



OPEN Detection and identification of centipedes based on deep learning

Weitao Chen¹, Zhaoli Yao², Tao Wang¹, Fu Yang¹, Weiwei Zu¹, Chong Yao²✉ & Liangquan Jia¹✉

The quantification of centipede populations is one of the key measures in achieving intelligent management of edible centipedes and promoting the upgrade of the rural centipede industry chain. However, current centipede counting techniques still face several challenges, including low detection accuracy, large model size, and difficulty in deployment on mobile devices. These challenges have limited existing network models to the experimental stage, preventing their practical application. To tackle the identified challenges, this study introduces a lightweight centipede detection model (FCM-YOLO), which enhances detection performance while ensuring fast processing and broad applicability. Based on the YOLOv5s framework, this model incorporates the C3FS module, resulting in fewer parameters and increased detection speed. Additionally, it integrates an attention module (CBAM) to suppress irrelevant information and improve target focus, thus enhancing detection accuracy. Furthermore, to enhance the precision of bounding box positioning, this study proposes a new loss function, CMPDIU, for bounding box loss. Experimental results show that FCM-YOLO, while reducing parameter size, achieves an improved detection accuracy of 97.4% (2.7% higher than YOLOv5s) and reduces floating-point operations (FLOPs) to 11.5G (4.3G lower than YOLOv5s). In summary, this paper provides novel insights into the detection and enumeration of centipedes, contributing to the advancement of intelligent agricultural practices.

Keywords Centipede, YOLOv5, Computer vision, Object detection, Counting

Centipedes, as a type of edible insect, are extensively used in the manufacture of traditional Chinese medicinal materials and as components in food products, particularly in the herbal medicine market of southern China, where their demand is significant. According to statistics, in 2021 alone, China's trade volume of centipedes reached approximately 304.49 tons, and this figure increased to 312.78 tons in 2022¹. Furthermore, due to the scarcity of centipedes as a resource, limited production scale, and low technical complexity, coupled with the scarcity of breeding stock, their market price remains high, indicating a promising future for the centipede breeding industry. However, with the continuous growth in centipede production, the demand for accurate centipede counting is also on the rise. Presently, centipede counting primarily relies on manual labor, which is not only time-consuming and inefficient but also results in the inefficient use of human resources. The repetitive and mechanical nature of this task often leads to deviations in centipede counting, making it challenging to ensure accuracy. This poses a particular challenge for large-scale centipede farms, where manual counting further increases operational costs. Therefore, it is imperative to conduct comprehensive research on centipede detection and counting to enhance the efficiency and accuracy of these processes.

In recent years, object detection has gained popularity as a prominent field within computer vision and digital image processing². It has found extensive applications in the agricultural sector. This approach, leveraging computer vision, has the potential to significantly reduce the demand for human resources, providing considerable practical benefits. As a result, object detection has emerged as a focal point in both theoretical and applied research, witnessing rapid advancements in its algorithms.

In the field of object detection, many researchers utilize traditional algorithms to detect and classify image data. The efficacy of these detection algorithms largely depends on the adequacy of the extracted and chosen features in representing the characteristics of the image, and the suitability of the selected classifier. However, these algorithms are often susceptible to external environmental factors. For instance, Rahim & Kamran³ proposed an algorithm that extracts geometric properties, color, and texture features from segmented hawthorn using the Gray Level Co-occurrence Matrix (GLCM) and various color spaces. This approach generates effective feature vectors through the QDA feature reduction method and classifies them using machine learning algorithms, achieving a 98.16% detection accuracy for hawthorn. Similarly, Kaewchote et al.⁴ employed image processing techniques

¹School of Information Engineering, Huzhou University, Huzhou 313000, China. ²Huzhou Central Hospital, Huzhou 313000, China. ✉email: ycu@hzhospital.com; 02426@zjhu.edu.cn

to identify shrimp targets, extracting local binary patterns and Red, Green, Blue (RGB) features of shrimp. They applied a random forest classifier for counting and identifying shrimp targets, achieving an accuracy of 98.50%. Nonetheless, these object detection algorithms have notable limitations, including high computational load, slow processing speed, numerous redundant windows, and a focus on single-context applications.

In contrast to conventional object detection techniques, a significant number of researchers are now pivoting towards the application of deep learning technologies for the purposes of object detection and recognition. This shift is attributed to the strong feature extraction and autonomous learning capabilities of deep learning, which have led to outstanding achievements in the field of object detection in complex natural environments^{5,6}. Deep learning techniques are particularly well-suited for application in the complex agricultural settings.

In two-stage object detection algorithms, the detection process is divided into two phases⁷. Initially, region proposal boxes are generated, followed by classification and regression correction using a convolutional neural network, as shown in Fig. 1(a). Two-stage detection algorithms are advantageous for their low false positive rate and reduced rate of missed detections. For instance, Zu et al⁸, used Mask R-CNN to detect and segment ripe green tomatoes, achieving an F1 score of 92.0% with an Intersection Over Union (IOU) threshold of 0.5. Similarly, Li et al⁹, introduced an integrated U-Net segmentation model for green apples, effectively combining U-Net's edge features with high-level features using the ASPP structure, significantly improving apple segmentation precision and the model's generalization ability. However, the slow speed and high computational complexity of two-stage object detection algorithms limit their application in real-time detection scenarios.

Single-stage object detection algorithms are regression-based detection methods¹⁰ that do not require a region proposal stage and directly produce the object's class probabilities and location coordinates. With just a single pass for detection, these algorithms can quickly provide the final detection results, as illustrated in Fig. 1(b). Prominent examples of such algorithms include Your Only Look Once (YOLO)¹¹, The Single Shot Detector (SSD)¹², and RetinaNet¹³. Due to their significantly faster detection speeds compared to traditional and two-stage object detection algorithms, single-stage object detection methods have become one of the most popular research directions in the field.

In recent years, the YOLO model has gained widespread attention due to its outstanding performance. While YOLO versions continue to evolve rapidly, current research primarily focuses on enhancing the performance of existing, reliable network models rather than developing novel ones. Wu et al¹⁴, employed an improved YOLOv3 model for cluster optimization, achieving rapid and accurate identification of banana fruits under natural lighting conditions with a 93% accuracy rate, striking a fine balance between speed and precision. Li et al¹⁵, proposed a green pepper recognition model based on YOLOv4-tiny, which integrated coordinate attention and multi-scale prediction mechanisms, and introduced an adaptive pyramid method to improve the detection accuracy of occluded and small targets, achieving a recall rate of 93.85% and an accuracy rate of 96.91%. To accurately detect tea buds, Xu et al¹⁶, combined the rapid detection capabilities of YOLOv3 with the high-precision classification ability of DenseNet201, significantly improving the detection accuracy of lateral tea buds by 10.60%.

To date, the YOLO algorithm has delivered satisfactory detection performance. However, the original model's large scale, numerous parameters, and high computational cost make its deployment on mobile devices expensive¹⁷. In response, researchers have shifted their focus towards developing lightweight models. Current efforts in model lightweighting primarily concentrate on reducing the number of model parameters, minimizing computational complexity, and shortening runtime. For instance, Xu et al¹⁸, proposed an apple leaf disease detection model based on ALAD-YOLO, which combines MobileNet-V3 basic blocks to compress the model size and utilizes group convolution in the Spatial Pyramid Pooling Cross-Stage Partial Convolution (SPPCSPC) module instead of ordinary convolution. This approach increased accuracy to 90.2% while reducing floating-point operations (FLOPs) to 6.1G. Wang & He¹⁹ introduced a channel-pruned YOLOv5s model for pre-thinned apple fruit detection, achieving accurate detection of apples in complex environments with 91.5% accuracy. Gui et al²⁰, integrated the Ghost_conv module into the YOLOv5 model and added a Bottleneck Attention Module (BAM) to suppress irrelevant information, resulting in a 9.66% increase in average precision and a reduction of 52.402 G in floating-point operations.

However, these lightweight neural network models were developed using datasets obtained from laboratories, characterized by simple data backgrounds, resulting in limited robustness of the models. They often show a marked decline in performance in complex background environments. To address this issue, we propose a rapid

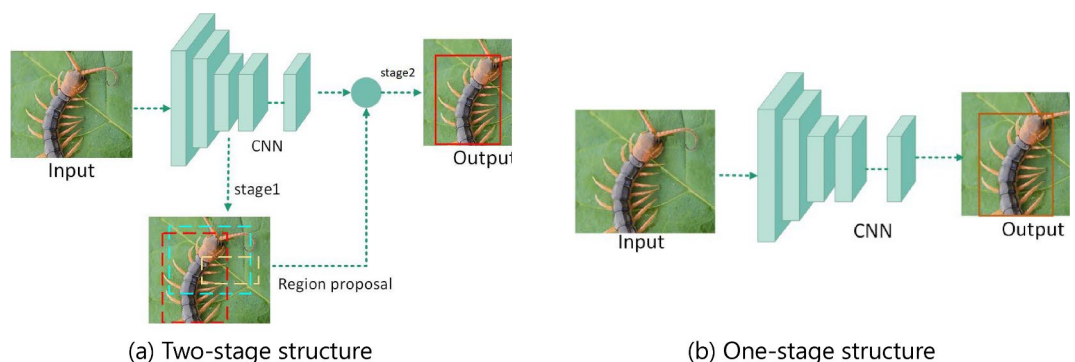


Fig. 1. Comparison between Two-Stage and Single-Stage Object Detection Algorithms.

and lightweight centipede detection algorithm, FCM-YOLO, based on an improved YOLOv5s. This algorithm marks the first application of a lightweight single-object detection algorithm in the field of centipede detection and counting, addressing the high costs of manual counting, the inefficiency of traditional algorithms, and the challenges of model deployment in current centipede counting practices. FCM-YOLO represents a lightweight network model that strikes a balance between detection precision and processing speed. It is more easily deployable on mobile devices without considering computational resources. The main contributions of this paper are as follows:

(1) To address the issue of large parameter and computational volume in existing models, which hinders ease of deployment, this study introduces a lightweight module, the C3FS module. This module not only diminishes the floating-point operations and parameter count of the model but also enhances the capability to extract features of occluded targets, significantly improving the model's recognition speed.

(2) To additionally improve the model's precision in recognition, this research proposes a novel loss function algorithm, CMPDIOU loss function, to supplant the existing CIOU loss function algorithm in the model. This change aims to achieve higher accuracy in evaluating the performance of centipede number detection.

(3) A CBAM attention module is integrated into the model. The CBAM attention module can more effectively capture target spatial and channel information without impacting the model's lightweight nature. It suppresses irrelevant information and extracts valuable information, thus enhancing the accuracy of the model's detection capabilities.

(4) To ensure optimal performance in centipede recognition, this study constructed a centipede target detection dataset in the laboratory and conducted multiple photography sessions according to real-world application scenarios to ensure the broad applicability of the data.

Materials and methods

yolov5 model

In the field of object detection, the YOLO algorithm has emerged as one of the most popular deep learning models to date. Despite the algorithm evolving to its YOLOv8 version²¹, YOLOv5 continues to be widely applied due to its superior performance. YOLOv5 is an open-source project developed by Ultralytics²². In contrast to previous object detection methods, YOLOv5 employs a novel detection strategy that divides images into smaller sections for prediction, significantly enhancing efficiency. The YOLOv5 model is comprised of three principal elements: the backbone, neck, and head.

The backbone component of the model utilizes Conv, C3, and SPPF structures for feature extraction from the input images. The Conv module, a fundamental convolution unit, operates throughout the architecture, sequentially performing 2D convolution, 2D normalization, and SiLU activation on the input. The C3 module, serving as the primary structure for feature extraction, is responsible for extracting and merging features, thereby enriching their semantic information. The SPPF module enriches the semantic information of features through pooling and feature fusion, endowing the deepest feature maps with rich semantic details.

The neck component focuses on merging three different scales of feature maps used for object detection with shallow features. This is achieved by incorporating a feature pyramid network with a bottom-up path enhancement characteristic, which merges the extracted feature maps with shallow features, ensuring that the feature maps possess both rich semantic information and accurate positional details of objects.

The head component convolves the three differently scaled feature maps, producing feature maps of dimensions $80 \times 80 \times 255$, $40 \times 40 \times 255$, and $20 \times 20 \times 255$, respectively. Different feature maps are utilized for detecting various target sizes; larger feature maps are used for detecting smaller objects, and smaller feature maps for larger objects. This approach effectively enables multi-scale object detection, enhancing the precision of the YOLO algorithm.

YOLOv5, based on network depth and feature layer width, is categorized into five distinct types: n, s, m, l, and x²³. With the increase in network depth and feature layer width, the model's performance improves. However, correspondingly, there is an increase in the number of parameters and computational requirements. This escalation also leads to higher demands on the hardware and longer detection times. Therefore, selecting an appropriate version of YOLOv5 is crucial. In this study, considering the trade-offs between the model's parameter count, floating-point operations, and mean accuracy, the YOLOv5s version was chosen as the base model for experimentation.

Design of FCM-YOLO

This section will detail the improvement strategies employed in our model. Our objective is to develop a simple and efficient network model for the detection and counting of centipede images. To achieve this, we introduce an innovative detection architecture that, while inheriting the base model, incorporates the C3FS module to reduce redundant computations significantly, thus lowering the model's computational requirements. Additionally, we have integrated the CBAM attention module, which effectively filters key information and suppresses irrelevant data. This approach not only further diminishes the parameter count but also augments the accuracy of the model's detection capabilities. Furthermore, we introduce a new loss function, the CMPDIOU loss function, which, as demonstrated by our experimental results, exhibits superior performance compared to other loss functions. The improved model's structural diagram is shown in Fig. 2.

CMPDIOU

The loss function is a crucial component of neural networks, primarily serving to measure the deviation between the network's predicted output and the desired output. A smaller deviation implies a lower value of the loss function²⁴. In the YOLO series, the loss function is divided into three parts. The first part is the class loss, used to predict the category of objects within the bounding boxes. The second part is the bounding box loss, responsible

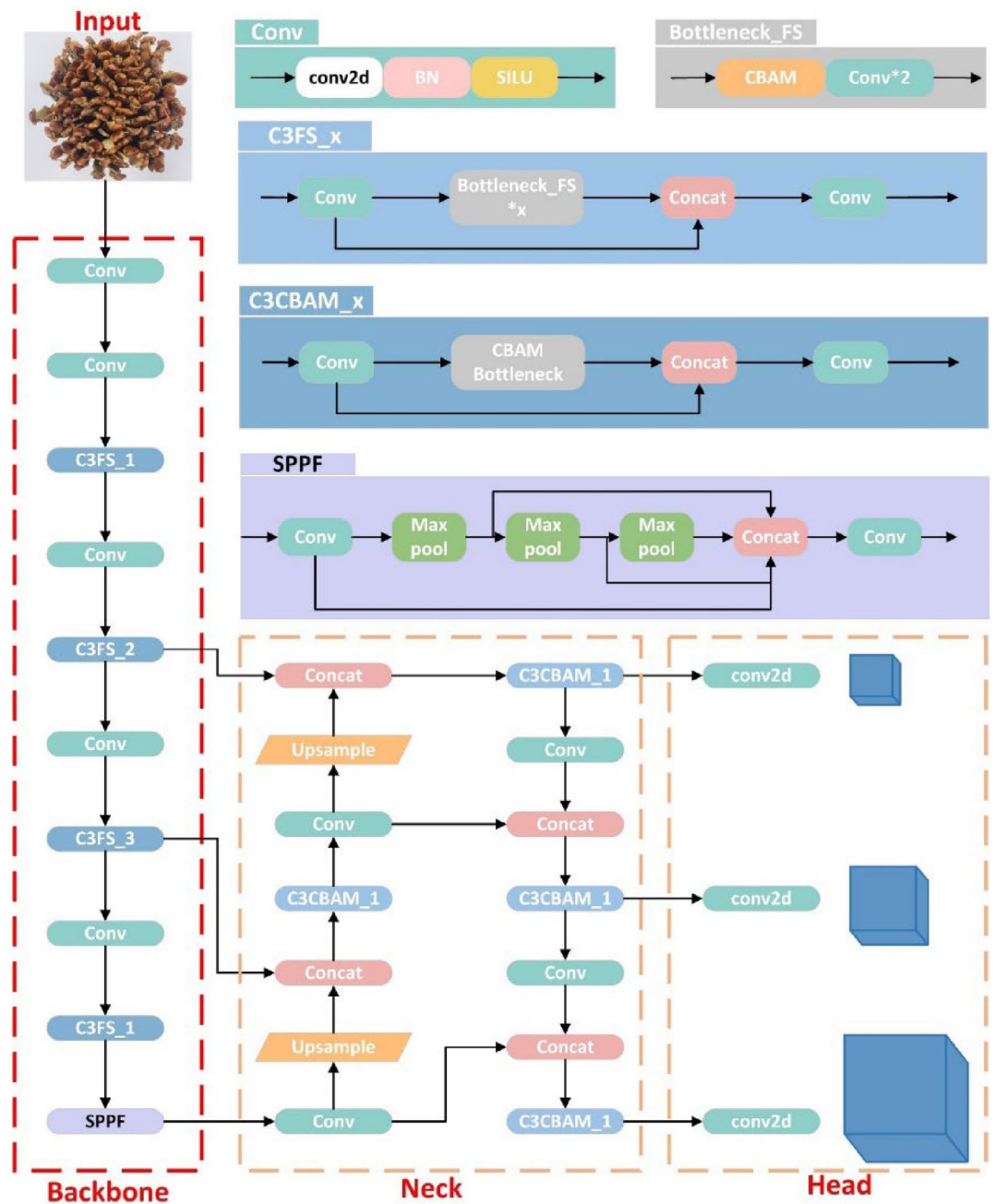


Fig. 2. Structural Model of the FCM-YOLO Network.

for calculating the error between the model's predicted bounding boxes and the actual bounding boxes. The third component is the confidence loss, assessing the precision of the model's certainty in its forecasts. In the YOLOv5 model, the confidence loss and class loss are computed utilizing the Binary Cross-Entropy (BCE) loss function. The bounding box loss is calculated using the CIoU loss function²⁵, the formula for which is as follows:

$$CIOU_Loss = 1 - IOU + \frac{\rho^2(\mathcal{B}, \mathcal{B}^{gt})}{c^2} + \alpha \nu \quad (1)$$

$$\alpha = \frac{\nu}{(1 - IOU) + \nu} \quad (2)$$

$$\nu = \frac{4}{\pi^2} \left[\left(\arctan \frac{w^{gt}}{h^{gt}} \right) - \left(\arctan \frac{w^{prd}}{h^{prd}} \right) \right]^2 \quad (3)$$

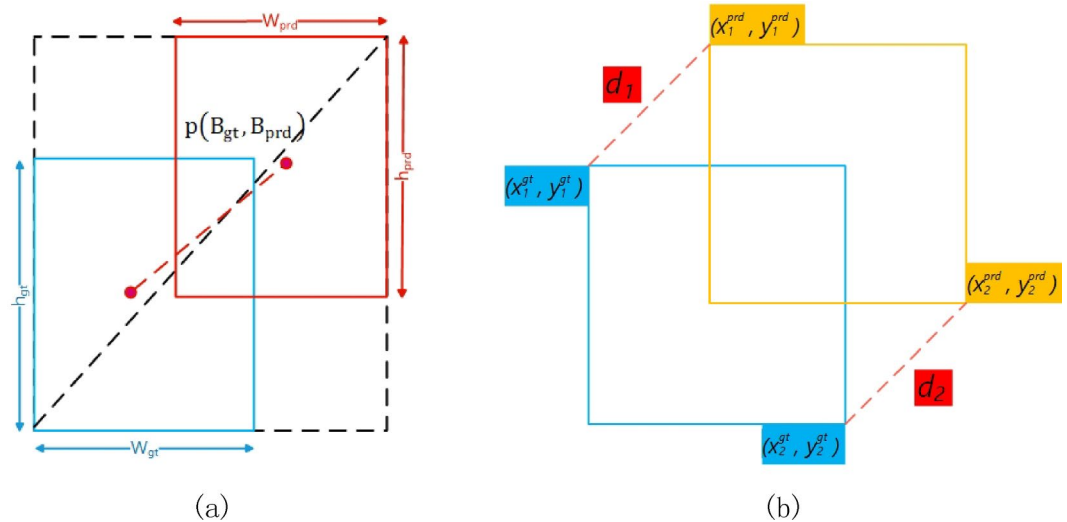


Fig. 3. Schematic Diagram of the Loss Function: **(a)** Schematic Diagram of the CIOU Loss Function; **(b)** Schematic Diagram of the MPDIOU Loss Function.

In this context, \mathcal{B} and \mathcal{B}^{gt} represent the center points of the predicted and actual bounding boxes, respectively. ρ denotes the straight-line distance between these two center points. c is the length of the diagonal of the smallest enclosing area between the predicted and actual bounding boxes. α is the weight factor of this function. ν measures the consistency of the aspect ratios between the two bounding boxes. h^{prd} and w^{prd} are the height and width of the predicted bounding box, while h^{gt} and w^{gt} correspond to the height and width of the actual bounding box.

In YOLOv5, the CIOU Loss considers three essential geometric aspects: the area of intersection, the distance between center points, and the proportion of dimensions. This function addresses the inclusion issue between the predicted and actual bounding boxes by directly regressing the distance between the centers of the two bounding boxes. The regression process also considers the aspect ratio of the bounding boxes. This approach not only enhances the accuracy of the bounding boxes but also improves the model's robustness in dealing with objects of varying sizes and shapes, as illustrated in Fig. 3.

Although CIOU Loss demonstrates remarkable efficacy in various aspects, it encounters two major challenges. Firstly, if the center points of the actual and predicted bounding boxes align precisely, but their widths and heights differ, CIOU Loss reverts to the conventional IOU loss function. Secondly, when the aspect ratios of the actual and predicted bounding boxes match, but their width and height values vary significantly, current bounding box regression loss functions struggle to optimize them efficiently. To overcome these limitations, the CIOU loss function has been substituted with the MPDIOU algorithm. MPDIOU is a boundary box similarity comparison metric based on the minimum point distance²⁶, encompassing all relevant factors considered in existing loss functions, as shown in Fig. 3. The MPDIOU loss function streamlines the assessment of resemblance between two bounding boxes, suitable for both overlapping and non-overlapping bounding box regression scenarios. This function eases the computation process by focusing on minimizing the direct point distances between the top-left and bottom-right corners of the predicted and the actual labeled boxes. The calculation method is as follows:

$$d_1^2 = (x_1^{prd} - x_1^{gt})^2 + (y_1^{prd} - y_1^{gt})^2 \quad (4)$$

$$d_2^2 = (x_2^{prd} - x_2^{gt})^2 + (y_2^{prd} - y_2^{gt})^2 \quad (5)$$

$$MPDIOU = \frac{A \cap B}{A \cup B} - \frac{d_1^2}{w^2 + h^2} - \frac{d_2^2}{w^2 + h^2} \quad (6)$$

In this context, w and h represent the width and height of the input image, respectively. The coordinates (x_1^{prd}, y_1^{prd}) and (x_2^{prd}, y_2^{prd}) denote the top-left and bottom-right points of the predicted box, respectively. Similarly, (x_1^{gt}, y_1^{gt}) and (x_2^{gt}, y_2^{gt}) represent the top-left and bottom-right points of the target box, respectively.

During practical application, this study observed that the loss function did not fully achieve the anticipated results. Upon deeper investigation, it was found that although the loss function accounted for the aspect ratio of the real and predicted bounding boxes, the algorithm primarily focused on distance loss rather than shape loss. This bias resulted in a slower convergence rate for the model compared to using the CIOU loss function. In light of this, we propose a new loss function algorithm, which we have named the CMPDIOU loss function algorithm. as shown in Fig. 4. Tests have shown that this algorithm has a faster convergence rate and demonstrates superior applicability in tasks involving the detection and counting of centipede numbers.

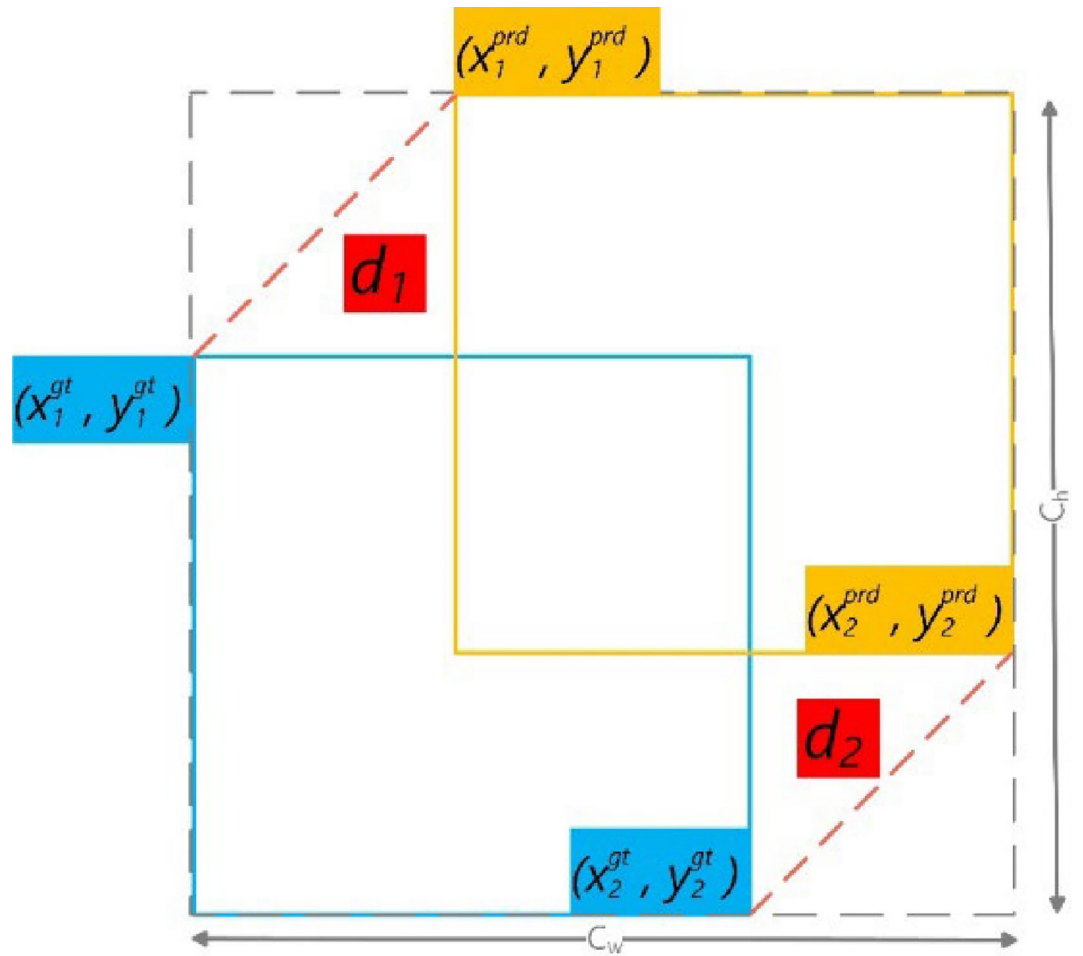


Fig. 4. Schematic Diagram of the CMPDIOU Loss Function.

CMPDIOU is a novel loss function algorithm proposed based on MPDIOU. This loss function adds a penalty factor for the length and width of the bounding box on top of what MPDIOU offers. This addresses the issue of slower convergence in terms of length and width observed in MPDIOU. In comparison to the CIoU loss function, CMPDIOU yields more accurate regression results. Since this loss function encompasses all relevant factors considered in existing loss functions, CMPDIOU can be considered an effective bounding box loss function algorithm for computer vision tasks. The calculation method is as follows:

$$d_1^2 = (x_1^{prd} - x_1^{gt})^2 + (y_1^{prd} - y_1^{gt})^2 \quad (7)$$

$$d_2^2 = (x_2^{prd} - x_2^{gt})^2 + (y_2^{prd} - y_2^{gt})^2 \quad (8)$$

$$rho_w = ((x_2^{prd} - x_1^{prd}) - (x_2^{gt} - x_1^{gt}))^2 \quad (9)$$

$$rho_h = ((y_2^{prd} - y_1^{prd}) - (y_2^{gt} - y_1^{gt}))^2 \quad (10)$$

$$CMPDIOU = \frac{A \cap B}{A \cup B} - \frac{d_1^2}{w^2 + h^2} - \frac{d_2^2}{w^2 + h^2} - \frac{rho_w}{C_w} - \frac{rho_h}{C_h} \quad (11)$$

In this context, w and h denote the width and height of the input image, respectively. C_w and C_h represent the width and height of the smallest external rectangle encompassing both the predicted box and the target box. The coordinates (x_1^{prd}, y_1^{prd}) and (x_2^{prd}, y_2^{prd}) indicate the top-left and bottom-right points of the predicted box, respectively. Similarly, (x_1^{gt}, y_1^{gt}) and (x_2^{gt}, y_2^{gt}) represent the top-left and bottom-right points of the target box, respectively. rho_w and rho_h represent the difference in width and height between the predicted box and the target box, respectively.

CBAM

The Convolutional Block Attention Module (CBAM)²⁷, presented at ECCV 2018, stands as a notable work in the field of attention mechanisms. This method models feature maps on both spatial and channel dimensions, effectively learning the specific contributions of each position and channel of the feature maps towards accurately predicting target categories during the training process. Based on the magnitude of these contributions, the mechanism allocates different weights, thereby enhancing the representation capability of key features while suppressing less significant ones, ultimately improving the model's overall predictive performance²⁸. The structure of CBAM primarily consists of two core components: the Channel Attention Module and the Spatial Attention Module. The specific working mechanism is illustrated in Fig. 5.

In this study, we have innovatively incorporated the CBAM attention mechanism into the existing model. This integration aims to reweight the feature maps within the model, allowing it to focus more on important features during the feature extraction phase.

During the operation of the Channel Attention Module, the input feature map F (HWC) is first introduced into the channel attention process. Here, the channel attention module employs global average pooling and global maximum pooling to spatially compress the feature map, resulting in $1 \times 1 \times c$ feature maps, where c represents the number of channels in the feature map. Subsequently, the two pooled features are fed into a shared Multilayer Perceptron (MLP) to extract deeper feature information. Following this, the feature maps processed by the MLP are summed and passed through a sigmoid activation function to obtain the final channel attention weights. The corresponding formula is as follows:

$$M_c(F) = \sigma (MLP (AvgPool (F)) + MLP (MaxPool (F))) = \sigma (W_1 (W_0 (F_{avg}^c)) + W_1 (W_0 (F_{max}^c))) \quad (12)$$

In this context, c represents the channel attention module, and σ denotes the sigmoid function. $AvgPool$ and $MaxPool$ refer to average pooling and maximum pooling, respectively. F_{avg}^c and F_{max}^c represent the features obtained from average pooling and maximum pooling. W_0 and W_1 are the weights of the Multilayer Perceptron (MLP).

To obtain attention features in the spatial dimension, the feature map output from the channel attention is also subjected to average pooling and maximum pooling based on the width and height of the feature map. This process yields two $wh \times 1$ feature maps. These pooled features are then concatenated and passed through a convolutional (Conv) layer for feature extraction. After sigmoid activation, the final spatial attention weights are obtained. The calculation process is as follows:

$$M_s(F) = \sigma (f^{7 \times 7} ([AvgPool (F); MaxPool (F)])) = \sigma (f^{7 \times 7} ([F_{avg}^s; F_{max}^s])) \quad (13)$$

In this context, s represents the spatial attention module. $AvgPool$ and $MaxPool$ correspond to average pooling and maximum pooling, respectively. $f^{7 \times 7}$ denotes a convolution operation with a filter size of 7×7 .

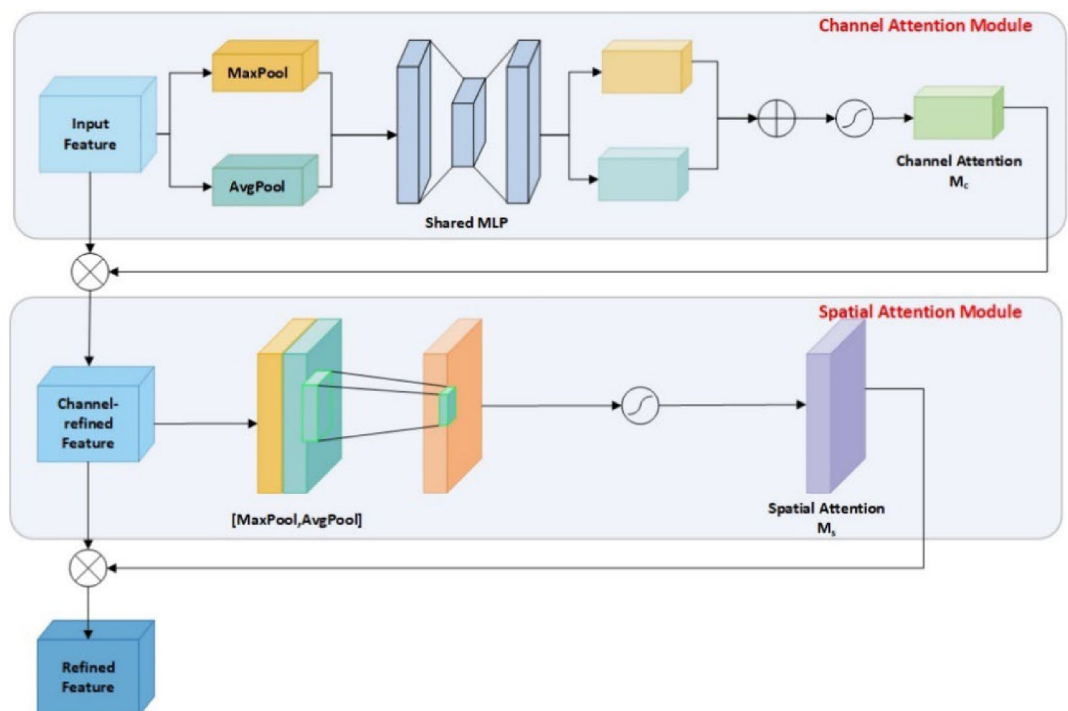


Fig. 5. Schematic Diagram of the CBAM Attention Mechanism.

C3FS module

The total number of model parameters directly impacts the computational resources required during the inference process. In practical application scenarios, selecting a model necessitates not only considering its excellence in accuracy but also its dependency on computational resources. Due to the limited storage space and computational capabilities of mobile terminals, it is essential to further reduce the computational demands and size of the model to facilitate future deployments on mobile devices. Within the context of the YOLOv5 framework, numerous developers have explored integrating streamlined backbone networks like MobileNet and GhostNet^{29,30}. Nevertheless, our evaluations indicated that while these adaptations markedly decreased the number of parameters, they concurrently led to a diminution in detection efficacy, rendering them less effective for mobile usage. To surmount these challenges, this study takes cues from the minimalist design ethos of FasterNet³¹ and introduces an innovative methodology. We have developed a C3FS module, aiming to significantly reduce the model's parameter load and computational intensity while preserving or enhancing its overall operational effectiveness.

In the YOLOv5 architecture, the C3 module, by incorporating multiple convolutional layers (Conv layers), effectively increases the depth of the network and the receptive field. This design significantly enhances the network's ability to extract complex features. However, the high number of parameters introduced by the Conv layers leads to redundant computations during inference, ultimately resulting in an increase in FLOPs, which to some extent hinders the model from achieving its goal of fast and lightweight detection. It is noteworthy that we observed that a reduction in FLOPs does not necessarily lead to a similar reduction in latency. This is mainly due to inefficiencies caused by low failure rates per second³².

To address this issue, we introduce a novel convolutional structure (PConv) and apply it to our model. The core idea of PConv is to apply traditional convolution to only a portion of the input feature map for spatial feature extraction while keeping the rest unchanged. During implementation, we prioritize continuous or distributed memory access patterns by selecting the first or last continuous channels of the feature map for computation to represent the entire feature map. This strategy, without compromising generality, ensures that the input and output feature maps maintain the same number of channels. Thus, PConv, by reducing redundant computations and memory access, can extract spatial features more efficiently. The specific structure is shown in the Partial Convolution (PConv) section in Fig. 6.

$$FLOPs_{Conv} = w \times h \times K_w \times K_h \times C_{in} \times C_{out} \quad (14)$$

$$C_p = \frac{1}{4}C_{in} = \frac{1}{4}C_{out} \quad (15)$$

$$FLOPs_{PConv} = w \times h \times K_w \times K_h \times C_p^2 = \frac{1}{16}FLOPs_{Conv} \quad (16)$$

Among them, $FLOPs_{Conv}$ represents the FLOPs value of the traditional convolution operation, and $FLOPs_{PConv}$ represents the FLOPs value of PConv. w and h are the width and height of the feature map, K_w and K_h denote the width and height of the convolution kernel, C_{in} and C_{out} represent the number of input and output channels, respectively. C_p denotes the number of input and output channels for PConv.

In PConv, since only 1/4 of the channels are involved in the computation, the FLOPs of PConv are only 1/16 of those in the conventional Conv, as shown in Eqs. (15) and (16).

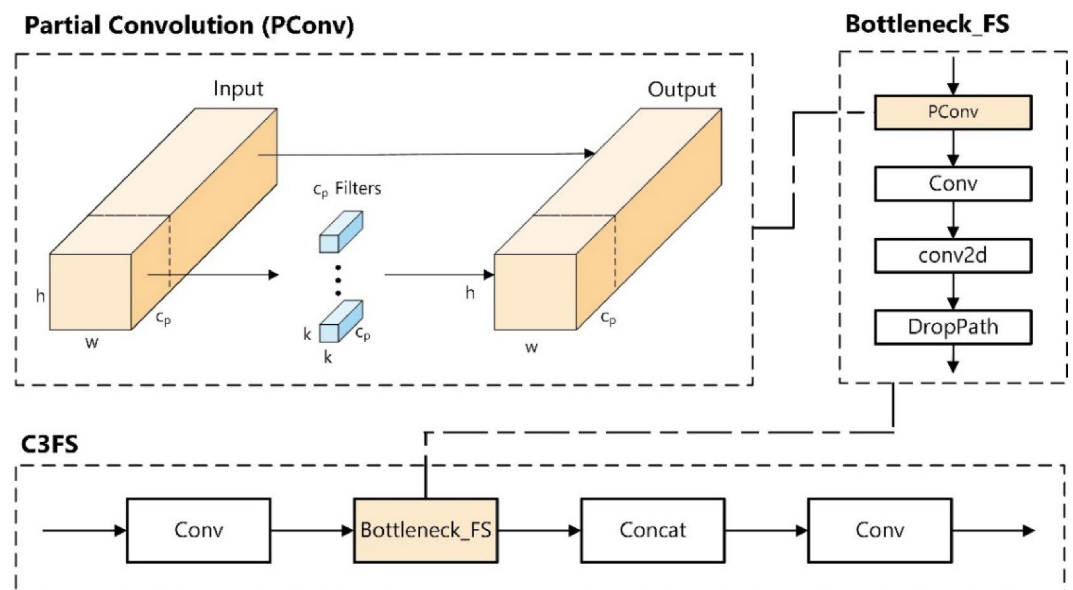


Fig. 6. Schematic Diagram of C3FS.

Items	values
Operating system	Window 11
CPU	I7-13650HX
GPU	RTX4060
Memory	8GB
Deep learning framework	Pytorch 1.11
Cuda version	11.7

Table 1. Main configuration of experimental equipment.

Hyperparameters	Value
lr0 (initial learning rate)	0.01
lrf (final OneCycleLR learning rate)	0.01
momentum (SGD momentum/Adam beta1)	0.937
box (box loss gain)	0.05
cls (cls loss gain)	0.5
cls_pw (cls BCELoss)	1.0
obj (obj loss gain)	1.0
obj_pw (obj BCELoss)	1.0
iou_t (IoU training threshold)	0.20
anchor_t (anchor-multiple threshold)	3
hsv_h (image HSV-Hue augmentation)	0.015
hsv_s (image HSV-Saturation augmentation)	0.7
hsv_v (image HSV-Value augmentation)	0.4
translate (image translation)	0.1
scale (image scale)	0.5
fliplr (image flip left-right)	0.5
mosaic (image mixup)	1.0

Table 2. Values of various parameters in YOLOv5s.

To strike a balance between detection accuracy and computational efficiency, we combined the PConv structure with the original C3 module, innovatively designing the C3FS module. As shown in Fig. 6, we reconstructed the Bottleneck module on the original foundation, naming it the Bottleneck_FS module. In the Bottleneck_FS module, we replaced part of the traditional convolution operations with PConv. This modification reduces the model parameters without affecting detection performance. The final results demonstrate that this improvement significantly reduces the computational cost during training without compromising final accuracy.

Experimental environment parameters and training parameters

Experimental Environment: All experiments in this study were conducted using the Pytorch deep learning framework and programmed in Python. The main specifications of the engineering machine used in this experiment are presented in Table 1.

Training Parameter Settings: Before training the model, hyperparameters were preset, as appropriate hyperparameters can enhance the model's performance. The YOLOv5 algorithm includes 28 hyperparameters, encompassing the learning rate, weights of different loss functions, weight decay coefficients, and various data augmentation parameters. The specific hyperparameters are detailed in Table 2. In this experiment, the improved FCM-YOLO was used as the training model for the centipede dataset. In consideration of deploying the final trained model in practical applications, we have set the parameters for the number of epochs, image size, and mosaic data augmentation to 300, 640×640, and 1, respectively. This setup aims to maximize the model's generalization capability without compromising image quality. Considering the performance of the experimental equipment, we set the batch size to 16. AP0.5 refers to the mean accuracy of all categories when the IOU threshold for accuracy evaluation is set to 0.5, while AP0.5-0.95 refers to the mean and weighted average accuracy when the IOU threshold for accuracy evaluation is increased from 0.5 to 0.95 in steps of 0.05.

Results

Model evaluation metrics

In order to evaluate the centipede detection model, this paper employs Precision, Recall, and Average Precision (AP) to assess the model's detection performance on the centipede dataset. AP is a crucial metric in object detection algorithms. It represents the mean precision at different Recall rates and can also be viewed as the

area under the precision-Recall curve. A higher AP value indicates better detection performance of the object detection model on the current dataset.

$$P = \frac{TP}{FP + TP} \quad (17)$$

$$R = \frac{TP}{FN + TP} \quad (18)$$

$$AP = \sum \int_0^1 P(R) dR \quad (19)$$

$$FLOPs_{Conv} = w \times h \times K_w \times K_h \times C_{in} \times C_{out} \quad (20)$$

Herein, P and R respectively represent Precision and Recall. TP (True Positive) refers to the number of correctly identified centipedes detected by the network model. FP (False Positive) denotes the count of incorrectly identified centipedes detected by the network model. FN (False Negative) indicates the actual number of centipedes that the network model failed to detect. w and h represent the width and height of the feature map, respectively. K_w and K_h denote the width and height of the convolution kernel, while C_{in} and C_{out} refer to the number of input channels and output channels, respectively.

Data collection and preprocessing

Data collection

Traditional centipede counting relies on laborious manual comparison, a method that is not only time-consuming and labor-intensive but also prone to errors, potentially leading to losses for farmers or businesses. Deep learning, especially the rapidly advancing field of object detection algorithms, has been widely applied in multiple relevant domains. However, deep learning models largely depend on suitable and effective datasets³³, and existing public datasets do not meet these requirements. To address this, our research team collected a large number of centipede images in the laboratory. The images were captured on September 24, 2023 (overcast conditions) and September 25, 2023 (sunny conditions). To ensure comprehensive data diversity and enhance the robustness and generalization capabilities of model training, we employed various shooting angles during image collection. To simulate real-world usage scenarios, we conducted the photography under both natural and artificial lighting conditions, ensuring the trained model's effective application in actual scenarios. The image data is illustrated in Fig. 7. Moreover, we manually selected images with clear and complete targets to ensure data quality, ultimately obtaining a total of 224 images, containing 22,400 detection targets.

To enhance the efficiency and convenience of dataset annotation, we utilized the online annotation platform MAKE-SENSE for dataset labeling and processing. MAKE-SENSE is a free online tool designed for annotating photos, which can be operated directly on a web browser without the need for downloading or installation. It is highly suitable for computer vision deep learning projects and supports multiple formats, making the dataset preparation process more straightforward and swift. To ensure the accuracy of dataset annotations, our research team selected professional dataset annotators from the group and established strict annotation standards for the dataset, effectively reducing precision loss due to human error in the model. In this experiment, to ensure the uniform distribution of various types of data during training, we randomly shuffled the order of samples in the dataset. Subsequently, the dataset was divided into training, validation, and test sets in the proportions of 80%, 10%, and 10%, respectively. The training set was used for model training, while the validation and test sets were utilized for evaluating the performance of the final model.

Data preprocessing

To enhance the generalization ability of the object detection model, enrich the image training set, effectively extract image features, and avoid overfitting, data augmentation was performed on the dataset before training in this experiment. Specifically, we employed the mosaic data augmentation technique³⁴, which randomly selects four images and performs operations such as random cropping, scaling, and rotation on them, eventually composing a single image. This approach significantly enriches the diversity of the background in which the detection objects are situated. The probability of image scaling and flipping was set at 50%, while the probabilities for adjusting hue, saturation, and brightness in color variation were 1.5%, 70%, and 40%, respectively. The probability of flipping was 10%. These measures collectively aim to improve the model's accuracy and robustness in processing diverse data.

Results and analysis

This paper utilizes a self-constructed dataset and conducts training for 300 epochs using an improved model. To prevent overfitting during the training process, this study plotted relevant curves during the training and validation phases. The specific results are illustrated in Fig. 8.

The two line charts in the first column display the trends of the bounding box loss function for the FCM-YOLO model during training, covering both the training and validation sets. The x-axis indicates the training epochs, and the y-axis shows the corresponding loss values. These charts reveal a continuous decrease in loss as training progresses, with values stabilizing at a relatively low level. This decreasing trend underscores the effectiveness of the CMPDIOU algorithm implemented in this experiment. The line charts for Precision and Recall are positioned in the fourth and fifth spots of the first row. Both metrics initially increase rapidly before gradually stabilizing, which indicates good convergence of the model without overfitting issues. The fourth and fifth line charts in the second row showcase AP50 and AP50-95, respectively. These metrics illustrate that the

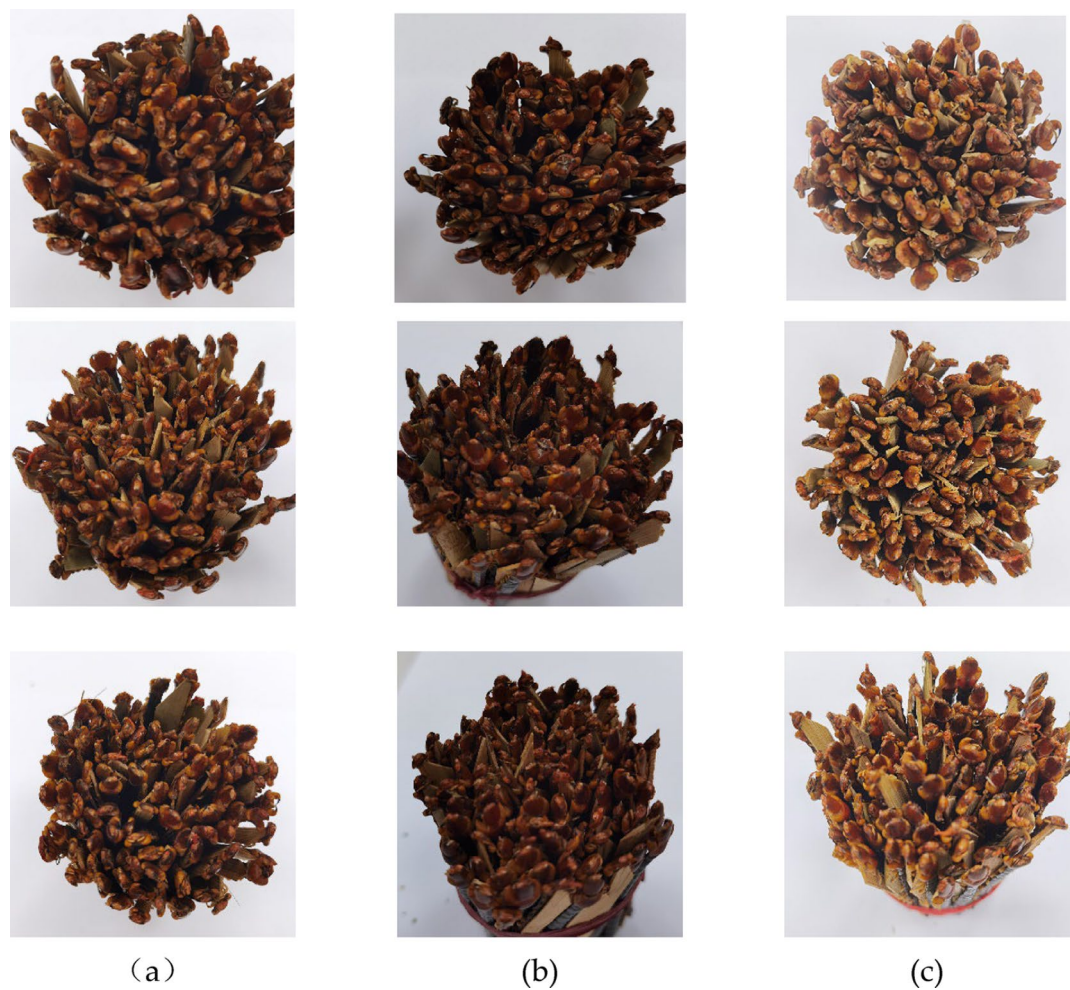


Fig. 7. Centipede Dataset. (a) Centipede image under sunny lighting conditions (b) Centipede image under overcast lighting conditions (c) Centipede image under artificial lighting conditions.

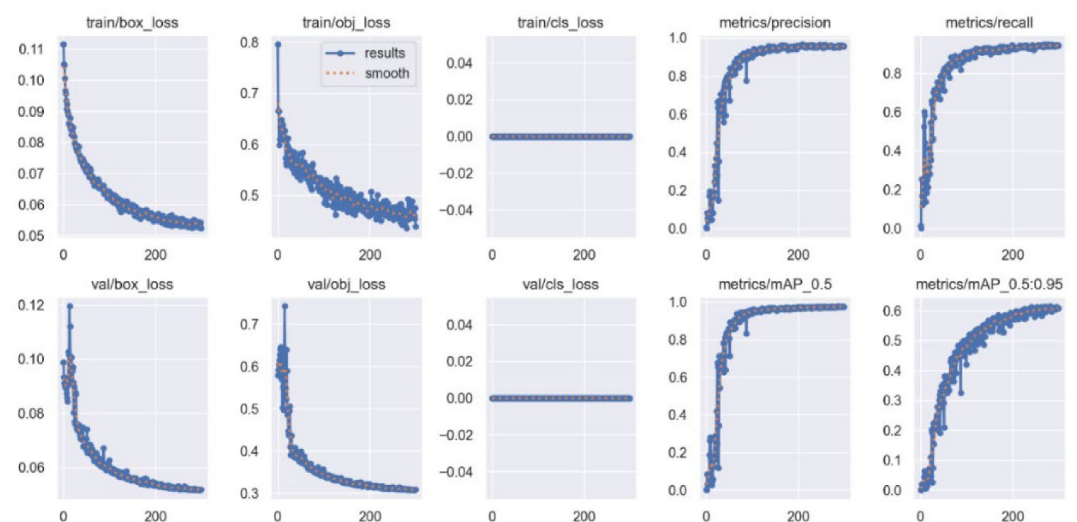


Fig. 8. Performance Metrics of the FCM-YOLO Model.

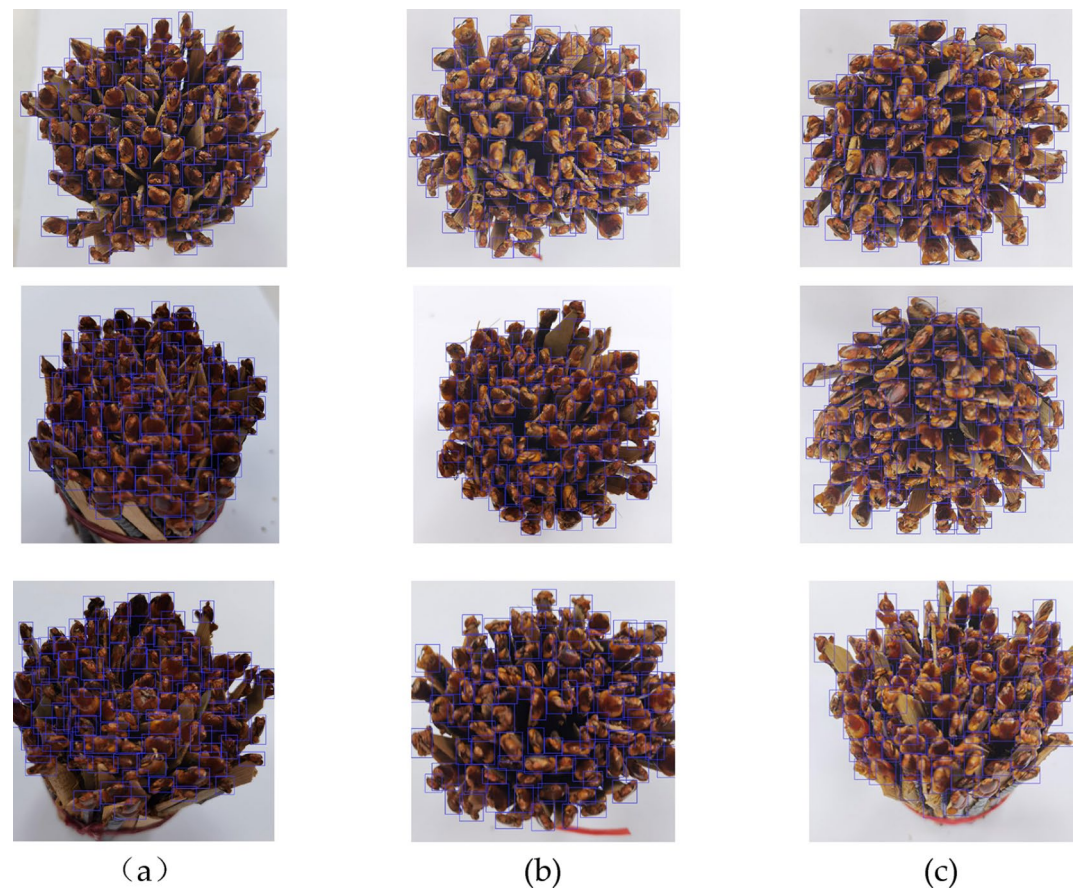


Fig. 9. Examples of Detection Results Based on the FCM-YOLO Model: (a) Image taken in overcast conditions. (b) Image taken under sunny conditions. (c) Images taken from different angles under indoor artificial lighting conditions.

Algorithms	C3FS	CBAM	CMPDIOU	Params(M)	FLOPs(G)	AP50(%)
YOLOv5s				7.01	15.8	94.7
Proposed Methods(1)	√			6.30	13.6	95.6
Proposed Methods(2)		√		5.99	13.8	96.4
Proposed Methods(3)			√	7.01	15.8	96.3
Proposed Methods(4)	√	√		5.28	11.7	96.6
Proposed Methods(5)	√		√	6.31	13.8	96.7
Proposed Methods(6)		√	√	5.99	13.7	96.8
Proposed Methods	√	√	√	5.28	11.5	97.4

Table 3. Ablation data.

model achieves a high and stable performance throughout the training process, confirming the success of the training strategy.

By inputting the images from the test set into the trained FCM-YOLO network model for detection, the results are shown in Fig. 9. From these results, it is evident that the proposed method can effectively identify and locate the vast majority of centipedes, demonstrating the model’s efficiency and accuracy in practical applications.

Comparison and experiment

Ablation experiment

In this study, we have innovatively improved the network structure and loss function of the Yolov5 detection algorithm. To comprehensively evaluate the impact of these improvements on performance, we conducted extensive ablation studies on the dataset constructed for the research. To ensure the reliability of the ablation experiments, all models were maintained with consistent hyperparameter configurations and operating environments. The experimental results are presented in Table 3.

IOU Loss Function	AP50(%)	AP50-95(%)	Precision(%)	Recall(%)
GIOU	95.9	60.8	94.4	92.2
DIOU	96.4	61.2	93.8	92.6
CIOU	96.6	60.5	95.5	91.3
MPDIOU	96.4	61.6	95.0	91.8
CMPDIOU	97.4	61.8	95.5	94.4

Table 4. Comparative results of different loss functions.

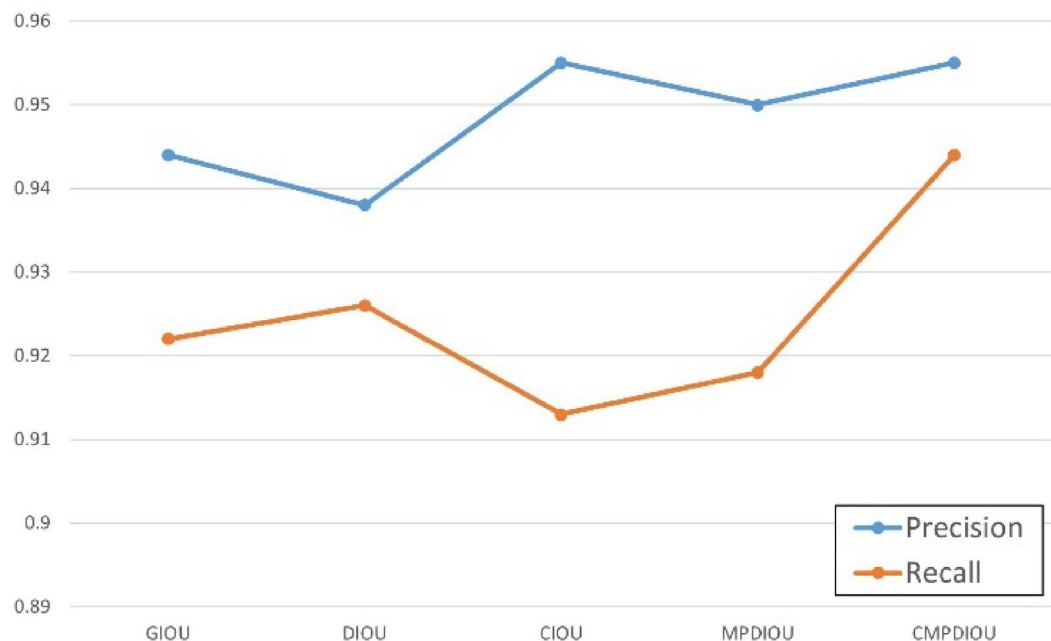


Fig. 10. Precision and Recall under Different Loss Functions.

The results indicate that modifications at different parts of the model have a positive impact. Compared to the original YOLOv5 model, replacing the original CIOU loss function algorithm with the CMPDIOU loss function algorithm, while keeping other parameters constant, increased the AP to 96.3%. The use of the CBAM attention mechanism reduced the number of parameters to 5.99 M, and simultaneously, the AP significantly increased to 96.4%. After adding the C3FS module to the original model, the AP rose to 95.6%, and the FLOPs substantially decreased, reaching 13.6G. The experimental results demonstrate that the various improvements made to the YOLOv5 model in this study play their intended role and offer a distinct advantage in detecting and counting centipedes in the dataset.

Comparison of different loss functions

This study further conducted a performance evaluation of the proposed CMPDIOU loss function algorithm by comparing it with several other loss functions through experimental analysis. Specifically, we examined the impact of various loss functions on the experimental results, including GIOU³⁵, DIOU³⁶, CIOU, MPDIOU, and CMPDIOU. The specific results of this comparison are presented in Table 4.

Compared to other loss functions, the CMPDIOU loss function achieved an accuracy of 97.4%. In terms of Precision and Recall, the performance is illustrated in Fig. 10. The model using the CMPDIOU loss function outperforms the other loss functions in both accuracy and recall rate. The experimental results indicate that our proposed CMPDIOU loss function surpasses the other four loss functions in every aspect. Therefore, we believe that employing the CMPDIOU loss function as the bounding box loss function for the FCM-YOLO model is the optimal choice, significantly enhancing the model's overall performance in object detection tasks.

Comparison of different models

To comprehensively validate the effectiveness of the FCM-YOLO algorithm, we compared and analyzed it against currently popular single-stage object detection algorithms. These comparative algorithms include the latest yolov7 algorithm³⁷, yolov8 algorithm, and lightweight improved algorithms such as the mobilenet+YOLOv5s algorithm³⁸, and the shufflenet+YOLOv5s³⁹ algorithm. To ensure the accuracy of the final experimental results, we trained and tested these eight networks using the same training set, validation set, and test set. The

Algorithms	Image Size	Params(M)	Precision(%)	Recall(%)	FLOPs(G)	AP50(%)
YOLOv5-Shufflenet2	640×640	3.17	89.8	83.4	5.8	91.5
YOLOv5-Mobilenet3-small	640×640	1.38	93.5	87.0	2.5	94.5
YOLOv5-Mobilenet3-large	640×640	5.08	94.1	92.4	11.2	96.0
YOLOv5-Mobilenet3-CARAFE	640×640	5.07	93.6	91.8	11.6	95.7
Yolov7	640×640	9.32	94.4	90.7	26.7	96.3
Yolov7-tiny	640×640	6.01	94.8	89.5	13.2	95.3
Yolov8-s	640×640	11.13	95.1	93.6	28.4	97.4
FCM-YOLO	640×640	5.28	95.5	94.4	11.5	97.4

Table 5. Comparative results of different models.

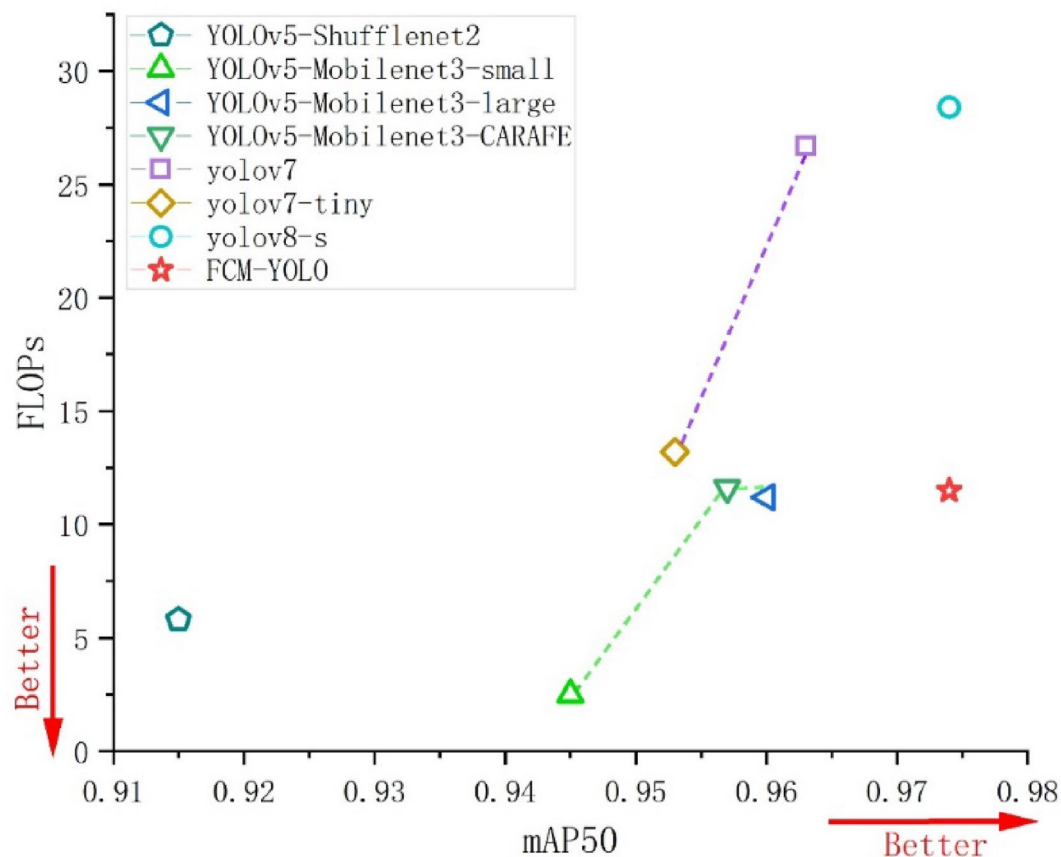


Fig. 11. Comparison of Different Models.

experimental results are presented in Table 5. It is noteworthy that, in this experiment, none of the models used pre-trained weights.

The experimental results demonstrate that the lightweight model proposed in this paper shows a notable improvement in accuracy compared to other lightweight models such as yolov5-shufflenet2 and yolov5-mobilenet3, with increases of 5.9% and 2.9% respectively. Compared to the current mainstream models like yolov7 and yolov8-s, the FCM-YOLO model still manages to significantly reduce the number of parameters and computational demands while maintaining accuracy, with reductions in FLOPs of 56.9% and 59.5% respectively. This endows the FCM-YOLO model with a distinct advantage in detection speed. Figure 11 visually compares the FLOPs and AP50 of the eight models. Among these models, yolov5-Mobilenet-small has the smallest computational load but relatively poorer performance. Yolov8-s has the highest accuracy, but its computational load is a substantial 28.4G, the highest among all models. The FCM-YOLO model, while only requiring 11.5G of computational resources, enhances the detection accuracy of the original model to 97.4%. Therefore, the FCM-YOLO model achieves the optimal balance between parameter size and accuracy in centipede detection and counting.

Discussion

In our research, we advanced the yolov5-s model, leading to the development of the FCM-YOLO model. This innovative model not only enhances detection precision but also accelerates the process, thereby enhancing its applicability for mobile device implementation. We embarked on a series of comparative analyses to ascertain the efficacy of FCM-YOLO. When benchmarked against the conventional YOLOv5s model, our FCM-YOLO demonstrates superior accuracy in detection, marked by an increased test set accuracy of 97.4%, which is 2.7% higher than that of YOLOv5s, and a substantial decrease in floating-point operations to 11.5G (4.3G less than yolov5s). Furthermore, FCM-YOLO exhibits an optimal balance of accuracy and compactness in comparison with other streamlined models such as YOLOv5-Mobilenet3-small, YOLOv5-Mobilenet3-large, YOLOv5-Mobilenet3-CARAFE, and YOLOv5-ShuffleNet2. This equilibrium positions FCM-YOLO as a more appropriate choice for tasks requiring both precision and efficiency.

In our in-depth study of the dataset and analysis of the test set results, we identified several issues within the model that warrant further exploration. Our constructed centipede dataset model covers a variety of scenarios encountered in practical applications, demonstrating the model's strong potential for application even in complex environments. However, in the process of actual application, we observed that the current targets to be detected may still experience missed or false detections due to various factors, such as camera shake or focus issues leading to unclear images, or obstructions of the target, as illustrated in Fig. 12. Therefore, in our subsequent research, we will focus on addressing these issues.

Additionally, when we applied the model in real-world production settings, we observed that the model's theoretical accuracy did not align with its actual detection precision. Various uncontrollable factors in practical usage, such as user habits and variations in the distance from which images are captured, tend to reduce the accuracy of the model's detections. Consequently, our research team plans to expand the dataset in subsequent studies to address this issue.

Additionally, the equipment and memory consumption required for mobile deployment are also crucial factors in practical use. The FCM-YOLO model, while reducing the number of parameters from the original model and employing the traditional attention mechanism CBAM module to enhance the model's focus on targets and improve detection capabilities, still has considerable room for improvement to further enhance efficiency and accuracy. In future research, we will actively explore more lightweight models to achieve broader application.

Conclusions

Exploring the implementation of centipede identification models, It's pivotal to assess not only the precision but also the speed of detection and the computational demand. This research introduces FCM-YOLO, a novel and more efficient centipede counting approach, as a modified iteration of yolov5. This advanced network model integrates diverse data enhancement strategies to improve the centipede dataset, thereby optimizing the process of model training and assessment. FCM-YOLO, an advanced yolov5 variant, incorporates the conventional CBAM attention mechanism to amplify the model's focus on relevant targets, thereby elevating its detection performance. This model replaces the CIOU loss with the more versatile CMPDIOU loss function, boosting its effectiveness and resilience in varied and complex environments. Additionally, the C3FS module is merged into the yolo framework, which further trims the model parameters and floating-point computations, yet preserves its accuracy. This significant reduction in the model size enables its deployment on portable devices. Empirical evaluations demonstrate that FCM-YOLO not only elevates the detection precision to 97.4% but also reduces the GFLOPs to 11.5G. When juxtaposed with leading detection models, FCM-YOLO excels in reduced computational demands, compact size, and superior detection precision. This study offers a fresh perspective on optimization strategies for related projects and contributes to the advancement of intelligent agriculture. Future investigations will aim at further refining the model to extend its applicability in diverse scenarios.



Fig. 12. Centipede Images Prone to Missed and False Detections (a) Partially obscured centipede image (b) Partially blurred centipede image due to camera shake or focus failure.

Data availability

The data presented in this study are available on request from the corresponding author.

Received: 28 January 2024; Accepted: 6 November 2024

Published online: 12 November 2024

References

1. YiWu. Abnormal climate reduces production of centipedes, with prices reaching historic highs. (2022).
2. Du, B. & Zhang, L. P. Target detection based on a dynamic subspace. *Pattern Recogn.* **47**(1), 344–358 (2014).
3. Azadnia, R. & Kheiralipour, K. Evaluation of hawthorns maturity level by developing an automated machine learning-based algorithm. *Ecol. Inf.* **71**, 101804 (2022).
4. Kaewchote, J., Janyong, S. & Limprasert, W. Image recognition method using local binary pattern and the Random forest classifier to count post larvae shrimp. *Agric. Nat. Resour.* **52**(4), 371–376 (2018).
5. Iandola, F. N. et al. *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size*. arXiv preprint arXiv:1602.07360, 2016.
6. Wu, Z., Shen, C. & Van Den Hengel, A. Wider or deeper: revisiting the resnet model for visual recognition. *Pattern Recogn.* **90**, 119–133 (2019).
7. Kinasih, F. et al. *Two-stage multiple object detection using CNN and correlative filter for accuracy improvement*. *Heliyon*, **9**(1). (2023).
8. Zu, L. L. et al. *Detection and segmentation of mature green tomatoes based on Mask R-CNN with Automatic Image Acquisition Approach*. *Sensors*, **21**(23). (2021).
9. Li, Q. W. et al. *A Novel Green Apple Segmentation Algorithm Based on Ensemble U-Net under Complex Orchard Environment*180 (Computers and Electronics in Agriculture, 2021).
10. Fu, K. et al. OSCD: a one-shot conditional object detection framework. *Neurocomputing*. **425**, 243–255 (2021).
11. Redmon, J. et al. *You only look once: Unified, real-time object detection*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. (2016).
12. Liu, W. et al. *Ssd: Single shot multibox detector*. in *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, Proceedings, Part I* 14. 2016. Springer. (2016).
13. Lin, T. Y. et al. *Focal loss for dense object detection*. in *Proceedings of the IEEE international conference on computer vision*. (2017).
14. Wu, F. Y. et al. *Multi-target Recognition of Bananas and Automatic Positioning for the Inflorescence Axis cutting point*. *Front. Plant Sci.*, **12**. (2021).
15. Li, X. et al. *Fast and Accurate Green Pepper Detection in Complex Backgrounds via an Improved YOLOv4-tiny Model*191 (Computers and Electronics in Agriculture, 2021).
16. Xu, W. et al. Detection and classification of tea buds based on deep learning. *Comput. Electron. Agric.* **192**, 106547 (2022).
17. Chen, S. et al. *An improved lightweight YOLOv5 algorithm for detecting strawberry diseases*. *IEEE Access.*, (2023).
18. Xu, W. S. & Wang, R. J. *ALAD-YOLO: an lightweight and accurate detector for apple leaf diseases*. *Front. Plant Sci.*, **14**. (2023).
19. Wang, D. & He, D. Channel pruned YOLO V5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning. *Biosyst. Eng.* **210**, 271–281 (2021).
20. Gui, Z. et al. A lightweight tea bud detection model based on YOLOv5. *Comput. Electron. Agric.* **205**, 107636 (2023).
21. Terven, J. & Cordova-Esparza, D. *A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond*. arXiv preprint arXiv:2304.00501, (2023).
22. Jocher, G. et al. *ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation*. Zenodo, (2022).
23. Ahmad, I. et al. *Deep learning based detector YOLOv5 for identifying insect pests*. *Appl. Sciences-Basel*, **12**(19). (2022).
24. Everingham, M. et al. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*. **88**, 303–338 (2010).
25. Gao, J. F. et al. *Detection of specific building in Remote sensing images using a novel YOLO-S-CIOU model*. *Case: Gas Station Identification*. *Sensors*, **21**(4). (2021).
26. Siliang, M. & Yong, X. *MPDIoU: A Loss for Efficient and Accurate Bounding Box Regression*. arXiv preprint arXiv:2307.07662, (2023).
27. Woo, S. et al. *Cbam: Convolutional block attention module*. in *Proceedings of the European conference on computer vision (ECCV)*. (2018).
28. Wang, Q. F. et al. *A deep Learning Approach Incorporating YOLO v5 and Attention Mechanisms for Field real-time Detection of the Invasive weed Solanum rostratum Dunal Seedlings*199 (Computers and Electronics in Agriculture, 2022).
29. Elfatimi, E., Eryigit, R. & Elfatimi, L. Beans Leaf diseases classification using MobileNet models. *Ieee Access*. **10**, 9471–9482 (2022).
30. Cao, M. L. et al. *Lightweight tea bud recognition network integrating GhostNet and YOLOv5*. *Math. Biosci. Eng.* **19**(12), 12897–12914 (2022).
31. Chen, J. et al. *Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks*. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. (2023).
32. Wang, Q. et al. *YOLO-SS-Large: a lightweight and high-performance model for defect detection in substations*. *Sensors*, **23**(19). (2023).
33. Zhu, Z. et al. *A review of the application of deep learning in intelligent fault diagnosis of rotating machinery*. *Measurement*, p. 112346. (2022).
34. Bochkovskiy, A., Wang, C. Y. & Liao, H. Y. M. *Yolov4: Optimal speed and accuracy of object detection*. arXiv preprint arXiv:2004.10934, (2020).
35. Rezaatofghi, H. et al. *Generalized intersection over union: A metric and a loss for bounding box regression*. in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. (2019).
36. Zheng, Z. et al. *Distance-IoU loss: Faster and better learning for bounding box regression*. in *Proceedings of the AAAI conference on artificial intelligence*. (2020).
37. Wang, C. Y., Bochkovskiy, A. & Liao, H. Y. M. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. (2023).
38. Yu, K. et al. *MobileNet-YOLO v5s: an Improved Lightweight Method for Real-Time detection of sugarcane stem nodes in Complex Natural environments*. *Ieee Access*. **11**, 104070–104083 (2023).
39. Xu, Y. L. et al. *Real-time object detection method of melon leaf diseases under complex background in greenhouse*. *J. Real-Time Image Proc.* **19**(5), 985–995 (2022).

Author contributions

Conceptualization, L.J. and W.C.; methodology, L.J. and W.C.; validation, L.J. and C.Y.; formal analysis, W.C.; investigation, L.J. and F.Y.; resources, Z.Y.; data curation, W.C.; writing—original draft preparation, W.C.; writing—review and editing, W.C.; visualization, W.Z.; supervision, L.J. and T.W.; project administration, L.J. and C.Y.; funding acquisition, Z.Y. All authors have read and agreed to the published version of the manuscript.

Funding

This research was supported by the Major Project for the Modernization of Traditional Chinese Medicine Jointly Established by the State Administration of Traditional Chinese Medicine and Zhejiang Province (GZY-ZJ-KJ-24050); Research on Quality Grade Evaluation and Application of Traditional Chinese Medicine Pieces Based on 'Optimal Shape-Quality-Efficacy' Correlation; Zhejiang Province Science and Technology Plan General Soft Science Research Project (2024C35128).

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to C.Y. or L.J.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2024