



# PS-YOLO: a small object detector based on efficient convolution and multi-scale feature fusion

Shifeng Peng<sup>1</sup> · Xin Fan<sup>1</sup> · Shengwei Tian<sup>1</sup> · Long Yu<sup>2</sup>

Received: 25 March 2024 / Accepted: 1 August 2024

© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2024

## Abstract

Compared to generalized object detection, research on small object detection has been slow, mainly due to the need to learn appropriate features from limited information about small objects. This is coupled with difficulties such as information loss during the forward propagation of neural networks. In order to solve this problem, this paper proposes an object detector named PS-YOLO with a model: (1) Reconstructs the C2f module to reduce the weakening or loss of small object features during the deep superposition of the backbone network. (2) Optimizes the neck feature fusion using the PD module, which fuses features at different levels and sizes to improve the model's feature fusion capability at multiple scales. (3) Design the multi-channel aggregate receptive field module (MCARF) for downsampling to extend the image receptive field and recognize more local information. The experimental results of this method on three public datasets show that the algorithm achieves satisfactory accuracy, prediction, and recall.

**Keywords** Small object detection · Multi-scale feature fusion · Downsampling

## 1 Introduction

Cameras have been among the most widely used devices in various industries and households, such as surveillance, transportation, medicine, drones, and autopilot. With the rapid development of science and technology, more and more picture materials for UAV aerial photography are available, which stimulates the development of UAV aerial photography image processing. However, the large scale, dense, overlapping small targets and uneven distribution of variable targets in UAV aerial images bring more difficulties and challenges to UAV image detection. Nowadays, the combination of object detection and deep learning techniques [1, 2] for UAV aerial images has become a

hot research direction. Since object detection entered the deep learning era, the development of CNNs-based algorithms has mainly been in two directions: two-stage and one-stage algorithms. Common two-stage algorithms such as R-CNN (region-based convolutional neural network) [3], Fast R-CNN series [4], Faster R-CNN series [5], etc. The R-CNN network predicts the feature maps in the last layer of the neural network. Due to the lack of fine-grained feature information in a single feature map, the standard R-CNN that outputs a single feature map is less effective in small object detection as the number of network layers deepens. To find a representation of multi-scale feature maps, Wang et al. [6] utilized the residual connection autoencoder multi-scale structure to address different scale targets. Based on the existing Faster R-CNN, Jakaria Rabbi et al. [7] used a dense residual block (RRDB) for image enhancement of multi-scale feature maps to achieve good small object detection. Due to the limitations of single-scale features, multi-scale feature fusion is getting more and more attention from academia and industry, and it has been proven effective in small object detection. These multi-scale feature fusion algorithms are also widely used in one-stage algorithms. Common one stage algorithms include SSD [8], YOLO [9], YOLOv3 [10], etc. SSD network uses a multi-scale feature mapping map to process

---

Communicated by Junyu Gao.

---

✉ Xin Fan  
vanxin011@163.com  
Shifeng Peng  
pengshifengpsf@163.com

<sup>1</sup> College of Software, Xinjiang University, Urumqi 830000, China

<sup>2</sup> College of Network Center, Xinjiang University, Urumqi 830000, China

feature information at different scales and Single Shot Detection to encapsulate localization and detection in the network's forward operation to increase the operation's speed. YOLOv3 combines FPN [11] proposed a real-time object detector, and the network better retains the information of various scale objects by the form of a feature pyramid. To summarize, the two-stage algorithm first needs to generate a candidate bounding box with a neural network to reduce the number of regions to be detected. Then, it performs the detection, which has high accuracy but is unsuitable for mobile deployment due to the slow detection speed. One-stage algorithms aim to simplify the detection task by extracting features directly in the network to predict the classification and location of objects. Although early one-stage algorithms did not have the same detection accuracy as two-stage algorithms, they are more suitable for mobile deployment due to their faster detection speed and ease of training. However, with the rapid development of one-stage algorithms, more breakthroughs in detection accuracy have occurred. Especially based on the improvement of the YOLO algorithm, the one-stage algorithm is comparable to the two-stage algorithm in terms of precision, recall, and other indicators.

The most popular YOLO series algorithm is the recently released YOLOv8 [12]. YOLOv8 consists of three main parts: the backbone, the neck, and the head parts. The backbone part is mainly used for feature extraction, the neck is used to enhance the feature information, and the head is used for detection. The algorithm comprehensively improves the detection accuracy of all sizes in general-purpose target detection. Furthermore, YOLOv8's feature fusion at the neck is improved by FPN and PANet [13], which enhances the information at different scales by continuous up-sampling and down-sampling. However, the model has obvious drawbacks for small object detection: (1) the shallow information it extracts at the backbone is not significant enough. The feature information is easily lost after the backbone part is computed. It is not enough to participate in and support the calculation of the neck part, especially for small objects in complex scenes, repetitive overlapping, and other problems, which makes it particularly difficult to extract features. (2) Although the neck part can enhance the information of different scales, it lacks the multi-level fusion of information in cross-scale feature fusion and does not sufficiently fuse cross-level information. The lack of mutual learning between shallow semantic information and high-level semantics is one of the reasons why YOLOv8 is not effective in small object detection. Therefore, we are inspired by GENERALIZED-FPN to introduce  $\log_2 n$ -link between backbone and neck. It aims to enhance the learning between shallow and advanced semantic information. This method can effectively solve the problem of YOLOv8 lacking cross-scale information fusion.

In order to solve the above problems, we propose a detector named PS-YOLO, a model that can better extract small object feature information, fully integrate cross-scale feature information, and improve the efficiency of small object detection. The main contributions of this paper are summarized as follows:

- The traditional C2f module is reconstructed to reduce the redundant computation to efficiently extract spatial features and reduce the loss of shallow information extracted by the backbone.
- Optimized the traditional feature extraction method by globally fusing multi-level features, fusing and extracting feature information across space. The fused information is injected to a higher level, enriching each scale map's feature information. It effectively solves the problem of losing low-level semantic details through a series of convolutions, causing the small object information features to be lost and decreasing detection accuracy.
- Multi-channel aggregated receptive field module (MCARF) is designed, a new downsampling module that replaces the traditional Conv module (CBS). MCARF aggregates the receptive fields of multiple channels to enhance the mutual learning between convolution and expand the image's receptive field.
- The present method has been experimentally validated several times on the publicly available and authoritative small object datasets VisDrone and TinyPerson. On the VisDrone dataset, ViSDrone (PS-YOLO-N) improves 4.1% and 2.5% on mAp50 and mAp50-95, respectively, compared to the baseline model (YOLOv8-N). On the TinyPerson dataset, it improved by 1.16% and 0.48% in the same comparison, respectively. In addition, to validate the model's generalization performance, we also conducted experiments on PASCAL VOC, a publicly available dataset containing various scales. We improved by 1.4% and 1.8% in the same comparison. Finally, we also provided five models N/S/M/L/X for different real-world scenarios and introduced GradCAM visualization to explain the present method's effectiveness.

## 2 Related work

### 2.1 Deep learning-based small object detection algorithms

The definition of a small object in the COCO dataset is an object of  $32 \times 32$  pixels, which is the absolute definition. However, in practice, other datasets have their definitions, such as objects occupying 5–10% of the image size. Small object detection is usually missed or misdirected because of the limited information provided in the input image, the

tendency to lose information after overlaying the network depth, and the limited information on the fused features. In recent years, many researchers in the field of small object detection have attempted to introduce deep learning techniques into small object detection tasks and have conducted many related studies. For example, researchers have improved the efficiency of small object detection by studying backbone networks: CNNs [14–16] structures, Transformer [17, 18] structures (e.g., ViT and derivative algorithms), CNNs combined with Transformer structures [19, 20], etc. The emergence of Transformers and their excellent performance in computer vision has gradually motivated researchers to move from CNN-based or hybrid systems to vision systems based entirely on Transformers. The emergence of Transformers and their outstanding performance in computer vision has gradually motivated researchers to move from CNN-based or hybrid systems to vision systems based entirely on Transformers, which are capable of capturing long-term dependencies in the data, with the advantage of being virtually free of inductive bias to make the models more flexible and adaptable to large amounts of data. Researchers have applied Transformer and various variant architectures to small object detection tasks, i.e., ViT and derived algorithms [21, 22], and have also achieved excellent performance. However, they still fall short compared to one-stage models based on CNNs. With the prevalence of one-stage models for object detection and inspired by other tasks in computer vision [23–26], models such as YOLOv5, YOLOv6, MAF-YOLO, YOLO-World, and Q-YOLO have been successively proposed [27–31]. Small object detection models based on one-stage architectures usually accomplish detection in a single forward propagation, which is more suitable for mobile deployment due to their relatively simple design and easier training process. The emergence of DETR [32] and DINO [33] has also motivated more CNNs to incorporate the Transformer structure. In this aspect of the research, these models first use CNN to extract the base features, flatten and then add positional encoding to get the sequence features, and then feed into the Transformer to do relationship modeling, and finally use the bipartite graph matching algorithm (Hungarian algorithm) to calculate the relationship between the prediction and the ground truth for the best match between prediction and ground truth. Although the performance of CNNs like DETR and DINO combined with the transformer structure is remarkable, the discrete type of Hungarian algorithm and the randomness of model training lead to the computation of ground-truth matching becoming an unstable process. This model type has a significant disadvantage: (1) Generally, it takes longer training time to reach convergence. (2) Regarding performance, DETR is compared to faster rcnn YOLO detection. Although it can achieve better detection of large objects, its performance is unsuitable for small object detection. In

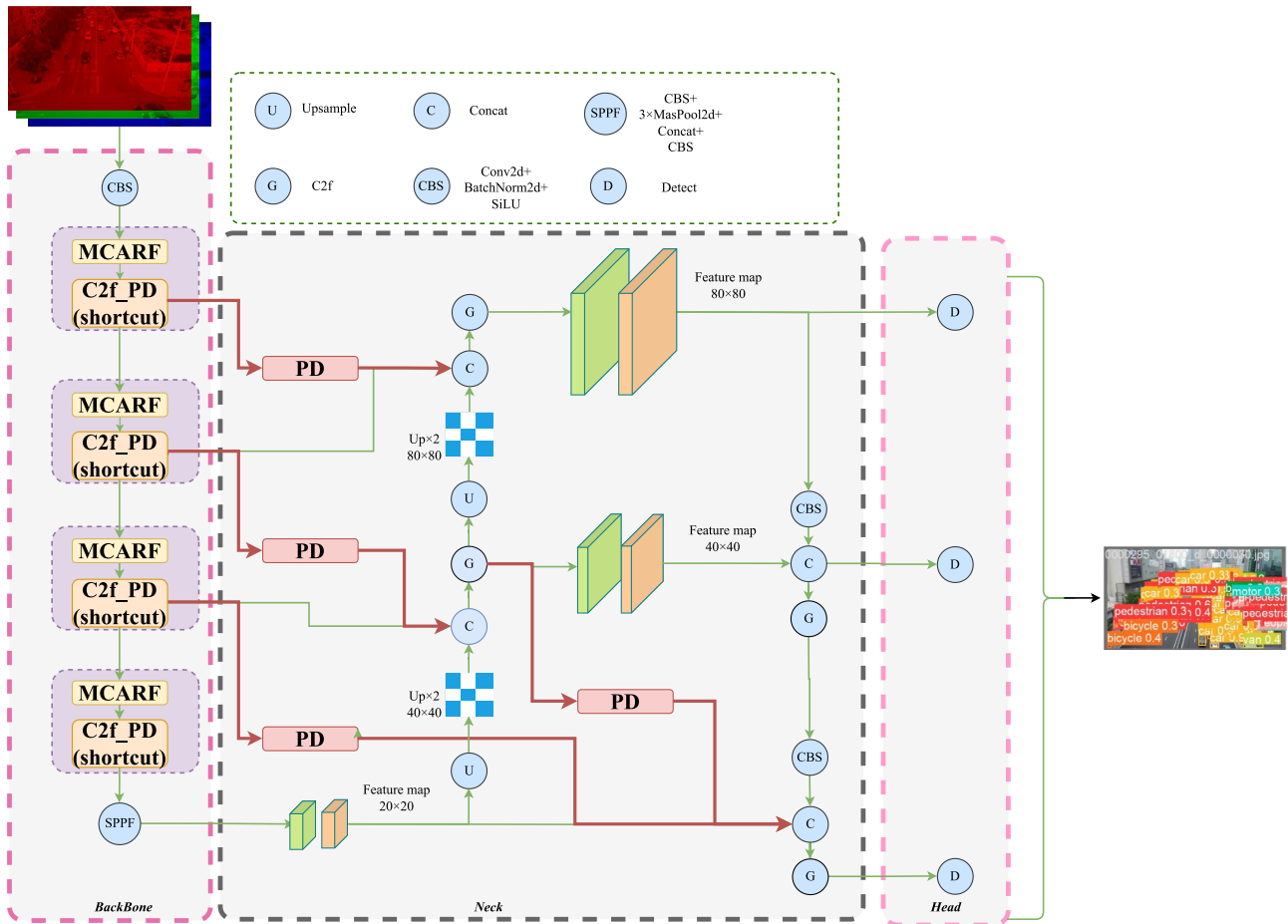
contrast, researchers widely use and study one-stage model detectors for their excellent performance.

## 2.2 Evolution of the feature pyramid

PANet enhances the entire feature hierarchy by augmenting bottom-up path features with accurate localization information in the lower layers. This structure shortens the information path between lower and higher-level features. It has been shown to enhance feature propagation and encourage information reuse, thus improving the expressiveness of the feature pyramid. Since its introduction to address the problem of hierarchical feature fusion in convolutional neural (CNNs), FPN has effectively enhanced problems in object detection tasks, enhancing the ability to detect objects at different scales. However, as the FPN structure loses too much information when transmitting information, YOLOv5 and YOLOv8 are inspired by both and combined into the FPN-PANet structure for multi-scale feature fusion at the neck, which enhances the fusion and mutual utilization of feature information at different scales. Researchers have extensively studied various efficient feature fusions in the past few years. Bidirectional FPN (BiFPN) [34] proposes a weighted bi-directional feature pyramid network and a hybrid scaling method that learns the importance of different feature maps by introducing learnable weights, which is a simple and efficient weighted bi-directional pyramid network. GENERALIZED-FPN [35] observed that previous FPN structures only focus on feature fusion and lack internal links. Therefore, a new path fusion, including jump layers and cross-space scale features, is proposed, reducing the gradient vanishing at the neck of the CNNs structure. Inspired by GENERALIZED-FPN, we introduce the  $\log_2 n - link$  method to improve the feature fusion in the neck of YOLOv8. The purpose is to allow more features to learn from each other and improve the multi-scale feature fusion capability of the model.

## 3 Method

As shown in Fig. 1, we optimize and improve the YOLOv8 network to obtain a high-mAP network named PS-YOLO, which: (1) Reconstructs the C2f module to enrich the gradient flow of the model. It enhances the ability to express features through dense and residual structures and mitigates the problem of small object features weakening or disappearing with frequent downsampling. (2) Feature fusion is applied to the neck part using the  $\log_2 n - link$  method, which can learn more shallow semantic information compared to YOLOv8 by skipping the layer-linking feature information collection and, at the same time, enhance the relationship between the low-level semantic information and high-level semantic information. (3) The C2f module is reconstructed to enrich



**Fig. 1** Optimization techniques used in the PS-YOLO structure. Instead of the original backbone, a new feature extraction network is constructed to capture effective feature information and output it by utilizing the multi-channel aggregate receptive field module

(MCARF) and power detect (PD) module. The  $\log_2 n$  - link method (red arrow in the neck part) is introduced to construct a new feature fusion network instead of the original neck layer. The blue circled module is the original structure (color figure online)

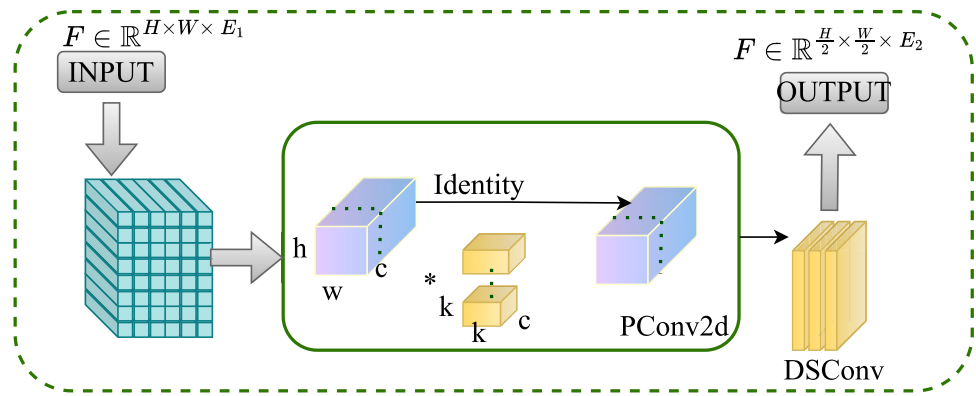
the gradient flow of the model. Information and high-level semantic information. This method can effectively reduce the heavy "giraffe" neck gradient disappearance in the previous YOLO series detectors. (4) A multi-channel aggregated receptive field module (MCARF) is proposed, which forms a new backbone with the reconstructed C2f module. This module can effectively aggregate the feature information of multiple channels and enhance the mutual learning between multiple-channel convolutions. It recognizes more effective local information and substantially improves the model's understanding of complex scenes.

### 3.1 Enrichment of gradient flow: introduction of the PD module

Compared with YOLOv5, YOLOv8 replaces the C3 module of v5 with the C2f module in the backbone part based on the CSPNet idea. However, we found that the C2f (shortcut) module is ineffective in enriching the gradient

flow information with the stacking of backbone network depth in the case of limited image resolution. The speed of extracting features is restricted due to frequent memory accesses. To address this problem, we combine partial convolution (PConv) [36] and distribution shifting convolution (DSConv) [37], named power detection (PD) module, as shown in Fig. 2, in an attempt to reduce redundant computations while speeding up feature extraction and enriching the gradient flow of the model. Because the feature maps have high similarity across channels, PConv applies the regular Conv module to a portion of the input channel for spatial feature extraction. For successive memory accesses, it computes the first or last  $C_p$  channel to represent the entire feature map while keeping the remaining channels constant. DSConv reduces memory usage through variable quantized kernel (VQK) and distribution shifts VQK. VQK is a quantized component of DSConv that allows faster multiplications to be performed. Our proposed PD module, which tries to mimic the original convolutional kernel's distribution,

**Fig. 2** Architecture used by the PD module. Where the inputs reduce sequential memory accesses through PConv and allow feature information to flow through all channels. Finally DSConv is used to perform faster multiplication and enrich the gradient flow of the model



can perform faster and memory-efficient multiplications. It is also able to reduce redundant computations and allow feature information to flow through all channels, enriching the gradient flow of the model. The expression for the PD is calculated by the following formula:

$$DSConv = SiLU(BatchNorm2d(DSConv2d(F_{in}))) \tag{1}$$

$$PD = DSConv(PConv2d(F_{in})) \tag{2}$$

where the input is  $F_{in}$ , we process the input first through PConv2d and then using DSConv (including DSConv2d, BatchNorm2d, SiLU). Then, we use the PD module to replace the Conv module (CBS) in the Bottleneck module, called the Bottleneck Power Detect module (*Bottleneck\_PD*), as shown in Fig. 3. The following formula calculates the expression for the *Bottleneck\_PD*:

$$Bottleneck\_PD = F_{in} + PD_{3 \times 3}(PD_{3 \times 3}(F_{in})), \tag{3}$$

where we use a  $3 \times 3$  convolutional kernel with a step size of 1 for the PD module to process  $F_{in}$  twice consecutively, and finally use residual connections to connect  $F_{in}$  and enrich the gradient flow. Finally, we use the PD module to replace the Conv module (CBS) in C2f, called CSPDarkNet Power Detect Module (*C2f\_PD* (shortcut)), as shown in Fig. 3. *C2f\_PD* (shortcut) can perform faster multiplication via VQK, enriching the gradient flow and mitigating the loss of feature

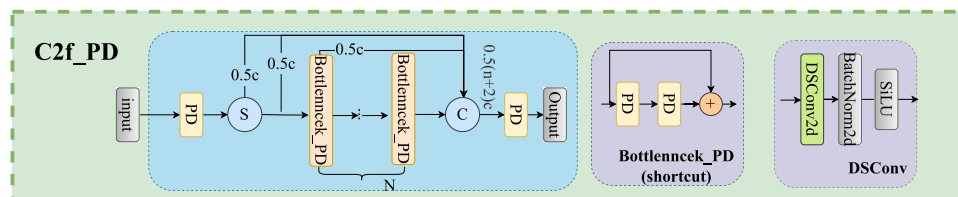
information. The expression for the *C2f\_PD* is calculated by the following formula:

$$A, B, K_0 = Split(PD_{1 \times 1}(F_{in})) \tag{4}$$

$$K_N = Bottleneck\_PD(K_{N-1}) \tag{5}$$

$$C2f\_PD = PD_{1 \times 1}(Concat(A, B, D_1, D_2, \dots, D_N)) \tag{6}$$

where the input image  $F_{in}$  is processed by the PD module of the  $1 \times 1$  convolution kernel, split into the duplicate three copies, and the outputs are A, B,  $K_0$ . A and B are received directly through the Concat, and  $K_0$  is processed recursively through N *Bottleneck\_PDs* ( $N \in (1, 2, \dots)$ ), and each time it is processed, it is added to the Concat. Finally, it is processed using the PD module of the  $1 \times 1$  convolution kernel. It is processed to output. Assuming the number of original input channels is E, the total number of channels after production is  $0.5 \times (n + 2) \times E$ . Compared with C2f (shortcut), *C2f\_PD* (shortcut) reduces redundant computation and frequent memory interrogation, enhances the ability to express features through dense and residual structures, and enriches the gradient flow of the model by linking more cross-layer branches, alleviating the problem of small object features that are weakened or disappeared with frequent downsampling.



**Fig. 3** C2f\_PD’s structure. Inputs speed up feature extraction by using the Power Detect module. Separation enhances the characterization through dense and residual structures that are able to enrich the gradient flow of the model

### 3.2 Multi-scale head-based feature fusion optimization

Multi-scale head detection is used in YOLOv8, as in Fig. 4a, which is capable of detecting images of various sizes. It extracts features by collecting important information from each previous layer. However, we observed that YOLOv8 does not capture feature information of small objects well when collecting large-scale feature maps. It is less efficient in dealing with the neck region because it cannot effectively deal with high-level and low-level information, leading to easy gradient vanishing problems, especially in the collection of cross-scale feature information. For example, Fig. 4a, layer  $C_1$  collects features from  $L_4$ . But if layer  $C_1$  wants to use the features of layer  $L_3$ , layer  $C_1$  can only call the feature information of layer  $L_4$  recursively. Similarly, if the  $C_4$  layer wants to use the feature information of  $C_1$  and  $L_4$ , it can only call the feature information of other layers. This invocation may lead to the loss of information during the computation process and does not fully fuse the low-level semantic information, so the overall effectiveness of information fusion is relatively inefficient. In order to effectively achieve the goal of sufficient multi-scale information exchange, we are inspired by GENERALIZED-FPN and introduce  $\log_2 n$ -link to improve the feature fusion method, as shown in Fig. 4b. The use of the  $\log_2 n$ -link feature fusion method reduces the gradient vanishing from the traditional “giraffe” neck. It can meet the challenge in large-scale change scenarios with small object datasets, ensuring that the model can effectively exchange high-level and low-level information. We use the PD module to capture the feature information of layer  $L_i$  and layer  $C_1$  in Fig. 4b, and the expression for the  $L_i$  is calculated by the following formula:

$$L_i = C2f(Conv_{3 \times 3}(L_{i-1})) \quad (i = 2, 3, 4) \tag{7}$$

where  $L_i$  represents the  $i$ th downsampled backbone layer in Fig. 4. Then, we fuse the information features of different layers and the same scale. We fuse the feature information of SPPF,  $L_4$  layer, and  $L_3$  layer processed by the PD module in  $C_1$  layer. The expression for the  $C_1$  is calculated by the following formula:

$$C_1 = \text{Concat}(SPPF, L_4, PD_{3 \times 3}(L_3)) \tag{8}$$

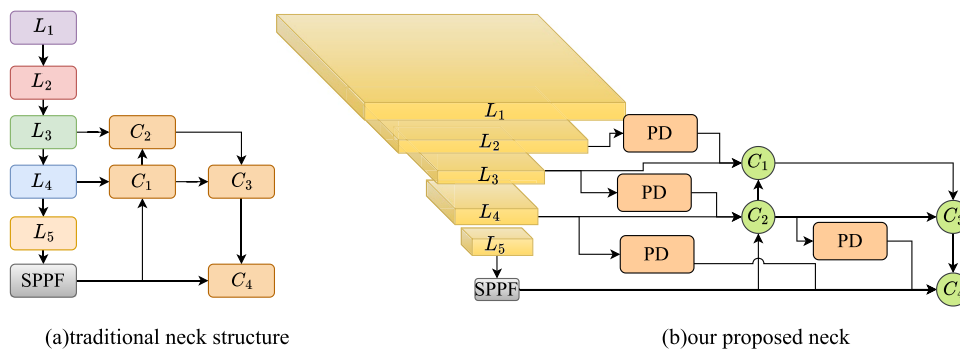
where SPPF is the original structure of YOLOv8,  $C_i$  denotes the Concat layer ( $i \in (1, 2, 3, 4)$ ). We processed the  $L_3$  layer using a  $3 \times 3$  convolutional kernel with a PD of step size 2. The feature information of the  $L_3$  layer is halved in height and width and added to the concat layer along with SPPF,  $L_4$ . Then we fuse the feature information of  $C_1$ ,  $L_3$  layer, and  $L_2$  layer processed by the PD module in the  $C_2$  layer. The expression for the  $C_2$  is calculated by the following formula:

$$C_2 = \text{Concat}(C_1, L_3, PD_{3 \times 3}(L_2)). \tag{9}$$

We use a  $3 \times 3$  convolutional kernel with a step size of 2 for PD to capture the feature information of the  $L_2$  layer. Then, we fuse the feature information of layer  $C_3$ , SPPF, layer  $L_4$  processed by the PD module, and layer  $C_1$  processed by the PD module at layer  $C_4$ . The expression for the  $C_4$  is calculated by the following formula:

$$C_4 = \text{Concat}(C_3, SPPF, PD_{3 \times 3}(L_4), PD_{3 \times 3}(C_1)) \tag{10}$$

Finally, the  $C_1$ ,  $C_3$ , and  $C_4$  layers are sent to the detection head for detection. The improved feature fusion path can effectively accept feature information at more scales. Moreover, the dense information is shared at different spatial scales and non-adjacent latent semantic levels and fused for mutual learning. It allows the model to have shallow spatial information and high-level semantic information of the same



**Fig. 4** (a) The traditional skeleton and neck feature fusion path of YOLOv8, where SPPF is the original structure of YOLOv8. (b) is the improved skeleton and neck feature fusion path, where the orange module is the power detect ( $k = 3, s = 2$ ) module. The improved feature fusion path can effectively accept feature informa-

tion at more scales and improve the learnability and stability of the network. Where (a), (b) in the figure  $L_i$  denotes the downsampling layer,  $i \in (1, 2, 3, 4, 5)$ , and  $C_i$  denotes the Concat layer,  $i \in (1, 2, 3, 4)$  (color figure online)

importance at the neck, effectively solving the problem of low-level semantic information being lost through a series of convolutions resulting in the loss of small object information features. At last, it also ensures the feature information of normal-sized objects, thus avoiding the problem of information loss and improving the learnability and stability of the network.

### 3.3 Multi-channel aggregate receptive field module

In recognition of small objects in complex scenes, small objects are prone to overlap each other, occlusion, dark environment, and other problems. It is challenging to gather information from multiple channels so that we can learn from each other during the training process. More obvious feature deviations lead to more difficulty recognizing partial information. In order to solve the above problems, we designed the multi-channel aggregated receptive field module (MCARF), as shown in Fig. 5, to replace the traditional Conv module (CBS). Based on guaranteeing the detection effect of the baseline, it can effectively aggregate the feature maps of over channels to learn from each other and solve the feature bias problem. Depthwise over-parameterized convolutional (DOConv) [38] uses multi-layer composite linear operation that can be collapsed into a compact single-layer representation after the training stage, which not only accelerates the training of the model but also continuously improves the performance of the converged model. We, therefore, perform additional pooling on the feature maps of DSConv, PConv, and DOConv by retaining their processed feature maps, respectively. The receptive fields of each channel are enhanced as much as possible in different degrees, and finally, the number of channels is merged. First, we split the dimension of the input image into thirds and assume that the number of channels is all  $E_1$ ,

respectively  $input_1, input_2, input_3$ , which is calculated by the following formula:

$$input_1, input_2, input_3 = \text{Split}(input). \tag{11}$$

The  $input_1$  is passed through DSConv, then PConv is used to reduce the redundancy calculation, and then Maxpool is used to increase the receptive field. The number of output channels is  $output_1$  of  $E_2$ , which is calculated by the following formula:

$$output_1 = \text{MAXPOOL}(\text{PConv2d}(\text{DSConv}_{1 \times 1}(input_1))). \tag{12}$$

The  $input_2$  is processed using a  $1 \times 1$  DOConv to add learnable parameters. Then, the redundant computation is reduced by PConv processing. Finally, the fully connected layer is replaced by AdaptiveAvgPool processing. The number of output channels is  $output_2$  of  $E_2$ , which is calculated by the following formula:

$$output_2 = \text{ADAPTIVEAVGPOOL}(\text{PConv}(\text{DOConv}_{1 \times 1}(input_2))). \tag{13}$$

The  $input_3$  input is processed using a  $3 \times 3$  convolution kernel with a step size 2 for DSConv. The number of output channels is  $E_2$  for output3, which is calculated by the following formula:

$$output_3 = \text{DSConv}_{3 \times 3}(input_3). \tag{14}$$

Concentrating more attention on large-scale feature maps further reduces the loss of feature information. Finally, the features extracted from the three outputs are concatenated and used in Conv (CBS module). The number of converted channels is  $E_2$ , the output output, which is calculated by the following formula:

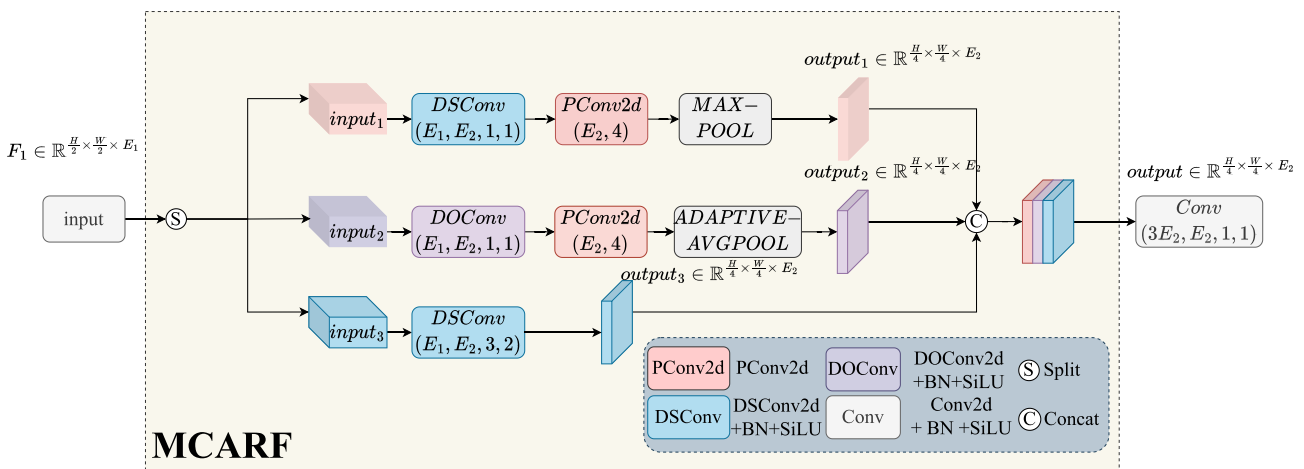


Fig. 5 Structure of multi-channel aggregate receptive field module (MCARF). We process the input through three different convolutions of three channels to get three outputs with the same aspect and num-

ber of channels. Then the three outputs are merged, and finally the number of channels is changed to one-third of the original merged output by the CBS (Conv) module

**Table 1** Comparison with two stage and one stage detectors on VisDrone 2019

Method	Backbone	mAP50 (%)	mAP50-95 (%)	P	R	Params (M)	FLOPS (G)	Latency (ms) (bs = 1)	FPS (ms) (bs = 1)	Latency (bs = 16)	FPS (bs = 16)
SSD	ResNet-50	10.9	5.7	-	-	34.3	106.9	-	-	-	-
EfficientDet	EfficientDet-D1	21.2	12.9	-	-	6.6	6.1	-	-	-	-
Retinanet	ResNet-50-FPN	26.2	15.7	-	-	34.0	248.0	-	-	-	-
Centernet	ResNet-50	29.0	14.0	-	-	31.8	206.0	-	-	-	-
Faster R-CNN	ResNet-50-FPN	36.3	20.6	-	-	41.0	207.0	-	-	-	-
YOLOv3-Tiny	DarkNet53	20.0	10.9	34.7	22.1	12.1	18.9	2.1	480.5	1.3	778.9
YOLOv3-Spp	DarkNet53	37.6	22.3	51.6	38.4	104.7	283.1	14.5	69.2	12.8	78.3
YOLOv3	DarkNet53	37.5	22.1	50.4	39.1	103.7	283.0	14.2	70.2	13.1	76.1
YOLOv5-N	CSPDarkNet	27.4	15.5	39.2	29.4	2.5	7.1	5.2	191.5	1.3	797.9
YOLOv5-S	CSPDarkNet	32.5	18.7	45.9	34.1	9.1	24.1	5.3	189.3	2.4	419.3
YOLOv5-M	CSPDarkNet	35.8	20.9	48.4	37.0	25.0	64.0	6.7	149.8	5.0	202.2
YOLOv5-L	CSPDarkNet	36.8	21.9	49.5	38.0	53.1	134.7	8.9	112.9	7.3	137.0
YOLOv5-X	CSPDarkNet	38.0	22.5	50.7	39.6	97.2	246.9	19.3	51.8	11.5	86.7
YOLOv6-N	EfficientRep	26.7	15.0	39.3	28.5	4.2	11.8	4.2	240.6	0.9	1062.6
YOLOv6-S	EfficientRep	32.3	18.0	45.2	33.8	16.3	44.2	4.2	239.0	2.1	475.6
YOLOv6-M	EfficientRep	34.9	20.6	49.1	35.5	51.9	161.2	9.0	111.8	6.3	157.9
YOLOv6-L	EfficientRep	36.4	21.5	50.5	37.2	110.8	391.9	15.7	63.6	12.0	83.2
YOLOv6-X	EfficientRep	36.8	21.8	50.0	37.5	172.9	610.3	25.0	40.0	21.2	47.3
YOLOv8-N	CSPDarkNet(C2f)	28.2	16.0	40.4	30.1	3.0	8.1	4.9	202.8	1.1	891.9
YOLOv8-S	CSPDarkNet(C2f)	33.2	19.2	46.2	34.7	11.1	28.4	4.7	211.6	2.1	472.4
YOLOv8-M	CSPDarkNet(C2f)	35.6	20.9	49.4	36.5	25.8	78.7	6.6	152.3	4.8	209.4
YOLOv8-L	CSPDarkNet(C2f)	36.8	21.9	49.7	38.2	43.6	165.4	9.5	105.5	7.9	126.5
YOLOv8-X	CSPDarkNet(C2f)	37.9	22.6	52.1	38.7	43.6	164.8	18.9	52.8	13.0	76.9
<b>PS-YOLO-N</b>	<b>(Ours)</b>	<b>32.3</b>	<b>18.5</b>	<b>45.4</b>	<b>33.9</b>	<b>5.0</b>	<b>10.0</b>	<b>6.3</b>	<b>186.8</b>	<b>1.9</b>	<b>760.8</b>
<b>PS-YOLO-S</b>	<b>(Ours)</b>	<b>36.0</b>	<b>21.0</b>	<b>49.8</b>	<b>36.4</b>	<b>19.4</b>	<b>35.4</b>	<b>6.4</b>	<b>160.3</b>	<b>3.7</b>	<b>384.4</b>
<b>PS-YOLO-M</b>	<b>(Ours)</b>	<b>37.6</b>	<b>22.3</b>	<b>50.2</b>	<b>38.9</b>	<b>42.2</b>	<b>88.0</b>	<b>9.1</b>	<b>104.6</b>	<b>5.1</b>	<b>175.3</b>
<b>PS-YOLO-L</b>	<b>(Ours)</b>	<b>39.3</b>	<b>23.3</b>	<b>52.0</b>	<b>40.5</b>	<b>69.3</b>	<b>176.0</b>	<b>13.3</b>	<b>84.7</b>	<b>10.4</b>	<b>99.1</b>
<b>PS-YOLO-X</b>	<b>(Ours)</b>	<b>39.7</b>	<b>23.6</b>	<b>52.5</b>	<b>41.0</b>	<b>108.3</b>	<b>274.5</b>	<b>27.8</b>	<b>45.2</b>	<b>20.2</b>	<b>56.7</b>
YOLOv10-N	-	27.1	15.1	40.0	29.3	2.3	6.7	7.7	129.3	2.5	393.4
YOLOv10-S	-	31.9	18.2	45.3	33.9	7.2	21.6	10.9	92.1	4.7	213.0
YOLOv10-M	-	34.8	20.3	48.7	36.2	15.4	59.1	14.7	68.1	8.9	112.3
YOLOv10-L	-	37.3	22.0	51.7	37.8	24.4	120.3	19.1	52.4	14.1	70.9
YOLOv10-X	-	37.1	21.7	51.7	37.8	29.5	160.4	25.5	39.2	20.3	49.3

Bold values highlight the performance of PS-YOLO across five different real-world scenarios: N, S, M, L, and X. PS-YOLO demonstrates superior performance metrics, such as mAP50 and mAP50-95, compared to other models within the same scenarios



$$\text{output} = \text{Conv}_{1 \times 1}(\text{Concat}(\text{output}_1, \text{output}_2, \text{output}_3)). \quad (15)$$

We use the Conv module (CBS) to convert the number of feature channels. The number of channels before conversion is  $3 \times E_2$ , and the number after conversion is  $E_2$ . The advantage of doing so is that it gathers feature information from multiple channels and enhances mutual learning between numerous channel convolutions. Resolve the feature bias in the fusion process and recognize more local information. This helps to improve the model's ability to understand complex scenes.

## 4 Experiments

### 4.1 Datasets

In this study, the detection performance of the proposed model is evaluated on two public UAV datasets and a publicly available generalized object detection dataset. The

proposed method is compared with existing classical object detection algorithms to demonstrate its superiority.

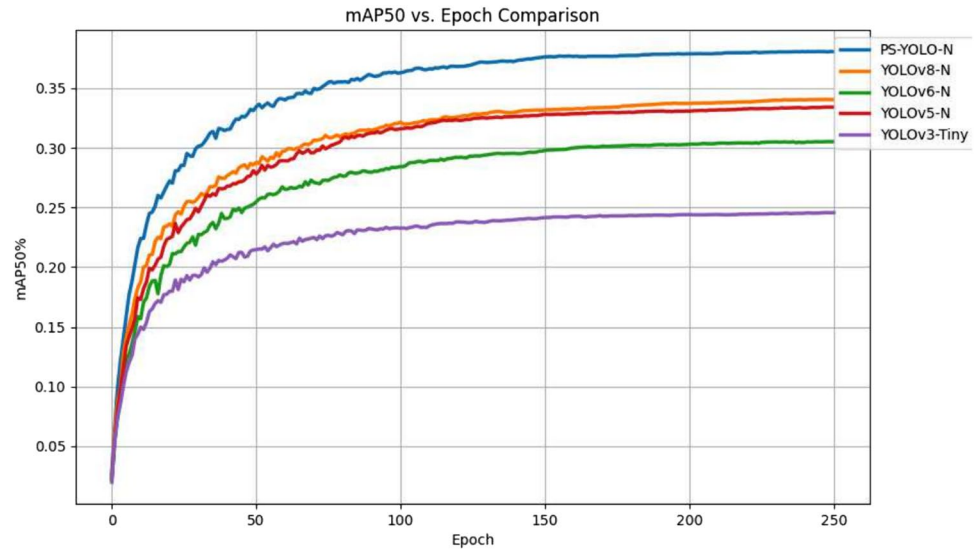
- The VisDrone2019 [39] dataset is collected by the AISKYEYE team at the Machine Learning and Data Mining Laboratory of Tianjin University. where the target size ranges from 12–4002 pixels, with an uneven distribution of target sizes at different scales. The dataset includes over 250 video clips, 25,000 frames, and 1000 still images. It covers a wide range of scenarios, including complex environments (from urban and rural areas), different locations (from 14 different cities in China), and so on. The dataset contains ten classes (pedestrians, people, cars, vans, buses, trucks, automobiles, bicycles, shade tricycles, tricycles). Some critical attributes, including scene visibility, object class, and occlusion, are also provided for better data utilization.

**Table 2** Comparison of detection performances of different methods on the TinyPerson dataset

Method	Backbone	mAP50 (%)	mAP50-95 (%)	P	R
YOLOv3-Tiny	DarkNet53	10.07	3.87	27.54	10.25
YOLOv3-Spp	DarkNet53	19.30	6.93	40.30	18.60
YOLOv3	DarkNet53	18.80	7.04	37.40	18.60
YOLOv5-N	CSPDarkNet	14.70	5.18	30.20	16.70
YOLOv5-S	CSPDarkNet	17.80	6.42	37.20	18.20
YOLOv5-M	CSPDarkNet	19.20	6.95	35.10	20.10
YOLOv5-L	CSPDarkNet	18.40	6.59	34.70	18.90
YOLOv5-X	CSPDarkNet	18.70	6.77	36.30	19.40
YOLOv6-N	EfficientRep	14.78	5.28	31.43	17.00
YOLOv6-S	EfficientRep	16.30	5.80	33.60	17.50
YOLOv6-M	EfficientRep	13.60	4.72	27.50	16.00
YOLOv6-L	EfficientRep	–	–	–	–
YOLOv6-X	EfficientRep	10.80	3.99	25.10	14.70
YOLOv8-N	CSPDarkNet(C2f)	16.74	5.95	29.34	19.91
YOLOv8-S	CSPDarkNet(C2f)	17.50	6.19	35.60	18.30
YOLOv8-M	CSPDarkNet(C2f)	19.20	6.73	32.10	20.70
YOLOv8-L	CSPDarkNet(C2f)	18.50	6.59	35.70	19.30
YOLOv8-X	CSPDarkNet(C2f)	19.20	6.69	34.30	20.30
<b>PS-YOLO-N</b>	<b>(Ours)</b>	<b>17.90</b>	<b>6.43</b>	<b>33.40</b>	<b>19.60</b>
<b>PS-YOLO-S</b>	<b>(Ours)</b>	<b>17.70</b>	<b>6.40</b>	<b>33.10</b>	<b>18.70</b>
<b>PS-YOLO-M</b>	<b>(Ours)</b>	<b>19.60</b>	<b>7.30</b>	<b>38.20</b>	<b>19.70</b>
<b>PS-YOLO-L</b>	<b>(Ours)</b>	<b>19.70</b>	<b>7.19</b>	<b>40.60</b>	<b>19.10</b>
<b>PS-YOLO-X</b>	<b>(Ours)</b>	<b>19.60</b>	<b>7.06</b>	<b>36.90</b>	<b>20.10</b>
YOLOv10-N	–	15.20	5.26	29.60	18.70
YOLOv10-S	–	17.20	6.03	34.90	19.60
YOLOv10-M	–	17.70	6.33	34.00	19.90
YOLOv10-L	–	16.80	5.86	33.90	19.00
YOLOv10-X	–	18.00	6.28	36.80	19.60

Bold values highlight the performance of PS-YOLO across five different real-world scenarios: N, S, M, L, and X. PS-YOLO demonstrates superior performance metrics, such as mAP50 and mAP50-95, compared to other models within the same scenarios

**Fig. 6** Comparing the training set on VisDrone with the training set of  $N$  models of other YOLO detectors with respect to the change curve of mAP50, our model far exceeds the  $N$  models of other YOLO detectors



- TinyPerson [40] is a tiny object detection dataset proposed in the context of rapid rescue. The average absolute size of a person in TinyPerson is around 18 pixels. It can better reflect the characters in long-distance video surveillance scenes. Regarding the labeling rules of the data, TinyPerson divides people into two categories: sea person and earth person. A person in a boat, lying in the water, or with more than half of his body in the water is defined as a sea person, and the rest as an earth person.
- The PASCAL VOC [41, 42] dataset is used in tasks including, but not limited to, classification, detection, segmentation, action recognition, etc. PASCAL VOC mainly consists of two datasets, PASCAL VOC 2007 and PASCAL VOC 2012. The pixel dimensions of its images vary in size, but the horizontal image is around  $500 \times 375$ , and the vertical image is around  $375 \times 500$ , basically not deviating by more than 100. PASCAL VOC 2007 contains about 2500 images in the training set, 2500 images in the validation set, and 5000 images in the test set, while PASCAL VOC 2012 contains about 6000 images in the training set and 6000 images in the validation set. We use a ratio of 13:2:5 to divide the training, validation, and test sets.

## 4.2 Implementation details

We use a batch size of 16, a total epoch of 500, a warm-up epoch of 3, a base learning rate of 0.01, stochastic gradient descent with momentum of 0.937, an SGD optimizer, and a weight decay of  $5e-4$ . The operating system used in this paper is Ubuntu version 20.04.1. The system's hardware facilities are a sheet of 24 GB of RAM and an NVIDIA A30 GPU. The software platform and version are torch 1.12.1, Python3.9.17, Anaconda, and Ultralytics 8.0.125.

## 4.3 Evaluation metric

The evaluation metrics used in this paper include P, R, and mAP (include mAP@50, mAP@[50:95]). Where P is Precision, R is Recall, and mAP is averaging the AP values for all categories. where these formulas can be shown as:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (16)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (17)$$

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^N AP_i \quad (18)$$

where  $AP_i$  is the AP in the  $i$ th class, and  $N$  is the total number of classes being evaluated.

## 4.4 Comparison with state-of-the-art methods

We focus on evaluating the P, R, and mAP of our model on the VisDrone, TinyPerson, and PASCAL VOC datasets. Additionally, we uniformly do not use pre-training weights for a fair comparison. To test and compare metrics in the same environment, we use batch size = 1 for the evaluation of four metrics on the test set.

Table 1. shows the results of PS-YOLO in VisDrone compared to other state-of-the-art single- and two-stage object detectors. Our PS-YOLO-N achieves 4.1%, 5.6%, 4.9%, and 12.3% improvement in mAP50 compared to the mAP50 of the same type of YOLO detectors, YOLOv8-N, YOLOv6-N, YOLOv5-N, and YOLOv3-Tiny, respectively. Compared with the mAP50 of the one-stage detector SSD, there is a substantial improvement in detection accuracy.

**Table 3** Comparison with other YOLO detectors on PASCAL VOC

Method	Backbone	mAP50 (%)	mAP50-95 (%)	P	R
v3-Tiny	DarkNet53	70.8	45.0	73.0	65.2
v3-Spp	DarkNet53	84.9	66.8	82.2	78.6
v3	DarkNet53	84.5	66.1	83.0	77.8
YOLOv5-N	CSPDarkNet	76.9	55.6	78.4	69.4
YOLOv5-S	CSPDarkNet	81.3	60.4	80.9	74.2
YOLOv5-M	CSPDarkNet	83.6	64.6	81.0	78.6
YOLOv5-L	CSPDarkNet	85.1	66.8	82.6	78.8
YOLOv5-X	CSPDarkNet	85.4	67.8	83.1	78.8
YOLOv6-N	EfficientRep	79.4	59.7	80.7	71.0
YOLOv6-S	EfficientRep	82.5	63.4	81.2	75.6
YOLOv6-M	EfficientRep	83.0	64.3	82.9	75.0
YOLOv6-L	EfficientRep	76.5	58.0	75.6	68.7
YOLOv6-X	EfficientRep	78.0	59.5	78.3	69.6
YOLOv8-N	CSPDarkNet(C2f)	78.9	58.4	79.3	72.1
YOLOv8-S	CSPDarkNet(C2f)	82.0	61.8	81.4	73.3
YOLOv8-M	CSPDarkNet(C2f)	82.9	65.1	81.4	77.1
YOLOv8-L	CSPDarkNet(C2f)	85.1	67.6	82.9	78.9
YOLOv8-X	CSPDarkNet(C2f)	85.9	68.2	83.4	78.8
<b>PS-YOLO-N</b>	<b>(Ours)</b>	<b>80.3</b>	<b>60.2</b>	<b>81.0</b>	<b>73.1</b>
<b>PS-YOLO-S</b>	<b>(Ours)</b>	<b>83.5</b>	<b>64.1</b>	<b>81.3</b>	<b>77.8</b>
<b>PS-YOLO-M</b>	<b>(Ours)</b>	<b>84.8</b>	<b>67.1</b>	<b>83.5</b>	<b>77.8</b>
<b>PS-YOLO-L</b>	<b>(Ours)</b>	<b>85.8</b>	<b>68.2</b>	<b>83.2</b>	<b>79.6</b>
<b>PS-YOLO-X</b>	<b>(Ours)</b>	<b>86.4</b>	<b>68.9</b>	<b>83.7</b>	<b>79.2</b>
YOLOv10-N	–	78.8	58.7	80.3	70.4
YOLOv10-S	–	82.2	62.7	81.8	74.7
YOLOv10-M	–	83.8	65.2	83.5	75.4
YOLOv10-L	–	84.6	66.2	82.5	78.4
YOLOv10-X	–	84.5	66.0	82.8	78.0

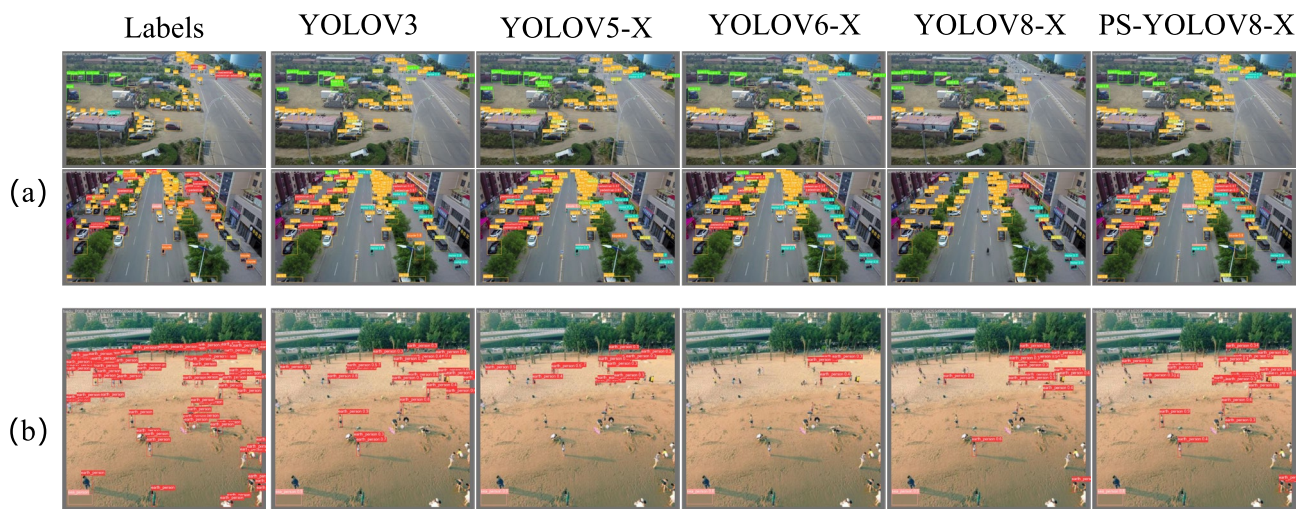
Bold values highlight the performance of PS-YOLO across five different real-world scenarios: N, S, M, L, and X. PS-YOLO demonstrates superior performance metrics, such as mAP50 and mAP50-95, compared to other models within the same scenarios

**Table 4** Ablation study of PS-YOLOv8. The test baseline is the A30 assessment of YOLOv8-N

A	B	C	mAP50	mAP50-95	P	R
✓			28.7	16.2	40.8	30.7
	✓		29.7	16.9	41.8	31.8
		✓	31.0	17.7	44.4	32.2
✓	✓		31.3	17.8	44.3	32.9
✓	✓	✓	<b>32.3</b>	<b>18.5</b>	<b>45.4</b>	<b>33.9</b>

PS-YOLO-N achieves 11.1%, 6.1%, and 3.3% improvement compared with the mAP50 of the two-stage detectors EfficientDet, Retinanet, and Centernet, respectively. In addition to this, we also compared it with the state-of-the-art detector—YOLOv10 [43]. mAP50 of PS-YOLO-N is improved by 5.2% compared to YOLOv10-N. Our S/M/L models for different practice scenarios also achieve different levels of improvement. Furthermore, we also compared the previously described metrics and FPS and Latency of all

single-stage detectors in this dataset. Although the FPS and Latency of PS-YOLO showed a small decrease compared to baseline, the mAP50 of PS-YOLO-X was 1.8%, 2.9%, 1.7%, and 2.0% higher than that of the same-type YOLO detectors YOLOv8-X, YOLOv6-X, YOLOv5-X, YOLOv3 by 1.8%, 2.9%, 1.7%, and 2.2%, and the evaluation indexes P and R of PS-YOLO-X were superior to other YOLO detectors of the same type, and PS-YOLO-X achieved a 3.4% improvement in the mAP50 compared to the mAP50 of the



**Fig. 7** Sample detection results from YOLOv3, YOLOv5-X, YOLOv6-X, YOLOv8-X and PS-YOLO-X. **a** VisDrone. **b** TinyPerson

two-stage detector, Faster-RCN. Besides, the experimental results show that PS-YOLO-X outperforms YOLOv10-X in mAP50 and mAP50-95 by 5.2%, 3.4%, and also achieves lower latency and higher FPS. The experimental results demonstrate the superior performance of the model we provide. This reflects that PS-YOLO can effectively enhance the sharing of dense information at different spatial scales and non-adjacent latent semantic levels and fuse them for mutual learning. Figure 6 shows the training set variation curves of PS-YOLO-N with YOLOv8-N, YOLOv6-N, YOLOv5-N, and YOLOv3-tiny, and the performance of our model PS-YOLO-N is far superior to other detectors of the same type.

The results of PS-YOLO in TinyPerson compared to other state-of-the-art single-stage object detectors are shown in Table 2. Our PS-YOLO-N achieves 1.16%, 3.12%, 4.32%, 7.83%, and 2.7% improvement in mAP50 compared to the mAP50 of the same type of YOLO detectors, YOLOv8-N, YOLOv6-N, YOLOv5-N, YOLOv3-Tiny, and YOLOv10-N, respectively, and the PS-YOLO N's evaluation indexes P and R are better than other YOLO detectors of the same type. Through the data in the table, we can find that the performance of each series of models of PS-YOLO, YOLOv8, YOLOv6, YOLOv3, and YOLOv10 fluctuates on this dataset to a different degree. The reason is that most of the objects in the images of this dataset are much smaller than the VisDrone so that many of the detectors under-perform on this dataset. However, Our N/S/M/L/X models for different practice scenarios outperform the same type of detectors in all evaluation metrics. And PS-YOLO-L achieves 0.5%, 3.4%, 0.5%, 0.4%, 1.9% improvement over the mAP50 of the best performing YOLOv8-X, YOLOv6-S, YOLOv5-M, YOLOv3-Spp, YOLOv10-X, respectively. This demonstrates

that PS-YOLO can efficiently aggregate feature information from multiple channels, learn from each other about them, and improve the model's understanding of extreme scenes.

Besides, to test the generalization of our model, we validate the generalization performance of our model on the generic object detection dataset PASCAL VOC 2007+2012. This dataset contains images of various scales, and the various sizes of objects contained in the images are sufficient to test the generalization ability of our proposed model, as shown in Table 3. Table 3 shows the results of PS-YOLO on PASCAL VOC in comparison with other state-of-the-art single-stage object detectors. By looking at the data in the table, we can see that our PS-YOLO-N achieves 1.4%, 0.9%, 3.4%, 9.5%, 1.5% improvement in mAP50 compared to the mAP50 of the same type of YOLO detectors, YOLOv8-N, YOLOv6-N, YOLOv5-N, YOLOv3-Tiny, and YOLOv10-N, respectively and the evaluation indexes P and R of PS-YOLO-N are better than other YOLO detectors of the same type. Our S/M/L models for different practice scenarios also achieve different degrees of improvement in all evaluation metrics. In addition, PS-YOLO-X achieves 0.5%, 8.4%, 1%, 1.9%, 1.9% improvement in mAP50 with the same type of YOLO detectors YOLOv8-X, YOLOv6-X, YOLOv5-X, YOLOv3, and YOLOv10-X. The experimental results fully verify that PS-YOLO can improve the stability of complex scene detection with excellent generalization ability.

In conclusion, the above three experimental results show that PS-YOLO has the ability to far outperform the above object detection algorithms in terms of detection accuracy compared to the two-stage detector as well as other YOLO series detectors.

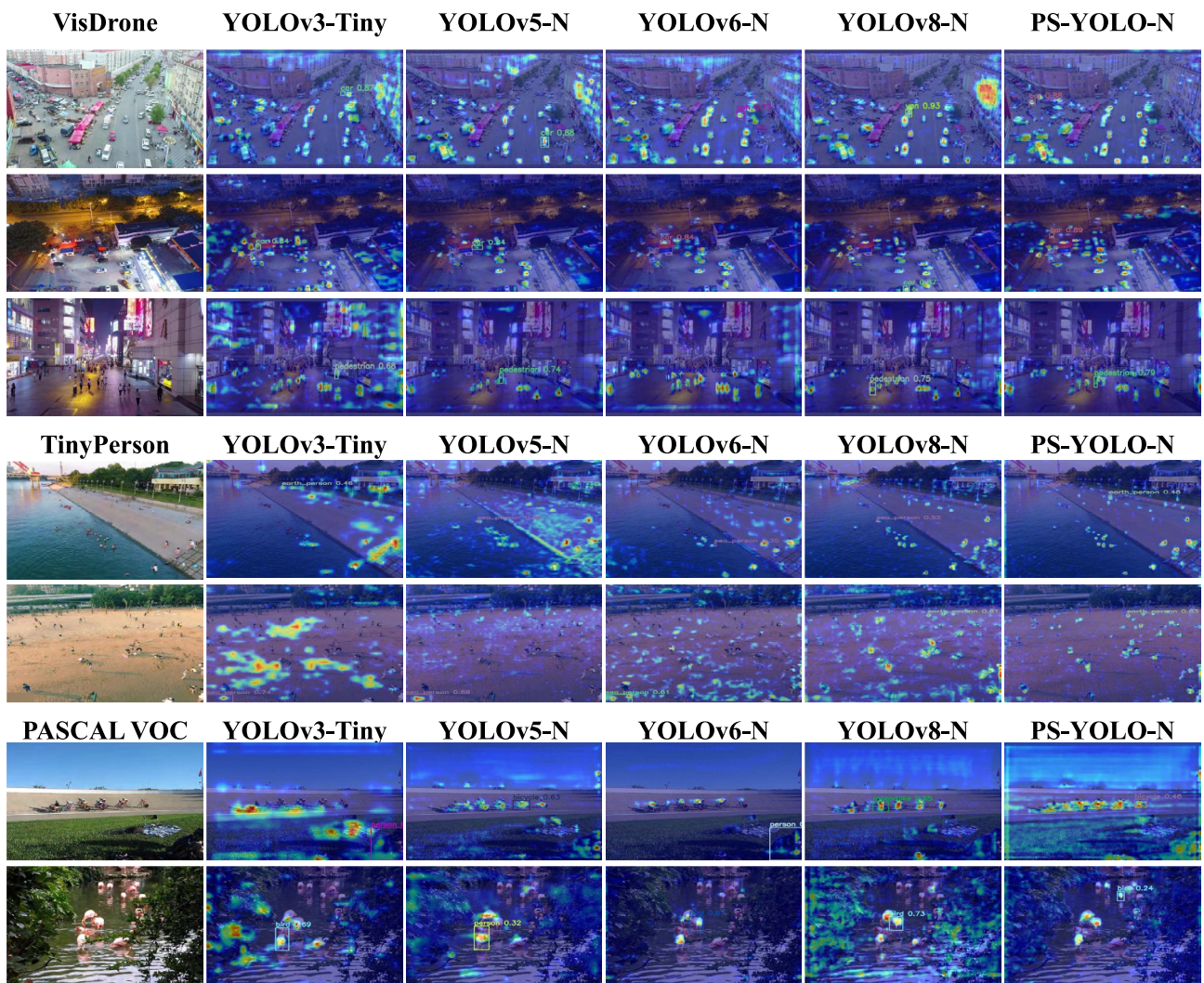


Fig. 8 Neck CAM visualization

### 4.5 Ablation study

In order to validate the effectiveness of the improved methods, this paper conducts ablation experiments at each improvement on the test set in the ViSDrone dataset. To ensure fairness, all experiments in this paper do not use pre-trained weights and compare them with YOLOv8-N. In order to be able to systematically analyze the performance improvement of each unit model, we sequentially define the improvement model A ( $C2f\_PD$ ), the improvement model B ( $featurefusion\_PD$ ), and the improvement model C (MCARF). To ensure the authenticity of the experiments, the accuracy (P), recall (R), and average precision (AP) are used as the evaluation metrics in this experiment. The test results are shown in Table 4. Improved model A shows that by replacing the Conv(CBS) module with the PD module, we can reduce the weakening or loss of the feature information with the depth of the network by

enriching the flow of the information gradient. The problem of feature information being weakened or lost as the network depth is superimposed. Improvement model B indicates that the feature bias problem in the fusion process can be effectively solved by improving the feature fusion part of the NECK, which improves 1.5% and 0.7% compared to the baseline mAP50 and mAP50-95, respectively. Improvement model C marked that the improvement can effectively gather information features from multiple channels and extract more local feature information more effectively. The results of both Improved Model A+B are better than Improved Model A and Improved Model B, indicating that the fusion of the two modules is effective. The four evaluation indexes of the improved model A+B+C are all better than the improved model A+B, respectively. The experimental results prove the effectiveness of each improvement measure.



**Fig. 9** Visual analysis of PS-YOLO's failed samples on VisDrone. Where green boxes are correct detections, red boxes are missed detections, and blue boxes are false detections (color figure online)

#### 4.6 Visualization

First of all, as shown in Fig. 7, we select the sample detection results of YOLOv3, YOLOv5-X, YOLOv6-X, and YOLOv8-X to compare with PS-YOLO-X. We note that the other detectors have different degrees of misdetection omission of small-size targets in small object images. Our method is able to reduce the false detection rate while improving the accuracy. The improved performance is mainly due to the fact that in this paper, we use the C2f Power Detect module,  $\log_2 n$  - link method, and multi-channel aggregated sensory field module, which better balance the feature information of different scales. It not only recognizes more local information but also reduces the possibility of interference for the target and improves the model's ability to understand the complex scene.

Secondly, below are the results of CAM visualization of the necks of YOLOv3, YOLOv5, YOLOv6, YOLOv8, and our PS-YOLOv8 on the VisDrone, TinyPerson, and PASCAL VOC datasets as shown in Fig. 8. Our model assigns higher weights to the detected regions of the object and has fewer false detections for the background. We can observe that the sensitivity of the feature maps of the other models to object locations is gradually reduced or lost as the depth of the network increases and the information interaction between different levels increases. And

the sensitivity to non-objects is accompanied by different degrees of increase. Our proposed PS-YOLO-N model is more sensitive to object locations and does not pay much attention to non-objects in the region.

Finally, we provide some visual analysis of the failed samples of our model to guide future research. As shown in Fig. 9, most of the targets can be detected correctly, and a small number of targets are missed. This may be because the targets are occluded (e.g., one row and four columns, three rows and four columns in the figure), including wall occlusion, character occlusion, boundary occlusion, etc., so that the occluded features are lost in the processing. Moreover, the target pixels are too small or blurred, and the contextual information around these smaller targets is not obvious enough (e.g., four rows and four columns in the figure) to make the model difficult to understand, which is also a possible cause of missed detection.

## 5 Conclusion

In this paper, we propose a new small object detector called PS-YOLO for small object detection. The architecture improves the C2f module, named  $C2f\_PD$ , based on YOLOv8, proposes a new downsampling module, MCARF,

and optimizes the traditional feature fusion method of YOLOv8, which is designed in such a way that the network can improve the model to capture the feature information of the small objects, fully fuses the shallow semantic information with the high-level semantic information, and learns the feature information from each other. The model is not only suitable for small object detection but also has some generalization ability in publicly available comprehensive object detection datasets. In the future, we will construct models with strong robustness and generalization based on more aspects (e.g., loss reduction use of excellent IOUs).

**Acknowledgements** This work is supported by the Natural Science Foundation of Tianshan Talent Training Program (NO. 2023TSY-CLJ0023), Major science and technology programs in the autonomous region (No. 2023A03001), Xinjiang Uygur Autonomous Region (NO. 2023D01C176), Xinjiang Uygur Autonomous Region Universities Fundamental Research Funds Scientific Research Project (NO. XJEDU2022P018).

**Author contributions** Shifeng Peng: Conceptualization, methodology, data organization, writing—original manuscript preparation, visualization. All authors reviewed the manuscript.

**Data availability** This study uses 3 publicly available datasets: (1) VisDrone primary login code is <https://github.com/VisDrone/VisDrone-Dataset>, (2) TinyPerson primary login code is <https://opendatalab.com/OpenDataLab/TinyPerson>, (3) PASCAL VOC main login code is <http://host.robots.ox.ac.uk/pascal/VOC/>.

## Declarations

**Conflict of interest** The authors declare no competing interests.

## References

- Liu, Y., Sun, P., Wergeles, N., Shang, Y.: A survey and performance evaluation of deep learning methods for small object detection. *Expert Syst. Appl.* **172**, 114602 (2021)
- Wen, L., Cheng, Y., Fang, Y., Li, X.: A comprehensive survey of oriented object detection in remote sensing images. *Expert Syst. Appl.* **224**, 119960 (2023)
- Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587 (2014)
- Girshick, R.: Fast r-cnn. In: *IEEE ICCV* (2015)
- Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems*, vol. 28 (2015)
- Wang, C., Bai, X., Wang, S., Zhou, J., Ren, P.: Multiscale visual attention networks for object detection in vhr remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **16**(2), 310–314 (2018)
- Rabbi, J., Ray, N., Schubert, M., Chowdhury, S., Chao, D.: Small-object detection in remote sensing images with end-to-end edge-enhanced gan and object detector network. *Remote Sens.* **12**(9), 1432 (2020)
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: single shot multibox detector. In: *ECCV* (2016). Springer
- Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: *IEEE CVPR* (2016)
- Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018). [arXiv:1804.02767](https://arxiv.org/abs/1804.02767)
- Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: *IEEE CVPR* (2017)
- Jocher, G., Chaurasia, A., Qiu, J.: Yolo by ultralytics (2023). <https://github.com/ultralytics/ultralytics>
- Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: Panet: few-shot image semantic segmentation with prototype alignment. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9197–9206 (2019)
- Tong, K., Wu, Y.: Small object detection using deep feature learning and feature fusion network. *Eng. Appl. Artif. Intell.* **132**, 107931 (2024)
- Kim, S., Hong, S.H., Kim, H., Lee, M., Hwang, S.: Small object detection (sod) system for comprehensive construction site safety monitoring. *Autom. Constr.* **156**, 105103 (2023)
- Ji, S.-J., Ling, Q.-H., Han, F.: An improved algorithm for small object detection based on yolo v4 and multi-scale contextual information. *Comput. Electr. Eng.* **105**, 108490 (2023)
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: *NeurIPS*, vol. 30 (2017)
- Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: hierarchical vision transformer using shifted windows. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022 (2021)
- Dai, Z., Cai, B., Lin, Y., Chen, J.: Up-detr: unsupervised pre-training for object detection with transformers. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1601–1610 (2021)
- Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: accelerate detr training by introducing query denoising. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13619–13627 (2022)
- Zhu, L., Wang, X., Ke, Z., Zhang, W., Lau, R.W.: Biformer: vision transformer with bi-level routing attention. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10323–10333 (2023)
- Xu, S., Gu, J., Hua, Y., Liu, Y.: Dktnet: dual-key transformer network for small object detection. *Neurocomputing* **525**, 29–41 (2023)
- Gao, J., Chen, M., Xu, C.: Vectorized evidential learning for weakly-supervised temporal action localization. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(12), 15949–15963 (2023). <https://doi.org/10.1109/TPAMI.2023.3311447>
- Gao, J., Xu, C.: Learning video moment retrieval without a single annotated video. *IEEE Trans. Circuits Syst. Video Technol.* **32**(3), 1646–1657 (2022). <https://doi.org/10.1109/TCSVT.2021.3075470>
- Gao, J., Zhang, T., Xu, C.: Learning to model relationships for zero-shot video classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(10), 3476–3491 (2021). <https://doi.org/10.1109/TPAMI.2020.2985708>
- Hu, Y., Gao, J., Dong, J., Fan, B., Liu, H.: Exploring rich semantics for open-set action recognition. *IEEE Trans. Multimed.* **26**, 5410–5421 (2024). <https://doi.org/10.1109/TMM.2023.3333206>
- Jocher, G., Chaurasia, A., Qiu, J.: Yolo by ultralytics (2022). <https://github.com/ultralytics/yolov5>
- Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., et al.: Yolov6: a single-stage object detection framework for industrial applications (2022). [arXiv:2209.02976](https://arxiv.org/abs/2209.02976)
- Yang, Z., Guan, Q., Zhao, K., Yang, J., Xu, X., Long, H., Tang, Y.: Multi-branch auxiliary fusion yolo with re-parameterization

- heterogeneous convolutional for accurate object detection (2024). [arXiv:2407.04381](https://arxiv.org/abs/2407.04381)
30. Cheng, T., Song, L., Ge, Y., Liu, W., Wang, X., Shan, Y.: Yolo-world: real-time open-vocabulary object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16901–16911 (2024)
  31. Wang, M., Sun, H., Shi, J., Liu, X., Cao, X., Zhang, L., Zhang, B.: Q-yolo: efficient inference for real-time object detection. In: Asian Conference on Pattern Recognition, pp. 307–321 (2023). Springer
  32. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020). Springer
  33. Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L.M., Shum, H.-Y.: Dino: detr with improved denoising anchor boxes for end-to-end object detection (2022). [arXiv:2203.03605](https://arxiv.org/abs/2203.03605)
  34. Tan, M., Pang, R., Le, Q.V.: Efficientdet: scalable and efficient object detection. In: IEEE CVPR (2020)
  35. Tan, Z., Wang, J., Sun, X., Lin, M., Li, H., et al.: Giraffedet: a heavy-neck paradigm for object detection. In: ICLR (2021)
  36. Chen, J., Kao, S.-h., He, H., Zhuo, W., Wen, S., Lee, C.-H., Chan, S.-H.G.: Run, don't walk: chasing higher flops for faster neural networks. In: IEEE CVPR, pp. 12021–12031 (2023)
  37. Nascimento, M.G.d., Fawcett, R., Prisacariu, V.A.: Dsconv: efficient convolution operator. In: IEEE CVPR (2019)
  38. Cao, J., Li, Y., Sun, M., Chen, Y., Lischinski, D., Cohen-Or, D., Chen, B., Tu, C.: Do-conv: depthwise over-parameterized convolutional layer. In: IEEE TIP, vol. 31 (2022)
  39. Du, D., Zhu, P., Wen, L., Bian, X., Lin, H., Hu, Q., Peng, T., Zheng, J., Wang, X., Zhang, Y., et al.: Visdrone-det2019: the vision meets drone object detection in image challenge results. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (2019)
  40. Yu, X., Gong, Y., Jiang, N., Ye, Q., Han, Z.: Scale match for tiny person detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1257–1265 (2020)
  41. Everingham, M.: The pascal visual object classes challenge 2007 (2009). <http://www.Pascal-network.org/challenges/VOC/voc2007/workshop/index.html>
  42. Everingham, M., Winn, J.: The pascal visual object classes challenge 2012 (voc2012) development kit. pattern analysis, statistical modelling and computational learning. Tech. Rep. **8**(5), 2–5 (2011)
  43. Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., Ding, G.: Yolov10: real-time end-to-end object detection (2024). [arXiv:2405.14458](https://arxiv.org/abs/2405.14458)

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

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.