Research

RSR-YOLO: a real-time method for small target tomato detection based on improved YOLOv8 network

Xiang Yue[1] · Kai Qi[1] · Fuhao Yang[1] · Xinyi Na[1] · Yanhua Liu[1] · Cuihong Liu[1]

Received: 20 March 2024 / Accepted: 24 April 2024
Published online: 14 May 2024

Abstract
In tomato producing fields, automated large-area detection method is critical for fruit health monitoring and harvesting. However, due to the limited feature information included in tomatoes, large-area detection across long distances results in more missing or incorrect detections. To address this issue, this research proposes an improved YOLOv8 network, RSR-YOLO, for long-distance identification of tomato fruits. Firstly, this paper designs a partial group convolution (PgConv) and furthermore an innovative FasterNet (IFN) module for feature extraction, taking into account the impact of split operations on the computational complexity of the backbone network. The IFN module is lightweight and efficient, which improves the detection accuracy and real-time detection performance of the model. Secondly, this research combines the Gather and Distribute mechanism (GD) and redesigns the feature fusion module to implement the extraction and fusion of various levels of tomato features, given the critical significance that low-dimensional features play in small target recognition and localization. Finally, Repulsion Loss is used in this paper to examine the impact of fruit overlap and leaf occlusion on detection outcomes. RSR-YOLO achieves precision, recall, F1 score, and mean average precision ($mAP_{@0.5}$) of 91.6%, 85.9%, 88.7%, and 90.7%, respectively, marking increases of 4.2%, 4%, 4.2%, and 3.6% compared to YOLOv8n. In addition, this paper designs a specialized Graphical User Interface (GUI) for the real-time detection task of tomatoes.

Article Highlights

1. Larger features comprising low-dimensional texture details and position favor detection of small target tomatoes.
2. Combining global feature information with low-dimensional local features enhances the detection of small target tomatoes.
3. Enhancing the effectiveness of spatial feature extraction is the convolution operation processing of continuous features in the input channel and constant mapping of the remaining features.

Keywords Tomato detection · GD mechanism · Lightweight · Partial convolution

✉ Cuihong Liu, cuihongliu77@syau.edu.cn | [1]College of Engineering, Shenyang Agricultural University, Shenyang 110866, China.

1 Introduction

Over the past decade, fruit detection has undergone extensive research and application. Traditional fruit detection methods include single-feature analysis methods [1] (such as those based on color, geometric shape, and texture features); multi-feature fusion analysis methods (such as tomato recognition algorithms that fuse geometric shape features with color features) [2]; fruit recognition algorithms based on the fusion of color, intensity, edge, and direction features [3]; and citrus fruit recognition algorithms based on color threshold segmentation and edge perimeter [4]. Unfortunately, low robustness and time consumption issues plague both single-feature analysis approaches and multi-feature fusion methods. Detection techniques are vulnerable to a variety of environmental influences in complex environments with poor appearance [5].

Article Highlights

In recent years, the application of deep learning in fruit detection has significantly improved detection accuracy and robustness and reduced time consumption compared to traditional algorithms. For instance, Wang et al. [6] introduced an enhanced Faster R-CNN model, leveraging the attention mechanism principle. This method overcomes difficulties such fruit overlap, stem and leaf shadowing, and fluctuating light levels to accurately identify early tomato fruits. Wang et al. [7] developed an advanced Faster R-CNN model for recognizing and detecting tomato ripeness. The integration of high-dimensional semantic and low-dimensional information, enabled by the Path Aggregation Network (PANet), facilitates tomato ripeness detection in complex contexts. Afonso et al. [8] proposed a tomato detection method utilizing the Mask R-CNN model. The experimental results demonstrate that this method attains 95% accuracy for ripe tomatoes and 94% accuracy for unripe tomatoes. Fruit detection using two-stage detectors [9] is characterized by its high accuracy and robustness. However, in regions of interest, consecutive feature localization and pooling processes lead to slower detection speeds and higher processing demands [10, 11]. Conversely, the single-stage detector using the YOLO framework offers an advanced solution, balancing accuracy with inference time.

In terms of real-time detection of objects, the YOLO series has undergone multiple revisions and improvements. YOLOv1-v3 [12, 13] established a foundational single-stage detection framework for objects of various sizes. YOLOv4 [14] incorporates Mish activation functions, PANet networks, and innovative data augmentation techniques. YOLOv5 [15] introduced additional data augmentation strategies and model variants, building on the enhancements made in YOLOv4 [14]. YOLOX [16] employs a novel design featuring anchor-free detection, multiple positives, and a decoupled head. YOLOv6 [17] introduced a reparameterization method and an efficient Rep backbone network along with the Rep-PAN neck. YOLOv7 [18] concentrates on optimizing gradient paths and introduces the E-ELAN structure. YOLOv8 [19] achieves the SOTA of the current YOLO family by combining the strongest features of earlier YOLO models.

The YOLO-Tomato network was presented by Liu et al. [20]. It uses circular bounding boxes instead of the more common rectangular bounding boxes and implements a sensing architecture to improve the tomato recognition accuracy of YOLOv3. Appe et al. [21] improved the feature fusion capabilities of the network by integrating the convolutional block attention module [22] into the neck module of YOLOv5. However, instead of enhancing network performance by increasing computational and parametric capacities, an increasing number of studies are advocating for a lightweight model design to optimize adaptation to real-time detection tasks [23, 24]. Zeng et al. [25] replaced the focus layer in YOLOv5s with a downsampling convolutional layer and designed a lightweight backbone network in combination with MobileNetV3 to further minimize the size of the model. The enhanced network attains an average precision of 96.9% for tomatoes at various ripening stages, with a 64.88% improvement in inference time compared to the original YOLOv5s. Mbouembe et al. [26] introduced a lightweight YOLOv4-tiny network for tomato fruit detection. Specifically, they utilized the bottleneck module and a simplified BottleneckCSP module in place of Cross Stage Partial (CSP)Networks in the backbone; additionally, the content aware reassembly of features replaced the upsampling operation in the neck module. This approach enhances the detection accuracy while reducing the parameter count and computational complexity of the model. Although the previously listed methods successfully identify and detect tomatoes, but do not account for situations like long-range detection or when the target takes up less pixels in the image, which could result in misses or incorrect identifications of small-target tomatoes. To enhance the accuracy of YOLO networks for small target detection, Wang et al. [27] developed SM-YOLOv5, which incorporates a small target detection layer to increase the accuracy of detecting small target tomatoes.

Based on research findings from researchers both domestically and globally, this paper presents a small target object method for tomatoes based on YOLOv8 network, taking into account the significance of low-dimensional information incorporated in larger features for small target detection [28].The following are the specifics of the work:

(1) In this study, PgConv is built to achieve the real-time detection performance of the model, while the IFN module is designed to achieve the effective extraction of tomato features. (2) By incorporating the GD mechanism, this method merges multi-level features globally and injects global information into upper layers, thereby facilitating efficient information exchange and improving the detection precision of small target tomatoes. (3) The adoption of Repulsion Loss has enhanced the ability of the network to accurately detect tomatoes in complex settings.

2 Materials and methods

2.1 Data acquisition and preparation

The tomato data collection was conducted at the tomato cultivation base of Shenyang Agricultural University (the coordinates of latitude 41.8° N). The dataset includes cracked tomatoes, frequently found among prematurely picked ripe tomatoes; these tomatoes hold both edible and economic value. Consequently, the algorithm could be applied in smart picking equipment to segregate cracked from healthy ripe tomatoes, potentially reducing farmers' cultivation costs. The photographs were captured with an iPhone 13 and saved in JPG format at a resolution of 1280 × 720. Images were captured at varying shooting distances, with a minimum threshold of 1.5 m for long distance shots and a maximum threshold of 0.5 m for normal distance shots. A total of 1896 tomato photographs were collected, with 1290 taken at larger distances, providing such data as small target tomato images, and the remaining 606 shot at normal distances, specifying such data as tomato images. The tomatoes in this study were classified into five maturity stages: ripe, half ripe, immature, young, and crack. Tomato photos captured at long and normal distances were allocated to the training and validation sets, respectively, at a 7:3 ratio. Figure 1 displays tomato images captured at normal (1-a) and longer distances (1-b, 1-c). The tomatoes in the dataset were labeled using LabelImg software [29], with label boxes added and saved as illustrated in 1-d.

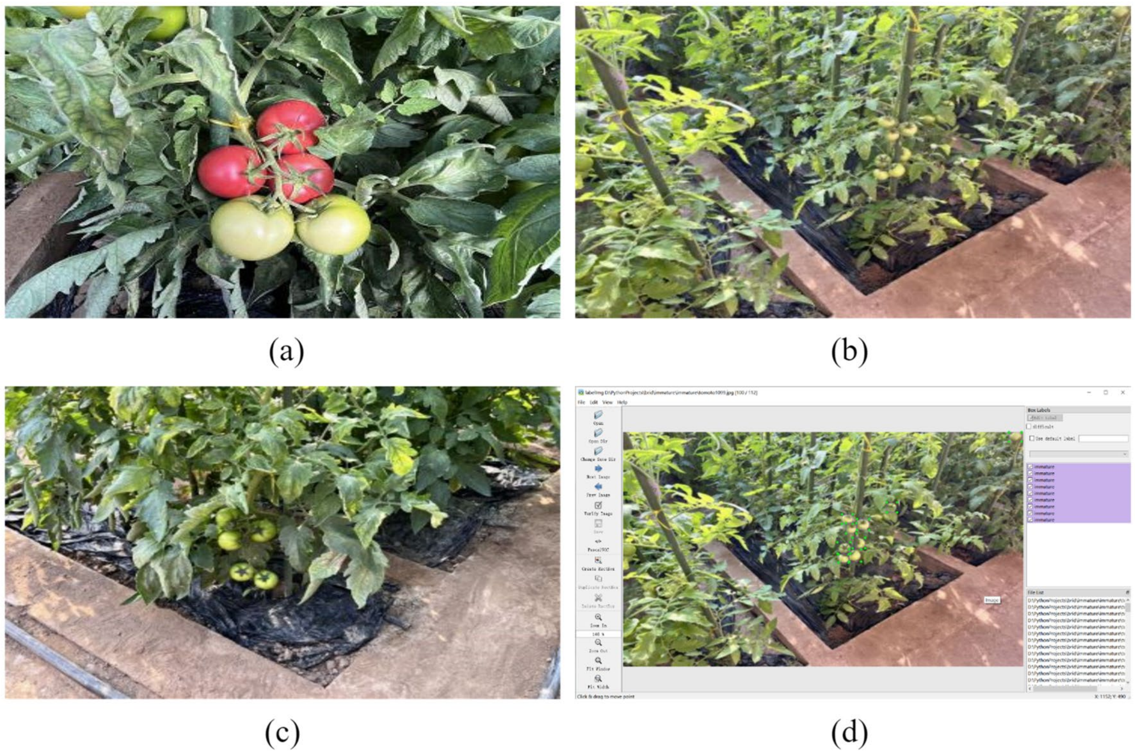


Fig. 1 Dataset and image annotation **a** normal distance capture **b**, **c** longer distance capture **d** image annotation

2.2 Data enhancement

The three channels of the RGB color space contain both luminance information and color information, hindering separate luminance processing in images [30]. In this study, the original images underwent two key processing steps: first, conversion from RGB to HSV color space; second, enhancement of luminance distribution using Contrast Limited Adaptive Histogram Equalization (CLAHE) to achieve more balanced and clear overall contrast. CLAHE is a more efficient method than global histogram equalization. Initially, the image is divided into small tiles, and then histogram equalization is performed separately for each tile. This localized approach enables CLAHE to boost the local contrast of the entire image without excessively enhancing specific areas, as illustrated in Fig. 2. After adjusting the brightness and contrast, the distinction between the foreground and background became more pronounced, aiding in tomato identification and detection.

To reduce the overfitting of the model during training and enhance its robustness and generalizability, this study expands the dataset by introducing zero-mean Gaussian noise, random rotation, and mirroring. The introduction of zero-mean Gaussian noise enhances the adaptability of the model to real-world noise by adding statistically balanced random disturbances, simulating environmental noise without altering the overall brightness of the image. Additionally, the dataset is augmented by randomly rotating images by 30°, 45°, and 90° and applying mirror transformations. This data augmentation process is depicted in Fig. 3. The structure of the augmented dataset is shown in Table 1.

2.3 YOLOv8

YOLOv8 offers five model scales (n, s, m, l, and x), with increasing network depth and width at each scale. The network structure of the YOLOv8 primarily comprises Backbone, Neck, and Head modules, as illustrated in Fig. 4.

2.3.1 Backbone

YOLOv8 employs a modified CSPDarknet53 [31] as its backbone network, yielding five distinct feature scales through five downsampling steps. The structure of the backbone network is depicted in Fig. 4-1. The backbone network replaces the CSP module with the C2f module, which is designed with additional hopping layer connections and split operations, enhancing gradient flow during backpropagation and thus improving model performance. The structure of the C2f module is illustrated in Fig. 4-6 (n denotes the number of bottlenecks). The CBS module processes the input information through convolution, followed by batch normalization, and SiLU activation is used to facilitate information flow, yielding the output results, as illustrated in Fig. 4-7. YOLOv8 utilizes the Spatial Pyramid Pooling Fast (SPPF) structure, which accelerates network operations by connecting three 5×5 pooling kernels in sequence, as depicted in Fig. 4-4.

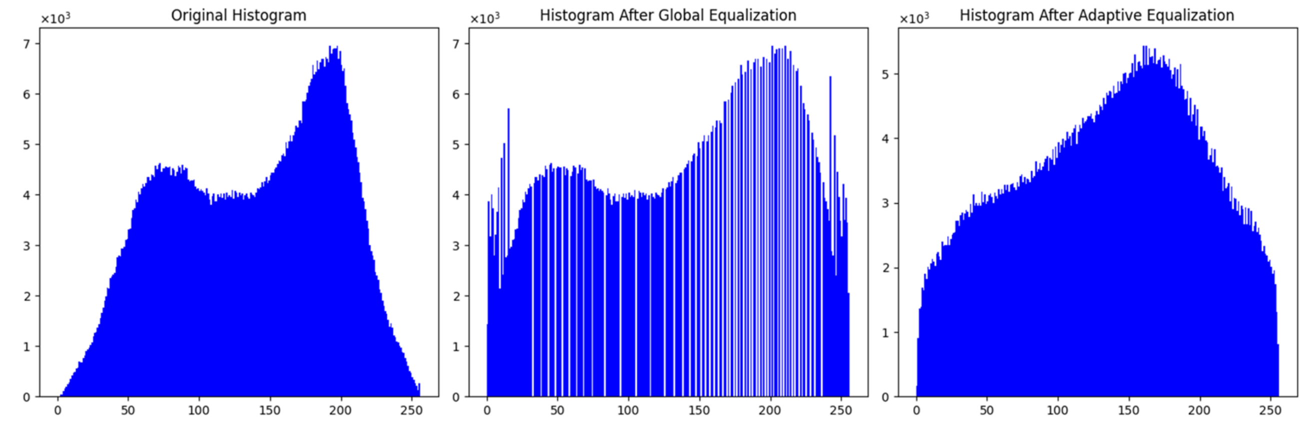


Fig. 2 Histogram equalization results

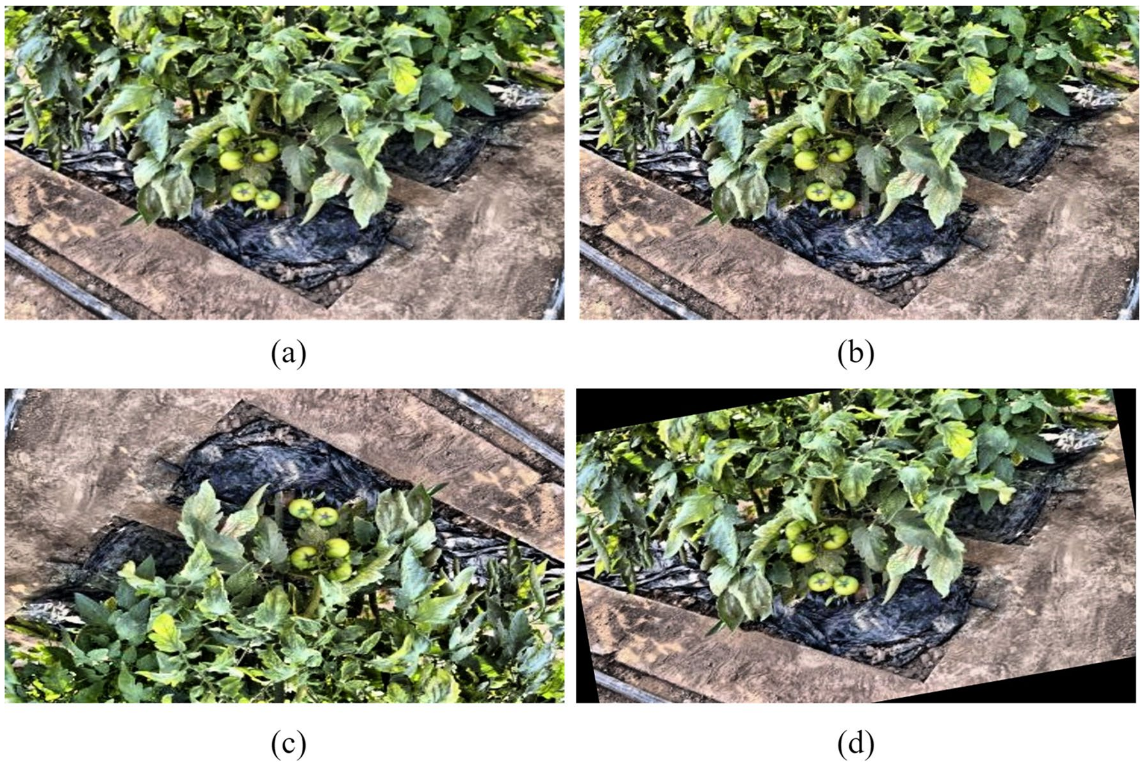


Fig. 3 Data augmentation. **a** CLAHE **b** noise addition **c** image mirroring **d** rotation by 30°

Table 1 Dataset structure

Type	Train	Validation	Train(enhancement)	Validation(enhancement)
TOM	424	182	1696	782
SOT	903	387	3612	1584
total	1327	569	5308	2276

TOM indicates images taken at normal distances; SOT indicates images taken at longer distances

2.3.2 Neck and head

Integrating PANet [32] with Feature Pyramid Network (FPN) [33] YOLOv8 incorporates the PAN-FPN structure in its neck component, as depicted in Fig. 4-2. In contrast to YOLOv5 [15] and YOLOv6 [34], YOLOv8 omits the 1×1 convolution prior to upsampling and directly merges features from various stages of the backbone network for feature integration. The PAN-FPN establishes both top-down and bottom-up network structures, achieving complementarity between shallow positional and deep semantic information through feature fusion, thus ensuring feature diversity and completeness. The head component employs the widely used decoupled head structure, as illustrated in Fig. 4-5. This decoupled head comprises two independent branches: one for object classification and the other for bounding box prediction regression. The following distinct loss functions are applied to these tasks: binary cross-entropy (BCE) loss for object classification and distribution focal loss (DFL) [35] along with the CIoU [36] for bounding box regression. The robustness of the model is increased by the use of the task aligned assigner of TOOD [37] for dynamic sample allocation in YOLOv8 as opposed to the static sample allocation technique of YOLOv5.

2.4 RSR-YOLO

In this work, the YOLOv8 network was carefully optimized to detect tomatoes on a large scale and accurately in the plantation. Initially, YOLOv8n, known for its smaller parameter scale and higher computational efficiency, was selected as the base framework, aligning with practical application scenarios and embedded system requirements. To facilitate real-time

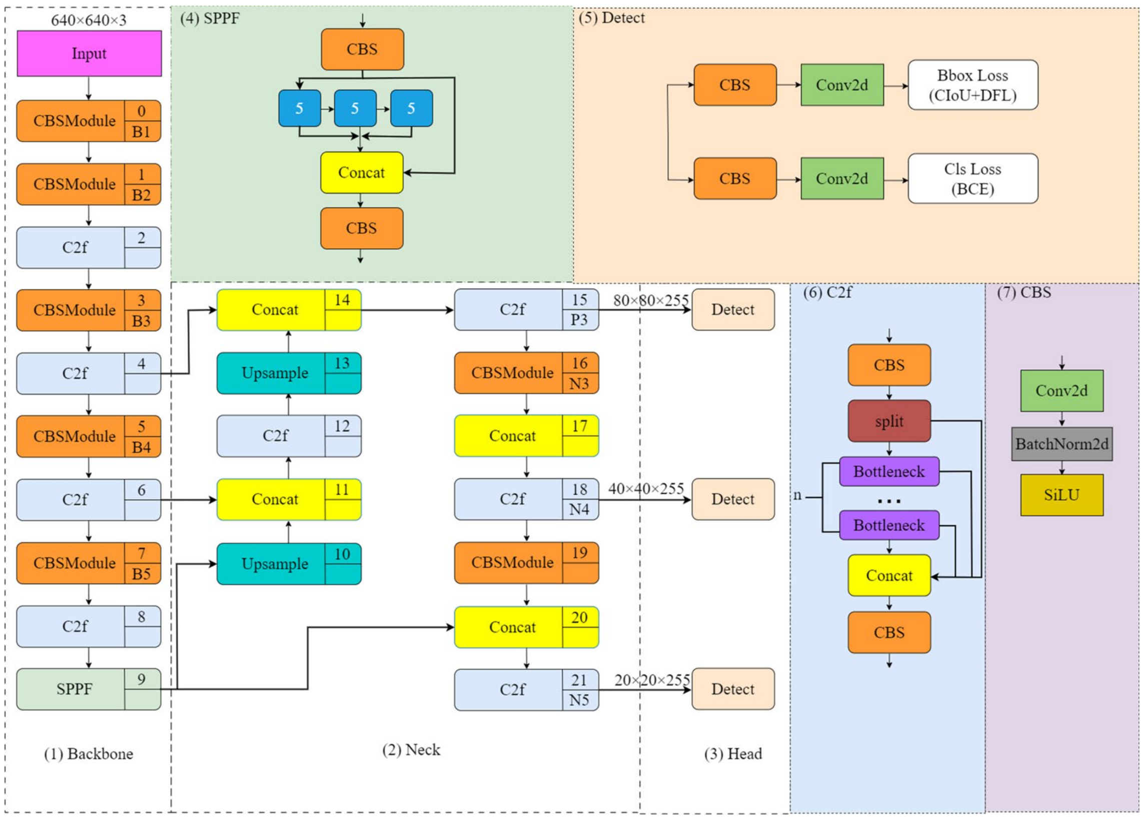


Fig. 4 YOLOv8 network structure

detection, a PgConv is introduced, reducing the computational load by replacing standard convolutions in PConv with grouped convolutions. Additionally, the innovative FasterNet module was developed to replace C2f in the backbone network, enhancing both the detection accuracy and computational efficiency. Second, tomatoes photographed from a distance appear with fewer pixels in the image, presenting challenges for feature extraction and fusion. Thus, by incorporating the GD mechanism, YOLOv8n achieves efficient information exchange through the global fusion of multilevel features and upward information injection. Additionally, recognizing the importance of low-dimensional features in detecting small targets, the redesigned C2f module replaces the reparametrized convolutional blocks (RepConv) module in the Low-stage information fusion module (Low-IFM) and Information injection module(inject) modules, enhancing the ability of the model to detect small targets. Finally, to address fruit overlap and leaf occlusion, this study introduces repulsion loss, boosting model accuracy in complex environments. The integration of these optimization strategies not only bolsters the model's ability to detect both regular and small tomatoes but also maintains computational efficiency and real-time performance, meeting the demands of practical applications. The structure of the RSR-YOLOv8 network is depicted in Fig. 5.

2.4.1 Lightweighting of the backbone network

To efficiently extract spatial features while minimizing redundant computations and memory usage, Chen et al. [38] introduced a streamlined approach, PConv, as illustrated in Fig. 6. PConv utilizes regular convolution for certain continuous features in the input channel, while other features are processed via constant mapping, preserving the integrity of the channel. In regular convolution, a convolution kernel spans all input channels to create an output channel, leading to complex interactions between the input and output channels that increase the computational load and parameter count. This paper introduces group convolutions in place of standard convolutions in PConv to further reduce the computational load of the backbone network. This modification leads to an enhanced version of PConv, termed Partial group Convolution (PgConv), illustrated in Fig. 6. The formula for channel allocation is delineated in Eqs. (1). Group convolution, a unique convolution operation, divides input channels into several groups, with each group independently performing

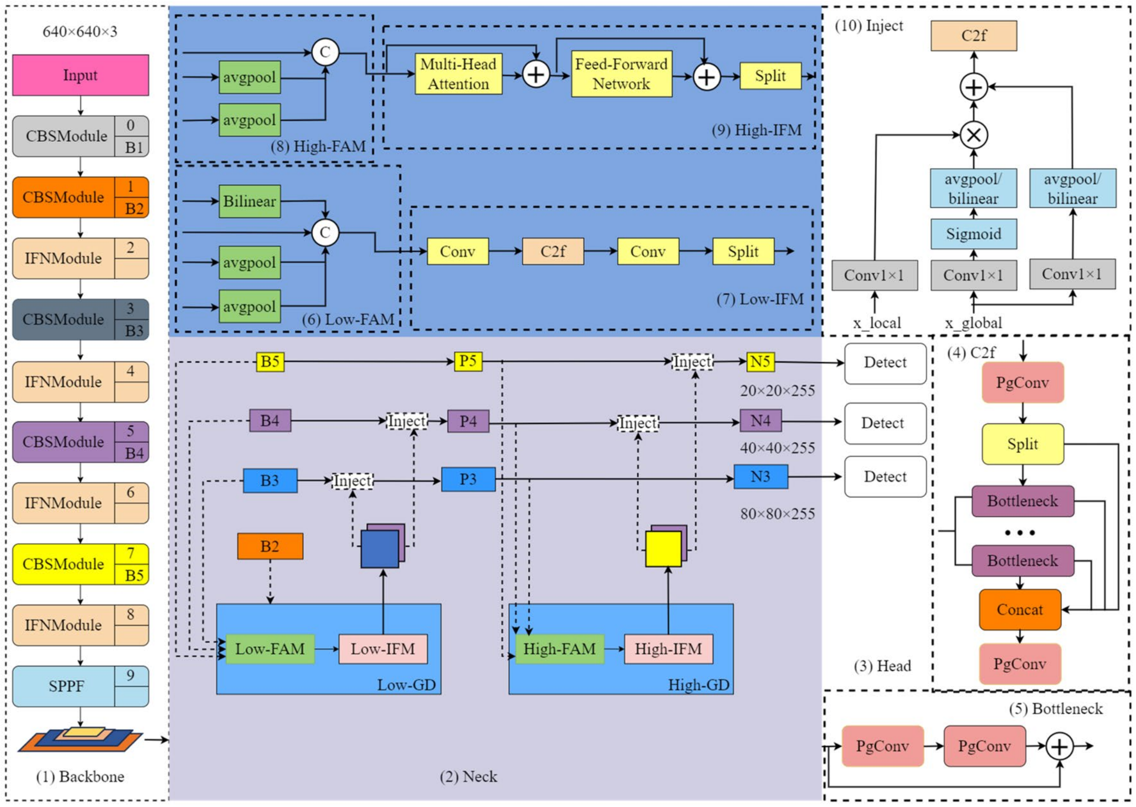
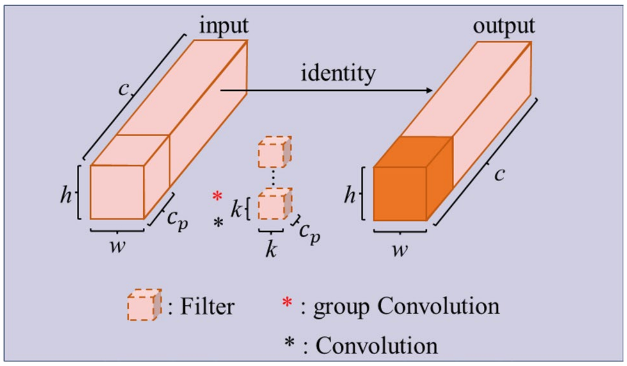


Fig. 5 The structure of RSR-YOLOv8 In this diagram 'x_local' denotes local feature injection information (produced by the current level) and 'x_global' indicates global feature injection information (generated by the IFM module)

Fig. 6 PgConv and PConv



convolution calculations. This approach reduces both computational demands and the number of parameters while preserving the effectiveness of the network.

$$C_{group} = \frac{C_{in}}{4} \quad (1)$$

where C_{group} is the total number of input channels and C_{in} is the number of channels after grouping.

The original FasterNet module comprises a PConv layer followed by two sequential 1×1 convolutional layer. It employs normalization and activation layers only after the second convolutional layer, effectively preventing the potential reduction in feature diversity due to excessive use of normalization and activation layers, as illustrated in Fig. 7a. However, the 1×1 convolution has a relatively small receptive field and lacks the ability to capture global features, unlike larger receptive fields that encompass a broader area of the feature and more effectively capture targets. Additionally, increasing the model depth with this structure could lead to model degradation and feature loss. Consequently, we redesigned

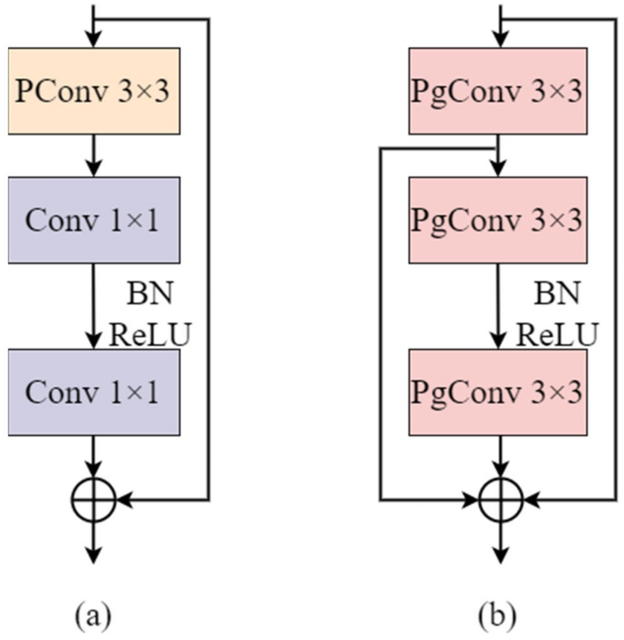


Fig. 7 Structure of the FasterNet module and innovative FasterNet module **a** FasterNet **b** improved FasterNet

the FasterNet module based on PgConv by replacing the two 1 × 1 regular convolutions with 3 × 3 PgConv and adding residual connections to the final two convolutional layers. This approach not only maintains computational speed and efficiency but also enlarges the receptive field, minimizes input feature loss, and enhances detection performance. The innovative FasterNet (IFN) model is depicted in Fig. 7b.

2.4.2 Improved GD mechanism for enhancing the accuracy of small target tomato detection

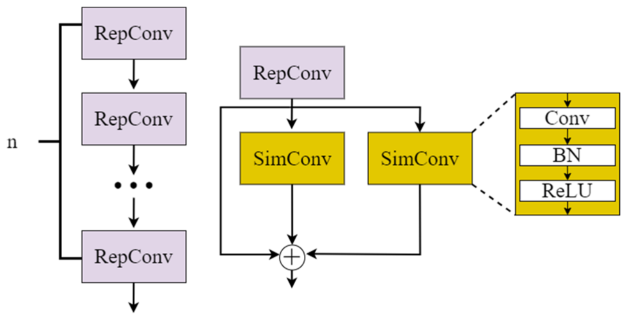
Traditionally, feature levels correspond to positional information for varying object sizes. Low-dimensional features encompass texture details and positional information of smaller objects, while high-dimensional features contain semantic information and positional attributes of larger objects. In this study, images of tomatoes captured from a longer distance possessed limited feature information. Consequently, the low-dimensional information within larger features is more conducive for the network to extract and fuse features of small target objects. To prevent significant information loss during computation and to fully utilize global information for feature fusion, the GD mechanism was incorporated [39]. The GD mechanism employs a unified module to gather and integrate information from each level, subsequently distributing it across various levels, thus enhancing the feature fusion capability of the neck without notably increasing latency. The GD mechanism is composed of two branches, a low-level distribution branch and a high-level distribution branch, which extract and fuse feature information via convolution-based and attention-based blocks, respectively. The low-stage gather-and-distribute branch includes the Low-stage Feature Alignment Module (Low-FAM), Low-IFM, and Information Injection Module (Inject); the high-stage branch comprises the High-stage Feature Alignment Module (High-FAM), High-stage Information Fusion Module (High-IFM), and Information Injection Module (Inject).

To enhance the extraction and fusion of smaller features and improve tomato fruit detection accuracy, this paper presents improvements to the low-stage gather-and-distribute branch, as depicted in Fig. 5-7. The specific improvements include the following steps: First, the C2f structure is redesigned, as illustrated in Fig. 5-4. This configuration enables the model to capture more detailed information on small targets. Second, the enhanced C2f module replaces the RepBlock module in Low-IFM, boosting its capacity to extract low-dimensional information from larger features. Finally, the improved C2f module replaces the RepBlock module in the Inject, enhancing its feature fusion capability and thereby improving the detection performance of YOLOv8n. The RepBlock module is presented in Fig. 8.

2.4.3 Repulsion loss

In this experiment, tomatoes were planted closely, with a spacing of 0.2 m between seedlings. In addition to fruit overlap and leaf shading within individual plants, mutual shading among plants presents a significant challenge for precise tomato

Fig. 8 RepBlock module



detection. To address this challenge, this study employed repulsion loss [40, 41] to enhance the positioning accuracy of the bounding box. This was achieved by integrating repulsion loss with the loss function of YOLOv8, as detailed in Eqs. (2)-(7). This enhancement is vital for addressing reduced detection accuracy issues caused by leaf occlusion, fruit overlap, and mutual occlusion among seedlings. Utilizing this method, we effectively enhance the ability of the model to identify tomatoes in complex environments, thereby improving its detection performance.

$$L = \lambda_{CIoU}L_{CIoU} + \lambda_{DFL}L_{DFL} + \lambda_{BCE}L_{BCE} \tag{2}$$

where λ_{CIoU}, λ_{DFL} and λ_{BCE} are the weighting coefficients assigned to the CIoU, DFL and BCE, respectively.

$$L_R = L + \alpha L_{RepGT} + \beta L_{RepBox} \tag{3}$$

where L is the original loss function for YOLOv8. L_{RepGT} is the loss value generated between.

the predicted bounding box and the adjacent ground-truth bounding boxes of the object; L_{RepBox} is the loss value generated between the predicted bounding box of the object and other adjacent predicted bounding boxes of other objects. The values of the weighting coefficients α and β are 0.5 and 2.0 respectively.

$$L_{RepGT} = \frac{\sum_{p \in p_1} Smooth_{\ln}(I_{IoG}(B^P, G^P_{RepP}))}{|P_1|} \tag{4}$$

$$I_{IoG}(B, G) = \frac{A_{area}(B \cap G)}{A_{area}(G)} \tag{5}$$

$$Smooth_{\ln} = \begin{cases} -\ln(1-x)x \le \delta \\ \frac{x-\delta}{1-\delta} - \ln(1-\delta)x > \delta \end{cases} \tag{6}$$

where $P1$ is the set of positive examples where the overlap between the predicted and actual bounding boxes reaches a predetermined threshold. B^P the predicted bounding box is derived through regression adjustment. G^P_{Rep} represents the ground truth bounding box that has the largest IoU ratio with the predicted bounding box, excluding the specified object. *Smoothln* is a smoothing function employed to modulate the loss function's sensitivity based on the intersection size; in this study, $\delta = 0.5$. $A_{area}()$ is an area calculation function.

$$L_{RepBox} = \frac{\sum_{i \ne j} Smooth_{\ln}(S_{IoU}(B_{P_i}, B_{P_j}))}{\sum_{i \ne j} 1[S_{IoU}(B_{P_i}, B_{P_j}) > 0] + \varepsilon} \tag{7}$$

where P_i and P_j are the predicted bounding boxes of different objects. B_{P_i} and B_{Pj} are the predicted bounding boxes of the objects regressed from the predicted bounding boxes P_i and P_j, respectively. $S_{IoU}(B_{P_i}, B_{Pj})$ is the IoU ratio between B_{P_i} and B_{Pj}. ε is a very small value set to prevent the divisor from being zero.

2.5 Performance evaluation of the models

The model was evaluated using five evaluation metrics: precision (P), recall (R), mAP, F1 score and frames per second (FPS) [42]. Equations (8)-(11) can be used to determine the values of precision, recall, F1 score and mAP; the higher the mAP is, the better the performance of the model. FPS indicates the number of image frames that can be processed per second. A higher value indicates that the model processes images faster.

$$P = \frac{TP}{(TP + FP)} \tag{8}$$

$$R = \frac{TP}{(TP + FN)} \tag{9}$$

$$F1 = \frac{2 \times P \times R}{(P + R)} \tag{10}$$

$$mAP = \frac{\sum_{i=1}^{C} AP_i}{C} \tag{11}$$

where true positive (*TP*) indicates that the actual sample is positive, and the prediction is also positive; false positive (*FP*) indicates that the actual sample is negative, but the prediction is positive; false negative (*FN*) indicates that the actual sample is positive, but the prediction is negative. The *AP* indicates the average precision; *mAP* is the average *AP* of the five categories; *C* indicates the number of categories.

3 Results and discussion

3.1 Experimental configuration

The examinations were performed on an Ubuntu 20.04 computer using an Intel® Xeon(R) Gold 6240 CPU @ 2.60 GHz and a TITAN RTX graphics card. The framework for deep learning was PyTorch 1.9.0 and CUDA 11.1, accelerated by cuDNN version 8.0.5. In this experiment, all the models were tested in the same environment and under the same hyperparameter settings: the learning rate, batch size, momentum, weight decay, and number of iterations were 0.01, 16, 0.937, 0.0005, and 500 epochs, respectively; the resolution was set to 640 pixels.

3.2 Tomato detection results based on the RSR-YOLO network

This experiment evaluates the detection and deployment performance of the RSR-YOLO model using a validation set. The RSR-YOLO model achieves the mAP$_{@0.5}$ of 90.7% and the FPS of 76, ensuring real-time detection capabilities along with high accuracy. Detection results of the RSR-YOLO model for tomatoes, including small target ones, are illustrated in Fig. 9. Images 9-b, 9-d, 9-n, and 9-p showcase the ability of model to detect tomato ripeness, while 9-f, 9-h, 9-j, and 9-l demonstrate detection of small target tomatoes. The detection abilities of the model for small target tomatoes obscured by leaves and overlapping fruits are shown in images 9-e, 9-g, 9-i, and 9-k. Additionally, images 9-a, 9-c, 9-m, and 9-o exhibit the model's capability to negate surface color effects and classify tomatoes by varying ripeness levels effectively. Table 2 presents the average precision for tomatoes at various stages of maturity. Table 2 reveals that the enhanced network adeptly detects tomatoes across various ripeness levels, exhibiting minimal variation in average precision across categories. The findings suggest that the RSR-YOLO model effectively mitigates the impact of surface color and features on detection outcomes, ensuring precise identification of tomatoes at varying ripeness levels.

3.3 Lightweight study of the RSR-YOLO backbone network

The lightweight design of network substantially lowers computational complexity, thereby decreasing reliance on high-performance hardware. The feature extraction capability of backbone is crucial for the final detection results. Thus,

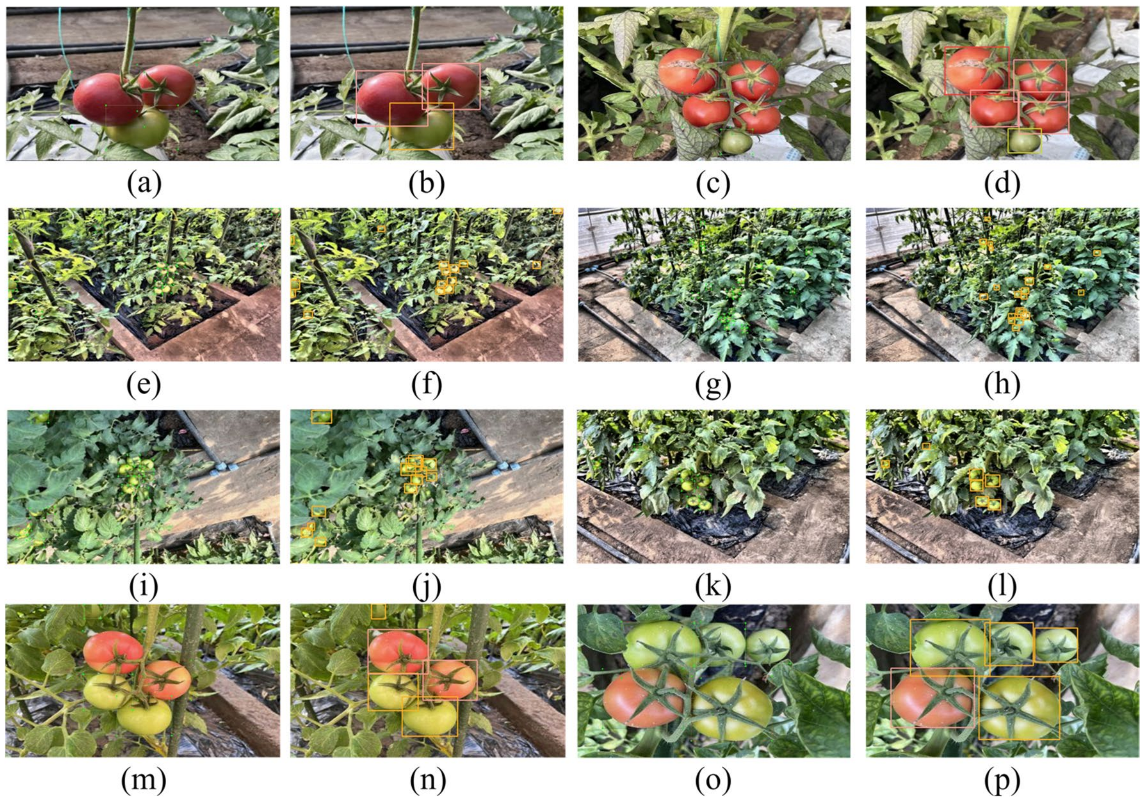


Fig. 9 Detection results

Table 2 Detection results for tomatoes at different maturities

Maturity	Crack	Ripe	Half ripe	Immature	Young fruit
AP (%)	90.9	92.0	90.1	89.9	90.8

AP denotes average precision, crack denotes cracked tomato; ripe denotes ripe tomato; half ripe denotes half ripe tomato; immature denotes immature tomato; young fruit denotes young tomato fruit

creating a lightweight yet powerful backbone network is essential to minimize computational complexity and boost detection speed. This study achieves a balance between detection and deployment performance by replacing the C2f module with the IFN module, significantly reducing computational complexity and enhancing accuracy. Comparative experiments with various numbers of IFN modules are detailed in Table 3. The RSR-YOLO model, utilizing IFN in its backbone network, achieves a 90.7% mAP$_{@0.5}$, marking a 0.3% increase and a 12.3% reduction in computational complexity compared to using C2f. Experimental results show that the lightweight PgConv significantly decreases the computational complexity, thereby enhancing deployment performance of network. The use of a 3×3 convolutional receptive field in the IFN structure and residual connections in the last two convolutional layers enhance the ability of the backbone network to capture features of tomatoes and reduces the loss of input features. Consequently, the use of IFN modules enhances the computational efficiency of model while maintaining detection accuracy.

3.4 Exploring the effect of the improved C2f module on the feature fusion capability of the Neck module

To enhance the feature extraction and fusion capabilities of the Neck component for small target tomatoes, this study utilizes the enhanced C2f to replace the RepBlock module in the Low-IFM and Inject. According to Table 4, the adoption of the C2f module in the RSR-YOLO network, as compared to the original GD mechanism, significantly enhances the extraction and fusion of small target tomato features. The improved GD mechanism results in a 1.1% increase in mAP$_{@0.5}$ score and a 21.8% reduction in computational complexity. Experimental results show that, in comparison to the RepBlock module formed by stacking RepConv, the C2f module, using skip connections and split operations, more

Table 3 Comparative experiments of IFN replacing C2f

+IFN	+IFN	+IFN	+IFN	mAP$_{@0.5}$ (%)	GFLOPs
				90.4	18.4
√				90.5	18.0
√	√			90.6	17.5
√	√	√		90.6	17.2
√	√	√	√	90.7	16.9

In this experiment, C2f was sequentially replaced with IFNs, specifically at layers 2, 4, 6, and 8 of the backbone

efficiently captures gradient flow information for small target tomatoes. This enhances the ability of GD mechanism to extract and integrate features of small target tomatoes. PgConv uses group convolution to convolve continuous features in the input channel, simultaneously applying identity mapping to the remaining features. This efficiently decreases the computational complexity of the enhanced C2f module, optimizing the deployment performance of network. In summary, the enhanced GD mechanism optimizes the detection performance of model without incurring additional delay.

3.5 Comparative experiment to verify the effectiveness of the RSR-YOLO network structure

This study conducted a series of comparative experiments using the YOLOv8n model to validate the effectiveness and feasibility of the RSR-YOLO model for detecting tomatoes and small target tomatoes. As indicated in Table 5, YOLOv8n + IFN achieves an mAP$_{@0.5}$ of 88.0%, indicating a 0.9% improvement over the original model. The results demonstrate that the lightweight IFN module enhances the feature extraction capacity of the backbone and computational efficiency while maintaining detection accuracy. Compared to YOLOv8n + IFN, the enhanced GD mechanism at the neck significantly improves the detection performance of the network, increasing the precision, recall, F1 score, and mAP$_{@0.5}$ by 1.4%, 4.1%, 2.8%, and 2.2%, respectively. The results indicate that the improved GD mechanism effectively boosts the detection performance of the model for small target tomatoes. RSR-YOLO achieves a precision, recall, F1 score, and mAP$_{@0.5}$ of 91.6%, 85.9%, 88.7%, and 90.7%, respectively, surpassing YOLOv8n + IFN + GD by 1.3%, 1.0%, 1.2%, and 0.5%, respectively. The increased precision highlights the effectiveness of repulsion loss in improving the detection of models of tomatoes with leaf shading and overlapping fruits. In conclusion, the network structure of the RSR-YOLO model is reasonable and effective. Figure 10 displays a graph comparing the mAP$_{@0.5}$ values with the sequential addition of each module. Figure 11 shows the results of the feature map visualization.

This paper uses the GradCAM method with a confidence setting of 0.6. After several training and testing sessions, heatmaps were generated from the output of the last C2f layer of YOLOv8n, YOLOv8n + IFN, YOLOv8n + IFN + GD, and RSR-YOLO. In Fig. 11a, f, k are the original images; (b), (g), and (l) are the output results of YOLOv8n; (c), (h), and (m) are the output results of YOLOv8n + IFN; (d), (i), and (n) are the output results of YOLOv8n + IFN + GD; (e), (j), and (o) are the output results of RSR-YOLO. The visualization of feature maps has progressively improved with model enhancements, and the performance of RSR-YOLO surpasses that of YOLOv8n, particularly in the detection of small target tomatoes.

Table 4 Performance comparison

+IFN	+C2f	+RepBlock	mAP$_{@0.5}$ (%)	mAP$_{@0.5–0.95}$ (%)	FPS	GFLOPs
√		√	89.6	78.3	80	21.6
√	√		90.7	80.2	76	16.9

Table 5 Comparison experiment

+IFN	+GD	+RL	P (%)	R (%)	F1(%)	mAP$_{@0.5}$ (%)	mAP$_{@0.5–0.95}$ (%)
			87.4	81.9	84.5	87.1	76.4
√			88.9	80.8	84.7	88.0	75.9
√	√		90.3	84.9	87.5	90.2	80.0
√	√	√	91.6	85.9	88.7	90.7	80.2

"YOLOv8n + IFN" indicates replacing C2f with IFN in the backbone, and "YOLOv8n + IFN + GD" refers to both replacing C2f with IFN in the backbone and incorporating the improved GD mechanism in the neck

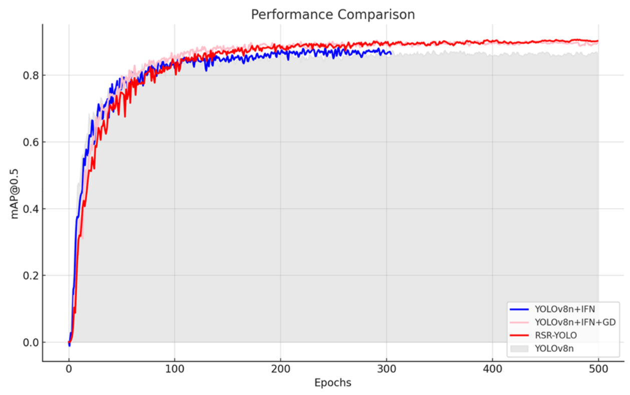


Fig. 10 Performance comparison

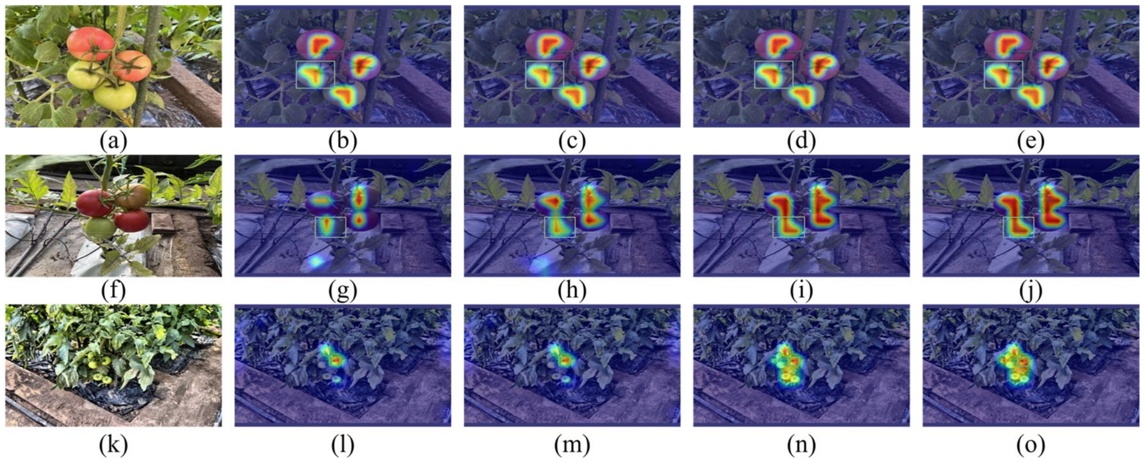


Fig. 11 Feature map visualization

3.6 Comparison of the RSR-YOLO results against those of other models

Target detection requires striking a balance between detection and deployment performance in order to assess the effectiveness of the algorithm [43]. The precision, recall, F1 score, and mAP were utilized in this work to evaluate the detection performance of RSR-YOLO, while the FPS and computing complexity were used to analyze the deployment performance. Comparative data for ten algorithms are presented in Table 6. The RSR-YOLO has a mAP$_{@0.5}$ of 90.7%, which is an improvement of 3.6%, 1.1%, 0.1%, 3.9%, 2.3%, 2.2%, 0.6%, and 0.3% compared to YOLOv8n, YOLOv8s, YOLOv8m, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv6s, and YOLOv6m, respectively. The FPS of the improved model increases by 50% and its computational complexity decreases by 83.8% when compared to YOLOv7, despite the fact that its mAP$_{@0.5}$ is 0.5% lower. RSR-YOLO significantly enhances deployment performance and reduces hardware dependency with minimal sacrifice in accuracy compared to YOLOv7. In conclusion, this paper shows that RSR-YOLO adeptly balances detection and deployment performance, outperforming other methods in real-time detection and accurate detection of both regular and small-target tomatoes. Figure 12 displays the mAP$_{@0.5}$ comparison graphs for the ten algorithms, while Fig. 13 shows their FPS and inference times.

The figure shows the mAP$_{@0.5}$ scores of 10 algorithms at epochs 0, 50, 100, and 499.The best mAP$_{@0.5}$ scores for the 10 algorithms are listed as follows: YOLOv5l: at epoch 356, mAP$_{@0.5}$ is 0.885; YOLOv5s: at epoch 377, mAP$_{@0.5}$ is 0.868; YOLOv5m: at epoch 385, mAP$_{@0.5}$ is 0.884; YOLOv8s: at epoch 412, mAP$_{@0.5}$ is 0.896; YOLOv8m: at epoch 421, mAP$_{@0.5}$ is 0.906; YOLOv6m: at epoch 456, mAP$_{@0.5}$ is 0.904; YOLOv6s: at epoch 437, mAP$_{@0.5}$ is 0.901; YOLOv7: at epoch 407, mAP$_{@0.5}$ is 0.912; YOLOv8n: at epoch 355, mAP$_{@0.5}$ is 0.871; RSR-YOLO: at epoch 480, mAP$_{@0.5}$ is 0.907. To better analyze the fluctuations in the mAP$_{@0.5}$ values of RSR-YOLO, an area graph is plotted in Fig. 12. The graph shows minimal fluctuation in the convergence process of the model, indicating a trend toward stabilization. In conclusion, RSR-YOLO exhibits robust detection capabilities.

Table 6 Performance comparison

Method	P (%)	R (%)	F1(%)	mAP$_{@0.5}$ (%)	mAP$_{@0.5-0.95}$ (%)	FPS	GFLOPs
YOLOv8n	87.4	81.9	84.5	87.1	76.4	126	8.1
YOLOv8s	89.5	85.9	87.6	89.6	79.0	90	28.6
YOLOv8m	90.8	84.3	87.4	90.6	79.4	64	78.7
YOLOv7	91.5	87.7	89.5	91.2	81.1	26	104.5
YOLOv6s	91.3	85.5	88.3	90.1	79.5	70	45.3
YOLOv6m	91.0	86.1	88.5	90.4	79.8	58	85.8
YOLOv5s	89.7	81.3	85.2	86.8	72.8	104	15.8
YOLOv5m	88.1	84.2	86.1	88.4	76.1	75	47.9
YOLOv5l	89.1	85.8	87.4	88.5	79.0	48	107.7
RSR-YOLO	91.6	85.9	88.7	90.7	80.2	76	16.9

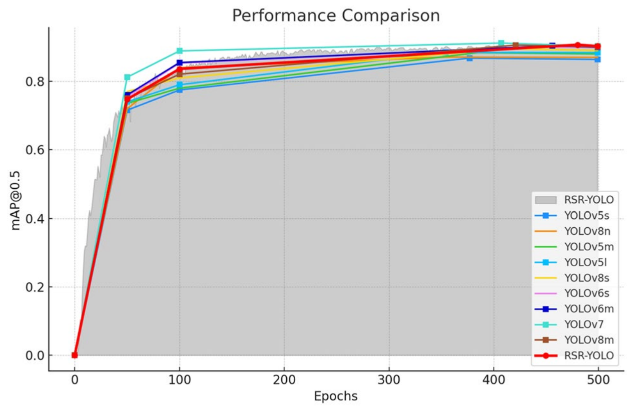


Fig. 12 Comparison of detection results

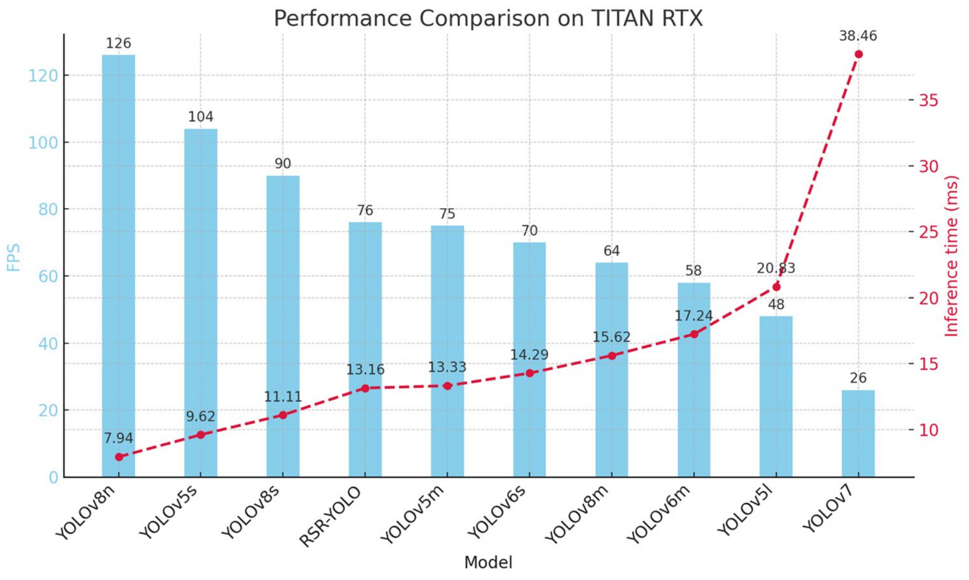


Fig. 13 Comparative chart of deployment performance

In the chart, FPS and inference time exhibit an inverse relationship: higher FPS values denote shorter inference times, reflecting accelerated model operation on hardware. From the figure, it can be seen that although RSR-YOLO is slightly inferior to YOLOv8n, YOLOv5s, and YOLOv8s in FPS and inference time (with a significant difference in mAP compared to these algorithms), its deployment performance is superior to YOLOv8m, YOLO6m, and YOLOv7 (whose mAP scores are relatively close to that of RSR-YOLO). In conclusion, RSR-YOLO excels in deployment performance while maintaining high accuracy.

3.7 Applications of RSR-YOLO

To better evaluate the performance of the model, this paper develops a GUI visualization interface and conducted field tests on a tomato plantation. The GUI is shown in Fig. 14. Figure 14a displays the general layout of the interface. Figure 14b shows the functional area setup of the interface, with the local button on the left for detecting local files or videos and the camera button for connecting a visual sensor. On the right, the Model button selects weight files for different models, the Conf button sets the confidence threshold, and the IoU button manually adjusts the intersection ratio threshold. Figure 14c illustrates the process of selecting a local file for detection. Figure 14d presents the tomato detection results, including statistics on tomato categories, count, and FPS displayed at the top of the area.

3.8 Limitations of the RSR-YOLO

This research proposes an RSR-YOLO model to achieve long-range small-target tomato detection. The RSR-YOLO model successfully balances detection performance with deployment performance, demonstrating excellent real-time detection performance while accurately detecting tomato fruit. RSR-YOLO does, however, have two drawbacks. One is that the model experiences issues like leaking and incorrect detection throughout the detection process because of limited features and anchor frames of the small target tomato. Secondly, the detection capability of the model in real-world applications is easily impacted by changes in illumination, which makes it difficult to accurately locate and identify tomato features. Based on the above two limitations, we will further design and optimize the network structure of the RSR-YOLO model to improve the detection ability of small target tomatoes and enhance the generalization ability of the network in the following work as follows: (1) Design a primary and secondary feature extraction network, the secondary feature extraction network focuses on extracting the texture and detail features of the small target tomatoes contained in the low-dimensional feature layer, and then fuses the features with the primary feature extraction network. network for feature fusion. (2) In the subsequent work, the diversity of the dataset will be further expanded to improve the generalization ability of the model in poor visual appearance environments. In addition, the idea of codebook priors to reconstruct images is borrowed to design an adaptive feature recovery network branch. When the IOU is lower than a certain threshold, the model will retrieve the pre-trained high-quality

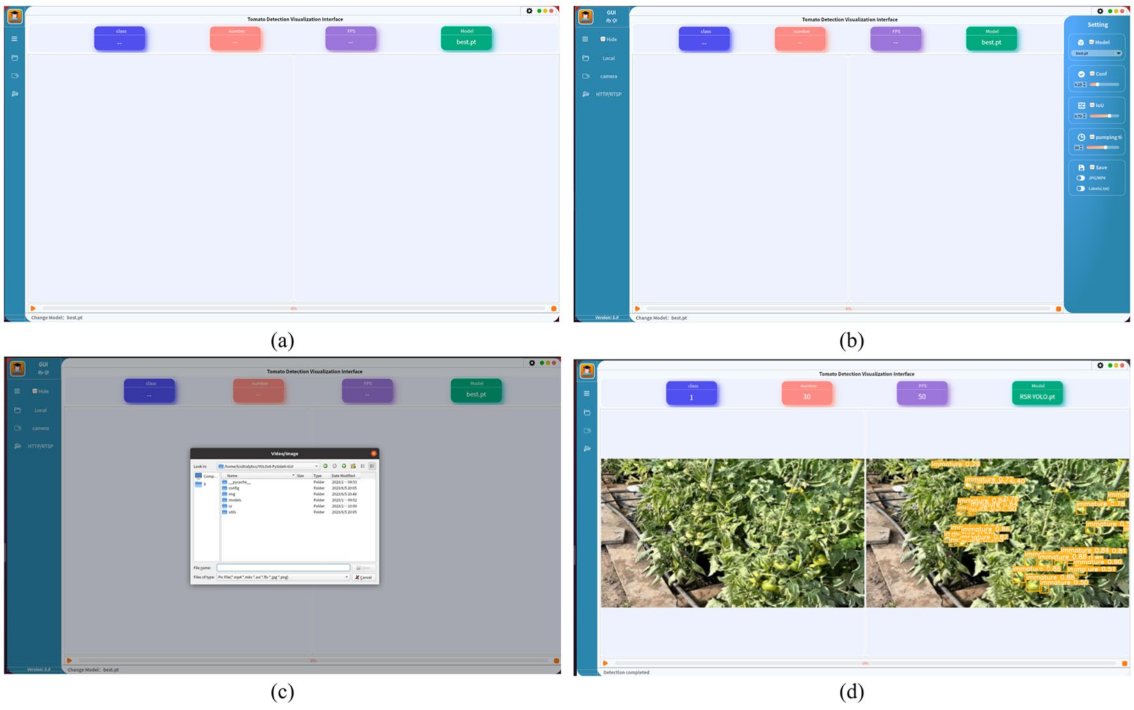


Fig. 14 GUI visualization interface

feature information in the codebook priors based on the contextual information of the environmentally affected features, thus further overcoming the influence of environmental factors on the generalization ability of the model.

4 Conclusions

This research proposes an RSR YOLO model to achieve precise tomato detection. First, YOLOv8n, which has a smaller parameter size and higher computing efficiency, is chosen as the fundamental framework for this experiment in consideration of real-world application situations and embedded system needs. This study designs PgConv, which replaces regular convolution in PConv with group convolution, thereby reducing the computational cost of the model and enabling real-time detection of the model. In addition, the IFN module is intended to take the place of the C2f module of the backbone network, guaranteeing detection accuracy while enhancing processing efficiency. Second, this research reconstructs the YOLOv8n neck module in conjunction with the Gather and Distribute (GD) mechanism, taking into account the critical role of low-dimensional features for small target tomatoes. To efficiently extract and merge the tomato features at different levels, this work employs the new C2f module, which ensures the lightness of the model while obtaining deeper information about small targets. Finally, Repulsion Loss is added to increase the accuracy of the model for detecting tomato leaf shade and fruit overlap. The RSR-YOLO has a $mAP_{@0.5}$ of 90.7%, which is an improvement of 3.6%, 1.1%, 0.1%, 3.9%, 2.3%, 2.2%, 0.6%, and 0.3% compared to YOLOv8n, YOLOv8s, YOLOv8m, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv6s, and YOLOv6m, respectively. Despite a slight decrease of 0.5% in $mAP_{@0.5}$ compared to YOLOv7, the Improved model demonstrates a 50 unit increase in FPS and an 83.8% reduction in computational complexity. RSR-YOLO significantly improves network deployment performance and minimizes hardware dependency with only a slight reduction in accuracy compared to YOLOv7. To sum up, the RSR-YOLO network exhibits outstanding real-time detection performance and reliably detects tomato fruit by skillfully balancing detection performance and deployment performance. Future work will concentrate on enhancing the detection of small-target tomatoes with low-dimensional information, designing a network better suited for small-target detection, and further optimizing deployment performance while increasing accuracy.

Acknowledgements This work was supported in part by the Youth Program of Liaoning Education Department under Grant LSNQN202025.

Author contributions Conceptualization, X. Y.; methodology, K. Q.; software, K. Q.; validation, K. Q. and F. Y.; formal analysis, K. Q.; investigation, X. N.; resources, K. Q.; data curation, Y. L.; writing—original draft preparation, X. Y. and K. Q.; writing—review and editing, X. Y.; visualization, K. Q.; supervision, C. L.; project administration, C. L.; funding acquisition, C. L.

Data availability Data will be made available upon request.

Declarations

Ethics approval and consent to participate This research is not a physiological study of tomatoes, but a visual method of detection. The study did not involve harvesting or damaging the plants, nor did it adversely affect the growth of the tomatoes.

Competing interests We have no affiliations with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.

References

1. Kelman E, Linker R. Vision-based localization of mature apples in tree images using convexity. Biosyst Eng. 2014;118:174–85. https://doi.org/10.1016/j.biosystemseng.2013.11.007.
2. Yin HP, Chai Y, Yang SX. Ripe tomato recognition and localization for a tomato harvesting robotic system. In: International conference of soft computing and patter recognition. 2009; 557–562. https://doi.org/10.1109/SoCPaR.2009.111
3. Hannan MW, Burks TF, Bulanon DM. A machine vision algorithm combining adaptive segmentation and shape analysis for orange fruit detection. Agric Eng Int CIGR J. 2009;6:1–17.

4. Patel HN, Jain RK, Joshi MV. Fruit detection using improved multiple features based algorithm. IJCA. 2011;13(2):1–5. https://doi.org/10.5120/1756-2395.
5. Gongal A, Amatya S, Karkee M, Zhang Q, Lewis K. Sensors and systems for fruit detection and localization: a review. Comput Electron Agric. 2015;116:8–19. https://doi.org/10.1016/j.compag.2015.05.021.
6. Wang P, Niu T, He D. Tomato young fruits detection method under near color background based on improved faster R-CNN with attention mechanism. Agriculture. 2021;11(11):1059. https://doi.org/10.3390/agriculture11111059.
7. Wang Z, Ling Y, Wang X, Meng D, Nie L, An G, Wang X. An improved Faster R-CNN model for multi-object tomato maturity detection in complex scenarios. Ecol Inform. 2022;72: 101886. https://doi.org/10.1016/j.ecoinf.2022.101886.
8. Afonso M, Fonteijn H, Fiorentin FS, Lensink D, Mooij M, Faber N, Wehrens R. Tomato fruit detection and counting in greenhouses using deep learning. Plant Sci. 2020;11: 571299. https://doi.org/10.3389/fpls.2020.571299.
9. Hu C, Liu X, Pan Z, Li P. Automatic detection of single ripe tomato on plant combining faster R-CNN and intuitionistic fuzzy set. IEEE Access. 2019;7:154683–96. https://doi.org/10.1109/ACCESS.2019.2949343.
10. Yue X, Qi K, Na X, Zhang Y, Liu Y, Liu C. Improved YOLOv8-Seg network for instance segmentation of healthy and diseased tomato plants in the growth stage. Agriculture. 2023;13(8):1643. https://doi.org/10.3390/agriculture13081643.
11. Widiyanto S, Wardani DT, Pranata SW. Image-based tomato maturity classification and detection using Faster R-CNN method. IEEE ISMSIT. 2021;130–134. https://doi.org/10.1109/ISMSIT52890.2021.9604534
12. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition 2016;779–788.
13. Redmon J, Farhadi A. Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017;7263–7271. https://doi.org/10.1109/CVPR.2017.690
14. Bochkovskiy A, Wang CY, Liao HYM. Yolov4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934. 2020. https://doi.org/10.48550/arXiv.2004.10934.
15. Glenn. ultralytics/ultralytics: v6.1. 2022. https://github.com/ultralytics/yolov5/releases/tag/v6.1.
16. Ge Z, Liu S, Wang F, Li Z, Sun J. Yolox: exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430. 2021. https://doi.org/10.48550/arXiv.2107.08430
17. Li C, Li L, Geng Y, Jiang H, Cheng M, Zhang B, Chu X. Yolov6 v3. 0: a full-scale reloading. arXiv preprint arXiv:2301.05586. 2023. https://doi.org/10.48550/arXiv.2301.05586
18. Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv preprint arXiv:2207.02696. 2022. https://doi.org/10.48550/arXiv.2207.02696
19. Glenn. ultralytics/ultralytics: v8.0.136 .2023.https://github.com/ultralytics/ultralytics
20. Liu G, Nouaze JC, Touko Mbouembe PL, Kim JH. YOLO-tomato: a robust algorithm for tomato detection based on YOLOv3. Sensors. 2020;20(7):2145. https://doi.org/10.3390/s20072145.
21. Appe SN, Arulselvi G, Balaji GN. CAM-YOLO: tomato detection and classification based on improved YOLOv5 using combining attention mechanism. PeerJ Comput Sci. 2023;9: e1463. https://doi.org/10.7717/peerj-cs.1463.
22. Woo S, Park J, Lee JY, Kweon IS. Cbam: convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). 2018;3–19. https://doi.org/10.48550/arXiv.1807.06521
23. Fei S, Zexu Z, Yanping Z, Tianhua L, Linlu Z. Detection of mature green tomato based on lightweight YOLO-v3. J Chin Agric Mech. 2022;43(3):132.
24. Yang G, Wang J, Nie Z, Yang H, Yu S. A Lightweight YOLOv8 tomato detection algorithm combining feature enhancement and attention. Agronomy. 2023;13(7):1824. https://doi.org/10.3390/agronomy13071824.
25. Zeng T, Li S, Song Q, Zhong F. Lightweight tomato real-time detection method based on improved YOLO and mobile deployment. Comput Electron Agric. 2023;205: 107625. https://doi.org/10.1016/j.compag.2023.107625.
26. Mbouembe PLT, Liu G, Sikati J, Kim SC, Kim JH. An efficient tomato-detection method based on improved YOLOv4-tiny model in complex environment. Front Plant Sci. 2023;14:1150958. https://doi.org/10.3389/fpls.2023.1150958.
27. Wang X, Wu Z, Jia M, Xu T, Pan C, Qi X, Zhao M. Lightweight SM-YOLOv5 tomato fruit detection algorithm for plant factory. Sensors. 2023;23(6):3336. https://doi.org/10.3390/s23063336.
28. Rekavandi AM, Xu L, Boussaid F, Seghouane AK, Hoefs S, Bennamoun M. A guide to image and video based small object detection using deep learning: case study of maritime surveillance. arXiv preprint arXiv:2207.12926. 2022. https://doi.org/10.48550/arXiv.2207.12926.
29. Tzutalin D. LabelImg. Git code. 2015. https://github.com/tzutalin/labelImg
30. Qi J, Liu X, Liu K, Xu F, Guo H, Tian X, Li Y. An improved YOLOv5 model based on visual attention mechanism: application to recognition of tomato virus disease. Comput Electron Agric. 2022;194: 106780. https://doi.org/10.1016/j.compag.2022.106780.
31. Redmon J, Farhadi A. Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767, 2018. https://doi.org/10.48550/arXiv.1804.02767
32. Liu S, Qi L, Qin H, Shi J, Jia J. Path aggregation network for instance segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018; 8759–8768. https://doi.org/10.1109/CVPR.2018.00913
33. Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S. Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017;2117–2125. https://doi.org/10.1109/CVPR.2017.106.
34. Li C, Li L, Jiang H, Weng K, Geng Y, Li L, Wei X. YOLOv6: a single-stage object detection framework for industrial applications. arXiv preprint arXiv:2209.02976. 2022. https://doi.org/10.48550/arXiv.2209.02976
35. Li X, Wang W, Wu L, Chen S, Hu X, Li J, Yang J. Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection. Adv Neural Inf Process Syst 2020 33:21002–21012. https://doi.org/10.48550/arXiv.2006.04388
36. Zheng Z, Wang P, Liu W, Li J, Ye R, Ren D. Distance-IoU loss: faster and better learning for bounding box regression. Proc AAAI Conf Artif Intell. 2020;34(07):12993–3000.
37. Feng C, Zhong Y, Gao Y, Scott MR, Huang W. Tood: task-aligned one-stage object detection. In: 2021 IEEE/CVF international conference on computer vision (ICCV). IEEE Computer Society. 2021;3490–3499. https://doi.org/10.1109/ICCV48922.2021.00349

38. Chen J, Kao SH, He H, Zhuo W, Wen S, Lee CH, Chan SHG. 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;12021–12031. https://doi.org/10.1109/CVPR52729.2023.01157
39. Wang C, He W, Nie Y, Guo J, Liu C, Wang Y, Han K. Gold-YOLO: efficient object detector via gather-and-distribute mechanism. arXiv preprint arXiv:2309.11331. 2023. https://doi.org/10.48550/arXiv.2309.11331
40. Chen H, Guan J. Teacher–student behavior recognition in classroom teaching based on improved YOLO-v4 and Internet of Things technology. Electronics. 2022;11(23):3998. https://doi.org/10.3390/electronics11233998.
41. Wang X, Xiao T, Jiang Y, Shao S, Sun J, Shen C. Repulsion loss: detecting pedestrians in a crowd. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018;7774–7783. https://doi.org/10.1109/CVPR.2018.00811
42. Yang W, Wu J, Zhang J, Gao K, Du R, Wu Z, Li D. Deformable convolution and coordinate attention for fast cattle detection. Comput Electron Agric. 2023;211: 108006. https://doi.org/10.1016/j.compag.2023.108006.
43. Mei L, Chen Z. An improved YOLOv5-based lightweight submarine target detection algorithm. Sensors. 2023;23(24):9699. https://doi.org/10.3390/s23249699.