall precision and recall values of each model with different deep learnings in the comparative experiments.

The number of parameters (Params) represents the computational complexity of the model. The greater the complexity of the model, the higher the computational cost. However, we must also consider the model detection performance metrics such as mAP_{50} and $mAP_{50:95}$. Overall, the top three highest numbers of parameters in this comparative experiment were obtained with YOLOv5, YOLOv8+CBAM, and YOLOv8, the parameter values being 70.2, 65.8, and 64.1 million parameters (M), respectively. YOLOv5 has a mAP_{50} value under 80%, whereas YOLOv8 has mAP_{50} of 4.76% higher than that of YOLOv5. The YOLOv8, YOLOv8+CBAM, YOLOv8+GhostNet, YOLOv9, and YOLOv10 models have mAP_{50} values between 80 and 89%. YOLOv9+CBAM and YOLO-Rice have mAP_{50} values of more than 90%: 90.02 and 91.95%, respectively. YOLOv5 has the lowest $mAP_{50:95}$ value of 55.24%. YOLOv8 and YOLOv8+CBAM have $mAP_{50:95}$ values of 60.72 and 64.34%, respectively. YOLO-Rice achieved the highest $mAP_{50:95}$ value of 77.53%.

Figure 12 shows the plot of mAP_{50} on the y-axis and number of parameters (Params/M) on the x-axis. The ideal condition is at the top-left position where the model has the highest value of mAP_{50} but a small value of Params/M. Params/M is the smallest value of 11.0M for the combination of YOLOv8+GhostNet. YOLO-Rice has a Params/M value of 12.1 M. However, the detection performance of YOLO-Rice is outstanding compared with those of the other models, exhibiting mAP_{50} of 91.95%, whereas YOLOv8+GhostNet has mAP_{50} of 87.32%. In the context of model efficiency, YOLO-Rice and YOLOv8+GhostNet show no significant gaps or difference in value. Meanwhile, in terms of detection performance represented by mAP_{50} and $mAP_{50:95}$, YOLO-Rice outperforms YOLOv8+GhostNet. Therefore, the comparative experiment results indicate that YOLO-Rice achieved the best performance in terms of both model performance and model efficiency.

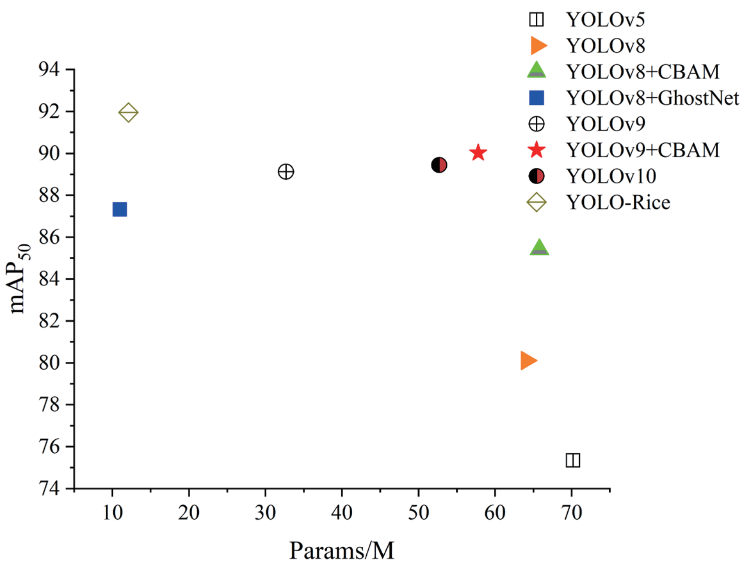

Fig. 12. (Color online) Model detection performance and model efficiency.

3.4 System deployment

The proposed model is deployed to use real-world data with the model embedded into an edge computing device. In this experiment, the hardware employed for the edge computing system is the NVIDIA Jetson Nano B01 version depicted in Fig. 13. Jetson Nano B01 enables several AI applications such as object identification, deep learning, and neural network applications. The advantages of this edge computing system are compact processing, low power consumption, and economical single-board computing solutions. It is compatible with several operating systems, including Linux, Ubuntu, and JetPack SDK. Jetson Nano is supported by the GPU of NVIDIA Maxwell architecture with the 128 NVIDIA CUDA® and CPU Quad-core ARM Cortex-A57 MPCore processor, with 4 GB of LPDDR4 64-bit, 1600 MHz 25.6 GB/s memory.

Figure 14 shows the configuration of system deployment where the input image, which is the current condition of the rice plant, is captured by camera vision sensing. The detected image is then transferred to the edge computing device, Jetson Nano B01, to be processed by the proposed model embedded into the edge device. Image processing and the object-detecting task are

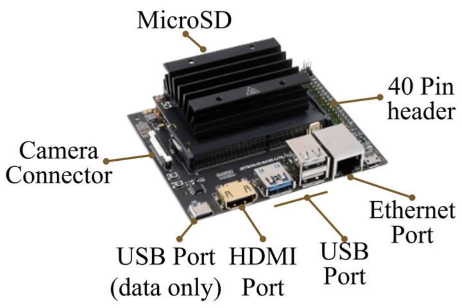

Fig. 13. (Color online) Jetson Nano B01.

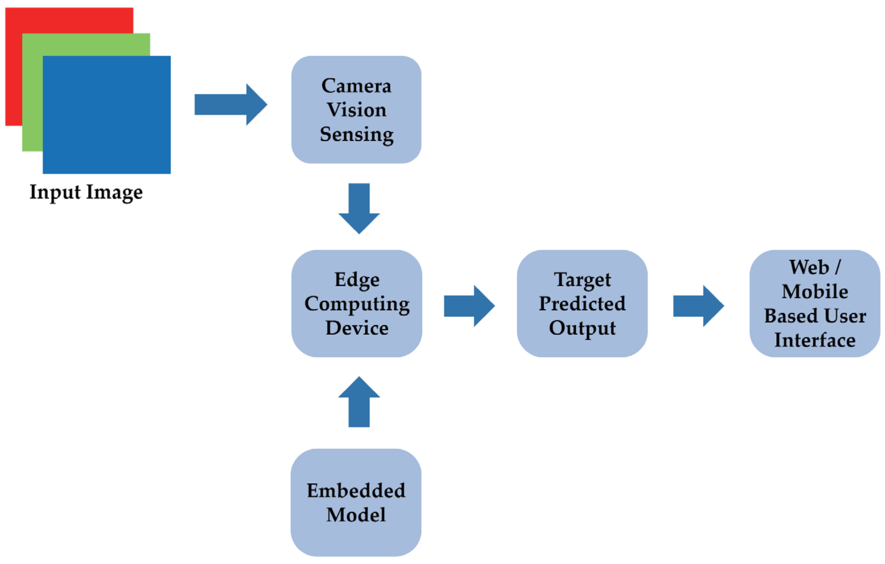

Fig. 14. (Color online) System deployment configuration.

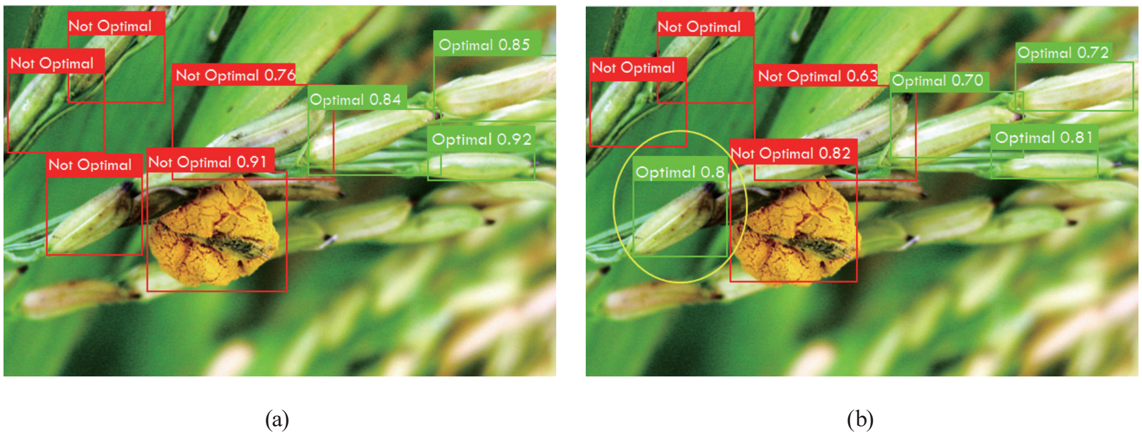

Fig. 15. (Color online) Practical application: (a) proposed model versus (b) baseline (YOLOv8).

conducted to determine the target object referring to the multiclassification image. The model performs prediction tasks through two aspects, classification and regression. The classification task is performed to determine the target object based on its context class, while the regression task is performed to generate and localize the bounding box onto the target object. To facilitate communication between the user and the system, mobile and web-based user interfaces are developed.

The results of practical application are depicted in Fig. 15, where two models were incorporated in practical application experiments: the baseline model (YOLOv8) and the proposed model (YOLO-Rice). The experiment was conducted using images of diseased rice plants, where rice grains have class labels "optimal" and "not optimal." Both models can detect optimal and not optimal grain conditions. However, YOLOv8 has misclassified data indicated by a yellow circle; the label should be the not-optimal condition. Meanwhile, the proposed YOLO-Rice accurately indicates the grain conditions. The inference times of YOLOv8 and YOLO-Rice are 84 and 45 ms, respectively, showing that YOLO-Rice requires less inference time than YOLOv8.

4. Conclusions

In this study, the multiclass intelligent object detection model called YOLO-Rice, aimed at detecting the current condition of rice plants, was developed. Experiments were carried out to validate the model performance, including the model performance on a dataset, ablation study, and comparative experiments. The proposed model exhibited the best performance in terms of model detection accuracy and model efficiency. The number of parameters obtained by YOLO-Rice indicated an efficiency 9% higher than that of the original model, YOLOv8. The $mAP_{0.5}$ values of YOLOv8 and YOLO-Rice were 80.11 and 91.95%, respectively, which indicates that the proposed model has outstanding detection performance. In this study, we implemented multiple classes consisting of the conditions of the rice leaves, grains, and the existence of insect pests, giving rise to the possibility of causing an imbalanced dataset. To address this issue, an image augmentation technique based on varying conditions such as light condition, object size, or angle of position is needed to produce a dataset that supports the model's robustness and increases the model detection accuracy. In future studies, datasets can be developed by enhancing classes such as the crop type class. In addition, other attention modules can be integrated to provide a powerful lightweight intelligent model for PA purposes.

Acknowledgments

This research was supported by the Department of Electronic Engineering of National Chin-Yi University of Technology. We would like to thank the National Chin-Yi University of Technology for supporting this research.

References

1 US Department of Agriculture, Foreign Agricultural Service: https://apps.fas.usda.gov (accessed March 2025).
2 Directorate-General of Budget Accounting and Statistics (R.O.C): Statistical Yearbook of The Republic of China 2022 (Directorate-General of Budget, Accounting and Statistic Executive Yuan - ROC (Taiwan), Taipei, 2023).
3 J. Syauqi, R.-K. Chen, A.-H. Cheng, Y.-F. Wu, C.-L. Chung, C.-C. Lin. H.-P. Chou, H.-Y. Wu, J.-Y. Jian, C.-T. Liao, C.-C. Kuo, S.-C. Chu, Y.-C. Tsai, D.-J. Liao, Y.-P. Wu, A. L. Abadi, L. Sulistyowati, and W.-C. Shen: Plant Dis. **106** (2022) 3187. https://doi.org/10.1094/PDIS-12-21-2806-RE
4 E. M. B. M. Karunathilake, A. T. Le, S. Heo, Y. S. Chung, and S. Mansoor: Agriculture **13** (2023) 1. https://doi.org/10.3390/agriculture13081593
5 L. Du, J. Zhu, M. Liu, and L. Wang: IET Image Proc. **19** (2024) 1. https://doi.org/10.1049/ipr2.13304
6 S. Jain, R. Sahni, T. Khargonkar, H. Gupta, O. M. Verma, T. K. Sharma, T. Bhardwaj, S. Agarwal, and H. Kim: Elecronics **11** (2022) 1. https://doi.org/10.3390/electronics11142110
7 X. Deng, L. Qi, Z. Liu, S. Liang, K. Gong, and G. Qiu: PLOS One **18** (2023) 1. https://doi.org/10.1371/journal.pone.0294709
8 A. K. Sangaiah, F.-N. Yu, Y.-B. Ling, W.-C. Shen, and A. Sharma: IEEE Trans. Network Sci. Eng. **11** (2024) 5201. https://doi.org/10.1109/TNSE.2024.3350640
9 Z. Li, W. Wu, B. Wei, H. Li, J. Zhan, S. Deng, and J. Wang: Sensors **25** (2025) 1. https://doi.org/10.3390/s25082494
10 R. K. Kasera, S. Gour, and T. Acharjee: Comput. Electron. Agric. **216** (2023) 108522. https://doi.org/10.1016/j.compag.2023.108522
11 C. G. V. N. Prasad, A. Mallareddy, M. Pounambal, and V. Velayutham: Int. J. Recent and Innovation Trends Comput. Commun. **10** (2022) 265. https://doi.org/10.17762/ijritcc.v10i1s.5848
12 A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao: Arxiv (2020) 1. https://arxiv.org/abs/2004.10934
13 M. Sohan, T. S. Ram, and C. V. R. Reddy: 2nd Int. Conf. Data Intelligence and Cognitive Informatics (ICDICI 2021) 529–545.
14 Ultralytics: https://docs.ultralytics.com/models/yolov8/ (accessed January 2025).
15 G. Chen, Y. Hou, T. Cui, H. Li, F. Shangguan, and L. Cao: Sci. Rep. **14** (2024) 1. https://doi.org/10.1038/s41598-024-65293-w
16 W.-T. Sung, I. G. T. Isa, C.-H. Tseng, and S.-J. Hsiao: IEEE Access **13** (2025) 46794. https://doi.org/10.1109/ACCESS.2025.3550539
17 K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu: 2020 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR, 2020) 1577–1586.
18 S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon: 2018 Eur Conf. Computer Vision (ECCV, 2018) 1–17.
19 D. W. Yang, M. K. Suh, and S. W. Seo: 2022 Asian Conf. Computer Vision (ACCV, 2022) 373–388.
20 J. Kang, W. Zhang, Y. Xia, and W. Liu: Appl. Sci. **13** (2023) 1. https://doi.org/10.3390/app131810441
21 M. Xu, S. Yoon, A. Fuentes, and D. S. Park: Pattern Recognit. **137** (2023) 109347. https://doi.org/10.1016/j.patcog.2023.109347
22 M. Hossin: Int. J. Data Mining Knowl. Manage. Process **5** (2015) 1. https://doi.org/10.5121/ijdkp.2015.5201

About the Authors

Wen-Tsai Sung (Member, IEEE) received his M.S. and Ph.D. degrees from the Department of Electrical Engineering, National Central University, Taoyuan City, Taiwan, in 2000 and 2007, respectively. He is a distinguished professor in the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung, Taiwan. He won the 2009 Journal of Medical and Biological Engineering (SCIE index) Best Annual Excellent Paper Award (Ph.D. Dissertation) and the Dragon Thesis Award (Master Thesis) sponsored by Acer Foundation. In October 2023, he was recognized as one of the world's top 2% scientists (Career Impact) (2001–2023) by Mendeley Data, a company owned by Elsevier. He has authored over 100 peer-reviewed journal papers and articles indexed by SCI/SCIE/EI. His research interests include artificial intelligence Internet of Things, intelligent computing and systems, and wireless sensor networks. (songchen@ncut.edu.tw)

Indra Griha Tofik Isa received his B.S. degree in informatics engineering from Pelita Bangsa University in 2011 and his M.S. degree in computer science from STMIK LIKMI Indonesia in 2014. He is currently a lecturer at Sriwijaya State Polytechnic and is continuing his Ph.D. studies at National Chin-Yi University of Technology (NCUT) in the fields of electrical engineering and computer science. His research interests are in deep learning, neural network applications, data science, image processing, and object detection (indraisa89@gm.student.ncut.edu.tw).

Sung-Jung Hsiao is an associate professor in the Department of Information Technology, Takming University of Science and Technology. He received his B.S. degree in electrical engineering from National Taipei University of Technology, Taiwan, in 1996, his M.S. degree in computer science and information engineering from National Central University, Taiwan in 2003, and his Ph.D. degree in electrical engineering from National Taipei University of Technology, Taiwan in 2014. He has work experience in research and design at Acer Universal Computer Co., Mitsubishi, and First International Computer. (sungjung@gs.takming.edu.tw)