Engenharia Agrícola

ISSN: 1809-4430 (on-line)

www.engenhariaagricola.org.br

Scientific Paper
Doi: http://dx.doi.org/10.1590/1809-4430-Eng.Agric.v43n4e20230065/2023

MDP-YOLO: A LIGHTWEIGHT YOLOV5S ALGORITHM FOR MULTI-SCALE PEST DETECTION

Jianghua Yu[1], Bing Zhang[2*]

[2*]Corresponding author. School of Intelligent Manufacturing Nanyang Institute of Technology, Nanyang, Henan 473306, China. E-mail: paulyujhp@163.com | ORCID ID: https://orcid.org/0000-0002-0698-8473

KEYWORDS

deep learning, YOLOv5s, object detection, pest detection, lightweight, attention mechanism.

ABSTRACT

Rice pest detection technology plays a crucial role in enabling food production, ensuring ecological balance, supporting sustainable agriculture, and promoting the health of farmers. However, existing technology is faced with challenges such as low detection accuracy, high computational complexity, and large model sizes, making it unsuitable for mobile deployment. This paper presents MDP-YOLO, a lightweight rice pest detection model based on the YOLOv5s model. It includes improvements such as the use of ShuffleNetV2 as the backbone network to significantly reduce the number of parameters and complexity of the model; the introduction of GhostConv to replace redundant convolutional layers, in order to further reduce the computational complexity and size; the integration of a large-scale feature extraction layer to enhance the ability of the algorithm to detect small rice pest objects; and the use of CBAM to increase the focus on the regions of interest. Tests on collected datasets show that the modified MDP-YOLO model increases the mean average precision by 5.5% and reduces the FLOPs, parameters, and model size by 72.4%, 89.2%, and 70.1%, respectively, compared to the original YOLOv5s model. Ablation experiments indicate that MDP-YOLO gives superior detection rice pest detection performance compared to other algorithms. MDP-YOLO can effectively detect rice pests, and provides theoretical and technical support for the deployment of lightweight rice pest detection models in practical scenarios.

INTRODUCTION

Rice is one of the most important food crops globally, with 518 million tons produced in 2019, according to the Food and Agriculture Organization of the United Nations. This plant is crucial for ensuring global food security and agricultural development. Rice pests are one of the main causes of reduced yield and death of rice plants, and the timely and efficient detection of these pests can reduce farmers' production costs and environmental pollution, improve the quality and safety of rice, maintain ecological balance, and promote sustainable agricultural development. However, the detection of rice pests is affected by various factors, including climate, light, humidity, nutrition, fertilizers, water management, and tillage conditions. Traditional manual identification methods are time-consuming, labor-intensive, and prone to misdiagnosis; in contrast, deep learning-based methods have higher accuracy, and are more suitable for crop pest detection than traditional methods. Research on the development of lightweight object detection algorithms based on deep learning therefore has practical value for the detection of crop pests, and is urgently needed to achieve agricultural automation and intelligence.

Following recent developments in information technology, deep learning has been widely applied in various fields such as food safety testing (Deléglise et al., 2022), crop yield estimation (Martínez-Ferrer et al., 2021), autonomous driving (Uhlemann, 2019), and precision agriculture (Alirezazadeh et al., 2021). The automatic detection of plant pests has become a topic of intense research. However, in complex natural environments, the robustness and generalization ability of these algorithms are poor, and the detection efficiency

[1] Jiujiang Precision Measuring Technology Research Institute, Jiujiang, Jiangxi, 332000, China.

Area Editor: Lucas Rios do Amaral
Received in: 4-17-2023
Accepted in: 9-29-2023

is too low to meet the need for rapid and accurate detection of agricultural rice pests with lightweight deployment. For example, Deng et al. (2022) proposed a detection technology for multiple pests based on federated learning (FL) and an improved faster region convolutional neural network (R-CNN). Their experimental results showed that the model had an average accuracy of 90.27% for the detection of multiple pests, and an speed of 20 frames per second (FPS). However, the model had high data storage and communication costs, and was difficult to adapt to complex detection environments. Wang et al. (2021) proposed an improved real-time object detection algorithm for early-stage tomato pests based on YOLOv3. The F1 score for this method was 94.77%, the AP value was 91.81%, the false positive rate was 2.1%, and the FPS was 18. However, this approach was found to be unreliable under adverse weather conditions.

Deep learning-based object detection algorithms use convolutional neural networks to extract features, as they have high speed, accuracy, and strong generalization capabilities. These algorithms are generally divided into two categories: the first are two-stage detection algorithms, such as Mask R-CNN (Xiao et al., 2022) and Faster R-CNN (Sun et al., 2018), while the second are one-stage detection algorithms, such as YOLO (Hu et al., 2021) and SSD (Liu et al., 2022a). Although the detection accuracy of one-stage algorithms is slightly lower than that of two-stage object detection algorithms, they have higher detection speeds and real-time detection capabilities. In this study, we therefore choose a one-stage object detection algorithm for the real-time detection of pests.

In recent years, with improvements in computer performance, a large number of deep learning-based algorithms for pest detection have been proposed, with significantly improved detection performance and speed (Liu & Wang, 2021). However, the higher computational requirements of complex networks limit the application of these approaches. To address this issue, some scholars have made efforts to create lightweight models that can achieve a balance between detection efficiency and accuracy. Li et al.(2021a) proposed a lightweight EfficientNet model for pest detection that achieved an accuracy of 93.73% in identifying and locating agricultural pests, with reduced equipment costs and accurate predictions of crop pest conditions. Li et al. (2023) proposed a fast and lightweight algorithm for detecting passion fruit pests, with an mAP of 96.51% and an average detection time of 7.7 ms, thus meeting the requirements for accuracy and real-time performance. Zha et al. (2021) proposed a lightweight YOLOv4 object detection algorithm for detecting forestry pests, using MobileNetv2 as the feature extraction network. The precision and recall of the proposed model were improved by 4.37% and 6.68%, respectively, compared to YOLOv4, and the size of the model was reduced to one sixth of that of YOLOv4. Cheng et al. (2022) proposed a lightweight crop pest detection method based on convolutional neural networks, which achieved a detection accuracy of 82.9% with only 9.8G FLOPS, corresponding to only 15% of the value for YOLOv3. The studies described above mainly focused on the detection of large individual pests or obvious experimental objects. Deep learning-based object detection algorithms can effectively extract high-dimensional image feature information in environments with less interference; however, individual rice pests have small sizes, some have inconspicuous color features, and rice leaves are dense, which poses certain challenges in terms of object detection in complex natural environments.

In this study, we investigate the growth characteristics of pests in complex natural environments and explore the object detection of rice pests using the YOLOv5s algorithm. The contributions of our work are as follows. Firstly, we select the improved ShuffleNetV2 lightweight network as the primary feature extraction network for YOLOv5, which reduces the complexity and computational cost of the network. Secondly, we incorporate large-scale feature extraction layers into the network structure to achieve fine-grained object detection, thereby improving the algorithm's ability to detect small objects. Thirdly, we use a lightweight GhostConv module to replace the standard convolutional layers in the neck layer. This reduces the number of parameters and computational complexity of the model while maintaining its accuracy, thus making the model more practical and lightweight. Finally, we incorporate a convolutional block attention module (CBAM) into the C3 module of the detection head. This helps the model to understand the features of different regions in the image, and enables it to predict the positions of objects more accurately.

The rest of this study is organized as follows. In Section 2, we introduce the materials and methods used in the experiments. Section 3 describes the experimental platform and presents a detailed analysis of the experimental results. In Section 4, we discuss some of the problems encountered during the research process and outline further research work. Finally, Section 5 provides a summary of this study.

MATERIAL AND METHODS

Acquisition and annotation of image data

The rice pest dataset used in this study includes nine key rice pests, and was finalized based on a comprehensive analysis of the pests by agronomists from the Agricultural Economy and Information Research Institute of Anhui Academy of Agricultural Sciences, China. It includes data on morphology, genetic characteristics, and ecological behaviors to determine the specific names of these nine pests. The nine types of pest are the egg, larval and adult stages of *Naranga aenescens Moore*, the larval, pupal and adult stages of the *Sesamia inferens*, and the three developmental stages of the *Nezara viridula*. The eggs of each species of *Naranga aenescens Moore* mark the beginning of its reproduction, with the larvae dealing with the pests in a parasitic manner and the adults carrying out reproduction and feeding. *Sesamia inferens* larvae are significant crop pests, while the pupae represent the transformation stage and adults are involved in reproduction and transmission. The *Nezara viridula* stink bug feeds on plant sap and may cause damage to crops, and its different developmental stages all have an impact on agricultural production.

The original dataset used in this study consisted of 1720 images (provided by the Agricultural Economy and Information Research Institute of Anhui Academy of Agricultural Sciences, China) with a resolution of 2000 × 1325 pixels. To simulate a more complex and realistic environment, we applied image augmentation techniques such as adding noise, enhancing contrast, applying random colors, and rotating the images through angles to expand the original dataset to 2120 images, thus increasing the diversity of the data. The processed results are shown in Figure 1.

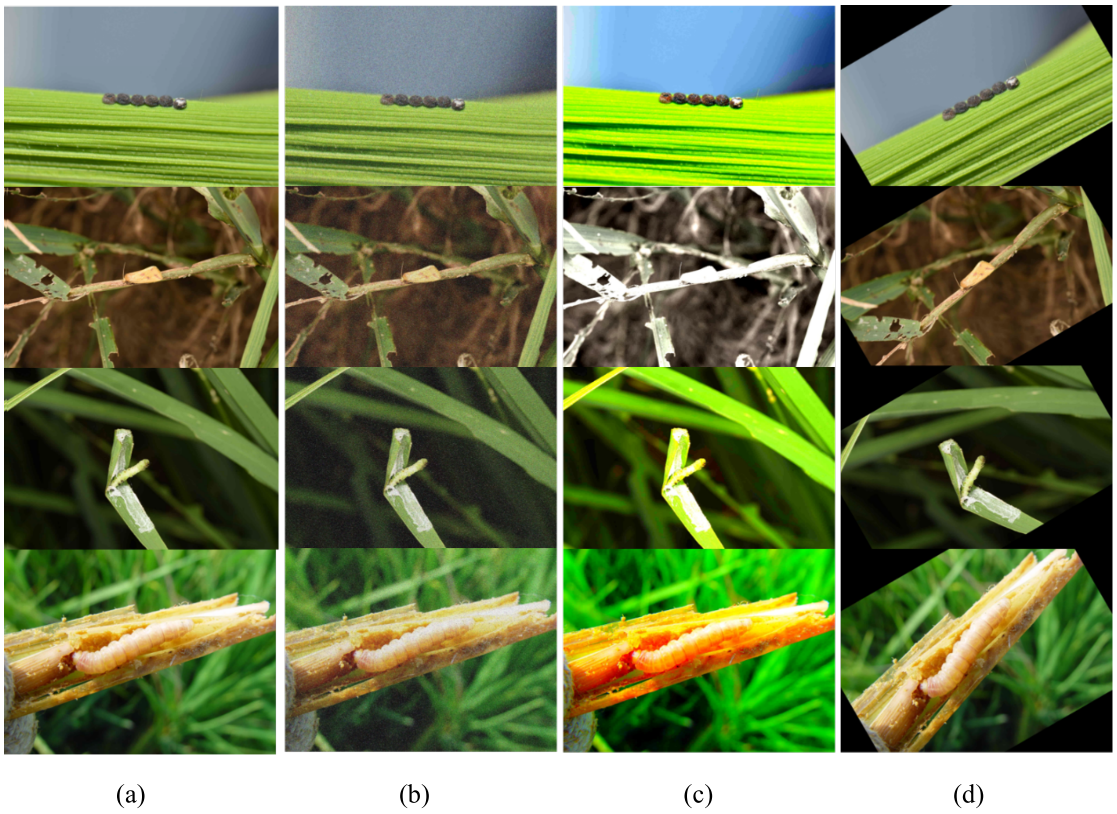


FIGURE 1. Results of data augmentation: (a) original image; (b) addition of noise; (c) application of random colors; (d) rotation through various angles.

In this study, the LabelImg annotation tool was used to label the images. The closest rectangle to the pest was considered to be the real box, and the center point coordinates were obtained as the ground truth. The labels were DMP1 (*Naranga aenescens Moore* eggs), DMP2 (*Naranga aenescens Moore* larvae), DMP3 (*Naranga aenescens Moore*), DDMP1 (*Sesamia inferens*), DDMP2 (*Sesamia inferens* pupa), DDMP3 (*Sesamia inferens* moth), DLP1 (*Nezara viridula* Linnaeus 1), DLP2 (*Nezara viridula* Linnaeus 2), and DLP3 (*Nezara viridula* Linnaeus 3). The numbers of each type of pest are shown in Figure 2.

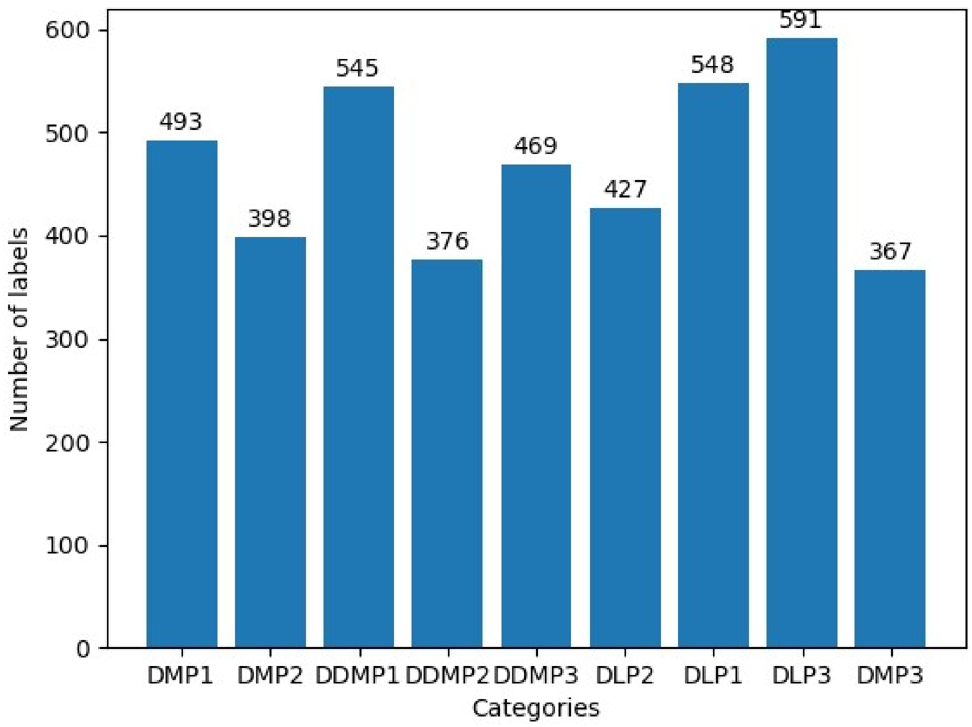
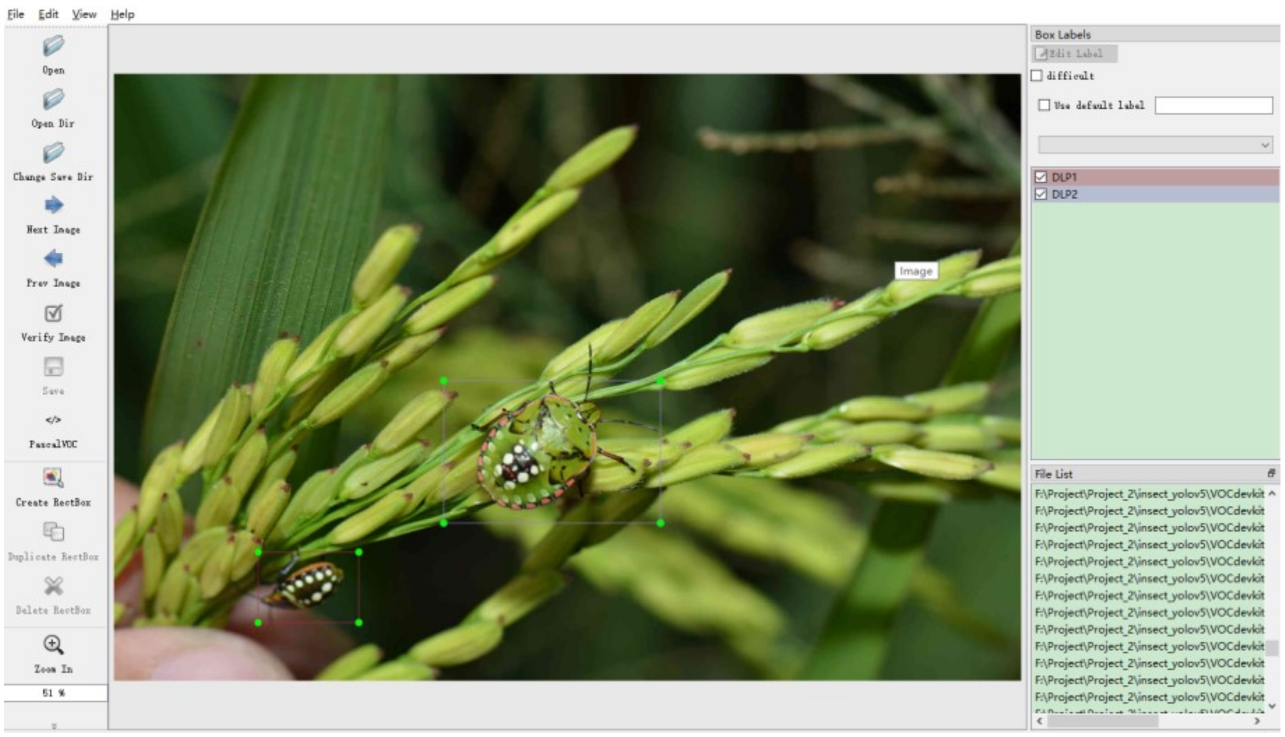


FIGURE 2. Number of pests by category.

The annotation results are shown in Figure 3. The size of the input to the model was set to 640 × 640, giving a multiple of 32 pixels. The labeled images were used to create a dataset in the VOC2007 format, which was then divided into training, validation, and test sets in the ratio 8:2:1. To meet the data format requirements of the YOLO network, the dataset in XML format was converted to TXT format and imported into the network for training.



F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit
F:\Project\Project_2\insect_yolov5\VOCdevkit

FIGURE 3. Label fabrication.

Principle of operation of the detection algorithm

YOLOv5 version 6.0 includes four network structures: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, in order of increasing depth and width (Li et al., 2021b). Although their recognition accuracy is continuously improving, their recognition speed is decreasing, and their hardware requirements are becoming ever higher. The parameters of the four YOLOv5 network structures are shown in Table 1. Of these, YOLOv5s is the most lightweight model, and is easy to deploy in practical application scenarios. To achieve better real-time recognition and reduce the training and deployment costs, we improved the network structure of YOLOv5s and conducted further experiments.

TABLE 1. Parameter settings for the YOLOv5s models.

Model	Depth multiple	Width multiple	Params (M)	Size of model (MB)	CPU time (ms)	GPU time (ms)
YOLOv5n	0.33	0.25	1.9	3.9	45	6.3
YOLOv5s	0.33	0.50	56.8	13.7	98	6.4
YOLOv5m	0.67	0.75	64.1	40.8	224	8.2
YOLOv5l	1.00	1.00	67.3	89.3	430	10.1
YOLOv5x	1.33	1.25	68.9	166.0	766	12.1

The YOLOv5s network architecture consists of three main parts: the backbone, neck, and head. The backbone is the main part of the YOLOv5 network, and is responsible for extracting high-level features from input images. YOLOv5s uses CSPDarknet53 as the backbone, which is a lightweight deep convolutional neural network containing multiple convolutional and pooling layers that can effectively extract features from images. The neck extracts additional useful information from the features extracted by the backbone. YOLOv5s uses PANet (Path Aggregation Network) as the neck, which can fuse feature maps with different resolutions to improve the accuracy and robustness of object detection. The head is the output stage of the YOLOv5s network, and contains multiple convolutional and fully connected layers to map the features to the output space for object detection, outputting information such as object categories, positions, and confidence scores. The head is mainly responsible for converting the feature map created by the neck to the object detection results. The specific workflow of the YOLOv5s network architecture is shown in Algorithm 1 below.

Algorithm 1 YOLOv5s 6.0 Network

```
1:  Input: Batch of images $X$
2:  Output: Predictions for object detection
3:  for each image in batch $X$ do
4:      $x \leftarrow$ Normalize image
5:      $x \leftarrow$ Convolutional layer with 32 filters, kernel size=3, stride=1
6:      $x \leftarrow$ Residual block with 64 filters
7:      $x \leftarrow$ Convolutional layer with 128 filters, kernel size=3, stride=2
8:      $x \leftarrow$ Residual block with 128 filters $\times$ 2
9:      $x \leftarrow$ Convolutional layer with 256 filters, kernel size=3, stride=2
10:     $x \leftarrow$ Residual block with 256 filters $\times$ 8
11:     $x \leftarrow$ Convolutional layer with 512 filters, kernel size=3, stride=2
12:     $x \leftarrow$ Residual block with 512 filters $\times$ 8
13:     $x \leftarrow$ Convolutional layer with 1024 filters, kernel size=3, stride=2
14:     $x \leftarrow$ SPPF block
15:     $x \leftarrow$ Convolutional layer with 512 filters, kernel size=1, stride=1
16:     $x \leftarrow$ PANet block
17:     $x \leftarrow$ Convolutional layer with 256 filters, kernel size=3, stride=1
18:     $x \leftarrow$ YOLOv5 head with anchor boxes and prediction outputs
19:     $y \leftarrow$ Post-processing of prediction outputs
20: end for
21: return $y$
```

Experimental platform

The experiments in this study were conducted on a test platform equipped with 8 x Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz (60G RAM) and NVIDIA GeForce RTX 3080Ti 12G. The software configuration environment consisted of CUDA 11.6.0, CUDNN 8.3.3, and Python 3.8.8. The parameter settings for training are shown in Table 2.

TABLE 2. Parameter settings used for training.

Parameter	Value
Momentum	0.95
Weight decay	0.0005
Batch size	16
Initial learning rate	0.01
Epochs	300
Threshold	0.5
Image size	640×640
Optimizer	SGD

The loading of pre-trained weights is an effective method of transfer learning. In this study, yolov5s.pt were used as pretrained weights for model training. This approach can accelerate the training speed, improve the performance of the model, and avoid the risk of overfitting during training.

Evaluation metrics for model performance

The YOLOv5s loss was used to evaluate the degree of inconsistency between the prediction results from the model and the ground truth. This loss consists of three parts: the bounding box loss (l_{bbox}), the object loss (l_{object}), and the classification loss ($l_{classification}$). To prevent underfitting or overfitting of the model, the training loss and validation loss of the training set were observed during the training process to obtain the best detection model.

$$loss = l_{bbox} + l_{object} + l_{classification} \tag{1}$$

In addition, we used objective evaluation metrics such as precision, recall, F1-score, and mean average precision (mAP) to evaluate the performance of the detection model. The mAP was measured at two thresholds, mAP@.5 and mAP@.5:.95, corresponding to intersection over union (IoU) thresholds of 0.5 and between 0.5 and 0.95 with a step size of 0.05, respectively. The larger the values of these metrics, the better the performance of the object detection model. The IoU between the predicted frame and the actual labeled frame was used to determine whether the object had been successfully predicted. The formulae used to calculate these metrics are as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall} \times 100\% \tag{4}$$

$$AP = \int_0^1 Precision \times Recall = \int_0^1 p(r)\mathrm{d}r \tag{5}$$

$$mAP = \frac{1}{m}\sum_{i=1}^{m} AP(i) \tag{6}$$

$$mAP@.5{:}.95 = \frac{(AP@.5 + AP@.55 + AP@.6 + \cdots AP@.95)}{N} \tag{7}$$

Where:

TP is true positive;

FP is false positive;

FN is false negative, and

m is the number of classes.

The precision, recall, and F1-score were obtained based on the above indicators. The mAP is the average value of the average precision (AP); the higher its value, the better the detection performance of the algorithm. The average precision at threshold *x* (*AP@x*) represents the average accuracy when the confidence threshold is set to *x*. *N* represents the number of cases within the confidence threshold range. In addition, the values for the floating point operations (FLOPs), parameters, and FPS represent the computational complexity, parameter quantity, and real-time detection performance of the model, respectively.

Proposed MDP-YOLO

To more effectively detect multi-scale rice pests in paddy fields, each 2000 × 1325 pixel image was normalized and resized to 640 × 640 pixels before training was carried out, to match the input size of the model. Although it was based on the lightweight YOLOv5s, the proposed MDP-YOLO model included four improvements, as shown in Figure 4: (i) the original CSPDarknet53 backbone network model was replaced with the ShuffleNetV2 network, which effectively reduced the number of parameters in the model's backbone network; (ii) new feature fusion layers were added to capture richer shallow information, thus enabling fine-grained detection of small objects; (iii) GhostConv modules were used to replace ordinary conv modules, thereby reducing the number of parameters and computational complexity of the model, and further improving the training and inference speed; and (4) a CBAM was incorporated in the C3 module of the detection head to enhance the model's understanding and expression ability of images.

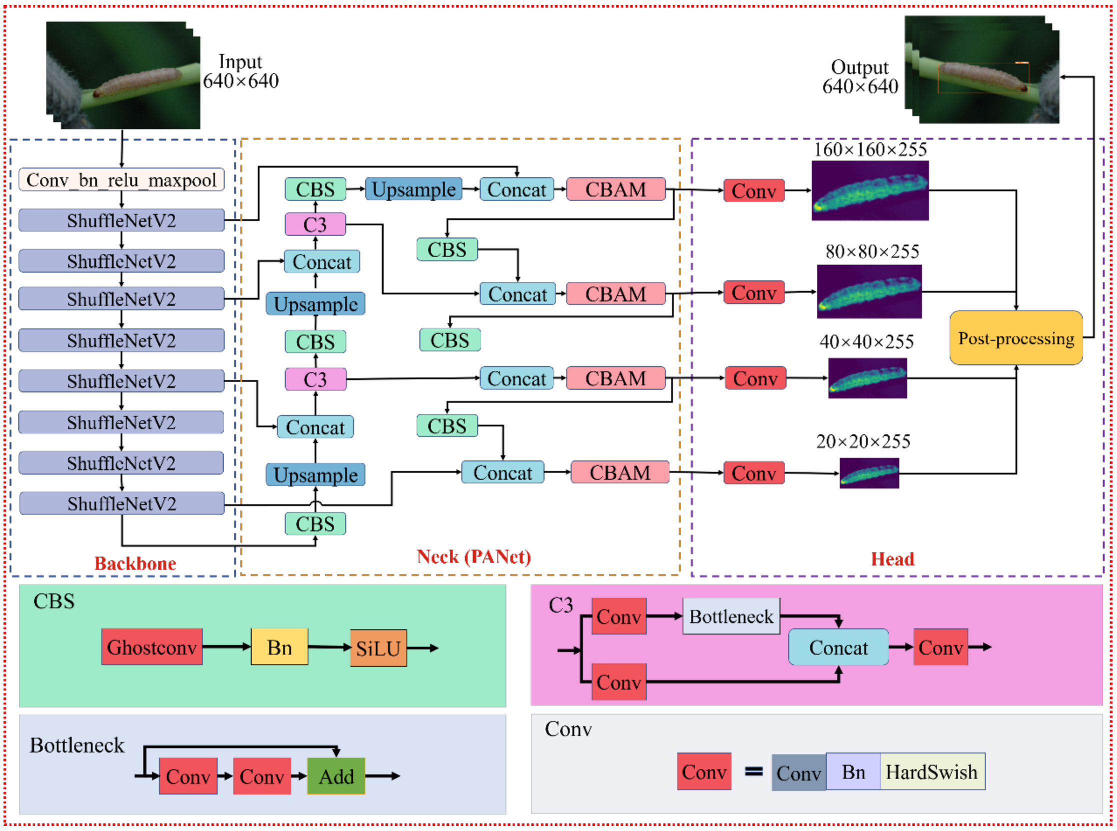


FIGURE 4. Structure of the MDP-YOLO network.

ShuffleNetV2 Block

ShuffleNet is a lightweight feature extraction network that was designed for use on mobile devices. It was developed from residual networks, and its lightweight design is based on a channel shuffling operation proposed by Zhang et al. (2018). The aim of the shuffle operation is to divide the input images into several groups according to channels, where each group is then convolved with a convolutional kernel. Compared with an algorithm where each convolutional kernel is convolved with each channel, this method greatly reduces the computational complexity of the neural network. At the same time, to solve the problem of the lack of communication between different channels, which leads to a reduction in the amount of learned content, this operation shuffles the channels within each group, thereby exchanging the content of the channels. This allows ShuffleNet to achieve a balance between lightweight design and accuracy. ShuffleNetV2 is mainly composed of two types of basic units stacked together (Ma et al., 2018), with a structure as shown in Figure 5.

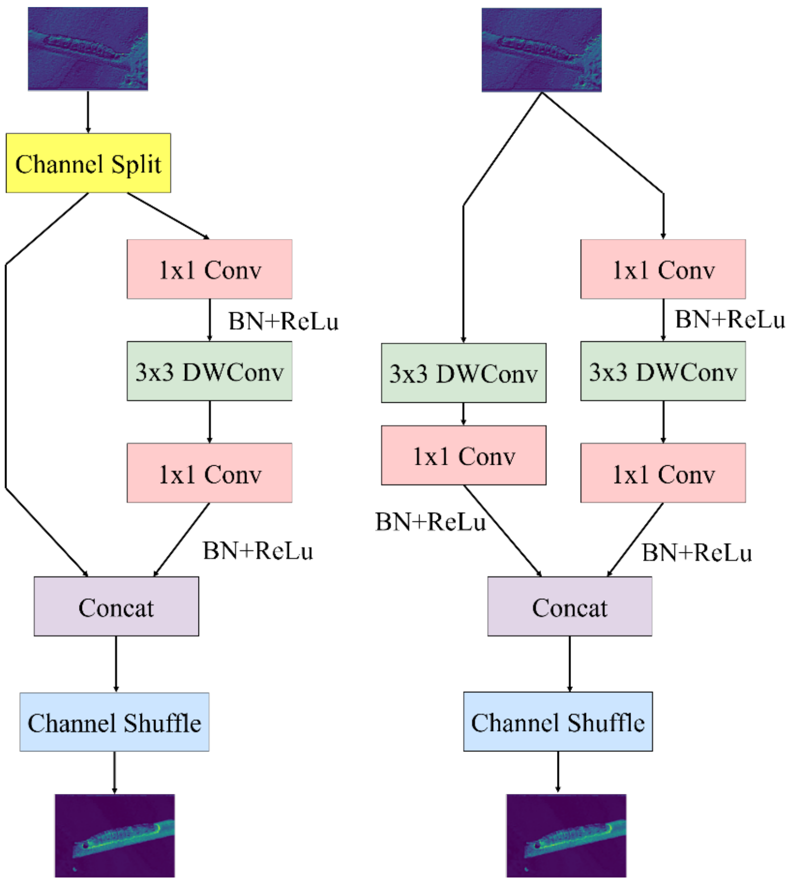


FIGURE 5. Structure of the basic unit of ShuffleNetV2.

In the basic unit with a stride of one, the input data are split into two parts according to the number of channels and are passed to two branches. The right branch performs feature extraction through two 1×1 convolution layers and a 3×3 depth convolution. The data are then merged with the input of the left branch through channel concatenation, and the order of the information is uniformly shuffled through channel shuffling. The basic unit with a stride of two mainly performs downsampling operations. Unlike the basic unit with stride one, it does not apply the step of separating channels, and a 1×1 convolution and 3×3 depth convolution are added to the left branch, thus increasing the number of output channels by a factor of two and reducing the size of the feature map by half. In the basic unit, batch normalization (BN) layers are used after the convolution operation to speed up the convergence of the network, and the ReLu function is applied after the 1×1 convolution to increase the nonlinear features of the model.

Small object detection layer

The low number of pixels occupied by each pest in the image poses a certain challenge for object detection models in terms of accurately detecting these small objects. The dimensions of the feature maps obtained by the original YOLOv5s detection layer are 20×20, 40×40, and 80×80, which cannot meet the requirements for subsequent detection and regression, meaning that the model has difficulty in achieving ideal results for the detection of rice pests in complex environments. To solve this problem, MDP-YOLO introduces an additional upsampling and downsampling process by adding a feature fusion layer, convolutional layer, and CSP module; this not only preserves the original detection scale of the model but also avoids reducing the learning ability of the original scale features. In this method, feature layers at the same scale, which retain more small object feature information in the backbone network, are input into the feature fusion layer for feature fusion. The final detection layer creates feature maps with scales of 20×20, 40×40, 80×80, and 160×160. The new detection layer is more sensitive to the features of small objects, and is more conducive to the detection of small rice pests in complex environments. The main purpose of upsampling is to enlarge the scale of the extracted high-level feature map and to fuse it with the low-level feature map to give a more informative feature map. The main purpose of downsampling is to reduce the size of the image in order to filter out redundant feature information while retaining key information. However, these image scaling operations not only fail to provide more information but also affect the quality of the image. As a first-stage algorithm, MDP-YOLO requires higher-resolution images to detect small objects. We therefore apply upsampling, which uses interpolation to improve and fill in the details of the image, and maps the image from low resolution to high resolution. The upsampling and downsampling process is shown in Figure 6

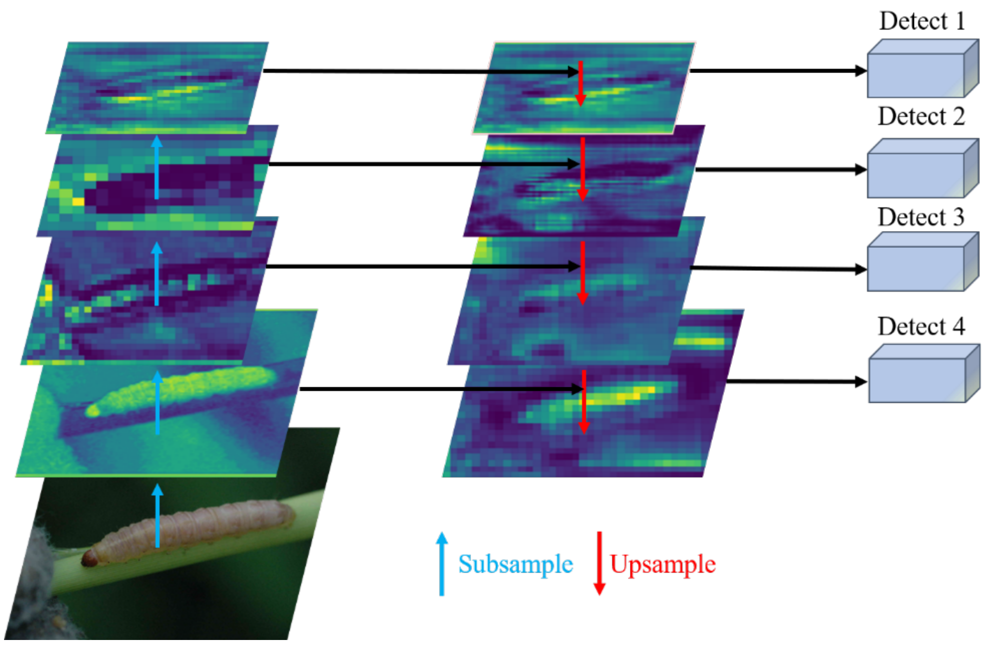


FIGURE 6. Schematic diagram of the upsampling and subsampling process.

GhostConv module

In the task of rice pest detection, some feature maps contain a lot of background information, and conventional convolution operations may produce a large number of redundant feature maps. These redundant intermediate feature maps have a limited impact on improving the accuracy of pest detection while increasing the consumption of computing resources, and it is therefore necessary to use efficient modules to improve the computational efficiency. Han et al. (2020) proposed a lightweight module called GhostConv that could extract sufficient features, reduce computational complexity, and even improve the performance of the original model. To solve this problem in YOLOv5s, and taking into account the characteristics of the rice pest dataset, the GhostNet module was used to replace the redundant feature maps generated by the conventional convolution, in order to reduce the number of model parameters and computational complexity, and to increase the detection speed.

GhostConv consists of two main parts, as shown in Figure 7. First, a regular convolution layer is used to extract intrinsic features with a small number of channels, and this is followed by a depthwise convolution layer to obtain redundant features. The depthwise convolution layer has a smaller computation cost and a kernel size of 1×1. It takes intrinsic features as input, and outputs redundant features. The intrinsic and redundant features are then concatenated along the channel dimension to obtain the final output feature. This method can reduce the amount of computation while maintaining (or even improving) the performance of the model. It is worth noting that GhostConv also has a certain regularization effect, which can effectively prevent overfitting.

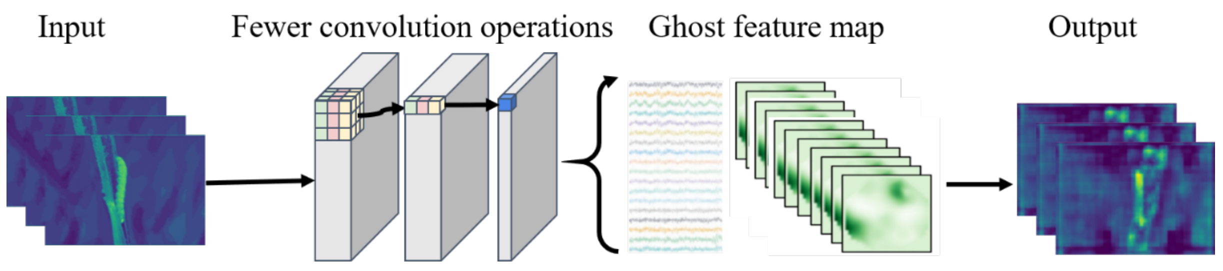


FIGURE 7. Schematic diagram of the GhostConv module.

CBAM

A CBAM is mainly used in convolutional neural networks (Woo et al., 2018). It includes two types of attention mechanisms, namely channel attention and spatial attention (Alirezazadeh et al., 2023). Channel attention is used to learn the importance of different channels, and calculates the importance weight of each channel by a weighted average of the different channels in the feature map. The structure used here is global average pooling and a fully connected network. Global average pooling is used to obtain the global information of each channel in the feature map; the weight of each channel is then calculated through the fully connected network, and the importance weight of each channel is finally obtained. The function of channel attention is to adjust the weight of each channel to ensure that the model focuses on important channels, thereby improving the performance of the model. Spatial attention is used to learn the importance of different positions. The importance weight of each position is calculated by a weighted averaging of the different positions in the feature map. The structure used here consists of a convolutional layer and a sigmoid function: the convolutional layer allows the weight of each position in the feature map to be learned, the value of each weight is then restricted to between zero and one through the sigmoid function, and the importance weight of each position is finally obtained. The function of spatial attention is to adjust the weight of each position to allow the model to focus more on the important positions, thereby improving the performance of the model. The structure of the CBAM attention mechanism is shown in Figure 8.

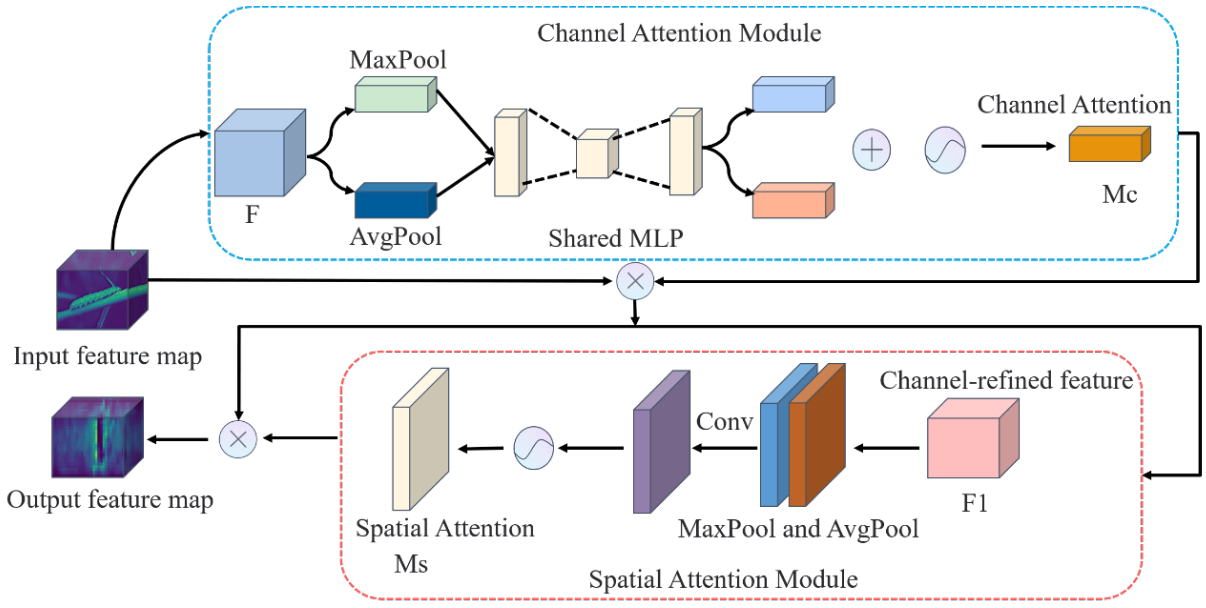


FIGURE 8. Structure of the CBAM network.

In CBAM, the input feature map F has a size of H×W×C. First, the feature map undergoes max-pooling and average-pooling operations to give two 1×1×C feature maps. These feature maps are then passed to a multi-layer perceptron (MLP); the output is a one-dimensional channel attention map Mc(F), which is obtained by applying a sigmoid activation function to the sum of the two feature maps. Mc(F) then undergoes element-wise multiplication with the input feature map F to obtain the channel attention-adjusted feature map F1. Next, F1 is subjected to max-pooling and average-pooling operations to give two H×W×1 feature maps, which are concatenated and passed through a convolutional layer to generate a two-dimensional spatial attention map Ms(F1). This is then multiplied element-wise with the feature map F1 to obtain the final attention-adjusted feature map.

CBAM combines channel attention and spatial attention to adaptively adjust the weights of the feature map, to improve the capture of useful features. This attention mechanism has been validated on numerous tasks and datasets, and its effectiveness in improving the performance of the model has been demonstrated.

Ablation study

An ablation study is an efficient method for studying causal relationships. To better understand the behavior of complex deep neural networks, it is often necessary to study the performance of a network by removing parts of it. To validate the effectiveness of the improvements made in this study to the YOLOv5s detection algorithm, ablation experiments were conducted on the constructed dataset, and the results are reported below.

A total of seven sets of ablation experiments were conducted with the aim of exploring the impact of the different improvements on the performance of the object detection model. First, the effects of different combinations of improvements were evaluated by ablating ShuffleNetV2, the detection layer, GhostConv, and CBAM. These experimental results were then compared with the original experimental results, which showed that the addition of the CBAM to the model and the replacement of the normal convolution with GhostConv had a positive impact on the performance. The MDP-YOLO model showed the best overall performance with a combination of all of the proposed improvements.

RESULTS AND DISCUSSION

Results

The training process of the model is shown in Figure 9. In the process of training, the value of the loss function intuitively reflects whether the model is converging steadily with an increase in the number of iterations. The changes in the values of the loss function during the training process of YOLOv5s and MDP-YOLO showed that both of these converged smoothly without obvious overfitting or underfitting, which verified the suitability of the rice pest dataset and the effectiveness of the experiments. In addition, from the loss curve, it can be seen that after 100 iterations, the loss value is stable and has decreased to around zero. Compared with the original YOLOv5s, MDP-YOLO has a faster regression speed, which further supports the better training process of the model. From the precision and recall curves, it can be seen that the values of P and R for the MDP-YOLO model gradually increase, and exceed those of YOLOv5s as the number of iterations increases, indicating that the model has a stronger ability to predict rice pests. The F1-score is a measure used for classification problems that consists of the harmonic mean of the precision and recall. From the curves below, it can be seen that the F1-score for MDP-YOLO is significantly higher than for YOLOv5s, indicating the superior performance of the MDP-YOLO model.

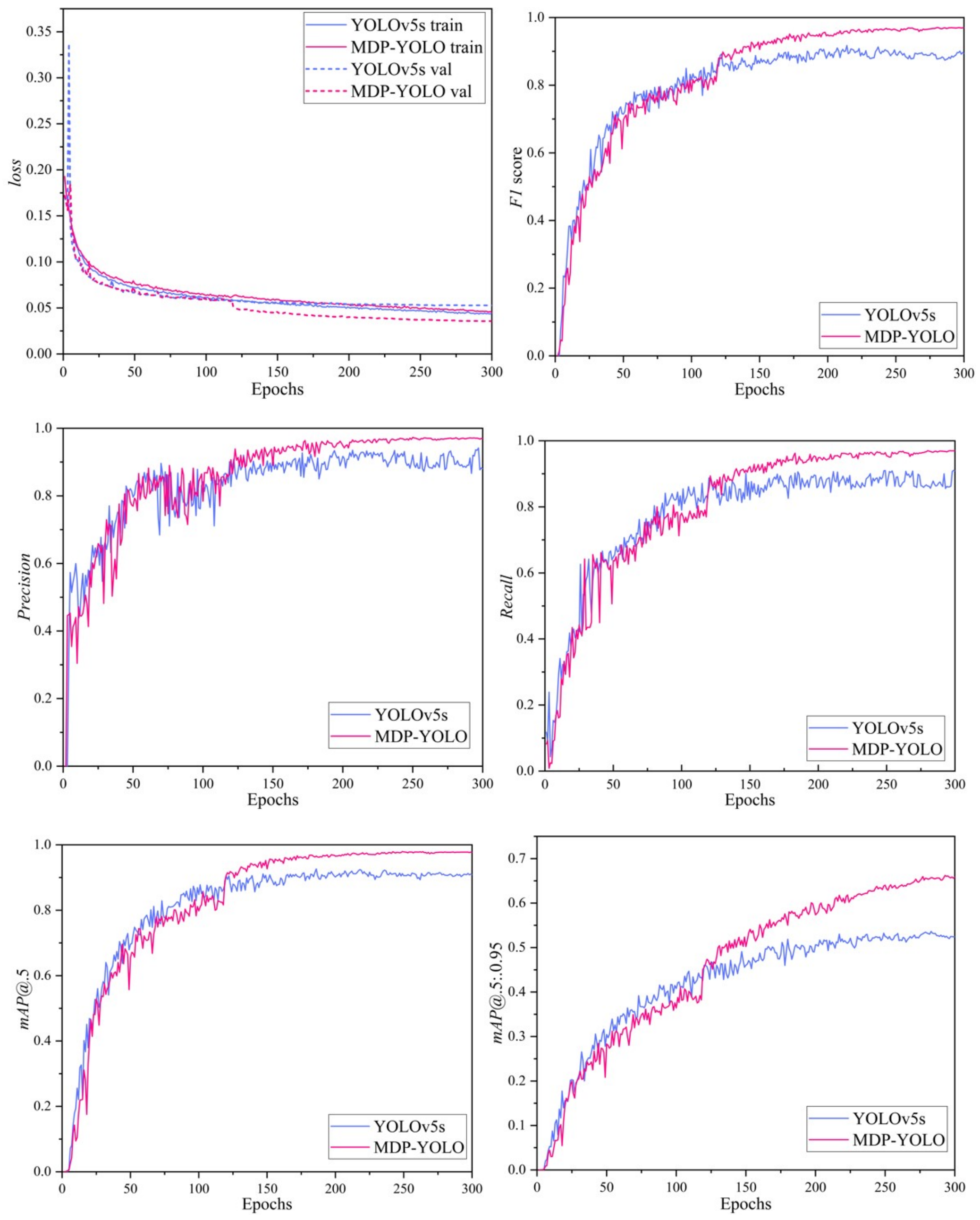
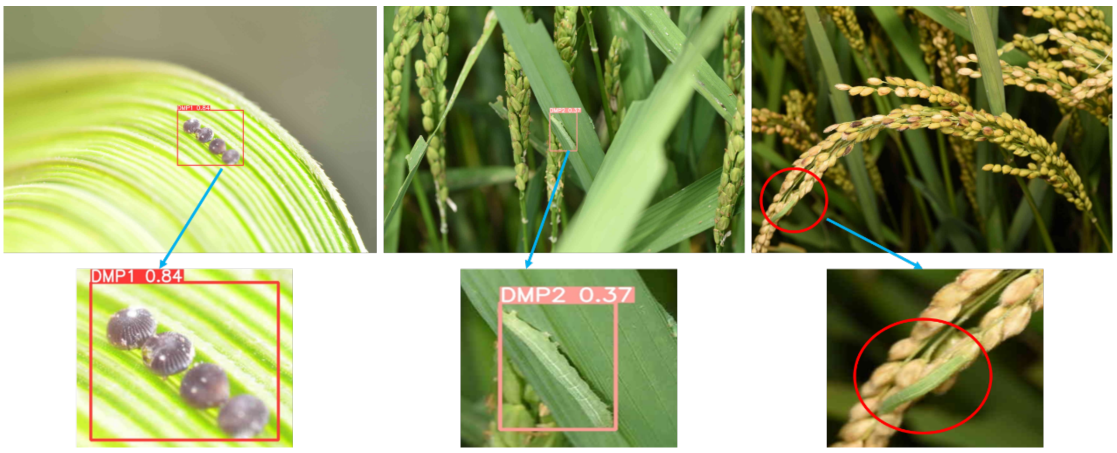
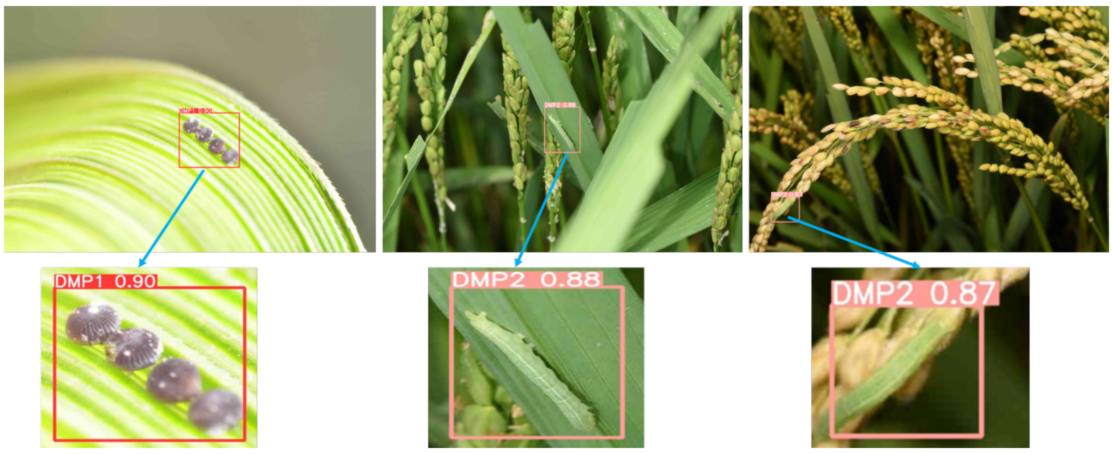


FIGURE 9. Training results for MDP-YOLO and YOLOv5s.

From Figure 9, it can be seen that the MDP-YOLO model achieves values for the mAP@.5 and mAP@.5:.95 of around 97% and 60%, respectively, after about 200 iterations; it then gradually stabilizes, and eventually reaches its maximum values of 97.8% and 66.2%. This indicates that the MDP-YOLO model has a high overall accuracy in terms of detecting rice pests, and that its performance exceeds expectations. In this study, the same rice pest images were input into both the YOLOv5s and MDP-YOLO models for inference, and the detection results are shown in Figure 10.



(a)



(b)

FIGURE 10. Examples of rice pest detection results using YOLOv5s and MDP-YOLO: (a)YOLOv5s; (b) MDP-YOLO.

The red ellipses indicate missed objects. From Figure 10, it can be seen that the accuracy of MDP-YOLO in detecting rice pests is significantly higher than that of YOLOv5s. YOLOv5s also showed missed detections during the detection process, which may have been due to occlusions in the sample data, objects with similar colors to the background, and insufficient learning of object features in small object regions, leading to false positives or false negatives. However, the MDP-YOLO model can adaptively learn feature maps at various scales, which enables it to focus more on the important features of the object and to learn object features more effectively. Hence, MDP-YOLO yields improved overall recognition accuracy, especially for complex environments and small objects.

Object detection heatmaps are usually displayed in the form of color mapping, which can reveal key areas. The different colors represent different scores or probability values; in general, the brighter the color of the heatmap, the higher the confidence of the model in terms of detecting the object. MDP-YOLO can create a 160×160 feature map through upsampling, and can detect small objects from higher granularity feature maps. From Figure 11, it can be seen that MDP-YOLO can accurately locate the regions of interest and extract features while excluding other interfering information, and can then perform post-processing on the extracted object feature information to accurately detect the rice pest objects.

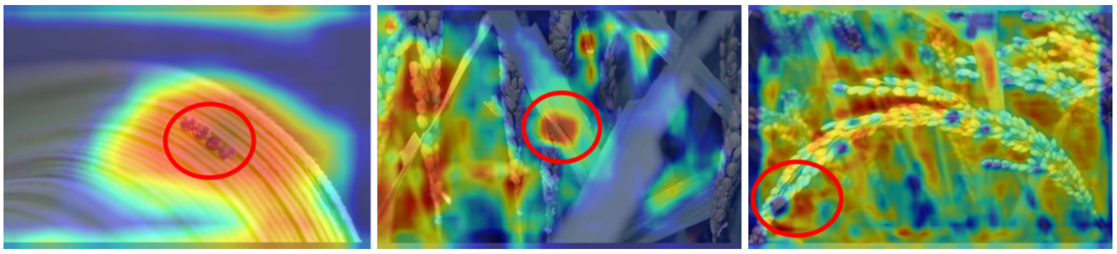


FIGURE 11. Visualization results in the form of heat maps for a large-scale detection head.

To further compare the performance of YOLOv5s and MDP-YOLO, the PR (precision and recall) curves for nine rice pests were obtained as shown in Figure 12. From the figure, it can be seen that MDP-YOLO achieves significantly higher detection accuracy for the nine rice pests compared to YOLOv5s. It can also be seen that the improvements to the lightweight YOLOv5s are effective, and that the introduction of the CBAM and small object detection layer can indeed improve the overall performance of the model.

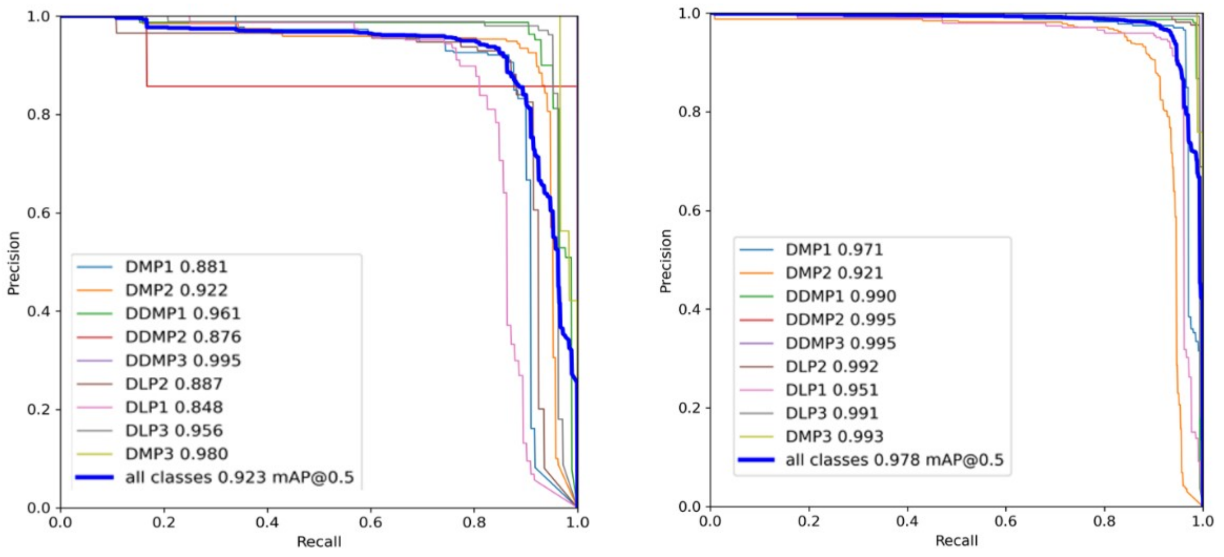


FIGURE 12. PR curves for YOLOv5s and MDP-YOLO.

The results of the ablation experiment are shown in Table 3, and it can be seen that the values of mAP@.5 and mAP@.5:.95 for YOLOv5s are 92.3% and 53.4%, respectively. First, when the lightweight network ShuffleNetV2 was used as the backbone network of YOLOv5s, the values of mAP@.5 and mAP@.5:.95 were decreased by 2.7% and 1.1%, respectively, as an inevitable result of a more lightweight network, and these values are acceptable. Then, new detection layers were added and GhostConv was introduced to replace the ordinary convolution layer, which removed the parameter redundancy and increased the small object detection capability. The values of mAP@.5 and mAP@.5:.95 for Model 4 increased by 3.2% and 1.0%, respectively, compared to Model 1, indicating that these improvements had a positive impact on rice pest detection. Finally, by introducing the CBAM based on Model 4, the attention paid by the model to the object area was improved, resulting in an increase in the values of mAP@.5 and mAP@.5:.95 of 5.5% and 12.2%, respectively, compared to YOLOv5s. This last network was the full MDP-YOLO model.

TABLE 3. Results of the ablation study.

Model	ShufflenetV2	Detection layer	GhostConv	CBAM	mAP@.5	mAP@.5:.95
YOLOv5s	×	×	×	×	92.3%	53.4%
Model 1	√	×	×	×	89.6%	52.3%
Model 2	√	√	×	×	89.3%	50.1%
Model 3	√	×	√	×	91.2%	52.8%
Model 4	√	√	√	×	92.8%	53.3%
Model 5	√	√	×	√	96.9%	65.0%
MDP-YOLO	√	√	√	√	97.8%	66.2%

The improvements to the model from the perspective of a lightweight network were also investigated, and the experimental results are shown in Table 4. After lightweight compression, the number of parameters, FLOPs, and model size of YOLOv5s were significantly reduced. Based on the lightweight model, the model slightly fluctuates in various parameters by adding a small object detection layer, where GhostConv was used to replace ordinary convolution, and the CBAM was added. Finally, compared with YOLOv5s, the parameters, FLOPs, and model size of MDP-YOLO were decreased by 72.4%, 89.2%, and 70.1%, respectively, while the precision and recall were increased by 6.7% and 5.0%, and the FPS was increased by 27. These results indicate that the proposed model achieved lightweight compression while ensuring the accuracy of rice pest recognition.

TABLE 4. Analysis of lightweight results.

Model	Params (M)	FLOPs (G)	Speed (FPS)	Size (M)	Layers	P (%)	R (%)
YOLOv5s	7.03	15.8	27	14.4	213	90.4	91.8
Model 1	3.33	1.9	59	6.9	218	90.7	88.2
Model 2	3.48	2.6	51	7.4	264	91.6	85.3
Model 3	2.89	1.8	57	6.0	234	93.9	89.0
Model 4	3.02	2.4	49	6.4	288	92.8	89.5
Model 5	2.40	1.9	57	5.2	222	95.3	94.4
MDP-YOLO	1.94	1.7	54	4.3	246	97.1	96.8

The improved MDP-YOLO model was then compred with other detection algorithm models, and the experimental results in Table 5 confirm its superior performance in terms of detecting rice pests. Compared to the other algorithms, the proposed model had the lowest number of parameters and FLOPs, thus effectively reducing the computational load. On the task of rice pest detection, we found not only that the precision, recall, and average precision (AP) were better than the other models, but that the size of the model was also smaller. The detection frame rate of the proposed algorithm model was 54 FPS, making it suitable for real-time detection in complex field environments. Compared with other models, this algorithm model had a high detection accuracy, with a mAP@.5 of 97.80% for rice pest detection.

TABLE 5. Comparison of results from alternative detection models.

Model	Params (M)	FLOPs (G)	Speed (FPS)	Size (M)	P (%)	R (%)	mAP@0.5
Faster-RCNN	137.11	79.20	14	108.0	86.21%	92.59%	94.79%
SSD	26.29	22.18	28	100.0	91.3%	87.20%	87.20%
YOLOv4	64.36	52.10	24	244.0	85.00%	84.00%	83.69%
YOLOv5s	7.03	15.80	27	14.40	90.40%	91.80%	92.30%
MDP-YOLO	1.94	1.70	54	4.30	97.10%	96.80%	97.80%

Discussion

YOLOv1 laid the foundation for the entire YOLO series, and the subsequent versions were improvements on the first version (Redmon et al., 2016). YOLOv1 applied an innovative approach to the tasks of classification and object localization, with a one-stage structure, but it had problems with spatial limitations and imprecise network losses (Ahmad et al., 2020). In YOLOv2, batch normalization was introduced and the fully connected layers were removed, which further improved the model's performance (Shi et al., 2021). YOLOv3 included bounding box prediction, and Darknet-53 was used to extract features based on the scheme used in YOLOv2 (Tian et al., 2019). In YOLOv4, data processing was optimized, and the training of the backbone network, the activation functions, and loss functions were to different extents based on the original YOLO object recognition architecture (Guo et al., 2021). YOLOv5 included some new improvements to YOLOv4, which greatly improved its speed and accuracy (Yuan et al., 2022).

To further explore the role of data enhancement in this experiment, two batches of datasets, with and without data enhancement, were used for comparative experiments before and after the proposed improvements to the model. The experimental results are shown in Table 6.

TABLE 6. Comparison of results with and without data enhancement.

Model	Data enhancement	Params (M)	FLOPs (G)	Speed (FPS)	Size (M)	P (%)	R (%)	mAP@0.5
YOLOv5s	√	7.03	15.80	27	14.40	90.40%	91.80%	92.30%
MDP-YOLO	√	1.94	1.70	54	4.30	97.10%	96.80%	97.80%
YOLOv5s	×	7.03	15.80	27	14.40	90.40%	86.80%	89.30%
MDP-YOLO	×	1.94	1.70	54	4.30	97.10%	94.30%	95.50%

As can be seen from Table 6, data enhancement was helpful for improving the performance of the model, and the data-enhanced YOLOv5s and MDP-YOLOv5s models had significantly improved precision, recall and mAP@.5 indices. In summary, the data enhancement step effectively improved the model performance, although this effect may be different for different models, and a specific analysis is required.

The results of this study provide significant theoretical and technical support for the lightweight deployment of rice pest detection models, which is essential for the advancement of intelligent agriculture (Hassan et al., 2023). In future work, we intend to continue our research and efforts in the following areas. Firstly, we will further optimize and enhance the lightweight performance of the rice pest detection model, which will involve adopting more advanced and efficient algorithms and architectures, further compressing and optimizing the models, and fully utilizing the hardware resources of mobile devices to improve the detection speed and accuracy (Liu et al., 2022b). Secondly, we will explore and develop more intelligent and adaptive rice pest detection algorithms and systems, including the use of emerging technologies such as deep reinforcement learning, to develop more intelligent rice pest detection systems in which adaptive learning and optimization are used to enhance the detection stability and robustness (Li et al., 2021c). Finally, we will expand and apply this rice pest detection technology to the actual context of agricultural production and management, by integrating it with other intelligent agricultural technologies, optimizing agricultural production processes, and improving agricultural production efficiency and quality (Cheng et al., 2017). Overall, the proposed lightweight deployment of rice pest detection technology will form a critical foundation for the development of intelligent agriculture, and further research and exploration are necessary in terms of both technology and applications to meet the practical needs of agricultural production and management.

CONCLUSIONS

The deployment of rice pest detection models on mobile devices requires careful consideration of factors such as the accuracy, speed, and computational capabilities of the model. To address these concerns, this study has presented a lightweight rice pest detection model based on YOLOv5s. The proposed model involves significantly fewer computational parameters and operations, with a greatly reduced size, making it ideal for deployment on mobile terminals with limited computational capabilities and storage space. The proposed model achieved a rice pest detection frame rate of 54 FPS, with a precision of 97.10%, recall of 96.80%, mAP@.5 of 97.8%, and mAP@.5:.95 of 66.2%. Compared with other mainstream detection models, the MDP-YOLO model offers advantages such as low computational requirements, small model size, and high detection accuracy. Our study provides both theoretical and technical support for the lightweight deployment of rice pest detection models in practical scenarios.

ACKNOWLEDGEMENTS

This study was financially supported by National Natural Science Foundation of China (61863027).

REFERENCES

Ahmad T, Ma Y, Yahya M, Ahmad B, Nazir S, Haq A ul (2020) Object detection through modified YOLO neural network. Scientific Programming 2020: 1-10. https://doi.org/10.1155/2020/8403262

Alirezazadeh P, Rahimi-Ajdadi F, Abbaspour-Gilandeh Y, Landwehr N, Tavakoli H (2021) Improved digital image-based assessment of soil aggregate size by applying convolutional neural networks. Computers and Electronics in Agriculture 191: 106499. https://doi.org/10.1016/j.compag.2021.106499

Alirezazadeh P, Schirrmann M, Stolzenburg F (2023) Improving deep learning-based plant disease classification with attention mechanism. Gesunde Pflanzen 75: 49-59. https://doi.org/10.1007/s10343-022-00796-y

Cheng X, Zhang Y, Chen Y, Wu Y, Yue Y (2017) Pest identification via deep residual learning in complex background 141: 351-356. https://doi.org/10.1016/j.compag.2017.08.005

Cheng Z, Huang R, Qian R, Dong W, Zhu J, Liu M (2022) A lightweight crop pest detection method based on convolutional neural networks. Applied Sciences 12: 7378. https://doi.org/10.3390/app12157378

Deléglise H, Interdonato R, Bégué A, d'Hôtel EM, Teisseire M, Roche M (2022) Food security prediction from heterogeneous data combining machine and deep learning methods. Expert Systems with Applications 190: 116189. https://doi.org/10.1016/j.eswa.2021.116189

Deng F, Mao W, Zeng Z, Zeng H, Wei B (2022) Multiple diseases and pests detection based on federated learning and improved faster R-CNN. IEEE Transactions on Instrumentation and Measurement 71: 1-11. https://doi.org/10.1109/TIM.2022.3201937

Guo F, Qian Y, Shi Y (2021) Real-time railroad track components inspection based on the improved YOLOv4 framework. Automation in construction 125: 103596. https://doi.org/10.1016/j.autcon.2021.103596

Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C (2020) Proceedings of the Computer Vision and Pattern Recognition (CVPR). Ghostnet: More features from cheap operations: 1580-1589. https://doi.org/10.1109/CVPR42600.2020.00165

Hassan SI, Alam MM, Illahi U, Suud MM (2023) A new deep learning-based technique for rice pest detection using remote sensing. PeerJ Computer Science 9: e1167. https://doi.org/10.7717/peerj-cs.1167

Hu X, Liu Y, Zhao Z, Liu J, Yang X, Sun C, Chen S, Li B, Zhou C (2021) Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network. Computers and Electronics in Agriculture 185: 106135. https://doi.org/10.1016/j.compag.2021.106135

Li J, Yu T, Yang B (2021c) Adaptive controller of PEMFC output voltage based on ambient intelligence large-scale deep reinforcement learning. IEEE Access, 9: 6063-6075. https://doi.org/10.1109/ACCESS.2020.3049072

Li K, Wang J, Jalil H, Wang H (2023) A fast and lightweight detection algorithm for passion fruit pests based on improved YOLOv5. Computers and Electronics in Agriculture 204: 107534. https://doi.org/10.1016/j.compag.2022.107534

Li K, Zhu J, Li N (2021a) Lightweight automatic identification and location detection model of farmland pests. Wireless Communications and Mobile Computing 2021: 9937038. https://doi.org/10.1155/2021/9937038

Li S, Li Y, Li Y, Li M, Xu X (2021b) Yolo-firi: Improved yolov5 for infrared image object detection. IEEE access, 9: 141861-141875. https://doi.org/10.1109/ACCESS.2021.312087

Liu J, Wang X (2021) Plant diseases and pests detection based on deep learning: a review. Plant Methods 17: 22. https://doi.org/10.1186/s13007-021-00722-9

Liu Q, Bi J, Zhang J, Bu X, Hanajima N (2022a) B-FPN SSD: an SSD algorithm based on a bidirectional feature fusion pyramid. The Visual Computer 1-13. https://doi.org/10.1007/s00371-022-02727-4

Liu S, Fan H, Ferianc M, Niu X, Shi H, Luk W (2022b) Toward full-stack acceleration of deep convolutional neural networks on FPGAs. IEEE Transactions on Neural Networks and Learning Systems 33: 3974-3987. https://doi.org/10.1109/TNNLS.2021.3055240

Ma N, Zhang X, Zheng H-T, Sun J (2018) Proceedings of the Computer Vision and Pattern Recognition (CVPR). Shufflenet v2: Practical guidelines for efficient cnn architecture design: 116-131. https://doi.org/10.1007/978-3-030-01264-9_8

Martínez-Ferrer L, Piles M, Camps-Valls G (2021) Crop yield estimation and interpretability with gaussian processes. IEEE Geoscience and Remote Sensing Letters 18: 2043-2047. https://doi.org/10.1109/LGRS.2020.3016140

Redmon J, Divvala S, Girshick R, Farhadi A (2016) Proceedings of the Computer Vision and Pattern Recognition (CVPR). You only look once: unified, real-time object detection: 779-788. https://doi.org/10.1109/CVPR.2016.91

Shi B, Li X, Nie T, Zhang K, Wang W (2021) Multi-object recognition method based on improved yolov2 model. Information Technology and Control 50: 13-27. https://doi.org/10.5755/j01.itc.50.1.25094

Sun X, Wu P, Hoi SCH (2018) Face detection using deep learning: an improved faster RCNN approach. Neurocomputing 299: 42-50. https://doi.org/10.1016/j.neucom.2018.03.030

Tian Y, Yang G, Wang Z, Wang H, Li E, Liang Z (2019) Apple detection during different growth stages in orchards using the improved YOLO-V3 model. Computers and electronics in agriculture 157: 417-426. https://doi.org/10.1016/j.compag.2019.01.012

Uhlemann E (2019) Trusting Autonomous Vehicles [Connected and Automated Vehicles]. IEEE Vehicular Technology Magazine 14: 121-124. https://doi.org/10.1109/MVT.2019.2905521

Wang X, Liu J, Zhu X (2021) Early real-time detection algorithm of tomato diseases and pests in the natural environment. Plant Methods 17: 43. https://doi.org/10.1186/s13007-021-00745-2

Woo S, Park J, Lee J-Y, Kweon IS (2018) Proceedings of the European conference on computer vision (ECCV). Cbam: Convolutional block attention module 3-19. https://doi.org/10.48550/arXiv.1807.06521

Xiao J, Liu G, Wang K, Si Y (2022) Cow identification in free-stall barns based on an improved Mask R-CNN and an SVM. Computers and Electronics in Agriculture 194: 106738. https://doi.org/10.1016/j.compag.2022.106738

Yuan S, Wang Y, Liang T, Jiang W, Lin S, Zhao Z (2022) Real-time recognition and warning of mask wearing based on improved YOLOv5 R6. 1. International Journal of Intelligent Systems 37: 9309-9338. https://doi.org/10.1002/int.22994

Zha M, Qian W, Yi W, Hua J (2021) A lightweight YOLOv4-Based forestry pest detection method using coordinate attention and feature fusion. Entropy 23: 1587. https://doi.org/10.3390/e23121587

Zhang X, Zhou X, Lin M, Sun J (2018) Proceedings of the Computer Vision and Pattern Recognition (CVPR). Shufflenet: an extremely efficient convolutional neural network for mobile devices: 6848-6856. https://doi.org/10.48550/arXiv.1707.01083