ISSN: 2664-9640

An Improved YOLOv8-Based Vehicle Detection Algorithm for Complex Traffic Scenes

Yunjuan Cheng, Botao Liu

College of Computer Science, Yangtze University, Jingzhou 434023, China

Abstract

To address illumination changes, occlusion, small objects, and the limited compute of edge devices in intelligent transportation scenarios, this paper proposes an improved detection framework built upon YOLOv8. A Centralized Feature Module (CFM) is introduced into the backbone to fuse global semantics with corner-level details and enhance small-object representation. A Dynamic Detection Head (DDH) is further designed that cascades scale, spatial, and task attentions to adaptively allocate features. In addition, an Adaptive Modulation Loss (ADLF) dynamically adjusts the IoU threshold and loss weights, improving robustness to occlusion and ambiguous boundaries. Experiments on UA-DETRAC and SODA10M demonstrate that, while maintaining real-time performance (≥220 FPS), the proposed method improves mAP50 from 90.9% to 93.1% on UA-DETRAC and from 69.7% to 72.3% on SODA10M, markedly reducing missed detections for small objects and in cluttered backgrounds. The model contains ~10.6M parameters and ~27.8G FLOPs-lower than the baseline-showcasing strong potential for lightweight edge deployment on in-vehicle and roadside devices.

Keywords

YOLOv8; Vehicle Detection; Attention Mechanism; Lightweight; Adaptive Loss.

1. Introduction

With the rapid development of intelligent transportation systems and vehicular networking, vehicle detection-an essential component of the perception layer-plays a crucial role in traffic flow monitoring, automatic inspection, intelligent signal control, and autonomous driving [1]. To enable accurate recognition and real-time tracking of vehicles, both detection accuracy and efficiency are critical [2]. Compared with cloud processing, on-device deployment on embedded or edge platforms can effectively reduce communication latency, save bandwidth, and improve system stability and security under specific conditions. However, due to the limited compute of edge devices, achieving high detection accuracy while maintaining real-time performance and lightweight deployment remains a key challenge.

In recent years, deep learning has driven rapid advances in object detection. One-stage detectors, represented by the YOLO [3] family and SSD, feature concise architectures and fast end-to-end inference, making them well suited for real-world traffic surveillance scenarios. The YOLO series has evolved from YOLOv1 to YOLOv8 [4], gradually improving the accuracy–speed trade-off; for example, YOLOv5 balances accuracy and speed via the CSP module, YOLOv7 strengthens small-object detection through head enhancements, and YOLOv8 adopts new feature extraction mechanisms and an anchor-free strategy to further boost performance (representative improvements retained here; detailed historical narration omitted).

Nevertheless, directly applying YOLOv8 to complex traffic environments still faces several challenges: (i) illumination changes, occlusion, and large scale variations degrade accuracy; (ii) on resource-constrained edge devices, there is room to improve inference speed and stability; and (iii) in dense traffic scenes, small vehicles are prone to missed and false detections. Existing

ISSN: 2664-9640

approaches in feature enhancement, attention mechanisms, and lightweight design share common limitations: the first two often introduce considerable computational and parameter overheads that hinder edge deployment, while lightweight schemes tend to sacrifice accuracy, especially for small objects and cluttered backgrounds.

To address these issues, this paper proposes, within the YOLOv8 framework: (i) a Centralized Feature Module (CFM) to fuse global and local information and strengthen representations for small objects under complex backgrounds; (ii) a Dynamic Detection Head (DDH) that cascades scale, spatial, and task attentions to adaptively focus on key features; and (iii) an Adaptive Modulation Loss (ADLF) that dynamically adjusts the IoU threshold and loss weights to enhance robustness for occluded and boundary-ambiguous samples. The proposed scheme improves detection performance while preserving high inference speed, making it more suitable for deployment on edge devices.

2. Related Work

In recent years, with the rapid progress of deep learning, convolutional-neural-network (CNN)–based object detectors have been widely adopted in intelligent transportation. Mainstream detection algorithms can be broadly categorized into two families: two-stage and one-stage methods.

Early two-stage approaches are represented by the R-CNN [5] family. Faster R-CNN, by introducing the Region Proposal Network (RPN), significantly improves the efficiency of proposal generation and often achieves higher accuracy. However, because these methods rely on a proposal-generation stage followed by per-region classification, inference is relatively slow and struggles to meet real-time requirements in traffic scenarios. Recently, Transformer-based detectors [6] (e.g., DETR, DINO) have achieved notable accuracy gains, but their substantial computational cost makes deployment on embedded or edge devices difficult.

By contrast, one-stage detectors-exemplified by the YOLO family and SSD-offer concise architectures and fast end-to-end inference, making them well suited for real-world traffic surveillance. The YOLO series has undergone multiple iterations since YOLOv1. For instance, YOLOv5 [7] balances accuracy and speed via an improved backbone with CSP modules; YOLOv7 further enhances the detection head to improve small-object performance; and YOLOv8 introduces new feature extraction mechanisms and an anchor-free strategy for another round of performance improvement. Compared with two-stage methods, YOLO-style models may sacrifice a small amount of accuracy but provide clear advantages in inference speed and deployment flexibility.

Building on the YOLO line, both academia and industry have proposed numerous improvement strategies. One stream focuses on feature enhancement and multi-scale fusion-for example, employing more sophisticated feature pyramids to strengthen sensitivity to objects at various scales [8]; however, such designs often increase computation substantially and hinder edge deployment. A second stream introduces attention mechanisms (channel, spatial, or hybrid) to amplify responses over key regions and improve detection accuracy [9], yet frequently at the cost of extra parameters and slower inference. A third stream pursues lightweight design-e.g., replacing the backbone, pruning, or quantization-to shrink model size, though usually with unavoidable accuracy drops, particularly for small objects or cluttered backgrounds. Overall, while these directions have achieved progress, it remains challenging in complex traffic scenes to balance accuracy, robustness, and real-time performance simultaneously. To this end, our method-within the YOLOv8 framework-enhances small-object and complex-background representations via a Centralized Feature Module (CFM), realizes multi-level adaptive allocation through a Dynamic Detection Head (DDH), and improves robustness to occlusion and

ISSN: 2664-9640

ambiguous samples using an Adaptive Modulation Loss (ADLF), thereby boosting accuracy while preserving high throughput for edge deployment.

3. Method

3.1. Overall Architecture of the Improved Detector

To enhance the detection performance of YOLOv8 in complex traffic scenarios, this paper conducts targeted optimizations based on the original structure. The overall framework is illustrated in Figure 1.

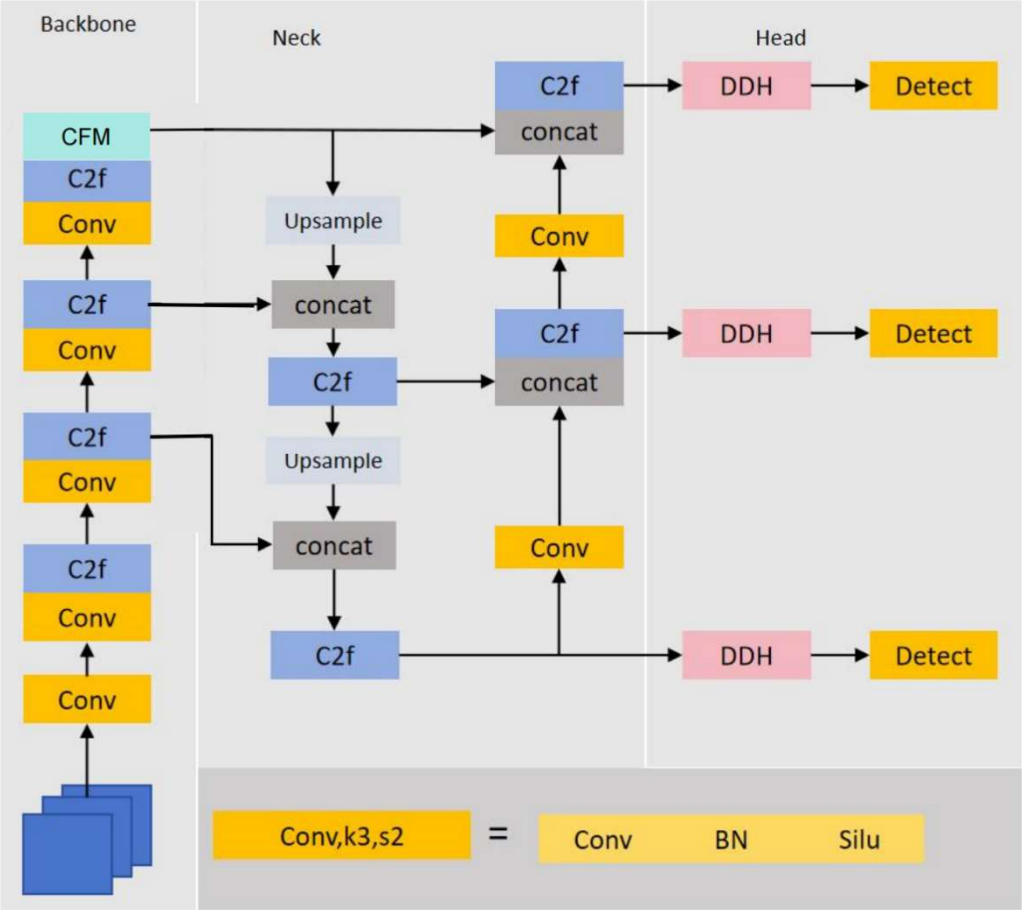

Figure 1. Improved YOLOv8 overall structure

Modular embedding is performed on the standard YOLOv8 Backbone–Neck–Head pipeline: the overall structure is shown in Figure 1. On the Backbone side, CFM (Centralized Feature Module) is added to the shallow/middle-layer feature branches to enhance fine-grained information such as edges/corners through a centralized feature fusion mechanism, while performing channel-level fusion with the original features to stabilize the distribution; the Neck end maintains a pyramid-style fusion strategy with multi-level upsampling and concatenation to aggregate cross-scale context; the Head end is replaced with DDH (Dynamic Detection Head) in a cascaded manner, allowing scale attention → spatial attention → task attention to act sequentially on three detection feature maps with different resolutions, thereby adaptively highlighting key regions and distinguishing classification and regression paths; during the training phase, ADLF (Adaptive Modulation Loss) is introduced to dynamically adjust thresholds and loss weights based on batch and historical IoU distribution to strengthen hard sample learning, while the inference phase fully reuses YOLOv8's forward and post-processing

ISSN: 2664-9640

diagrams without adding extra latency or deployment complexity. The above design follows the division of labor of "shallow enhancement, head self-adaptation, training-side scheduling", enabling the model to achieve better accuracy-speed trade-off and robustness in complex traffic scenarios.

3.2. Centralized Feature Module (CFM)

The traditional YOLOv8 Backbone often suffers from limitations in receptive field when extracting small target features, leading to insufficient response to local edges and corner information. To this end, this paper adds a Centralized Feature Module (CFM) to the shallow feature extraction stage of the Backbone to enhance the model's perception of fine-grained targets in complex backgrounds. CFM is embedded in the shallow/middle-layer feature paths of YOLOv8, consisting of a lightweight MLP branch and a visual center mechanism in parallel, and completes channel-level fusion with the original features at the output to stabilize the distribution and enhance key region responses: its overall structure is shown in Figure 2 (upper for MLP, lower for visual center).

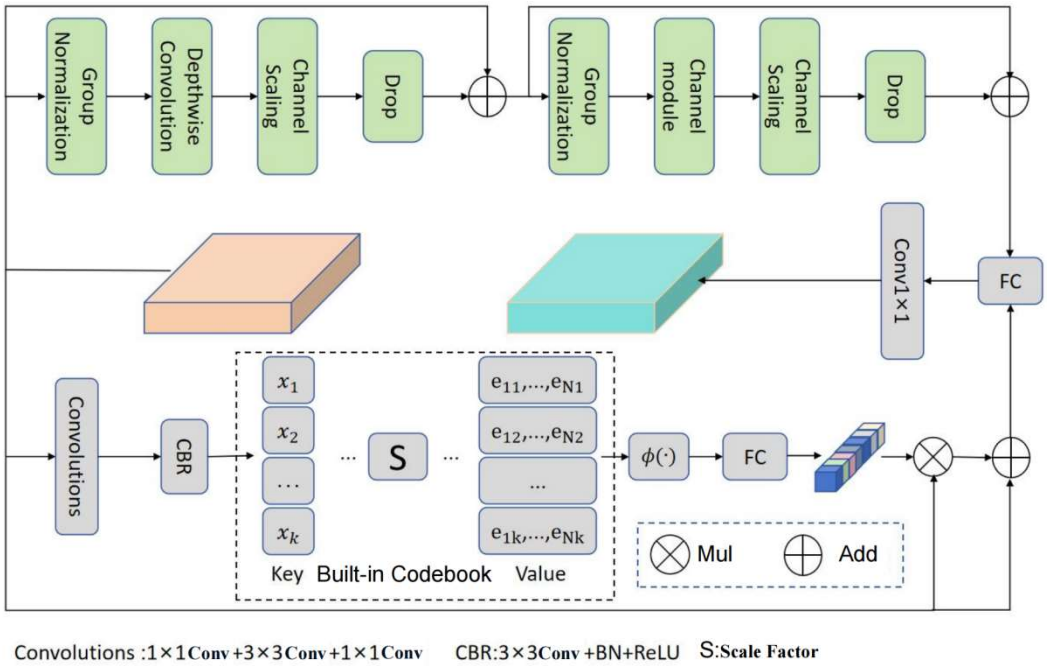

Figure 2. Overall structure of CFM (MLP on top, visual center mechanism on bottom)

The MLP branch first performs group normalization (GN) on the input feature $F \in \mathbb{R}^{H\times W\times C}$, followed by depthwise separable convolution, channel scaling, and DropPath to form a spatial MLP, expressed as equation (1):

$$F_1 = DConv\big(GN(F)\big) + F \tag{1}$$

where $DConv(\cdot)$ is a 1×1 depthwise separable convolution, GN is group normalization, channel scaling refers to a learnable channel weight vector, and DropPath adopts a random depth drop strategy with a drop rate of 0.1.

Compared to spatial MLP, channel MLP can not only effectively reduce computational complexity but also meet the requirements of general visual tasks. Structurally different from spatial MLP, it only replaces the depthwise separable convolution with a channel module, followed by residual connection to form the MLP layer. This part can be expressed as:

ISSN: 2664-9640

$$\mathrm{MLP(F)} = \mathrm{CM}\big(\mathrm{GN}(\mathrm{F_1})\big) + \mathrm{F_1} \tag{2}$$

where CM represents the channel module.

The visual center mechanism is an encoder module with a built-in dictionary mechanism, mainly composed of a built-in codebook and a set of learnable scaling factors. The built-in codebook is denoted as $B = \{b_1, b_2, \dots, b_k\}$, where k is the total number of visual centers. The learnable scaling factors are denoted as $S = \{s_1, s_2, \dots, s_k\}$, used to adjust the response intensity of each visual center. Similar to MLP, assume the input feature is $F \in R^{H \times W \times C}$, where H, W, and C represent the height, width, and number of channels of the feature map, respectively. The input feature is first encoded through a module consisting of multiple groups of convolution layers, which includes a 1×1 convolution, a 3×3 convolution, and another 1×1 convolution. After that, the encoded feature F is further processed through a CBR block, which contains a 3×3 convolution, a batch normalization (Batch Normalization, BN) layer, and a ReLU activation function for nonlinear feature refinement. After obtaining the encoded features, the features are input into the codebook. To enhance the response of key semantic prototypes in the feature space, we introduce a learnable visual codebook. Specifically, for each pixel position xi, the information between it and the k-th visual center bk is calculated through the following formula:

$$e_k = \sum_{i=1}^{N} \frac{e^{-s_k \|x_i - b_k\|^2}}{\sum_{j=1}^{K} e^{-s_k \|x_i - b_j\|^2}} (x_i - b_k) \tag{3}$$

where x_i represents the feature at the i-th spatial position, b_k is the k-th learnable visual center, and s_k is the corresponding scaling factor. The x_i-b_k in the expression describes the relative relationship between the pixel and the codebook. Subsequently, all e_k are processed through a fusion function $\phi(\cdot)$ to obtain the global visual response feature of the entire image under k codewords. This process is expressed as:

$$e = \sum_{k=1}^{K} \phi(e_k) \tag{4}$$

where ϕincludes BN layer, ReLU activation function, and averaging operation for fusing local feature information. After obtaining the fused feature e, it is fed into a fully connected layer and a 1×1 convolution layer to further generate feature maps that highlight key semantic categories. Next, the result is fused with the original Stem module's input feature F using channel-wise multiplication, specifically expressed as:

$$Z = F \otimes \delta(Conv_{1\times1}(e)) \tag{5}$$

where $\delta(\cdot)$ represents the Sigmoid activation function, $\otimes$represents channel-level element-wise multiplication, and $\mathrm{Conv}_{1\times1}$ represents $1 \times 11 \times 1$ convolution operation. Finally, channel-level addition is used to add the fused feature Z with the original input feature F to obtain the final output:

$$Out = Z \oplus F \tag{6}$$

where$\otimes$represents channel-level element-wise addition. Through the above mechanism, CFM can significantly enhance the feature's response to key regions while maintaining the original spatial structure, thereby improving the overall model's perception accuracy for target regions.

ISSN: 2664-9640

3.3. Detection Head Design (DDH)

In actual traffic scenarios, the original detection head of YOLOv8 adopts a single-scale feature map strategy, which cannot fully utilize multi-scale complementary information and lacks deep modeling of context, leading to a decline in detection accuracy under large size differences or complex backgrounds. To break through this bottleneck, this paper proposes a Dynamic Detection Head (DDH) module, which integrates three major mechanisms-scale attention, spatial attention, and task attention-to build a hierarchical feature enhancement system (as shown in Figure 3).

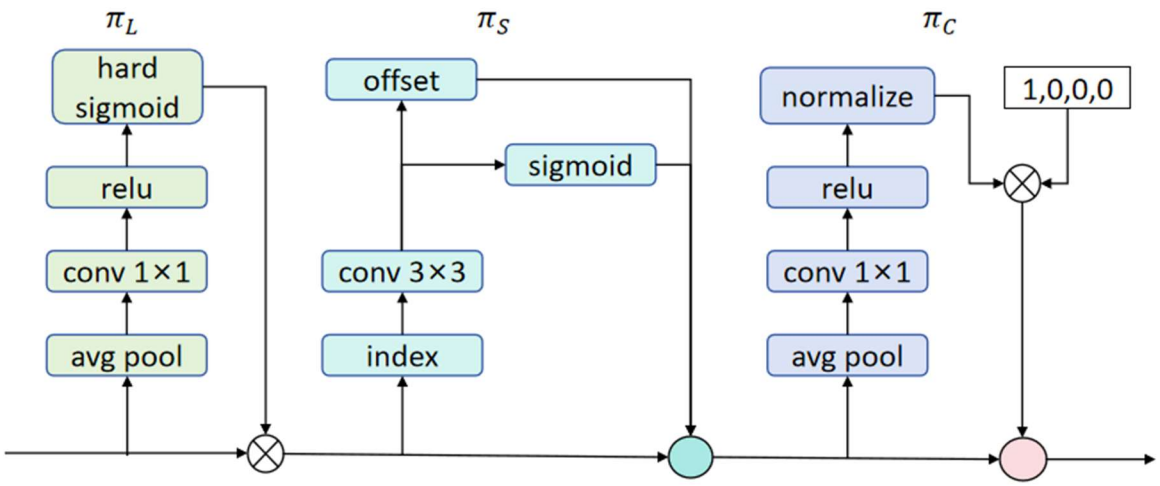

Figure 3. Overall structure of DDH

DDH adopts cascaded attention computation for the input three-dimensional feature tensor F:

$$W(F) = \pi_C(\pi_S(\pi_L(F) \cdot F) \cdot F) \cdot F \tag{7}$$

where F represents the input three-dimensional feature tensor, and $\pi_L(\cdot)$, $\pi_S(\cdot)$, and $\pi_C(\cdot)$ correspond to different modules for scale perception, spatial perception, and task perception, respectively.

Scale Perception Attention Module: This module adaptively adjusts feature weights along the scale dimension. First, average pooling is performed on F in scale and channel dimensions to extract global scale information; subsequently, a nonlinear transformation function f(·) and activation function σ(·) generate attention weights, which are element-wise weighted on F:

$$\pi_L(F) = \sigma(f(\frac{1}{SC}\sum_{S,C} F)) \cdot F \tag{8}$$

This design highlights key scale responses, improving detection robustness for small/large targets.

Spatial Perception Attention Module: For the spatial dimension, this module adopts a sparse sampling strategy, selecting limited positions and weighted summation to strengthen target-related regions:

$$\pi_S(F) \cdot F = \frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K} \mathcal{W}_l, k \cdot F(l; p_k + \Delta p_k; c) \cdot \Delta m_k \tag{9}$$

ISSN: 2664-9640

where L represents the number of selected reference positions, K is the number of sampling points around each reference position; w_l is the adaptively learned weight parameter used to control the contribution of each sampling point to the output feature; $p_k+\Delta p_k$ represents the new sampling position after introducing offset Δp_k on the original sampling point p_k, thereby allowing the model to focus on discriminative regions; Δm_k is the importance weight at the offset position, further adjusting the influence of each feature response.

(3)Task Perception Attention Module: To enhance information sharing between classification and regression, this module dynamically adjusts responses in the channel dimension: first, global average pooling on F to extract channel context; then two layers of linear transformation + normalization + Sigmoid activation to generate weights:

$$\pi_C(F) \cdot F = max(\alpha^1(F) \cdot F_c + \beta^1(F), \alpha^2(F) \cdot F_c + \beta^2(F)) \tag{10}$$

where F represents the input feature map; F_c represents the feature subset of the c-th channel; $\pi_c(\cdot)$ represents the attention weight function acting on the c-th channel; $\alpha^1(\cdot)$, $\alpha^2(\cdot)$, $\beta^1(\cdot)$, and $\beta^2(\cdot)$ are two learnable parameter functions about the input feature map F; max is the maximum operation, which compares the results in the expression and finally obtains the weight value used to adjust the features of the c-th channel, thereby dynamically adjusting the activation degree of each channel under different tasks.

3.4. Adaptive Modulation Loss Function Design (ADLF)

The traditional YOLOv8 loss function (such as cross-entropy and IoU loss) is easily affected by low recall of small targets, boundary blur, and background interference in complex traffic scenarios, leading to unstable detection. To enhance model robustness, this paper proposes an Adaptive Modulation Loss Function (ADLF), which achieves real-time perception and optimization of target characteristics through dynamic IoU thresholds and weight adjustment, and the dynamic IoU threshold adjustment formula in this mechanism is as follows:

$$IoU_{mean} = \lambda \times IoU_{mean,prev} + (1 - \lambda) \times IoU_{current} \tag{11}$$

where IoU_{mean} represents the updated IoU threshold in the current iteration, λ is the decay control parameter used to adjust the participation degree of historical information in the current threshold update, $IoU_{mean,prev}$ represents the IoU mean value calculated last time. And $IoU_{current}$ represents the average IoU of samples in the current training batch. Through this update mechanism, the model can dynamically fuse historical detection trends and current prediction performance, thereby achieving smooth adjustment and stable update of the IoU threshold, enhancing the adaptability of the loss function to sample scale and distribution changes. The calculation formula for the decay control parameter λ is as follows:

$$\lambda(x) = decay \cdot (1 - e^{-\frac{x}{\tau}}) \tag{12}$$

where x is the current model training iteration count, decay and τ are the decay rate constant and time constant, respectively. In addition, ADLF further combines the IoU matching degree between the predicted box and the real target to dynamically adjust the loss weights, thereby effectively alleviating the adverse impact of noise interference on detection performance under complex background conditions. Through sensitive modeling of IoU distribution, this mechanism can adaptively improve the model's discrimination ability for boundary blur and unclear target samples, enhancing robustness in occlusion and multi-target dense

ISSN: 2664-9640

environments. To this end, ADLF constructs a set of modulation loss weight calculation methods, with the expression as follows:

$$Loss_{modulated} = Loss_{base} \times M(IoU, IoU_{mean}) \tag{13}$$

where $Loss_{base}$ is the original model's cross-entropy loss function. $M(IoU, IoU_{mean})$ is the loss weight factor, which adjusts with changes in the IoU value between the prediction result and the real target. The specific implementation of the loss weight factor is as follows:

$$M(x) = \begin{cases} 1 & IoU < IoU_{mean} - 0.1 \\ e^{(1-IoU_{mean})} & IoU_{mean} > IoU > IoU_{mean} - 0.1 \\ e^{(1-IoU)} & IoU \geq IoU_{mean} \end{cases} \tag{14}$$

4. Experiments

4.1. Experimental Setup

The experimental setup is as follows: the experimental platform is built on the Ubuntu 20.04 operating system, with the graphics card using NVIDIA GeForce RTX 4090. In terms of software environment, PyTorch 1.13.1 is used as the deep learning framework, combined with NVIDIA CUDA 11.7 toolkit and cuDNN acceleration library.

In terms of training parameter settings, the training batch size is set to 8, the optimizer selects momentum stochastic gradient descent (SGD), the total number of training epochs is 300, and a cosine annealing strategy is used for dynamic adjustment of the learning rate.

4.2. Evaluation Metrics

This paper evaluates the model from two dimensions: detection accuracy and inference efficiency, using Precision (P), Recall (R), and mAP (including mAP@0.5, denoted as mAP50) to measure detection accuracy; FPS is used to measure inference speed. IoU calculation and AP/mAP definitions are consistent with general object detection evaluation. This paper uniformly reports P/R/mAP50 and FPS on both datasets, and synchronously provides changes in key metrics in the ablation study.

4.3. Dataset Introduction

To systematically evaluate the detection performance and generalization ability of the proposed method in complex traffic scenarios, this paper selects two representative public datasets, UA-DETRAC and SODA10M, for experiments, and divides them into training set, validation set, and test set in a 7:1:2 ratio on both.

(1) UA-DETRAC: This dataset is constructed from real traffic surveillance videos, covering common complex situations in urban roads, including small target vehicles, occluded vehicles, dense traffic flows, etc., and includes various weather and time conditions such as normal lighting, low light, and rain/snow, with high real-world challenges, suitable for testing the model's robustness in varying environments.

(2) SODA10M: Released jointly by Huawei Noah's Ark Lab and Sun Yat-sen University, the images are collected from 32 cities across the country, covering multiple road types and traffic environments, providing massive images with bounding box annotations, with diverse scenes and wide regional coverage, suitable for evaluating the model's cross-scene generalization ability.

ISSN: 2664-9640

4.4. Ablation Experiments

To verify the actual improvement effect of the proposed improvement method on vehicle target detection performance, this section conducts ablation experiments on UA-DETRAC and SODA10M datasets, performing combination analysis on the three components: Centralized Feature Module (CFM), Dynamic Detection Head (DDH), and Adaptive Modulation Loss Function (ADLF), and evaluating their impact on detection accuracy. The ablation experiment results are shown in Table 1.

Table 1. Ablation experiments of each component on UA-DETRAC and SODA10M datasets

Models	CFM	DDH	ADLF	UA-DETRAC				SODA10M			
				P/%	R/%	mAP/%	FPS	P/%	R/%	mAP/%	FPS
1	-	-	-	90.5	81.7	90.9	254	72.7	64.1	69.7	254
2	√	-	-	91.3	83.0	91.8	248	74.1	65.3	70.8	248
3	√	√	-	91.5	84.2	92.5	236	74.9	66.1	71.7	236
4	√	√	√	91.8	84.9	93.1	224	75.3	66.7	72.3	221

The ablation experiments show that the three improvement modules bring progressive and complementary gains: on the baseline YOLOv8, adding CFM increases mAP on UA-DETRAC from 90.9%→91.8%, and on SODA10M from 69.7%→70.8%; continuing to stack DDH reaches 92.5%/71.7% respectively; adding ADLF finally improves to 93.1%/72.3%, while Precision and Recall also rise slightly synchronously, verifying the synergistic effect of "feature enhancement-detection head adaptation-loss modulation". Further analysis shows that the introduction of CFM mainly improves the recall rate of small vehicles (area<32×32 pixels), from the baseline 65.2% to 71.5%, confirming its enhancement role on detail features; DDH's cascaded mechanism alleviates multi-scale overfitting, improving mid-target Precision by 2.1%; ADLF's dynamic weights focus on occluded samples, further increasing Recall by 1.3%, overall collaboratively optimizing robustness in complex scenarios.

4.5. Comparative Experiments

To comprehensively evaluate the detection performance and practical value of the proposed vehicle target detection model in diversified traffic scenarios, this paper selects multiple representative mainstream detection algorithms as comparison models and conducts experimental comparisons on UA-DETRAC and SODA10M datasets. The comparison models cover single-stage detectors SSD[6], YOLOv5, YOLOv7, YOLOv8, two-stage detectors Faster R-CNN, Anchor-Free mechanism-based FCOS, as well as recent advanced structures such as ASF-YOLO and Transformer architecture DINO. The experimental results are shown in Tables 2 and 3.

The comparative experiments show that the proposed method achieves the best on UA-DETRAC with mAP@0.5=93.1%/224 FPS, superior to YOLOv8 (91.4%), YOLOv7 (91.2%), and ASF-YOLO (92.9%), and while slightly higher in accuracy than the Transformer architecture DINO (91.8%), it is much faster than its 46 FPS; on the more challenging SODA10M, it also exceeds YOLOv8 (69.7%) and YOLOv7 (70.4%) with mAP@0.5=72.3%/221 FPS, approaching and slightly higher than DINO (72.0%) but significantly faster (45 FPS). Further analysis shows that ASF-YOLO's mAP 69.3% < YOLOv8 69.7% on SODA10M, possibly due to overfitting of its attention module on large-scale diverse datasets, while our DDH alleviates this problem through cascading and sparse design, improving generalization. At the same time, the proposed model's Params 10.6M/FLOPs 27.8G is 5% lighter than YOLOv8 (11.2M/28.6G), proving the advantage in balancing accuracy and real-time performance, more suitable for edge deployment.

ISSN: 2664-9640

Table 2. Comparative experimental results of various models on the UA-DETRAC dataset

Models	P/%	R/%	mAP50/%	FPS	Params(M)	FLOPs(G)
FCOS	82.5	78.3	76.2	53	32.5	66.8
Faster R-CNN	57.2	60.3	55.5	16	41.7	180.2
SSD	81.0	33.1	52.8	98	26.3	68.4
YOLOv5	91.9	80.1	90.4	346	7.2	16.5
YOLOv7	92.8	82.7	91.2	282	6.2	13.7
YOLOv8	91.1	82.1	91.4	254	11.2	28.6
ASF-YOLO	91.1	84.3	92.9	328	11.5	29.2
DINO	92.3	83.5	91.8	46	86.4	198.5
Ours	91.8	84.9	93.1	224	10.6	27.8

Table 3. Comparative experimental results of various models on the SODA10M dataset

Models	P/%	R/%	mAP50/%	FPS	Params(M)	FLOPs(G)
FCOS	70.1	57.2	63.5	52	32.5	66.8
Faster R-CNN	56.8	59.2	44.2	16	41.7	180.2
SSD	66.8	22.4	40.1	97	26.3	68.4
YOLOv5	74.2	60.5	66.8	341	7.2	16.5
YOLOv7	75.4	62.4	70.4	285	6.2	13.7
YOLOv8	72.7	64.1	69.7	254	11.2	28.6
ASF-YOLO	72.0	63.9	69.3	326	11.5	29.2
DINO	76.1	65.8	72.0	45	86.4	198.5
Ours	75.3	66.7	72.3	221	10.6	27.8

4.6. Visualization Results

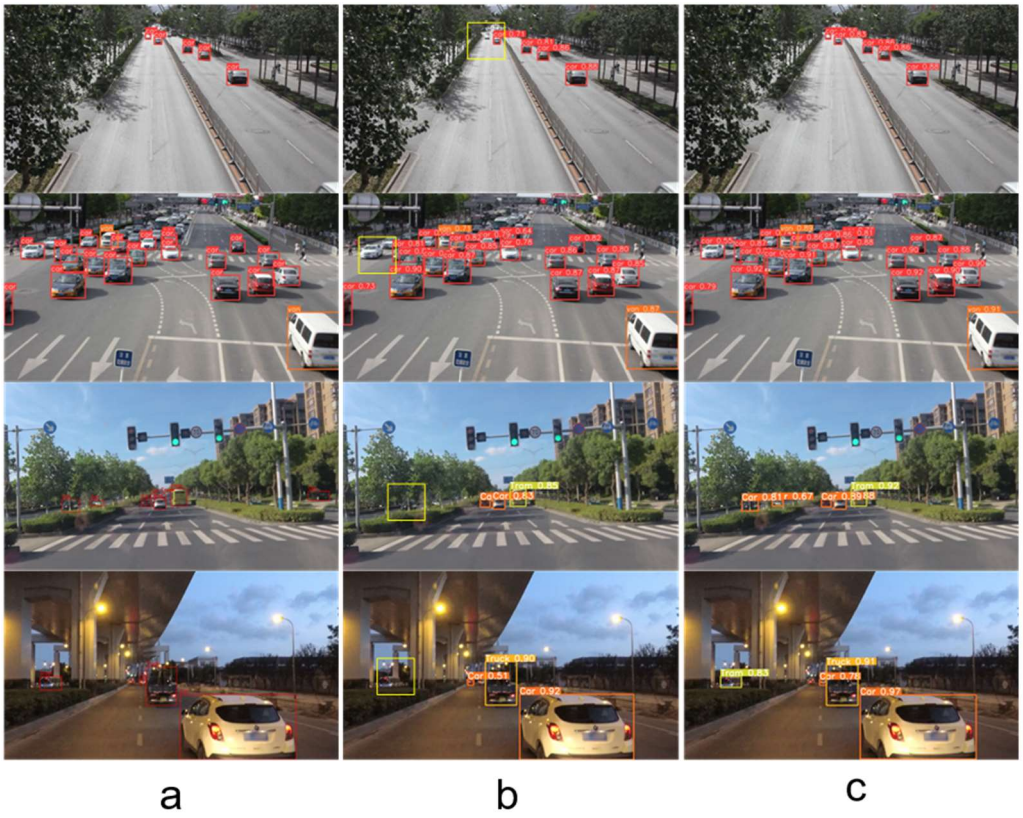

Figure 4. Visualization results comparison. (a) Ground truth; (b) YOLOv8 baseline detection results; (c) Our method detection results. Yellow boxes denote missed detections.1

ISSN: 2664-9640

To comprehensively evaluate the effectiveness of the proposed method, Figure 4 shows the visualization detection result comparisons in typical scenarios. Among them, the first column is the manually annotated real target information, the second column is the detection results of the YOLOv8 model, and the third column is the visualization output of the proposed method.

It can be seen from the yellow box marked missed detection areas that the proposed method effectively reduces missed detections on distant small vehicles (first row) and partially occluded vehicles (second row). Under the same threshold, missed detections are significantly reduced, and boundaries are more stable. This confirms the enhancement of CFM on small targets and corner information, the focusing of DDH on key regions, and the strengthening of ADLF on fuzzy sample training.

5. Conclusion

This paper addresses the vehicle detection problem in complex traffic scenarios and proposes Centralized Feature Module (CFM), Dynamic Detection Head (DDH), and Adaptive Modulation Loss (ADLF) on the YOLOv8 framework, synergistically improving detection performance from three levels: shallow feature enhancement, multi-scale attention self-adaptation, and training objective scheduling. The experimental results show that the method achieves significant gains compared to the baseline on UA-DETRAC and SODA10M datasets, while maintaining high inference speed and low parameter amount and computational overhead, with potential for real-time application on edge devices.

Future work will focus on the following directions:1.For extreme conditions such as heavy rain/fog, nighttime backlight, and snowy weather, conduct generalization ability research (data augmentation, domain adaptation, and robustness evaluation protocols);2.Advance model compression and quantization (structured/unstructured pruning, distillation, INT8/mixed precision quantization) to further reduce latency and power consumption;3.Complete systematic deployment experiments and optimization on edge hardware platforms (inference engine selection, operator fusion, memory/parallel scheduling), and evaluate end-to-end performance in real scenarios; 4 Explore uncertainty estimation and open-set detection to enhance robustness and interpretability when long-tail classes and anomalous targets appear.

References

[1] GIRSHICK R. Fast R-CNN[C]// Proceedings of the IEEE International Conference on Computer Vision. Piscataway, NJ: IEEE, 2015: 1440–1448.

[2] TERVEN J, CÓRDOVA-ESPARZA D M, ROMERO-GONZÁLEZ J A. A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLO-NAS[J]. Machine Learning and Knowledge Extraction, 2023, 5(4): 1680–1716.

[3] LIU W, ANGUELOV D, ERHAN D, et al. SSD: Single Shot Multibox Detector[C]// European Conference on Computer Vision. Cham: Springer, 2016: 21–37.

[4] GIRSHICK R. Fast R-CNN[C]// Proceedings of the IEEE International Conference on Computer Vision. Piscataway, NJ: IEEE, 2015: 1440–1448.

[5] VIOLA P, JONES M J. Robust real-time face detection[J]. International Journal of Computer Vision, 2004, 57(2): 137–154.

[6] LOWE D G. Distinctive image features from scale-invariant keypoints[J]. International Journal of Computer Vision, 2004, 60(2): 91–110.

[7] FELZENSZWALB P F, GIRSHICK R B, MCALLESTER D, et al. Object detection with discriminatively trained part-based models[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 32(9): 1627–1645.

[8] HASELHOFF A, KUMMERT A. A vehicle detection system based on Haar and triangle features[C]// IEEE Intelligent Vehicles Symposium. Piscataway, NJ: IEEE, 2009: 261–266.

ISSN: 2664-9640

[9] GAO T, LIU Z, GAO W, et al. Moving vehicle tracking based on SIFT active particle choosing[C]// International Conference on Neural Information Processing. Berlin, Heidelberg: Springer, 2008: 695–702.