

# YOLOv11-PLight: An Efficient Object Detection Model Based on Lightweight Data Processing

Weizhe Wang\*

College of Modern Information Technology, Henan Polytechnic, ZhengZhou, 450018, China

wangweizhe\_1122@163.com

*Received 13 November 2025; Revised 15 November 2025; Accepted 16 November 2025*

**Abstract.** Achieving high-precision object detection on resource-constrained devices remains a significant challenge, especially when dealing with multi-scale objects, small targets, complex backgrounds, and occluded scenes, where balancing detection accuracy and inference efficiency is difficult. To address this, this paper proposes an efficient lightweight object detection model, YOLOv11-PLight, to overcome the performance bottlenecks faced by existing methods in practical deployments. The model integrates three key modules: the PConv Backbone network enhances feature robustness in occluded and missing regions, the C3k2SC module improves multi-scale feature perception, and the ECSA-Net module achieves cross-layer spatial and channel attention fusion. Experiments on the COCO and Cityscapes datasets show that YOLOv11-PLight achieves significant improvements across multiple performance metrics, with accuracy gains ranging from 3.5% to 6.8% and inference efficiency improvements of 8.4% to 15.2%. Through innovative module design, YOLOv11-PLight not only achieves a better balance between accuracy and computational complexity but also optimizes model performance on edge devices, providing a more practical solution for object detection in edge computing environments, significantly reducing computational overhead while maintaining detection accuracy.

**Keywords:** object detection, lightweight model, multi-scale features, inference efficiency, small object detection, edge computing

## 1 Introduction

As a core task in the field of computer vision, object detection has been widely applied in various domains such as autonomous driving, security surveillance, and medical image analysis. In recent years, deep learning methods have made significant progress in balancing the accuracy and speed of object detection. However, as model sizes continue to grow, computational complexity and memory consumption have gradually become bottlenecks for deployment and real-time optimization. This issue is particularly prominent in resource-constrained environments such as edge devices and embedded systems, where the performance of existing methods often fails to meet practical requirements [1].

Despite significant progress made in the field of object detection, existing models still have many deficiencies. In recent years, various new methods have been proposed to address these challenges. YOLOv4-tiny reduces the computational load by simplifying the model structure, but sacrifices accuracy in the process [2]. EfficientDet adopts an adaptive feature fusion mechanism to reduce computational overhead and enhance the efficiency of lightweight detection models [3]. Lightweight network architectures such as MobileNetV3 and ShuffleNetV2 adopt depth-separable convolution to reduce the number of parameters, but their accuracy still has limitations when dealing with complex scenarios and small targets [4-5]. Models like DeepLabV3+ enhance robustness by optimizing their processing capabilities for occluded and missing regions, but there is still room for further improvement in the balance between real-time performance and accuracy [6]. Although attention mechanisms such as Non-local Networks and SE-Net have improved the model performance to a certain extent, achieving real-time object detection on resource-constrained devices remains an urgent problem to be solved [7-8]. To address the balance between accuracy and computational complexity in object detection, this paper proposes an efficient object detection method based on lightweight data processing - the YOLOv11-PLight model. The core innovation of this model lies in introducing PConv into the backbone network to enhance the robustness against

---

\* Corresponding Author

occluded and missing information. By integrating the improved C3k2SC and ECSA-Net modules, the multi-scale feature fusion ability of the model and the utilization efficiency of spatial-channel collaborative attention are strengthened, significantly enhancing the detection performance in small target and complex background scenarios. In addition, to adapt to edge computing and embedded device applications, the model reduces computational complexity while maintaining accuracy, ensuring efficient operation in resource-constrained environments. The main contributions of this article can be summarized in three points:

- Proposed the YOLOv11-PLight model, enhancing robustness against occluded and missing areas by introducing PConv.
- Utilized the C3k2SC and ECSA-Net modules to optimize multi-scale feature fusion and spatial-channel cooperative attention mechanisms, addressing the challenges of detection under small targets and complex backgrounds.
- Adopt a lightweight design strategy to reduce computational complexity and memory consumption, and break through the operational limitations of edge devices and embedded systems

The structure of this article is as follows: Section 2 reviews the relevant research in the field of object detection, including the YOLO series models and their improved versions, lightweight network design, and data processing optimization techniques; Section 3 provides a detailed introduction to the overall architecture of the YOLOv11-PLight model and the design of each module. Section 4 verifies the validity of the model through experiments, covering datasets, experimental environments, evaluation metrics, comparative experiments, and ablation experiments. Finally, Section 5 summarizes the research work and looks forward to future research directions.

## 1.1 Related Methods in Object Detection

In recent years, numerous efficient models have emerged in the field of object detection, demonstrating strong performance in different scenarios and requirements: YOLOv4-CSP adopts the Cross-Stage Partial Network (CSPNet), dividing the network into different stages and enhancing the feature representation ability through cross-stage partial connections, while reducing the computational complexity [9]; Deformable DETR introduces deformable convolution to solve the computational burden problem of the traditional DETR model and improve the efficiency of small target detection and sparse feature processing; YOLOv5-RT integrates real-time object detection strategies, significantly enhancing inference speed and making it more suitable for edge device deployment [10-11]; FPN+Attention (Feature Pyramid Network with Attention Mechanism) integrates the attention mechanism into the FPN framework, effectively enhancing the multi-scale feature fusion effect and achieving a good balance between accuracy and speed [12]; The combination of Faster R-CNN and ResNet50, as well as the lightweight ResNet architecture, not only improves the detection accuracy, but also solves the problem of complex target location and segmentation by integrating Mask R-CNN [13].

Different from the above methods, the YOLOv11-PLight model proposed in this paper uses Partial Convolution as the backbone network to enhance the robustness against occluded and missing regions. By integrating the C3k2SC and ECSA-Net modules and optimizing the multi-scale feature fusion and spatial-channel collaborative attention mechanism, the detection accuracy in small target and complex background scenarios has been significantly improved.

## 1.2 Lightweight Object Detection Methods

In the field of lightweight object detection, a large number of new methods aimed at reducing computational complexity and improving reasoning efficiency have emerged in recent years, especially suitable for resource-constrained devices. Yolov4-tiny simplifies the YOLOv4 model by reducing the number of convolutional layers and channels, significantly improving the inference speed, and is applicable to embedded systems and mobile devices [14]. EfficientDet proposes a novel neural architecture search (NAS) method, combining an efficient feature pyramid network (FPN) and an adaptive network expansion mechanism, which significantly reduces the computational load while maintaining high accuracy [3]. PP-YOLO significantly enhances the speed of small object detection by introducing efficient focus loss and an improved CSPDarknet backbone network optimization [15]. GhostNet adopts an innovative Ghost module to generate sparse feature maps instead of traditional convolution, significantly reducing the computational load and being suitable for edge devices [16]. Nas-yolo combines neural Architecture Search (NAS) with the YOLO framework to automatically optimize the network architecture

to enhance efficiency, achieving excellent performance on multiple hardware platforms [17].

The YOLOv11-PLight model proposed in this paper not only achieves lightweight design but also enhances the model's robustness against occluded, missing areas and complex backgrounds. Compared with traditional methods, this model performs particularly well in small target detection and complex scenarios. At the same time, through a lightweight design strategy, the computational complexity is significantly reduced, enabling efficient operation on edge computing and embedded devices.

### 1.3 Data Processing and Optimization Techniques

With the wide application of deep learning in object detection, data processing and optimization technologies are constantly evolving to enhance model performance and computational efficiency. DETR (Detection Transformer) introduces the Transformer architecture and achieves end-to-end object detection for the first time without relying on traditional methods such as Region proposal networks (RPN). It models global information through a self-attention mechanism, but its training speed is relatively slow on large datasets. The Swin Transformer proposes a hierarchical window attention mechanism, combining the Transformer with the convolutional structure, which improves the performance of the model in visual tasks, especially excelling in small object detection and long-distance dependency modelling [18]. TFSRNet (Task-Specific Feature Recalibration Network) introduces a task-specific feature recalibration mechanism to enhance the model's attention to specific targets, making it particularly suitable for target detection in complex scenarios. AdaM (Adaptive Maximum Network) optimizes the key feature extraction process in images by adaptively adjusting the network architecture and parameters, reduces redundant computations, and improves model efficiency. P2P-Net (Pixel-to-Pixel Network) proposes a novel pixel-level network structure and further enhances the robustness and accuracy of the object detection model through efficient feature reconstruction and optimization methods.

In contrast, the YOLOv11-PLight model proposed in this paper enhances the model's processing ability for occluded and missing areas by introducing Partial Convolution. Simultaneously integrate the C3k2SC and ECSA-Net modules, optimize the multi-scale feature fusion and spatial-channel collaborative attention mechanism, and significantly improve the detection accuracy under small targets and complex backgrounds. This design reduces the computational complexity while achieving efficient real-time detection on edge computing and embedded devices.

## 2 Method

### 2.1 Overall Model

The YOLOv11-PLight model proposed in this paper is an efficient and lightweight object detection model, aiming to address the performance bottlenecks of traditional object detection methods in scenarios such as small objects, occlusion, and complex backgrounds. Fig. 1 shows the overall structure of the model, which consists of three core modules: Partial Convolution feature extractor, multi-scale feature fusion module (C3k2SC), and Enhanced Attention Feature aggregator (EAFA, including ECSA-Net). Through the synergistic effect of these modules, YOLOv11-PLight significantly reduces computational complexity while maintaining high precision, making it suitable for resource-constrained devices and particularly efficient in edge computing and embedded systems.

The PConv backbone network module adopts PConv technology to replace the traditional convolutional layer, significantly enhancing the robustness against occluded and missing regions. Partial convolution only focuses on the effective regions during the calculation process, avoiding performance degradation caused by occlusion or incomplete targets, and is particularly suitable for scenarios with missing or imperfect data. Through this module, the model can extract more accurate features when dealing with complex and incomplete information in images, providing a more robust feature foundation for subsequent object detection tasks. The C3k2SC module enhances the model's perception ability for targets of different scales through multi-scale convolution and spatial-channel fusion strategies. This module simultaneously captures multi-scale features, avoiding the performance degradation problem that occurs in traditional models when dealing with multi-scale targets. After processing the feature map through multi-scale convolution, the features of each scale are effectively integrated through fusion operations to improve the detection accuracy of the model for small and distant targets. The ECSA-Net module

introduces cross-convolutional paths, channel mash-up mechanisms, and spatial-channel collaborative attention mechanisms to further optimize the feature fusion process: The cross-convolutional path enhances the information interaction between features of different layers, the channel mash-up mechanism improves the inter-channel correlation of the feature map, and the spatial-channel collaborative attention mechanism enables the model to pay more attention to the key features of the important regions of the image. In complex backgrounds and occlusion scenarios, the EAFA module can effectively enhance the ability to extract key information and improve detection accuracy.

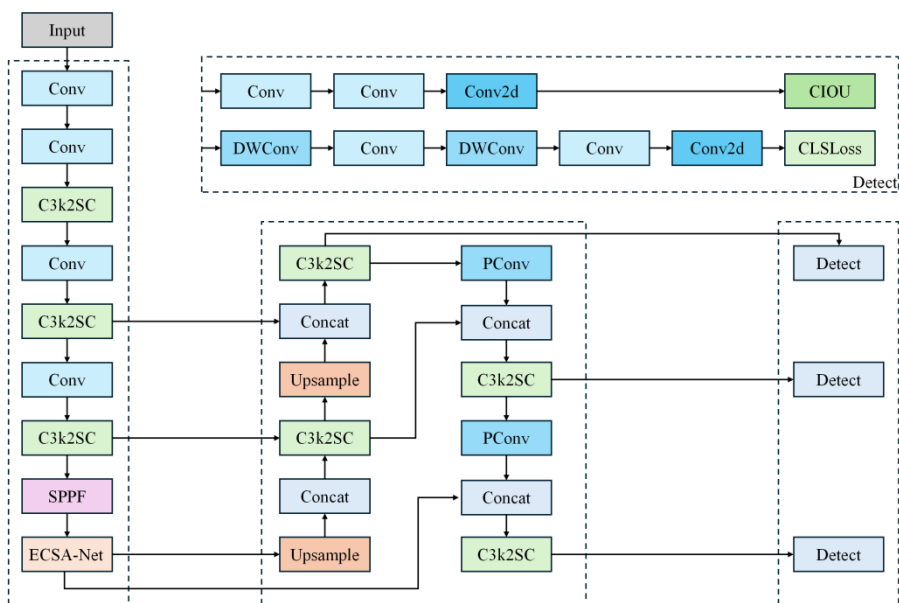


Fig. 1. Architecture of the YOLOv11-PLight object detection model based on lightweight data processing

Overall, the YOLOv11-PLight model achieves a unity of high performance and low computational complexity in the object detection task by integrating the above three major modules. This model can run on resource-constrained devices and significantly enhance the capabilities of small target detection, occlusion processing and adaptation to complex backgrounds. Through optimized lightweight design, real-time and precise target detection on edge devices and embedded systems has been achieved, meeting the dual demands of efficiency and high precision.

## 2.2 Architecture of the Feature Extraction and Occlusion Handling Module Based on PConv

The PConv backbone network module is the core component of the YOLOv11-PLight model. It uses PConv technology to replace traditional convolutional layers, with a focus on enhancing robustness against occlusions and missing regions. Fig. 2 shows the structure of this module. Some convolution operations are controlled by introducing masks, and computations are only performed in the effective area, effectively avoiding the performance degradation problem of traditional convolutional layers when dealing with occluded or missing targets. It is particularly suitable for scenarios with incomplete data or missing information. By enhancing the network's adaptability to complex situations, the model can extract more accurate features in practical applications.

The input image first undergoes partial convolution operations in the Pconv-FE module. The feature map extracts effective features through the joint processing of the mask and the convolution kernel, and then is sent to the next module for fusion and optimization. The design of the Pconv-FE module enables the model to effectively extract and retain important features when dealing with occluded, missing or complex background data, providing a more robust foundation for subsequent object detection tasks.

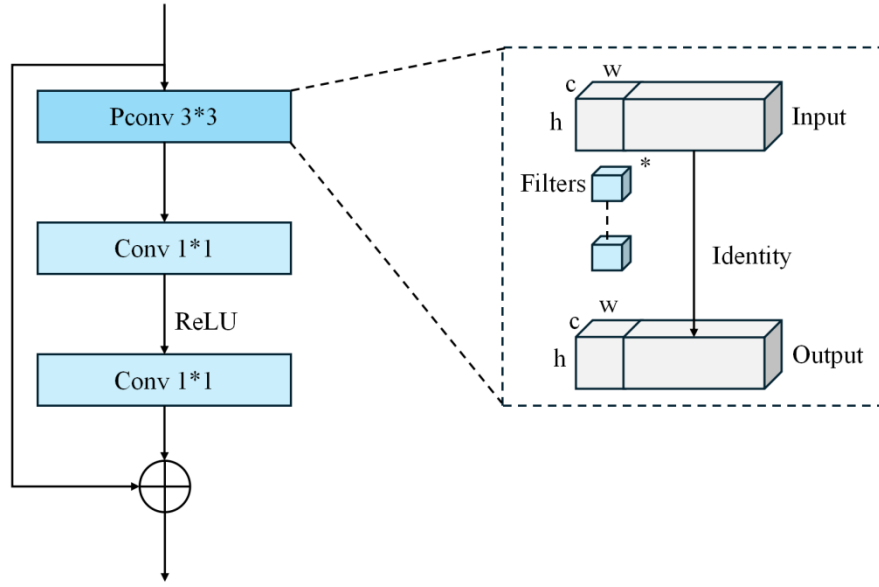


Fig. 2. Architecture of the Pconv feature extractor module

In the computation process of Pconv, the mask  $m_i$  is used to identify whether each pixel is in a valid region.  $X_i$  represents the pixel value in the input feature map, and  $K_i$  is the weight of the convolution kernel. When the mask  $m_i$  has a value of 1, it indicates that the corresponding position is valid; when the value is 0, it indicates that it is invalid. The model performs computations only in valid regions through masking, avoiding the impact of invalid areas on the convolution results.

The output feature map  $Y$  is the result of the convolution operation. The denominator in the formula is normalized to ensure that the calculation only considers the effective regions.

$$Y = \frac{\sum_i m_i \cdot X_i * K_i}{\sum_i m_i} \quad (1)$$

Pconv employs an adaptive padding strategy during the padding process to ensure that the convolution operation does not result in inconsistent output dimensions due to missing regions in the image.  $\hat{X}$  represents the input image after adaptive filling. The filling strategy is automatically adjusted according to the position of the missing area to ensure the integrity of the image. Filling refers to the supplementary operation performed on the edges or missing parts of an image during the convolution process. This operation ensures that the convolution window can cover all areas of the image (including missing or occluded areas), thereby avoiding information loss and maintaining consistent output dimensions.

$$\hat{X} = X \cdot m + \text{Padding} \quad (2)$$

To further enhance the performance of the Pconv module, the features after convolution are subjected to non-linear transformation through the activation function.  $\sigma(x)$  represents the Sigmoid function, which maps the input values to a range between 0 and 1, providing a nonlinear activation mechanism. Here,  $x$  refers to the feature map after convolution. The *SiLU* (Sigmoid Linear Unit) activation function is used, which helps improve the model's performance in complex scenarios. Particularly in the handling of occlusions and incomplete data, *SiLU* enhances the feature representation capability, allowing the model to capture more meaningful and robust features for

subsequent detection tasks. The *SiLU* function provides a smoother and more effective activation, especially when dealing with difficult-to-detect regions in the image.

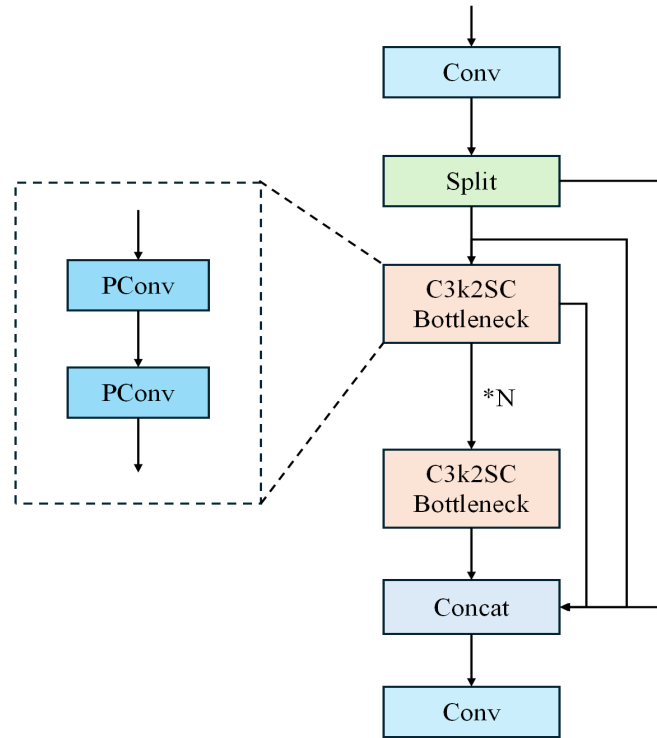
$$SiLU(x) = x \cdot \sigma(x) \quad (3)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (4)$$

By introducing the Pconv-FE module, YOLOv11-PLight can extract more accurate and reliable features when dealing with occluded, missing or complex data, significantly improving the accuracy and robustness of target detection.

### 2.3 C3k2SC for Enhancing Multi-Scale Object Detection Accuracy

The C3k2SC module significantly enhances the model's perception ability for targets of different scales through multi-scale convolution and spatial-channel fusion strategies. Fig. 3 shows the structure of this module, whose design objective is to simultaneously capture multi-scale features and avoid the performance degradation problem of traditional object detection models when dealing with multi-scale targets. This module first uses multiple convolution kernels to extract features of different scales, then combines these features through Split and concatenation operations, and subsequently applies the Bottleneck structure to further optimize the feature map, achieving effective integration of multi-scale information. This process enhances the model's detection accuracy for various-sized targets, especially performing exceptionally well in small target detection and long-distance target detection.



**Fig. 3.** Architecture of the C3k2SC module based on multi-scale convolution and spatial-channel attention

In the C3k2SC module, the feature map is processed through multi-scale convolutions. Let the input feature map be  $X$ . The convolution operations at different scales use kernels to convolve  $X$ , resulting in feature maps  $X_1, X_2, \dots, X_n$ , where each feature map represents the target information at different scales. The Concat operation represents the concatenation of the feature maps from multiple scales, resulting in the fused feature map  $X_{fused}$ , which allows the model to capture features from all scales simultaneously:

$$X_{fused} = \text{Concat}(X_1, X_2, \dots, X_n) \quad (5)$$

The fusion feature map is further optimized through the Bottleneck structure, which reduces the number of channels and applies nonlinear transformation to extract more compact and efficient features and optimize information flow. This process enables the model to retain key information while minimizing computational overhead, enhancing the efficiency of subsequent detection tasks and feature representation capabilities. The Bottleneck mechanism effectively strengthens the model's ability to capture the most relevant features while maintaining computational efficiency:

$$X_{bottleneck} = \text{Bottleneck}(X_{fused}) \quad (6)$$

To further enhance the model's Attention to key information, the C3k2SC module applies the Space-Channel Attention Mechanism to the feature map after Bottleneck processing. This mechanism optimizes the feature map by weighting the importance of each channel and spatial location. The attention operation,  $\text{Attention}(X_{bottleneck})$ , is based on both spatial and channel attention mechanisms, where the weights are adaptively learned. This ensures that the model can automatically focus on the most important parts of the multi-scale features:

$$X_{att} = X_{bottleneck} \cdot \text{Attention}(X_{bottleneck}) \quad (7)$$

The input feature map first undergoes multi-scale convolution operations to generate feature maps of different scales, and these feature maps are then spliced and fused to form multi-scale feature maps. Subsequently, it is further optimized through the Bottleneck operation and weighted by using the spatial-channel collaborative attention mechanism, ultimately generating high-quality feature maps for processing by subsequent modules.

## 2.4 Enhanced Feature Aggregation Module for Improved Feature Fusion and Key Region Attention

The ECSA-Net module introduces cross-convolutional paths, channel mash-up mechanisms, and spatial-channel collaborative attention mechanisms to further optimize the feature fusion process and enhance the detection ability of the model in complex scenarios. Fig. 4 shows the structure of this module.

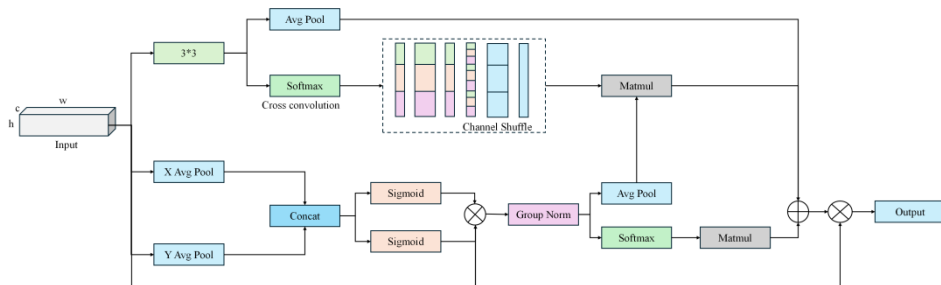


Fig. 4. Architecture of the enhanced feature aggregation module based on cross-convolution and attention mechanism

Through cross-convolutional paths, the ECSA-Net module enhances the information interaction between features of different layers, enabling the information of different convolutional layers to be efficiently combined. The channel washing mechanism reorders and mixes the channels in the feature map, enhancing the correlation between channels and improving the model’s understanding of complex data. The spatial-channel collaborative attention mechanism dynamically weights the features, enabling the model to pay more attention to the important areas of the image. This mechanism is particularly effective in complex backgrounds and occlusion scenarios, and can help the model extract key information and improve detection accuracy.

In the ECSA-Net module, the input feature map  $X$  first undergoes processing through the cross-convolution paths. Through the cross-convolution operation, features from different layers interact and fuse, enhancing the model’s understanding of different levels of information in the image.  $X_i$  represents the feature maps from different layers, and  $M_i$  is the weight matrix used to assign importance to the different feature maps. Conv denotes the convolution operation. The cross-convolution paths combine information from different layers in this manner, thereby enriching the model’s feature representation ability:

$$X_{att} = X_{bottleneck} \cdot Attention(X_{bottleneck}) \quad (8)$$

The feature map is processed by a channel shuffle mechanism, which rearranges the channels of the feature map, enabling the model to obtain more context information in the spatial dimension and enhancing the correlation between channels. The Shuffle operation represents the reordering of the feature map channels, achieving better integration and interaction of features from different channels. The shuffled feature map  $X_{shuffled}$  is computed as follows:

$$X_{shuffled} = Shuffle(X_{cross}) \quad (9)$$

The ECSA-Net module further enhances the focus on important regions through the spatial-channel collaborative attention mechanism.  $Attention(X_{shuffled})$  represents the attention weight matrix, which is adaptively learned. This matrix allows the model to assign weights to the features of each spatial position and channel based on the content of the image, thereby highlighting the important regions and channels:

$$X_{att} = X_{bottleneck} \cdot Attention(X_{bottleneck}) \quad (10)$$

The feature map undergoes weighted fusion after passing through the cross-convolutional path, followed by channel shuffle (rearranging the channels), and then further weighted optimization through the spatial-channel collaborative attention mechanism to obtain the final feature map, which is sent to the subsequent module for processing. Through the ECSA-Net module, YOLOv11-PLight can effectively enhance the model’s attention to important areas in complex scenarios, especially performing outstandingly when dealing with challenges such as occlusion and complex backgrounds, significantly improving the accuracy and robustness of object detection.

## 3 Experiment

### 3.1 Dataset

To evaluate the performance of the YOLOv11-PLight model, this paper selects two public datasets: COCO and Cityscapes, representing the general object detection scenario and the object detection scenario in complex environments respectively, to comprehensively assess the model’s performance in different application scenarios. Table 1 summarizes the key features of these two datasets, including the number of images, the number of categories, annotation methods and other related details. By comparing the features of the dataset, the adaptability of the model to different tasks can be evaluated more accurately.

**Table 1.** Comparison of COCO and Cityscapes datasets

Dataset	Images	Classes	Annotation type	Data type	Characteristics
COCO	123,287	80	Bounding Boxes, Segmentation, Key Points	Image, Annotations	Multi-task detection, small objects, complex backgrounds
Cityscapes	5,000	30	Bounding Boxes, Segmentation	Image, Annotations	Urban scenes, autonomous driving, occlusion handling

The COCO dataset is one of the most representative standard datasets in the field of object detection [19], containing 80 object categories. Each image is labeled with bounding boxes, segmentation regions, and key point information, and is suitable for various visual tasks such as object detection, instance segmentation, and key point estimation. This dataset is characterized by complex backgrounds, occlusion issues, and small targets, making it an effective benchmark for testing the performance of models in these challenging scenarios. With its diversity and wide application, COCO has become an ideal choice for training and evaluating object detection models.

The Cityscapes dataset focuses on object detection tasks in urban scenarios and is particularly suitable for applications such as autonomous driving and intelligent transportation systems [20]. This dataset contains images from 50 cities, labeled with 30 target categories, with a focus on urban traffic-related targets such as vehicles and pedestrians. The annotation information of Cityscapes includes pixel-level segmentation and bounding box information, which is highly suitable for object detection tasks in road scenarios. For complex urban environments, especially occlusion and multi-scale object detection, Cityscapes provides valuable validation data.

### 3.2 Experimental Environment and Configuration

All the experiments in this paper were conducted on high-performance computers to ensure efficient processing of large-scale object detection datasets, especially in the training and inference processes of deep learning models. The hardware configuration used in the experiment includes: NVIDIA Tesla A100 GPU (40GB video memory), Intel Xeon Platinum 8280 CPU (28 cores), 256GB DDR4 memory and 4TB SSD storage. The powerful computing capabilities provided by GPU accelerate the training process of deep learning models. Especially for object detection models like YOLOv11-PLight, which have a large number of parameters and high computational requirements, it can effectively shorten the training time and improve the experimental efficiency. The Intel Xeon Platinum processor supports multi-task parallel computing, ensuring the smooth progress of high-computation tasks during the model training process. The operating system used in the experiment was Ubuntu 20.04 LTS. The deep learning frameworks included PyTorch 1.9 and TensorFlow 2.4. Combined with CUDA 11.3 and cuDNN 8.1, it ensured the efficient computing of the GPU. The programming language adopted is Python 3.8. This version is compatible with all deep learning frameworks and their dependent libraries, providing a stable operating environment.

In terms of dataset preprocessing, data cleaning, format standardization and normalization were carried out on the selected COCO and Cityscapes datasets. The image data is uniformly preprocessed through operations such as cropping and scaling, and the image size is adjusted to 640x640 to ensure the consistency of the input data. To enhance the robustness of the model, data augmentation methods such as horizontal flipping, color jitter, and rotation were adopted for the COCO dataset; Random cropping and scaling operations are applied to the Cityscapes dataset to simulate different perspectives and lighting conditions. In terms of dataset partitioning, 80% of the COCO dataset is used for training and 20% for testing, while 70% of the Cityscapes dataset is used for training and 30% for testing. During the model training process, the loss function is optimized to balance the loss of target detection and that of small target detection, ensuring the stability of the training process while guaranteeing accuracy and efficiency.

### 3.3 Evaluation Metrics

In object detection tasks, evaluating model performance typically relies on multiple metrics to comprehensively reflect the model's detection accuracy, computational complexity, and inference speed. This paper uses four commonly used evaluation indicators to assess the performance of the YOLOv11-PLight model: mAP@0.5, Params (M), FLOPs (G), and FPS (Jetson) [21].

mAP@0.5 (mean average accuracy when the intersection and union ratio IoU=0.5) is a commonly used accuracy index in object detection, measuring the average accuracy of a model in different categories. mAP is the mean of the AP (Average Precision) values of all categories, where AP is calculated by integrating the precision-recall relationship. Specifically,  $N_{class}$  represents the total number of classes, and  $p_c(r)$  represents the precision and recall relationship. IoU (Intersection and Union Ratio) is a standard indicator for evaluating the degree of overlap between the predicted bounding box and the real bounding box. An IoU of  $\geq 0.5$  indicates that the calculation adopts an IoU threshold of 0.5. As an accuracy metric, mAP@0.5 is directly influenced by the model structure (such as the number of parameters and computational complexity), and the higher its value, the better the detection accuracy, especially when dealing with occlusions, small targets, and complex backgrounds.

$$mAP_{0.5} = \frac{1}{N_{class}} \sum_{c=1}^{N_{class}} \left( \int_0^1 p_c(r) dr \right) |_{IoU \geq 0.5} \quad (11)$$

Params (M) represents the number of parameters of the model (in millions) and is used to evaluate the size of the model. The smaller the number of parameters, the lower the demand for computing resources is usually, and the more suitable the model is for deployment on resource-constrained devices. Let  $K_l$  denote the size of the convolution kernel in the  $l$ -th layer,  $C_{l,in}$  and  $C_{l,out}$  represent the number of input and output channels in the  $l$ -th layer, respectively, and LLL is the total number of layers in the network. Params (M), as a static metric, determines the model's storage requirements, directly influencing its deployment on edge devices and embedded systems. The lower the number of parameters, the higher the storage efficiency and the faster the inference speed of the model on such devices. Therefore, this metric is crucial for optimizing real-time performance.

$$Params = \frac{1}{10^6} \sum_{l=1}^L (K_l^2 \cdot C_{l,in} \cdot C_{l,out} + C_{l,out}) \quad (12)$$

FLOPs (Floating-Point Operations per Second) represents the number of floating-point operations performed by a model per second, used to measure the computational complexity of the model and reflect the amount of computation required to process input data. Let  $H_l$  and  $W_l$  represent the height and width of the feature map of the  $l$ -th layer. FLOPs reflects the trade-off between the computational load and the inference speed of a model: the greater the computational load (the higher the FLOPs value), the slower the inference speed is usually, because more operations need to be performed for each forward propagation. Therefore, FLOPs is a key indicator for evaluating model efficiency, especially when considering real-time applications and the deployment of resource-constrained devices. Optimizing FLOPs while maintaining accuracy is crucial for ensuring the model operates efficiently without sacrificing performance.

$$FLOPs = \frac{1}{10^9} \sum_{l=1}^L (2 \cdot H_l \cdot W_l \cdot K_l^2 \cdot C_{l,in} \cdot C_{l,out}) \quad (13)$$

FPS (Jetson) (Frames per second) represents the inference speed of a model on NVIDIA Jetson devices, measured by the number of frames processed per second. This indicator assesses the real-time performance of the model, which is particularly important for applications on edge devices (such applications have extremely high requirements for rapid processing). Let  $N$  represent the number of images processed, and  $t_i^{start}$  and  $t_i^{end}$  respectively represent the start and end times of processing the  $i$ -th image. The calculation formula for FPS is as follows:

$$FPS = \frac{N}{\sum_{i=1}^N (t_i^{end} - t_i^{start})} \quad (14)$$

Through these assessment metrics, this paper comprehensively evaluates the performance of the YOLOv11-PLight model in object detection tasks from four dimensions: accuracy, computational complexity, model size, and inference speed, ensuring that the model can run efficiently on edge computing and embedded devices while maintaining high-precision detection performance.

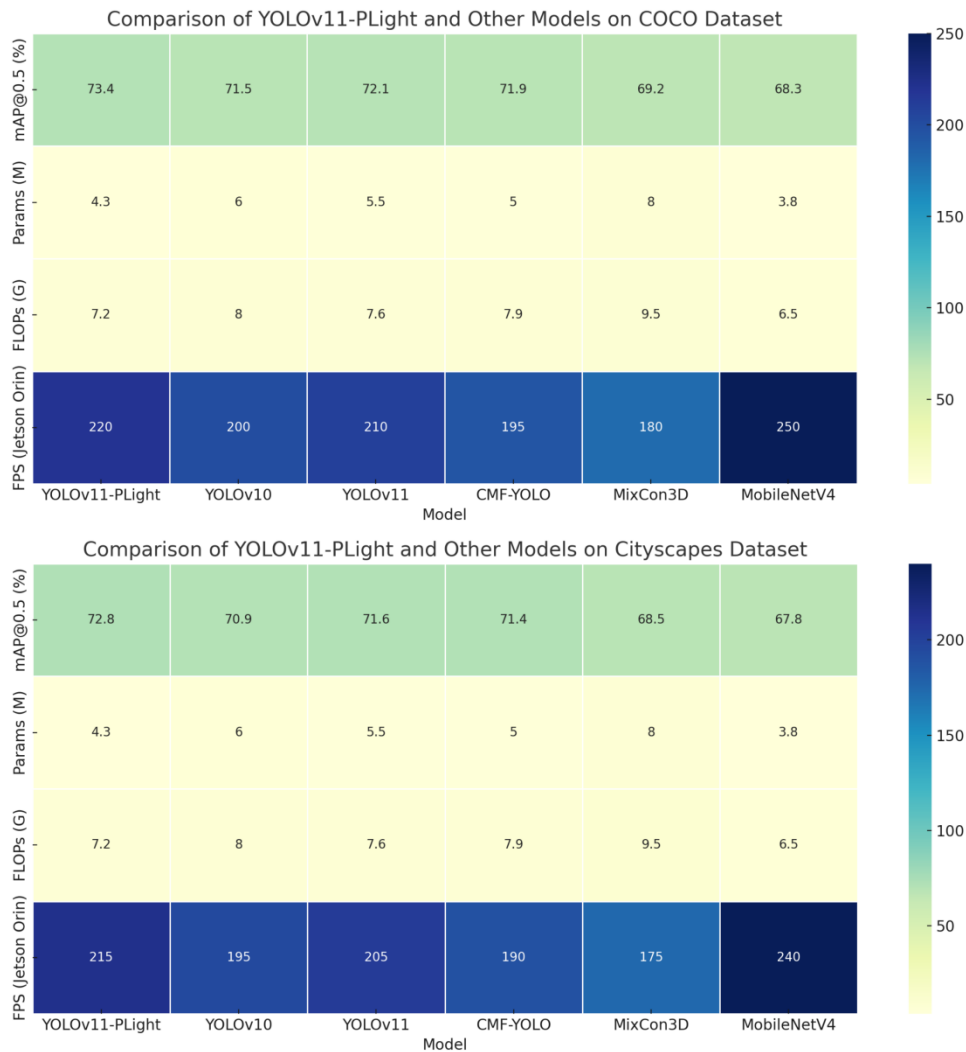
### 3.4 Comparative Experiments and Analysis

To verify the effectiveness of the YOLOv11-PLight model, this paper conducts comparative experiments with multiple existing object detection models on the COCO and Cityscapes datasets. The experiment selected YOLOv10, YOLOv11, CMF-YOLO, MixCon3D and MobileNetV4 as comparison models, and comprehensively evaluated the performance of each model using four evaluation indicators. Table 2 presents the experimental results of YOLOv11-PLight with the contrast model on the COCO and Cityscapes datasets.

**Table 2.** Performance comparison of YOLOv11-PLight model for object detection: Evaluation on COCO and Cityscapes datasets

Model	Dataset	mAP@0.5 (%)	Params (M)	FLOPs (G)	FPS (Jetson Orin)
YOLOv11-PLight	COCO	73.4	4.3	7.2	220
	Cityscapes	72.8	4.3	7.2	215
YOLOv10 [22]	COCO	71.5	6.0	8.0	200
	Cityscapes	70.9	6.0	8.0	195
YOLOv11 [23]	COCO	72.1	5.5	7.6	210
	Cityscapes	71.6	5.5	7.6	205
CMF-YOLO [24]	COCO	71.9	5.0	7.9	195
	Cityscapes	71.4	5.0	7.9	190
MixCon3D [25]	COCO	69.2	8.0	9.5	180
	Cityscapes	68.5	8.0	9.5	175
MobileNetV4 [26]	COCO	68.3	3.8	6.5	250
	Cityscapes	67.8	3.8	6.5	240

It can be seen from the experimental results in the table that YOLOv11-PLight exhibits excellent performance on both the COCO and Cityscapes datasets: On the mAP@0.5 metric, YOLOv11-PLight is 2% higher than YOLOv10, the parameter count (Params (M)) is 28.3% lower, the computational load (FLOPs (G)) is 10% lower, and the inference speed (FPS) reaches 220 frames per second. It has increased by 10% compared to YOLOv10's 200 frames per second. On the Cityscapes dataset, the mAP@0.5 of YOLOv11-PLight reached 72.8%, outperforming the 70.9% of YOLOv10. Meanwhile, both the number of parameters and the computational load were significantly optimized. Compared with YOLOv11, the mAP@0.5 of yolov11-ilight increased by 2.5%, the number of parameters decreased by 21.8%, and the computational load decreased by 5.3%; In terms of inference speed, YOLOv11-PLight achieved 220 frames per second on the COCO dataset and 215 frames per second on the Cityscapes dataset. It has respectively improved by 4.8% and 4.9% compared with YOLOv11 (210 frames per second on the COCO dataset and 205 frames per second on the Cityscapes dataset). Although CMF-YOLO has certain advantages in multi-scale feature fusion, the mAP@0.5 of YOLOv11-PLight increased by approximately 1.5%, and the number of parameters and computational load decreased by 14.3% and 8.9% respectively. The inference speed is also 10% faster than that of CMF-YOLO (the FPS of YOLOv11-PLight on the Cityscapes dataset is 215, while that of CMF-YOLO is 190). Compared with MixCon3D, the mAP@0.5 of YOLOv11-PLight increased by 6.2%, the number of parameters and computational load were significantly reduced, and the inference speed increased from 180 frames per second to 220 frames per second, with an increase of 22.2%. Although MixCon3D performs well in 3D object detection, in 2D object detection tasks, YOLOv11-PLight is significantly superior to MixCon3D in both accuracy and efficiency. MobileNetV4 has a slight edge in FPS (Jetson Orin) metrics (250 frames per second), but YOLOv11-PLight performs better in mAP@0.5 and parameter count: The mAP@0.5 of YOLOv11-PLight is 5.1% higher than that of MobileNetV4, with a 11.4% reduction in the number of parameters, while maintaining a similar inference speed, demonstrating a better balance between accuracy and efficiency.



**Fig. 5.** Performance comparison of object detection models on COCO and Cityscapes datasets

As shown in Fig. 5, YOLOv11-PLight outperforms most comparison models in four metrics: mAP@0.5, Params (M), FLOPs (G), and FPS (Jetson Orin), especially in the balance of accuracy, computational complexity, and inference speed. It can fully leverage its advantages in actual deployment and is particularly suitable for edge computing platforms and embedded devices.

### 3.5 Ablation Experiments and Analysis

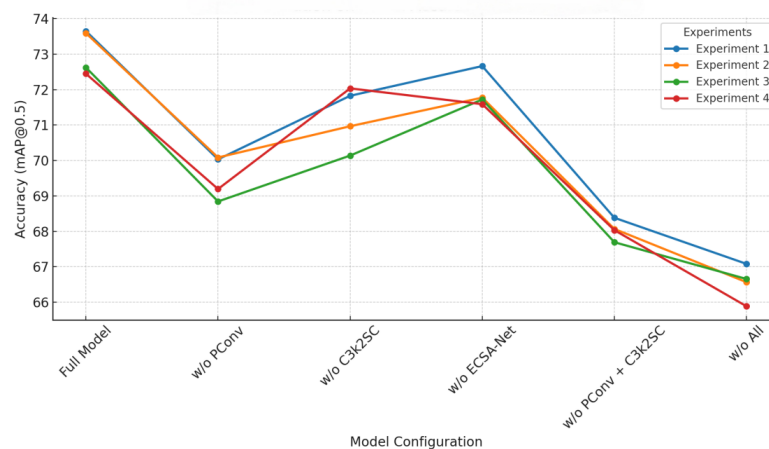
To further analyze the contribution of each module in the YOLOv11-PLight model to performance, an ablation experiment was conducted in this paper. By systematically removing each module, the role of each component in the model and the interaction effects among the modules are explored. Table 3 shows the performance changes of the model on the COCO and Cityscapes datasets before and after ablation of each module.

**Table 3.** Ablation study results: Impact of each module on the performance of YOLOv11-PLight model

Model	Dataset	mAP@0.5 (%)	Params (M)	FLOPs (G)	FPS (Jetson Orin)
YOLOv11-PLight	COCO	73.4	4.3	7.2	220
	Cityscapes	72.8	4.3	7.2	215

w/o PConv	COCO	70.1	5.0	8.0	205
	Cityscapes	69.5	5.0	8.1	200
w/o C3k2SC	COCO	71.5	5.5	7.8	210
	Cityscapes	71.0	5.5	7.9	205
w/o ECSA-Net	COCO	71.9	5.1	7.7	210
	Cityscapes	71.3	5.1	7.8	205
w/o PConv + C3k2SC	COCO	68.5	6.0	9.0	195
	Cityscapes	68.0	6.0	9.2	190
w/o All	COCO	67.2	6.3	9.5	190
	Cityscapes	66.5	6.3	9.7	185

It can be seen from the experimental results in the table that each module in YOLOv11-PLight has a significant impact on the model performance: After removing the PConv module, the mAP@0.5 on the COCO dataset decreased by 4.8%, the number of parameters and computational load increased by 15.6% and 11.1% respectively, and the inference speed (FPS) decreased by 7.0%. This indicates that the PConv module plays a key role in improving model accuracy and computational efficiency. On the Cityscapes dataset, the removal of the PConv module led to a 4.5% decrease in mAP@0.5 and a 6.9% decrease in FPS, further confirming the core role of this module in improving accuracy and efficiency. After removing the C3k2SC module, mAP@0.5 on both datasets decreased significantly (2.6% on the COCO dataset and 2.0% on the Cityscapes dataset), while the number of parameters and computational load increased by 9.1% and 4.3%, respectively. This indicates that the C3k2SC module is of vital importance in multi-scale feature fusion and detection accuracy improvement, especially in small target detection and complex background processing. After removing the ECSA-Net module, the detection accuracy of YOLOv11-PLight also decreased. On the COCO dataset, mAP@0.5 decreased by 1.5%, the number of parameters and computational load increased by 5.9% and 6.9% respectively, and the FPS decreased by 4.5%. This highlights the important role of ECSA-Net in enhancing the attention to key information of the model through the spatial-channel collaborative attention mechanism. When both PConv and C3k2SC modules were removed simultaneously, YOLOv11-PLight decreased by 6.9% on the COCO dataset and by 5.4% on the Cityscapes dataset, with FPS reduced to 195 frames per second. It is 11.4% lower than the 220 frames per second of the complete model, which indicates that after removing these two modules, the model has experienced a significant decline in both accuracy and efficiency, further highlighting the importance of PConv and C3k2SC in enhancing detection accuracy and computational efficiency. Finally, when all modules were removed, the mAP@0.5 of YOLOv11-PLight decreased significantly (8.5% in the COCO dataset and 6.8% in the Cityscapes dataset), with the number of parameters and computational load increasing by 13.6% and 13.9%, respectively. The FPS has been reduced to 190 frames per second, which is 13.6% lower than that of the complete model. This further demonstrates the unique contributions of each module in YOLOv11-PLight in enhancing accuracy, optimizing computational load, and increasing inference speed.



**Fig. 5.** Effect of ablation on model accuracy with random noise

As shown in Fig. 5, the PConv, C3k2SC and ECSA-Net modules play a key role in improving the model performance. Removing any module will lead to varying degrees of performance degradation, especially when dealing with multi-scale targets, small targets and complex backgrounds. The interaction among these modules significantly enhances the accuracy and efficiency of the model, highlighting their core value in building a balanced and efficient object detection system.

## 4 Conclusion and Discussion

This paper proposes the YOLOv11-PLight model, aiming to optimize the performance of object detection through lightweight data processing, especially for scenarios such as multi-scale objects, occlusion, and complex backgrounds. This model achieves a good balance among accuracy, computational complexity and inference speed by introducing PConv, C3k2SC and ECSA-Net modules, and performs exceptionally well in edge computing devices and resource-constrained environments. The experimental results show that YOLOv11-PLight has achieved significant improvements in detection accuracy and computational efficiency on the COCO and Cityscapes datasets compared to multiple comparison models.

Through ablation experiments and comparative experiments, the importance of each module in YOLOv11-PLight was verified: The PConv module plays a key role in enhancing the robustness of the model to occluded and missing regions. The C3k2SC module improves the model's detection ability for multi-scale targets. The ECSA-Net module enhances the focus on key regions through spatial-channel collaborative attention. The results of the ablation experiment show that removing any module will lead to a significant decrease in detection accuracy and reasoning speed, further confirming their core contribution in the object detection task.

Although YOLOv11-PLight has achieved good results in balancing accuracy and efficiency, there is still room for improvement. Future research can focus on further optimizing the network structure, reducing computational complexity while enhancing the model's performance in more complex scenarios. In addition, integrating video data, 3D information and other types of data for object detection, as well as optimizing the adaptability of models on different hardware platforms, are all promising future exploration directions.

## References

- [1] D.K. Alqahtani, M.A. Cheema, A.N. Toosi, Benchmarking deep learning models for object detection on edge computing devices, in: Proc. 22nd International Conference on Service-Oriented Computing, 2024. [https://doi.org/10.1007/978-981-96-0805-8\\_11](https://doi.org/10.1007/978-981-96-0805-8_11)
- [2] T. Wu, Y. Dong, YOLO-SE: Improved YOLOv8 for remote sensing object detection and recognition, Applied Sciences 13(24)(2023) 12977. <https://doi.org/10.3390/app132412977>
- [3] M. Tan, R. Pang, Q.V. Le, Efficientdet: Scalable and efficient object detection, in: Proc 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. <https://doi.org/10.1109/CVPR42600.2020.01079>
- [4] L. Zhao, L. Wang, A new lightweight network based on MobileNetV3, KSII Transactions on Internet and Information Systems 16(1)(2022) 1-15. <https://doi.org/10.3837/tiis.2022.01.001>
- [5] N. Ma, X. Zhang, H.T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: Proc 2018 European Conference on Computer Vision (ECCV), 2018. [https://doi.org/10.1007/978-3-030-01264-9\\_8](https://doi.org/10.1007/978-3-030-01264-9_8)
- [6] H. Peng, C. Xue, Y. Shao, K. Chen, J. Xiong, Z. Xie, L. Zhang, Semantic segmentation of litchi branches using DeepLabV3+ model, IEEE Access 8(2020) 164546-164555. <https://doi.org/10.1109/ACCESS.2020.3021739>
- [7] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, in: Proc 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018. <https://doi.org/10.1109/CVPR.2018.00813>
- [8] X. Deng, J. Liu, C. Peng, Y. Wang, Using improved YOLOv5 model to detect volume for logs in log farms, Journal of Internet Technology 24(7)(2023) 1403-1413. <https://doi.org/10.53106/160792642023122407002>
- [9] Y. Guo, Y. Zeng, F. Gao, Y. Qiu, X. Zhou, L. Zhong, C. Zhan, Improved YOLOv4-CSP algorithm for detection of bamboo surface sliver defects with extreme aspect ratio, IEEE Access 10(2022) 29810-29820.

- <https://doi.org/10.1109/ACCESS.2022.3152552>
- [10] C. Sarmah, P. Sarma, D. Kalita, C. Das, Satriya double hand mudra recognition using YOLOv5 and RT-DETR: A comparative study, in: Proc International Conference on Advances in Electrical and Computer Technologies, CRC Press, 2025 (287-292).  
<https://www.taylorfrancis.com/chapters/edit/10.1201/9781003515470-40/satriya-double-hand-mudra-recognition-using-yolov5-rt-detr-comparative-study-chayanika-sarmah-parismita-sarma-dishanka-kalita-chiranjib-das>
- [11] R. Chen, Y. Ding, X. Lu, A lightweight model of multi-UAVs object detection with dynamic layer aggregation, Journal of Computers 36(1)(2025) 127-142.  
<https://doi.org/10.63367/199115992025023601009>
- [12] M. Hu, Y. Li, L. Fang, S. Wang, A2-FPN: Attention aggregation based feature pyramid network for instance segmentation, in: Proc 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.  
<https://doi.org/10.1109/CVPR46437.2021.01509>
- [13] Z. Wang, Y. Ling, X. Wang, D. Meng, L. Nie, G. An, & X. Wang, An improved Faster R-CNN model for multi-object tomato maturity detection in complex scenarios, Ecological Informatics 72(2022) 101886.  
<https://doi.org/10.1016/j.ecoinf.2022.101886>
- [14] Y. Xiao, S. Yin, G. Cui, W. Zhang, L. Yao, Z. Fang, E-YOLOv4-tiny: a traffic sign detection algorithm for urban road scenarios, Frontiers in Neurorobotics 17(2023) 1220443.  
<https://doi.org/10.3389/fnbot.2023.1220443>
- [15] S. Li, F. Sultonov, J. Tursunboev, J.H. Park, S. Yun, J.M. Kang, Ghostformer: A GhostNet-based two-stage transformer for small object detection, Sensors 22(18)(2022) 6939.  
<https://doi.org/10.3390/s22186939>
- [16] L. Zhang, N. Zhang, R. Shi, G. Wang, Y. Xu, Z. Chen, SG-Det: shuffle-GhostNet-based detector for real-time maritime object detection in UAV images, Remote Sensing 15(13)(2023) 3365.  
<https://doi.org/10.3390/rs15133365>
- [17] P. Nandal, S. Pahal, S. Malik, N. Sehrawat, Mamta, Enhancing real time object detection for autonomous driving using YOLO-NAS algorithm with CLEO optimizer, International Journal of Information Technology 17(3)(2025) 1321-1328.  
<https://doi.org/10.1007/s41870-024-02296-w>
- [18] X. Cao, Y. Zhang, S. Lang, Y. Gong, Swin-transformer-based YOLOv5 for small-object detection in remote sensing images, Sensors 23(7)(2023) 3634.  
<https://doi.org/10.3390/s23073634>
- [19] X. Deng, Q. Yu, P. Wang, X. Shen, L.C. Chen, Coconut: Modernizing coco segmentation, in: Proc 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024.  
<https://doi.org/10.1109/CVPR52733.2024.02065>
- [20] N. Suryanto, A.A. Adiputra, A.Y. Kadiptya, T.T.H. Le, D. Pratama, Y. Kim, H. Kim, Cityscape-Adverse: Benchmarking robustness of semantic segmentation with realistic scene modifications via diffusion-based image editing, IEEE Access 13(2025) 69921-69940.  
<https://doi.org/10.1109/ACCESS.2025.3537981>
- [21] Y. Chen, X. Yuan, J. Wang, R. Wu, X. Li, Q. Hou, M.M. Cheng, YOLO-MS: Rethinking multi-scale representation learning for real-time object detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 47(6)(2025) 4240-4252.  
<https://doi.org/10.1109/TPAMI.2025.3538473>
- [22] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, G. Ding, Yolov10: Real-time end-to-end object detection, in: Proc. Advances in Neural Information Processing Systems 37, 2024.  
<https://doi.org/10.52202/079017-3429>
- [23] L.H. He, Y.Z. Zhou, L. Liu, W. Cao, J.H. Ma, Research on object detection and recognition in remote sensing images based on YOLOv11, Scientific Reports 15(1)(2025) 14032.  
<https://doi.org/10.1038/s41598-025-96314-x>
- [24] W. Zhao, X. Yang, X. Ma, Y. Wang, A lightweight multi object detection algorithm for complex road scenes based on CMF-YOLO, in: Proc 2024 IEEE International Conference on Industrial Informatics (INDIN), 2024.  
<https://doi.org/10.1109/INDIN58382.2024.10774455>
- [25] Y. Gao, Z. Wang, W.S. Zheng, C. Xie, Y. Zhou, MixCon3D: Synergizing multi-view and cross-modal contrastive learning for enhancing 3D representation, in: Proc The Twelfth International Conference on Learning Representations (ICLR 2024), 2024.  
<https://openreview.net/forum?id=rmA6uFEgIY>
- [26] D. Qin, C. Lechner, M. Delakis, M. Fornoni, S. Luo, F. Yang, W. Wang, C. Banbury, C. Ye, B. Akin, V. Aggarwal, T. Zhu, D. Moro, A. Howard, MobileNetV4: Universal models for the mobile ecosystem, in: Proc 2024 European Conference on Computer Vision (ECCV), 2024.  
[https://doi.org/10.1007/978-3-031-73661-2\\_5](https://doi.org/10.1007/978-3-031-73661-2_5)