ISSN:2790-167X Volume-13-(2025)

Optimization and Performance Evaluation of Deep Learning-Based Target Detection Model for Autonomous Driving

Yuhui Jin

Vortek Inc

2711 Lyndon B Johnson Fwy, Dallas, TX, 75234

yuhui.jin@vorteklab.com

Abstract. To enhance the accuracy and real-time performance of object detection in autonomous driving scenarios, an optimized detection network is constructed based on the YOLOv5 model. The network integrates a BiFPN feature fusion structure, CIoU loss function, and data augmentation strategies to strengthen multi-scale object perception. Experiments conducted on the KITTI and nuScenes datasets demonstrate that the optimized model improves the mAP@0.5 from 87.6% to 91.2% and the recall rate from 82.9% to 88.3%, while reducing the inference latency to 20.1 ms. These results show significant improvements in both precision and efficiency over the original model, validating its effectiveness and adaptability in complex traffic environments.

Keywords: autonomous driving; target detection; deep learning; model optimization.

1. Introduction

The rapid development of autonomous driving technology puts forward higher requirements for target detection accuracy and real-time performance, and deep learning shows significant advantages in this field. Due to the complex road environment, multi-target interference and edge computing resources, the traditional detection model is difficult to meet the actual deployment requirements [1]. In this paper, we focus on the structural optimization and performance evaluation of the target detection model, and construct an optimization strategy and a multi-dimensional evaluation system suitable for autonomous driving scenarios, with a view to improving the accuracy, stability, and engineering adaptability of the detection system, and promoting its wide application in intelligent transportation.

2. Fundamentals of Autonomous Driving Target Detection

2.1 Overview of Autonomous Driving Systems

Autonomous driving system is a complex intelligent system integrating sensing, decision-making and execution, which mainly includes perception layer, decision-making layer and control layer. The perception layer collects environmental information through sensors such as cameras, lidar, millimeter wave radar, etc., the decision-making layer performs path planning and behavioral judgment based on the perception results, and the control layer executes the corresponding vehicle operation commands [2]. Target detection, as the core task of the perception layer, directly affects the system's recognition accuracy of obstacles, pedestrians, traffic signs and other targets, and is a

key link in realizing the safety and reliability of autonomous driving. The schematic diagram is shown in detail in Fig. 1.

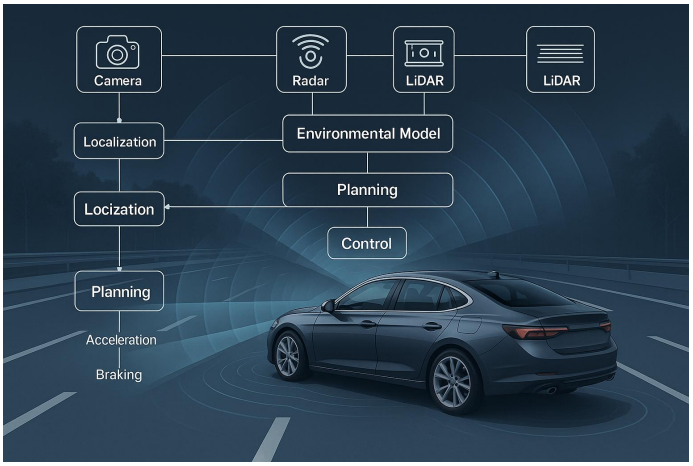


Fig 1 Autopilot Schematic

2.2 Introduction to the target detection task

Target detection is a computer vision task that combines target recognition and localization in identifying specific classes of targets in an image and labeling their bounding boxes. In autonomous driving, target detection is mainly used to identify key targets such as vehicles, pedestrians, traffic signs, signals, etc., to ensure the safety and real-time driving decisions [3-4]. This task needs to deal with light changes, target occlusion, scale differences, etc. in complex environments, which puts higher requirements on the accuracy and speed of detection algorithms, and is one of the core technologies in the automatic driving perception system.

2.3 Application of Deep Learning in Target Detection

Deep learning, by virtue of building a deep neural network system, significantly enhances the accuracy and robustness of feature extraction and target discrimination in automatic driving target detection, and the mainstream methods have a two-stage model based on candidate frames, YOLOv5, as a typical representative object, adopts CSPDarkNet as the backbone network, and then introduces the constituent structures of PANet, FPN, etc., to realize the multiscale feature aggregation, which Greatly improve the small target detection capability [5].

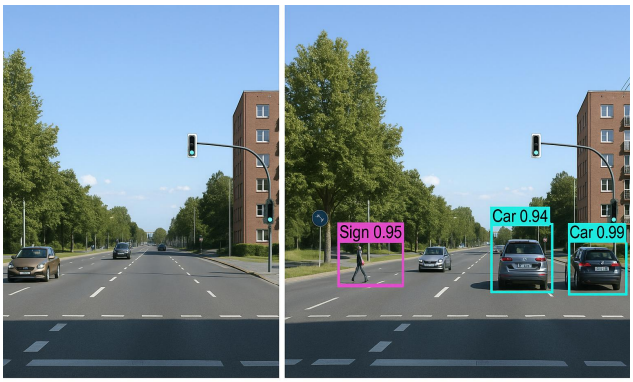
The target detection task is essentially to predict the category c , bounding box location (x, y, w, h) and confidence p of each target in the image. A typical one-stage loss function is:

$$L = \lambda_{cls} \cdot L_{cls} + \lambda_{loc} \cdot L_{bbox} + \lambda_{obj} \cdot L_{obj} \quad (1)$$

where L_{cls} is the category loss, L_{bbox} is the bounding box regression loss (e.g., GIoU), and L_{obj} is the target existence probability loss.

Detection results based on YOLOv5 in an autonomous driving scenario (Fig. 2). In a typical urban road environment, the model is able to accurately detect and label pedestrians, vehicles, and traffic signs [6]with confidence levels all above 0.9, showing excellent real-time performance and accuracy. According to the KITTI dataset experiments, YOLOv5 的 mAP@0.5 达 到 92.1%, the

inference speed is more than 45 FPS, which meets the real-time demand of the automatic driving system.



Original Image Detection Result

Fig. 2 Effectiveness of deep learning target detection

2.4 Comparative Analysis of Mainstream Detection Models

Mainstream target detection models are mainly divided into two categories: two-stage and one-stage. Two-stage models, represented by Faster R-CNN, perform classification and regression after generating candidate regions first, with high accuracy, suitable for scenes requiring high detection accuracy; one-stage models, such as the YOLO series and SSD, regress the target category and bounding box directly from the image, with faster inference speed, more suitable for real-time autonomous driving tasks [7-8]. In recent years, YOLOv5 and YOLOv8 strike a balance between speed and accuracy and are widely used in real road scenarios.RetinaNet improves the recognition of small and hard-to-detect targets by introducing Focal Loss, while DETR fuses the Transformer structure to achieve end-to-end modeling with stronger contextual understanding. Each model has its own focus in terms of accuracy, speed, complexity, and ease of deployment, and it is necessary to choose an adapted solution according to the actual needs of autonomous driving scenarios [9].

3. Deep Learning Target Detection Model Optimization

The optimization of deep learning target detection models mainly focuses on three aspects: network structure improvement, loss function design and training strategy optimization to enhance detection accuracy, speed and robustness [10]. In terms of structure optimization, feature pyramid network (FPN) or bidirectional feature pyramid (BiFPN) is often introduced to enhance the multi-scale target detection capability. In YOLO series, residual structure and cross-stage partial connection (CSP) are utilized to enhance gradient flow transfer and parameter utilization[11].

Loss function optimization is a key part, and the commonly used bounding box regression loss is GIoU (Generalized IoU):

$$L_{GIoU} = 1 - IoU \frac{\left|C-(A \cup B)\right|}{\left|C\right|} \quad (2)$$

Where, C is the minimum closure region, A and B are the prediction frame and the real frame, and GIoU penalizes the non-overlapping region on the basis of IoU to improve the localization accuracy.

To cope with the category imbalance problem, Focal Loss is widely used in categorization loss:

$$L_{Focal} = -\alpha_t (1 - p_t)^{\gamma} \log(p_t) \quad (3)$$

where p_t is the predicted probability and γ is the parameter that regulates the focus of difficult cases.

Data enhancement methods such as Mosaic and MixUp, as well as cosine annealing learning rate scheduling strategy are also introduced in the training to effectively improve the model generalization ability and convergence speed, thus better adapting to the complex scenarios in automated driving [12].

4. Performance evaluation methodology design

4.1 Performance assessment indicators

Commonly used metrics include Mean Average Precision (mAP), which reflects the overall accuracy of detection; Precision and Recall are used to evaluate false and missed detections; and F1-score takes into account the balance between the two [13]. In terms of real-time metrics, the frame rate (FPS) measures the model inference speed, while the number of model parameters and inference latency reflect its deployment efficiency. False detection rate (FPR) and missed detection rate (FNR) are particularly important for safety-sensitive autonomous driving systems and are key metrics in performance evaluation.

4.2 Evaluation Process and Test Platform

The evaluation process mainly consists of five steps: data preparation, model loading, inference execution, metrics calculation and results analysis (see Table 1). First, select a standard test dataset, unify the input size and preprocessing method; load the model to be tested and configure the inference parameters; then execute the testing process, record the prediction bounding box and category labels; calculate the core metrics, such as mAP, Recall, FPS, etc., to comprehensively evaluate the model performance [14]. The test platform usually chooses deep learning frameworks with GPU acceleration capability, such as PyTorch or TensorFlow, supplemented with tools such as TensorRT for deployment verification.

Table 1 Typical test platforms and their application scenarios

Platform Type	Tool/Environment	Applicable Scenario
Training & Validation Platform	PyTorch, MMDetection	Model development and comparison experiments
Inference & Deployment Platform	TensorRT, ONNX	Real-time performance testing and edge deployment
Simulation Verification Platform	CARLA, Apollo	Simulation testing in autonomous driving scenarios

4.3 Experimental design for comparing the performance of different models

Different model performance comparison experiments need to be conducted under a unified dataset, evaluation index and hardware platform to ensure comparable results. The experimental process includes model selection, parameter standardization, unified input preprocessing, output result comparison and quantitative analysis [15]. The main comparison metrics include mean average precision mean (mAP), inference speed (FPS), false positive rate (FPR) and model complexity (parametric quantity).

The average accuracy is calculated as follows:

$$mAP = \frac{1}{N}\sum_{i=1}^{N}\int_{0}^{1} \Pr ecision_i(r)d_r \tag{4}$$

where N is the total number of categories and $\Pr ecision_i(r)$ is the precision-recall curve for category i .

The experimental setup needs to cover a variety of scenarios (daytime, nighttime, rainy day see Fig. 3) with different target densities to comprehensively reflect the model's adaptability in complex traffic environments [16].

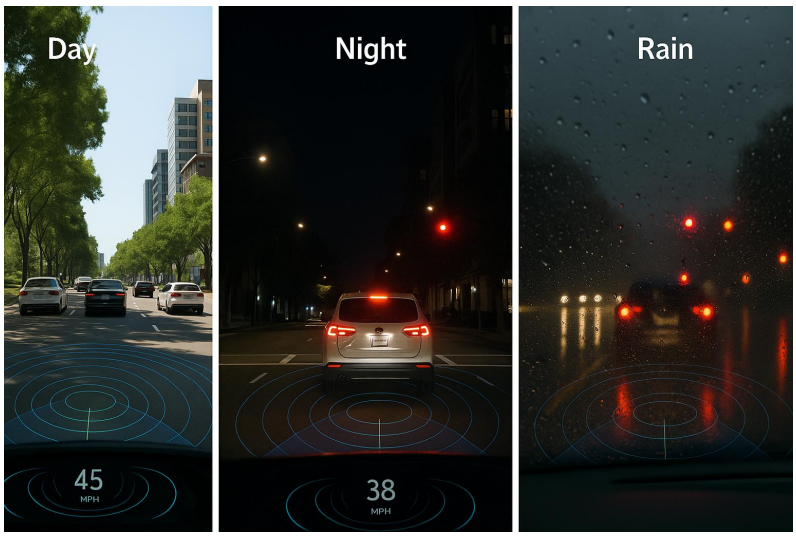


Fig 3 Perceived Adaptation of Models in Different Environmental Conditions

4.4 Evaluation of data set selection and construction criteria

Commonly used public datasets include KITTI, nuScenes, Waymo Open Dataset, etc., covering a wide range of weather, lighting and traffic density conditions, which are suitable for evaluating the robustness of the model. The dataset should contain high-quality images, accurate bounding box labeling, and target category information to ensure a balanced distribution of multi-category targets (e.g., vehicles, pedestrians, and traffic signs) in the sample [17]. If a self-built dataset is used, the parameters of the image acquisition equipment, annotation specifications (e.g., using Pascal VOC or COCO format), and quality control processes should be clarified to ensure data consistency and reproducibility. The dataset division should follow the principle of 70% training, 15% validation, and 15% testing to avoid the crossover of training and testing samples and to improve the objectivity and generalization of the evaluation results.

5. Experimental design and analysis of results

5.1 Experimental environment and configuration

The experimental environment is based on Ubuntu 20.04 operating system with PyTorch 1.13 deep learning framework, equipped with NVIDIA RTX 3090 GPU with CUDA version 11.7 and 24 GB of video memory. the processor is Intel Core i9 with 64 GB of RAM, which can satisfy the demand of high concurrency data loading and model inference [18]. During the training process, the batch size was set to 16, the initial learning rate was 0.001, the optimizer was Adam, the number of training rounds was set to 100, and the early stop mechanism was enabled to prevent overfitting. The datasets used for the experiments include KITTI and nuScenes, and the image resolution is uniformly adjusted to 640×384 to ensure the stability of the model in different scenes.

5.2 Model Training and Validation Process

Before model training, the dataset is divided into training set, validation set and test set in 7:2:1, and the image size is unified and normalized. YOLOv5 architecture is used for training, and Mosaic, MixUp and other data enhancement strategies are introduced to improve the model generalization ability. The optimizer selects Adam, sets the initial learning rate and weight decay, and uses the learning rate scheduling mechanism to adjust the parameters dynamically. The mAP and loss changes on the validation set are continuously monitored during training, and the Early Stopping mechanism is applied to prevent overfitting. The optimal model is saved after each round of training for subsequent performance evaluation and visualization analysis.

5.3 Comparison of model performance before and after optimization

In order to verify the effectiveness of the proposed optimization strategy, the changes in several key performance indicators of the target detection model before and after optimization are compared and analyzed, and the results are shown in Table 2.

Table 2 Comparison of model performance before and after optimization

Model Version	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Recall (%)	Precision (%)	FPS (1080P)	Model Parameters (M)	Inference Latency (ms)
Original YOLOv5s	87.6	55.3	82.9	84.5	46.2	7.5	21.7
Optimized YOLOv5s+	91.2	60.8	88.3	89.6	48.9	8.1	20.1

The optimized model YOLOv5s+ outperforms the original YOLOv5s across multiple dimensions. The mAP@0.5 increases from 87.6% to 91.2%, and mAP@0.5:0.95 rises to 60.8%, indicating a significant enhancement in overall detection accuracy. The recall improves from 82.9% to 88.3%, and precision from 84.5% to 89.6%, effectively mitigating false positives and missed detections. With only a slight increase in model parameters to 8.1M, the inference frame rate increases from 46.2 FPS to 48.9 FPS, and the inference latency is reduced to 20.1 ms. These results further demonstrate that the optimization not only boosts detection performance but also maintains

excellent real-time capability and computational efficiency, making it suitable for deployment in latency-sensitive autonomous driving systems. See Figure 4 for details.

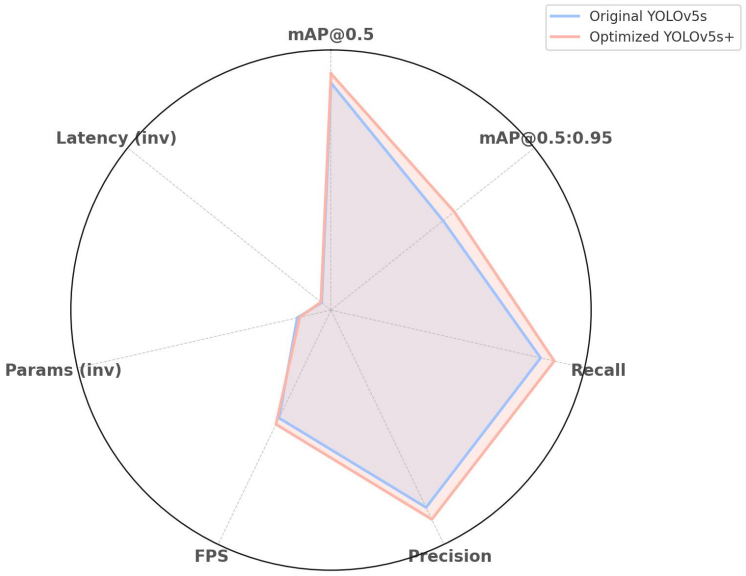


Fig. 4 Comparison of model performance before and after optimisation

5.4 Misdetection and missed detection analysis

In order to comprehensively assess the robustness of the model in the actual road environment, this paper analyzes the two dimensions of false detection rate and leakage rate respectively, and the results are shown in Tables 3 and 4.

Table 3 Comparison of False Detection Rates by Target Category

Category	False detection rate of original model (%)	Optimised model false positive rate (%)
Pedestrian	6.4	3.1
Small Vehicle	4.9	2.7
Large Vehicle	5.5	2.9
Traffic Sign	7.1	3.8
Traffic Light	6.7	3.6

As can be seen in Table 3, the optimized model's false detection rates on all target categories are significantly reduced. The false detection rate of pedestrian category is reduced from 6.4% to 3.1%, traffic signs from 7.1% to 3.8%, and signals from 6.7% to 3.6%, which shows that the model's ability to discriminate small-sized targets is significantly enhanced. The decrease in the overall false detection rate reflects that the optimization strategy has a good effect in improving the accuracy of the model discrimination boundary and reducing false detections, which helps to improve the stability and decision-making reliability of the automated driving system in complex urban traffic environments.

Table 4 Comparison of leakage detection rate under different scenario conditions

Category	False detection rate of original model (%)	Optimized model false positive rate (%)
Daytime Urban	3.2	1.5

Road		
Night Road	7.8	4.2
Rainy Weather	8.9	5.4
Tunnel/Shadow Area	10.3	6.7

As shown in Table 4, the optimized model significantly improves the leakage detection in complex environments. The leakage detection rate is reduced from 7.8% to 4.2% for nighttime scenarios, from 8.9% to 5.4% for rainy scenarios, and from 10.3% to 6.7% for tunnels or shaded areas. Among them, the improvement of adaptability to low-light environment is especially obvious. This result indicates that the optimized detection model maintains a strong target sensing ability under edge conditions such as low light and weather changes, which further improves the adaptability and safety of the autonomous driving system to multiple scenarios. The details are shown in Fig. 5.

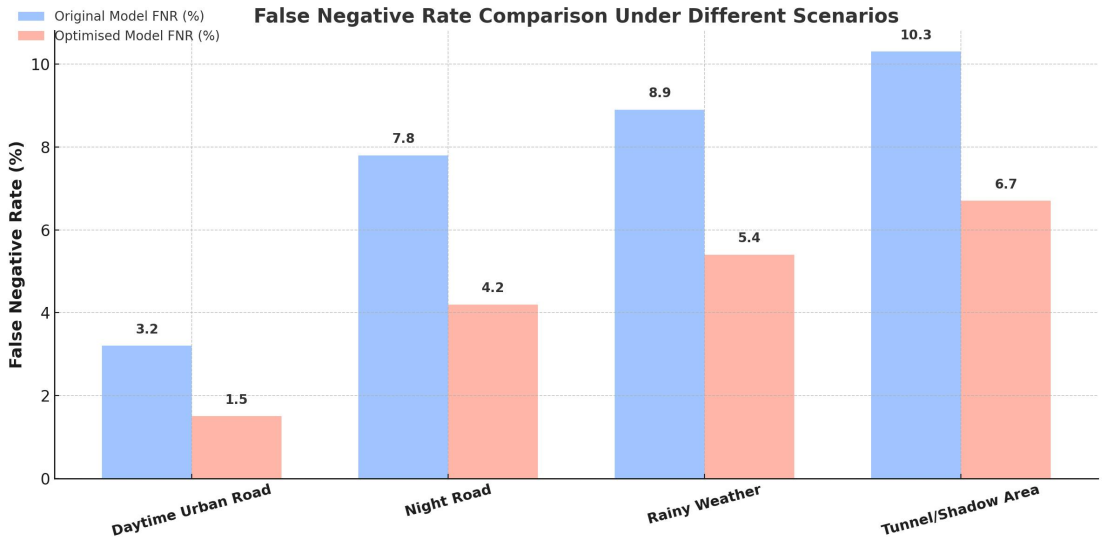


Fig 5 False Negative Rate Comparison Under Different Scenarios

6. Conclusion

This paper focuses on the target detection task in automatic driving scenarios, systematically explores the specific methods of deep learning models in terms of structure optimization, loss function improvement and training strategy adjustment, and constructs a multi-dimensional performance evaluation system. The experimental results show that the optimized model has significant improvement over the baseline model in terms of precision, recall and inference efficiency, and exhibits stronger robustness and adaptability in complex traffic environments.

Future research can further integrate the Transformer structure with the multimodal perception mechanism to improve the recognition effect of the detection model on low-light, small and occluded targets; exploring a more efficient lightweight network structure and edge inference acceleration scheme for the computational constraints of in-vehicle deployment environments will be an important direction to promote the application of autonomous driving systems on the ground.

References

[1] Upadhyaya P D ,Cakir G ,Ramat S , et al. A Multihead Attention Deep Learning Algorithm to Detect Amblyopia Using Fixation Eye Movements [J]. Ophthalmology Science, 2025, 5 (5): 100775-100775.

[2] Dahiya R ,G M ,Jawale S S , et al. Deep learning-based multi-brain capsule network for Next-Gen Clinical Emotion recognition using EEG signals [J]. Neuroscience Informatics, 2025, 5 (2): 100203-100203.

[3] Li R ,Wang Q ,Gao R , et al. Sepsis Important Genes Identification Through Biologically Informed Deep Learning and Transcriptomic Analysis. [J]. Clinical and experimental pharmacology & physiology, 2025, 52 (7): e70031.

[4] Shi K ,Wang K ,Wang X , et al. Quantitative evaluation of grooves in fuel ice layers of ICF based on deep learning and x-ray phase retrieval [J]. Nuclear Fusion, 2025, 65 (6): 066013-066013.

[5] Singh P ,Pranav P ,Dutta S . Bi-GAN-LDA for cybersecurity: a hybrid deep learning framework for advanced network anomaly detection [J]. Engineering Research Express, 2025, 7 (2): 025238-025238.

[6] Ma J ,Zuo Y ,Du H , et al. Interactive Output Modalities Design for Enhancement of User Trust Experience in Highly Autonomous Driving [J]. International Journal of Human–Computer Interaction, 2025, 41 (10): 6172-6190.

[7] Wu Y ,Mu X ,Shi H , et al. An object detection model AAPW-YOLO for UAV remote sensing images based on adaptive convolution and reconstructed feature fusion. [J]. Scientific reports, 2025, 15 (1): 16214.

[8] Peng H ,Hu Y ,Yu B , et al. TCAINet an RGB T salient object detection model with cross modal fusion and adaptive decoding [J]. Scientific Reports, 2025, 15 (1): 14266-14266.

[9] Song Z ,Zhang X ,Tan P . YOLO-Fast: a lightweight object detection model for edge devices [J]. The Journal of Supercomputing, 2025, 81 (5): 724-724.

[10] Şahin D ,Torkul O ,Şişci M , et al. Real-Time Classification of Chicken Parts in the Packaging Process Using Object Detection Models Based on Deep Learning [J]. Processes, 2025, 13 (4): 1005-1005.

[11] Wang M ,Yuan G ,He H , et al. Multi-category solar radio burst detection based on task-aligned one-stage object detection model [J]. Astrophysics and Space Science, 2025, 370 (3): 23-23.

[12] Gao P ,Li Z . YOLO-S3DT: A Small Target Detection Model for UAV Images Based on YOLOv8 [J]. Computers, Materials & Continua, 2025, 82 (3): 4555-4572.

[13] Wang C ,Yi H . DGBL-YOLOv8s: An Enhanced Object Detection Model for Unmanned Aerial Vehicle Imagery [J]. Applied Sciences, 2025, 15 (5): 2789-2789.

[14] Yılmaz A ,Yurtay Y ,Yurtay N . AYOLO: Development of a Real-Time Object Detection Model for the Detection of Secretly Cultivated Plants [J]. Applied Sciences, 2025, 15 (5): 2718-2718.

[15] Hong Y ,Wang L ,Su J , et al. Enhanced foreign body detection on coal mine conveyor belts using improved DLEA and lightweight SARC-DETR model [J]. Signal, Image and Video Processing, 2025, 19 (5): 349-349.

[16] Soans R ,Fukumizu Y . Custom Anchorless Object Detection Model for 3D Synthetic Traffic Sign Board Dataset with Depth Estimation and Text Character Extraction [J]. Applied Sciences, 2024, 14 (14): 6352-6352.

[17] Guoqing X ,Xiaolong X ,Honghao G , et al. FP-RCNN: A Real-Time 3D Target Detection Model based on Multiple Foreground Point Sampling for Autonomous Driving [J]. Mobile Networks and Applications, 2023, 28 (1): 369-381.

[18] Gang Z ,Shan L ,Juan W Z , et al. A Lightweight Mobile Outdoor Augmented Reality Method Using Deep Learning and Knowledge Modeling for Scene Perception to Improve Learning Experience [J]. International Journal of Human–Computer Interaction, 2021, 37 (9): 884-901.