Computer Life
ISSN: 1819-4818 | Vol. 13, No. 3, 2025

Analysis of vehicle and pedestrian recognition algorithm based on improved YOLO

Shugang Liu [a], Yuhe Zhang [b]

School of Computer Science, NORTH CHINA ELECTRIC POWER University, Baoding 071003, China
[a] lsg69@qq.com, [b] 1253387547@qq.com

Abstract: This paper expounds a vehicle pedestrian recognition algorithm based on improved YOLO. It analyzes the shortcomings of current algorithms and proposes improved schemes such as Feature Pyramid Network (FPN) and loss function adjustment. It extracts multi-layer image features through convolutional layers, performs feature fusion and multi-scale detection, and introduces Focal Loss instead of the original classification loss on the basis of the existing loss function to improve the current detection performance and solve the problem of inaccurate recognition of small targets.

Keywords: Intelligent technology; Multi-layer images; YOLO; Vehicle pedestrian recognition.

1. Introduction

The YOLO algorithm is renowned for its efficiency and real-time performance, demonstrating exceptional detection capabilities in current vehicle recognition tasks. However, it shows notable limitations when processing small targets and complex backgrounds. [1] To address these shortcomings in vehicle-pedestrian detection, this study proposes optimization strategies to enhance detection accuracy and robustness. [2] The continuous refinement of this technology holds both theoretical significance and practical value in advancing intelligent systems.

2. YOLO algorithm principle and improvement scheme design

2.1. Network Structure Optimization

To address the network architecture challenges in vehicle-pedestrian detection using the YOLO algorithm, this study employs a Feature Pyramid Network (FPN) for optimization. The FPN effectively enhances detection capabilities across different scales, particularly improving accuracy for small vehicles and pedestrians at long distances. Its core mechanism involves constructing multi-scale feature representations: extracting high-resolution shallow features through bottom-up pathways while generating low-resolution deep features via top-down pathways, followed by feature fusion through vertical integration. The specific implementation steps are as follows [3].

(1) Multi-layer feature extraction. The shallow feature map, with high resolution, captures detailed information and is advantageous for small object detection; the deep feature map, featuring lower resolution, contains richer semantic information and is suitable for large object recognition. Three feature maps C2, C3, and C4 are established, each corresponding to different convolutional layers. Each layer can be defined through convolution operations as shown in Equation (1).

$$\mathbf{C}_l = f(\mathbf{C}_{l-1}; \mathbf{W}_l) \tag{1}$$

In this formula, f (•) represents the convolution operation; W l represents the kernel parameter of the l-th layer; C l-1 represents the feature map output of the previous layer.

The path generation process involves up-sampling high-level feature maps and performing pixel-wise additive fusion with corresponding low-level feature maps. The fused feature map retains rich detail information while preserving high-level semantic features. The fused feature map P l is defined by Equation (2), where C l represents the original feature map and P l+1 denotes the up-sampled high-level feature map.

$$\mathbf{P}_l = \mathbf{C}_l + \mathrm{UpSample}(\mathbf{P}_{l+1}) \tag{2}$$

In the formula, (P (l+1)) represents the upsampled result of P (l+1), matching its resolution with C (l). The upscaling operation is usually implemented using bilinear interpolation or inverse convolution (transposed convolution).

(2) Feature fusion. Through a top-down path, deep features are up-sampled step by step and pixel-wise added with corresponding shallow feature maps for fusion. [4] The study adopts a lateral connection method to achieve the fusion of feature maps at different levels, as shown in Equation (3).

$$\begin{aligned} P_4 &= C_4 \\ P_3 &= C_3 + \mathrm{UpSample}(P_4) \\ P_2 &= C_2 + \mathrm{UpSample}(P_3) \end{aligned} \tag{3}$$

In this formula, P2, P3 and P4 are the fused feature maps respectively, and UpSample represents the up sampling operation.

(3) Multi-scale detection. Object detection is performed on the fused feature map. By applying convolution operations on feature maps at different scales, bounding boxes and category predictions are generated. This approach enhances detection accuracy for small targets (such as distant pedestrians and compact vehicles) while maintaining capability for large targets (like nearby vehicles). The specific operation is illustrated in Equation (4).

$$\mathrm{Detection} = \mathrm{Conv}(P_2) + \mathrm{Conv}(P_3) + \mathrm{Conv}(P_4) \tag{4}$$

In vehicle and pedestrian detection, the integration of Fast Proposal Network (FPN) addresses the limitations of YOLO in small object detection. The high-resolution shallow feature maps effectively capture detailed information of distant small vehicles and pedestrians, while the rich semantic information from deep feature maps enhances recognition capabilities for large objects in complex backgrounds. Through feature fusion

and multi-scale detection, FPN improves YOLO's performance across various target scales, particularly demonstrating significant enhancements in detecting distant pedestrians and small vehicles in urban traffic scenarios.

2.2. Loss Function Adjustment

In the YOLO vehicle and pedestrian recognition task, the original loss function includes localization loss, confidence loss and classification loss. However, there are problems such as category imbalance and small target detection difficulty in practical applications. [5] Therefore, the following improvement schemes are proposed:

(1) The focal loss is introduced to replace the original classification loss and solve the problem of category imbalance. The focal loss increases the weight of difficult-to-classify samples while reducing the weight of easy-to-classify ones, thereby improving the detection accuracy of small targets and minority classes. As shown in Equation (5).

$$\text{Focal Loss} = -\alpha_t (1-p_t)^{\gamma} \log(p_t) \quad (5)$$

In this formula, α t is the balancing factor, γ is the adjusting factor, and p t is the predicted category probability.

(2) To address the issue of localization loss, we replace the original mean squared error (MSE) loss with Intersection-Union (IoU) loss to improve the accuracy of bounding box regression by directly measuring the overlap between predicted boxes and real boxes. In the specific design, for a given predicted box Bp and real boxBg, IoU is calculated Can be defined as Equation (6).

$$\text{IoU} = \frac{|B_p \cap B_g|}{|B_p \cup B_g|} \quad (6)$$

In the formula, |Bp∩Bg| represents the area of intersection between predicted boxes and real boxes; |Bp ∪ Bg| represents the area of union between predicted boxes and real boxes.

In order to calculate the IOU, we need to calculate the area of the intersection and union of the two boxes.

The intersection calculation is shown in Equation (7).

$$|B_p \cap B_g| = \max(0, x_{\min}^{\text{int}} - x_{\max}^{\text{int}}) \times \max(0, y_{\min}^{\text{int}} - y_{\max}^{\text{int}}) \quad (7)$$

In this formula, x min int and y min int respectively represent the upper left corner coordinates of the intersecting rectangular frame; x max int and y max int respectively represent the lower right corner coordinates of the intersecting rectangular frame.

The combined calculation method is shown in Equation (8).

$$|B_p \cup B_g| = |B_p| + |B_g| - |B_p \cap B_g| \quad (8)$$

IoU Loss directly measures the overlap between predicted and actual bounding boxes, which better aligns with the localization accuracy requirements of object detection tasks compared to traditional Mean Squared Error (MSE) Loss. By minimizing IoU Loss, the algorithm can more precisely adjust prediction boxes to better match real-world bounding boxes, thereby improving detection accuracy—particularly demonstrating significant effectiveness in boundary box regression tasks.

(3) The confidence loss is improved and the robustness of border box localization is further enhanced by introducing GIoU Loss, which considers the minimum envelope between predicted boxes and real boxes to provide meaningful gradient information when the border boxes do not overlap.

Combined with the improvement of FPN, multi-scale detection, IoU Loss and focal loss, the objective function of YOLO model can be expressed as Equation (9).

$$L = L_{cls} + \lambda_{box} L_{box} + \lambda_{iou} L_{iou} + \lambda_{focal} L_{focal} \quad (9)$$

In equation (9), L cls represents the classification loss; L box denotes the bounding box regression loss; L iou stands for the IoU loss; L focal refers to the focal loss. λ box, λ iou, and λ focal respectively indicate the weight coefficients of each component in the losses.

3. Experimental and result analysis

3.1. Experimental design

To validate the effectiveness of the YOLO-based vehicle and pedestrian recognition algorithm, we selected two datasets with extensive annotated data: COCO and KITTI. The COCO dataset contains 80 object categories, making it suitable for multi-scenario object detection. It provides a large number of vehicle and pedestrian samples to enhance model generalization capabilities. The KITTI dataset focuses on autonomous driving scenarios, delivering high-quality annotations for both vehicles and pedestrians.

The experiment was conducted on the NVIDIA Tesla V100 GPU platform using the PyTorch deep learning framework. The hardware configuration included 32 GB of video memory and high-performance computing resources to ensure high performance for large-scale data sets

Training methodology. The software environment incorporates CUDA and cuDNN libraries to accelerate deep learning model computations. Specific hyperparameter configurations for the experiments are detailed in Table 1. Loss values and accuracy metrics were monitored during training, with an early stopping strategy implemented to prevent overfitting. Model performance was evaluated using cross-validation.

Table 1. Hyperparameter design

Hyperparameter	Setting Value	Selection Reason
Learning Rate	0.001	Can converge quickly in the initial stage of training and maintain stability in the later stage, avoiding unstable training or slow speed.
Batch Size	16	The batch size determines the number of samples used for each parameter update. 16 is the optimal choice under the current hardware configuration.
Optimizer Parameter β_1	0.9	The default setting of β_1 for the Adam optimizer is 0.9, which has been proven to provide good convergence in most cases.
Optimizer Parameter β_2	0.999	The default setting of β_2 for the Adam optimizer is 0.999, which has been proven to provide good convergence in most cases.
Weight Decay	0.0005	Used for regularization to prevent the model from overfitting.
Momentum	0.9	Momentum parameters help accelerate the convergence of gradient descent.
Learning Rate Decay Strategy	Decay by 0.1 times every 10 Epochs	The learning rate decay strategy allows the model to converge quickly in the initial stage and be fine-tuned in the later stage to ensure the stability and accuracy of the final model.
Adjustment Factor γ	2.0	A common default value for dealing with class imbalance issues.
Balance Factor α	0.25	Improve the detection accuracy of minority classes.

3.2. Experimental Result

The performance metrics of the enhanced YOLO algorithm in vehicle and pedestrian detection tasks were evaluated, with results shown in Table 2. Detection effectiveness across different scenarios is detailed in Table 3. The improved algorithm demonstrates significant enhancements in both precision and recall rates, particularly excelling in complex background environments and small object detection scenarios. Specifically, it achieves approximately 12% improvement in precision and 10% increase in recall for distant small targets. In crowded urban traffic environments, precision and recall rates improve by about 9% and 8%, respectively. These outcomes indicate that through network architecture optimization and loss function adjustments, the refined YOLO algorithm can further enhance detection performance and robustness in vehicle and pedestrian recognition tasks.

Table 2. Performance index evaluation

Algorithm Version	Precision (P)	Recall (R)	F1 Score	Mean Average Precision (mAP)
Before Improvement	78.5%	75.3%	76.9%	72.4%
After Improvement	85.2%	82.1%	83.6%	80.7%

Table 3. Detection results under different scenarios

Scenario Type	Indicator	Before Improvement	After Improvement
Detection of Small Targets at Long Distances	Precision	68.3%	80.3%
	Recall	65.4%	75.8%
Congested Urban Traffic Environment	Precision	70.5%	79.5%
	Recall	68.2%	76.2%

Through this experiment, the following conclusions were drawn: (1) The introduction of FPN enabled YOLO to perform object detection across multiple scales of feature maps, particularly enhancing small object detection at lower-level high-resolution feature maps. These high-resolution feature maps retain more spatial detail information, allowing the model to demonstrate stronger detection capabilities for distant small objects such as pedestrians and compact vehicles. (2) The multi-scale detection strategy enables the model to simultaneously identify targets of varying sizes within a single image through convolution operations on feature maps at different scales. For small targets, their sparse pixel distribution in original images often leads to underrepresentation in low-resolution feature maps, whereas high-resolution feature maps better capture these smaller objects. (3) The focal loss function is introduced to address class imbalance issues, particularly for distant small targets that are frequently underrepresented in datasets. By weighting harder-to-detect samples, this approach enhances detection accuracy for minority classes.

To conduct a thorough analysis of computational costs before and after algorithm optimization, this study designed an experimental comparison between two machine learning algorithms in real-world applications. By running both the original YOLO and its optimized version on identical hardware configurations, we measured their inference time, memory usage, GPU utilization rate, and processing frame rate respectively. This methodology enabled us to determine the actual computational costs and performance improvements achieved through the optimizations. The hardware environment is described as follows.

GPU: NVIDIA Tesla V100 with 32 GB of video memory
CPU: Intel Xeon E5-2680 v4,28 cores 2.40 GHz
Memory: 128 GB DDR4
Storage: 2 TB SSD
Operating system: Ubuntu 20.04 LTS

Specific test items include: reasoning time (ms/ frame), memory usage (GB), CPU usage (%) and processing frame rate (fps).

The results are presented in Table 4. The improved solution shows a slight increase in inference time (by +18.4%) compared to YOLOv5, primarily due to the introduction of a feature pyramid network (FPN) and a sophisticated multi-scale detection strategy. Despite this, it still outperforms YOLOv5 in specific tasks. The enhanced YOLO model demonstrates a significant memory footprint increase (+51.1%), mainly attributed to additional feature maps and deeper convolutional layers. However, this trade-off brings notable improvements in small object detection capabilities and stronger background robustness. The optimized approach utilizes more GPU resources, achieving a 26.5% higher GPU utilization rate than YOLOv5. Although resource-intensive, this reflects the solution's efficient resource utilization in complex computational tasks. While frame rate decreases by 17.9% compared to YOLOv5 due to increased computation, the improved solution maintains relatively stable performance in handling complex scenarios, making it practical for high-precision small object detection applications.

Table 4. Test results

Indicator	YOLOv3	YOLOv5	Improved Scheme	Change of YOLOv5 vs Improved Scheme
Inference Time/ (ms/frame)	53.8	46.2	54.7	+18.4%
Memory Usage/ GB	5.4	4.7	7.1	+51.1%
GPU Utilization Rate/%	69.2	63.5	80.3	+26.5%
Processing Frame Rate/fps	18.5	21.7	17.8	-17.9%

4. Conclusion

By optimizing the network architecture of the YOLO algorithm and fine-tuning its loss function, this enhanced version demonstrates significantly improved performance in vehicle-pedestrian detection. The optimized algorithm exhibits superior accuracy and robustness when processing complex backgrounds and small targets, with enhanced detection capabilities that boost both adaptability and generalizability. These improvements not only strengthen the original algorithm's effectiveness but also provide robust support for advancing intelligent transportation systems and

autonomous driving technologies. The refined loss function proves particularly effective in addressing class imbalance issues and small target detection challenges within vehicle-pedestrian recognition tasks.

References

[1] Xue Z, Guo M, Fan H, et al. CorrBEV: Multi-View 3D Object Detection by Correlation Learning with Multi-modal Prototypes[C]//Proceedings of the Computer Vision and Pattern Recognition Conference. 2025, p. 27413-27423.

[2] Wang J H, Li B M. Research on real-time traffic flow collection system based on YOLO model[J]. Computer Technology and Development. Vol. 34 (2024) No. 9, p. 209-214.

[3] Song Cunli, Chai Weiqin, Zhang Xuesong. Road small object detection based on improved YOLO v5 algorithm [J]. Journal of System Engineering and Electronic Technology, 2024,46(10):3271-3278.

[4] Hong L, Zeng X J. Infrared target detection algorithm in complex street scenes based on improved YOLOv8[J]. Infrared Technology. Vol. 47 (2025) No. 5, p. 591-600..

[5] Li X, Li X R, Wang C, et al. Surface defect detection model of aero-engine based on improved YOLOv5[J]. Laser & Optoelectronics Progress. Vol. 60 (2023) No. 16, p. 1615007-1 - 1615007-10.