Attention Mechanism-Based Method for Intrusion Target Recognition in Railway

SHI Jiang[1], *BAI Dingyuan*[2], *GUO Baoqing*[2*], *WANG Yao*[2], *RUAN Tao*[2]

1. CHN Energy ShuoHuang Railway Development Company Ltd, Beijing 100080, P. R. China;

2. School of Mechanical and Electronic Control Engineering, Beijing Jiaotong University, Beijing 100044, P. R. China

(Received 7 May 2024; revised 10 August 2024; accepted 13 August 2024)

Abstract: The detection of foreign object intrusion is crucial for ensuring the safety of railway operations. To address challenges such as low efficiency, suboptimal detection accuracy, and slow detection speed inherent in conventional comprehensive video monitoring systems for railways, a railway foreign object intrusion recognition and detection system is conceived and implemented using edge computing and deep learning technologies. In a bid to raise detection accuracy, the convolutional block attention module (CBAM), including spatial and channel attention modules, is seamlessly integrated into the YOLOv5 model, giving rise to the CBAM-YOLOv5 model. Furthermore, the distance intersection-over-union_non-maximum suppression (DIoU_NMS) algorithm is employed in lieu of the weighted non-maximum suppression algorithm, resulting in improved detection performance for intrusive targets. To accelerate detection speed, the model undergoes pruning based on the batch normalization (BN) layer, and TensorRT inference acceleration techniques are employed, culminating in the successful deployment of the algorithm on edge devices. The CBAM-YOLOv5 model exhibits a notable 2.1% enhancement in detection accuracy when evaluated on a self-constructed railway dataset, achieving 95.0% for mean average precision (mAP). Furthermore, the inference speed on edge devices attains a commendable 15 frame/s.

Key words: foreign object detection; railway protection; edge computing; spatial attention module; channel attention module

CLC number: TN925 **Document code:** A **Article ID:** 1005-1120(2024)04-0541-14

0 Introduction

With the continuous expansion of the railway network across the nation, its pivotal role in the daily lives of individuals is progressively accentuated. Nonetheless, the constant threat of foreign object intrusion poses a significant challenge to railway safety operations. Urgency mounts for the rapid and precise detection of foreign objects to ensure the secure operation of trains. The method of foreign object detection based on image processing primarily entails differential processing of the target image and the background image to yield a grayscale representation capturing motion entities. Subsequently, employing apt thresholds, the grayscale image undergoes filtration to isolate regions exhibiting motion characteristics[1-2]. Cui et al.[3] devised a swift sparse detection approach, attaining superior detection outcomes in real-time surveillance footage. Wang et al.[4] delineated moving targets from the background via frame differencing and background subtraction, thereby instituting a mechanism for alerting pedestrian intrusion. These efforts, despite depending on extensive manual feature annotation, struggle to adapt to changing environments. They work very well in good lighting conditions, but their performance drops significantly in bad weather.

The exponential advancement of deep learning within the realm of object detection has markedly elevated the precision and dependability of object de-

*Corresponding author, E-mail address: bqguo@bjtu.edu.cn.

How to cite this article: SHI Jiang, BAI Dingyuan, GUO Baoqing, et al. Attention mechanism-based method for intrusion target recognition in railway[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2024, 41(4): 541-554.

http://dx.doi.org/10.16356/j.1005-1120.2024.04.010

tection[5-7].

For example, in the field of road traffic, Wang et al.[8] developed a monocular depth estimation network called SABV-Depth based on self-attention mechanisms. This network retains multi-scale information by enhancing information exchange between different layers of the network and includes an internally connected decoder module designed to recover depth maps with sharp edge contours, thus allowing for more accurate target identification within the depth map. Additionally, Wang et al.[9] improved the traditional feature pyramid network by integrating an adaptive attention module (AAM) and a feature enhancement module (FEM) and combined it with the YOLO model to enhance the detection accuracy of traffic signs.

However, there are significant differences between road traffic and railway traffic. For instance, in railway scenarios, it is challenging to obtain ground truth for distances, and the scales of the targets to be identified vary greatly. These factors make the direct application of the aforementioned methods difficult in railway contexts. Nevertheless, their approaches to feature enhancement offer valuable insights for this research.

In the field of rail traffic, Yang[10] introduced the feature fusion enhancement network (FFE-Net) architecture, significantly enhancing the detection prowess pertaining to distant small-scale and elongated targets. Shi[11] integrated the focal loss mechanism and DIoU Loss function into the single shot MultiBox detector (SSD) network framework, ameliorating the model's detection efficacy and fortifying the detection accuracy when foreign objects encroaching upon railway domains.

The object detection method based on deep learning creates a multi-layered detection network that can autonomously extract data features. However, deployment typically requires substantial computational resources, limiting them to operation on powerful servers. Conversely, cameras stationed at the forefront may contend with issues such as data loss and transmission latency attributable to constraints in network bandwidth during the conveyance of image data[12]. Presently, the bulk of edge-based object detection methodologies cater to diminutive environments[13-14], mandating a restricted monitoring scope and subdued detection velocity. Conversely, railway surveillance encompasses intricate settings and engenders copious volumes of monitoring data during routine operations, thereby imposing heightened requisites on detection accuracy and velocity.

Integrating deep learning algorithms with edge computing aims to achieve real-time performance for foreign object detection systems while maintaining high accuracy levels. By employing enhancement techniques like attention modules and model compression, on-site processing of image data enables real-time detection of intrusion targets.

1 Intrusion Target Detection Model Based on Attention Mechanism and Model Pruning

The traditional image processing method for target detection faces challenges such as lower detection accuracy and limited ability to generalize. In this study, we propose a novel target detection model founded on deep learning algorithms. The schematic overview of the algorithmic framework is delineated in Fig.1. Initially, an attention module is incorporated to bolster the detection precision of the model. Subsequently, network pruning is executed on the refined model to enhance the detection velocity.

1. 1 Feature enhancement module

Due to the complex and variable terrain along railway tracks, images captured by cameras contain a large amount of noise, leading to potential false positives and false negatives. Therefore, this paper introduces an attention mechanism to effectively filter out noise information, automatically adjust the detection focus, and enhance the model's ability to perceive targets in complex scenarios.

The convolutional block attention module (CBAM)[15] is adopted as the feature information enhancement module for intrusion target detection models in railway scenes. Its structure, as shown in Fig.2, iteratively calculates attention weights from

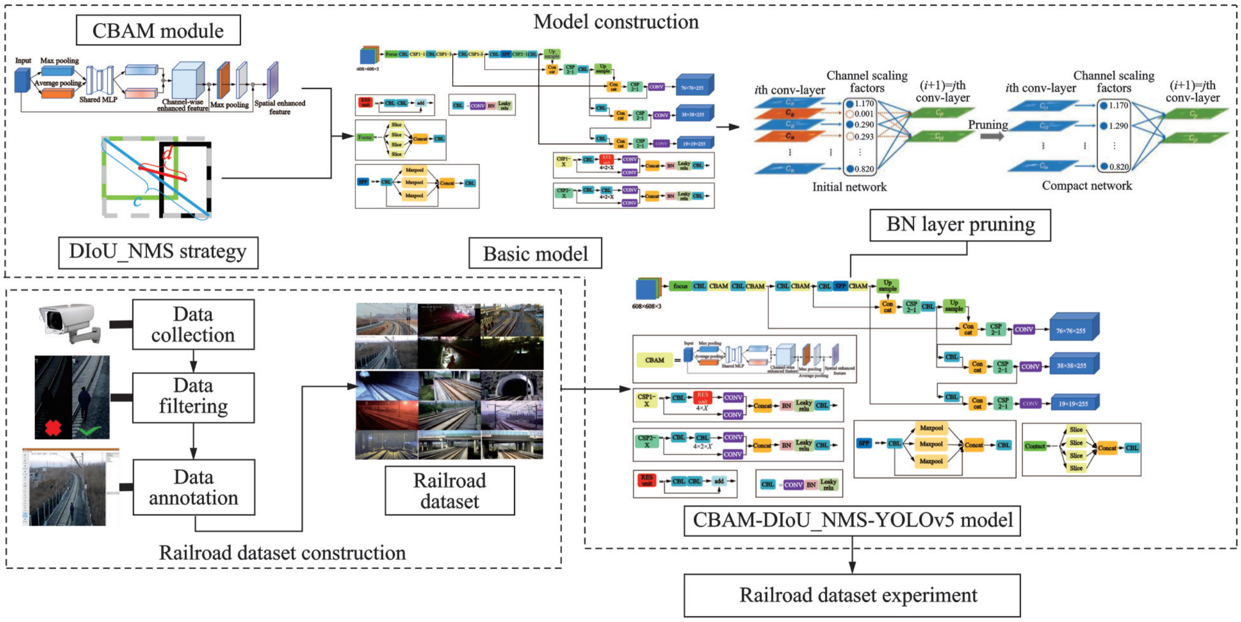


Fig.1 Overall structure of the intrusion target detection model

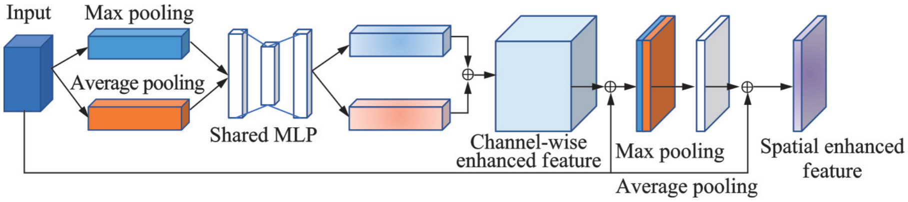


Fig.2 Schematic diagram of feature enhancement module

both spatial and channel dimensions. After the computation, the obtained attention weights are applied to the input feature maps through parameter-wise multiplication, adaptively adjusting the input features. This enables the module to simultaneously focus on both channel and spatial information in the feature maps, thereby comprehensively capturing the importance of features.

The channel attention module operates on the input feature tensor $F_1 \in \mathrm{R}^{H \times W \times C}$. Upon two successive global average pooling and max pooling operations, two distinct sets of channel information are derived from this feature, each possessing dimensions of $1 \times 1 \times C$. Subsequently, these two sets of channel information undergo input into a shared convolutional layer for further feature extraction. The initial layer of the network comprises C/r neurons, while the subsequent layer comprises C neurons. Following this, features obtained from the two layers of the network are amalgamated via summation. The resultant features are then subjected to activation by the sigmoid function, thereby generating attention weight coefficients denoted as M_c. These coefficients are subsequently applied to the original features F through element-wise multiplication, thereby yielding scaled new features. Eq.(1) delineates the computation method for the channel attention weight coefficients.

$$M_c(F) = \sigma(\mathrm{MLP}(\mathrm{AvgPool}(F)) + \mathrm{MLP}(\mathrm{MaxPool}(F))) = \sigma(\boldsymbol{W}_1(\boldsymbol{W}_0(F^c_{\max})) \quad (1)$$

where σ represents the Sigmoid activation function, MLP the multilayer perceptron operation, AvgPool the average pooling operation, MaxPool the max pooling operation, and $F^c_{\max}$ the feature obtained after the input feature F undergoes max pooling operation along the channel dimension. $\boldsymbol{W}_0$ and $\boldsymbol{W}_1$ are the shared weights of the MLP, $\boldsymbol{W}_0 \in \mathrm{R}^{C/r \times C}$, $\boldsymbol{W}_1 \in \mathrm{R}^{C \times r/C}$.

The input feature for the spatial attention module is denoted as $F_2 \in \mathrm{R}^{H \times W \times C}$. Initially, the input feature undergoes pooling operations, encompassing average pooling and max pooling, aimed at diminishing the feature scale. Subsequently, the Sigmoid function is employed to activate and derive the weight coefficients M_s. These coefficients are then applied to the input feature F_2 via element-wise multiplication, thereby yielding the spatial attention features. The explicit procedure is encapsulated within Eq.(2).

$$\begin{aligned} M_s(F)= \\ &\sigma\left(f^{7\times7}([\mathrm{AvgPool}(F),\mathrm{MaxPool}(F)])\right)= \\ &\sigma\left(f^{7\times7}([F^{S}_{\mathrm{avg}};F^{S}_{\max}])\right) \end{aligned} \tag{2}$$

where $f^{7\times7}$ represents a convolutional layer with a 7×7 convolutional kernel, F^{S}_{avg} the feature obtained after the input feature F undergoes average pooling along the spatial dimension, and $F^{S}_{\max}$ the feature obtained after the input feature F undergoes max pooling along the spatial dimension.

Following the integration of the attention mechanism, the network architecture depicted in Fig.3 is presented: Leveraging the foundational YOLOv5 network model, the CBAM-YOLOv5 model is derived by substituting the CSP1_1, CSP1_2, and CSP1_3 modules, alongside the CSP2_1 and CBL modules within the neck network, with CBAM modules. The CBAM module prioritizes the exploration of feature channel relationships to yield channel attention maps. To streamline the computation of channel attention, the algorithm initiates by downsizing the spatial dimensions of the input feature map. Subsequently, the module executes cascaded pooling operations on the input feature map to collate spatial features. Following this, the average features and spatial features are fed into a weight-sharing network composed of MLP to procure preliminary channel attention embeddings. Spatial attention is employed as an augmentation mechanism for an alternate dimension, where the algorithm initially generates feature representations via average pooling and max pooling operations along the channel axis, and subsequently formulates spatial attention embeddings through convolutional layers. Upon acquiring the two attention embeddings, the model amalgamates the channel information of the feature map through pooling operations, subsequently concatenates the features, and executes convolution operations, culminating in the generation of enhanced attention feature maps.

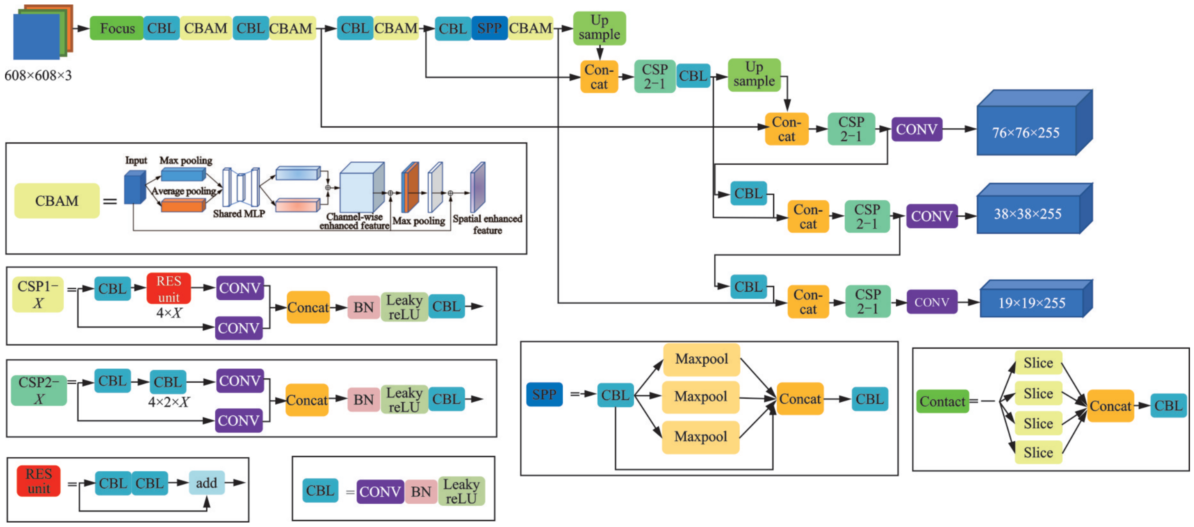


Fig.3 CBAM-YOLOv5 network architecture diagram

1.2 DIoU_NMS strategy

In object detection, the non-maximum suppression (NMS) operation is commonly employed for post-processing. The original post-processing technique utilized in YOLOv5 is weighted NMS[16], wherein the model computes a weighted average of confidence between the highest-confidence anchor box H and candidate boxes surpassing a certain threshold to determine whether to retain or discard the candidate boxes. The weighted NMS algorithm exhibits enhanced stability and contributes to improvements in the model's detection accuracy and recall. Nonetheless, its sequential processing mode can potentially hamper detection speed. Furthermore, practical detection scenarios have unveiled instances of missed detections when two objects of the same class are concurrently present, particularly when one is significantly occluded.

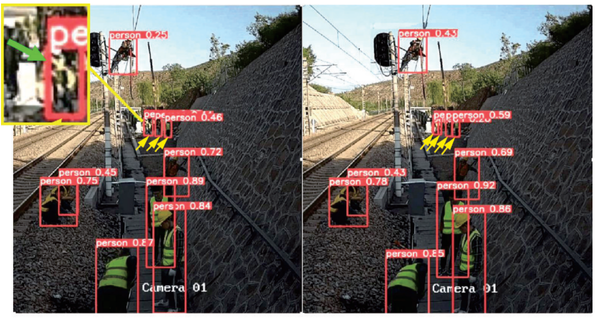
In order to mitigate missed detections and augment detection accuracy, this study embraces DIoU_NMS as the post-processing step for the model. Serving as a more balanced approach, it not only considers the traditional overlap area between predicted boxes but also introduces the calculation of the distance between the centers of two boxes. The calculation methodology is delineated as

$$S_i=\begin{cases} S_i & \mathrm{IoU}-R_{\mathrm{DIoU}}(H,B_i)<\varepsilon \\ 0 & \mathrm{IoU}-R_{\mathrm{DIoU}}(H,B_i)\geqslant\varepsilon \end{cases} \tag{3}$$

$$R_{\mathrm{DIoU}}=\frac{\rho^2(b,b^{\mathrm{gt}})}{c^2} \tag{4}$$

where S_i represents the class score, b^{gt} the center point position of the box with the maximum confidence score, and b the center point position of the box being analyzed. $\rho(\cdot)$ is used to calculate the Euclidean distance, and c the diagonal length of the minimum enclosing rectangle containing the two boxes.

As illustrated in Fig.4, the utilization of DIoU_NMS notably enhances the detection performance of occluded targets. While the algorithm employing the weighted_NMS strategy manages to identify three maintenance personnel, the network leveraging DIoU_NMS successfully detects the fourth person, despite being partially occluded.



(a) Weighted_NMS (b) DIoU_NMS

Fig.4 Comparison of model detection effects based on different NMS strategies

1.3 Model lightweight design based on BN layer pruning

Expanding the number of neural network layers and augmenting learnable parameters can indeed yield substantial enhancements in the detection accuracy of object detection models. However, this enhancement comes at the expense of diminished detection speed and escalated consumption of computational resources. Consequently, there is a pressing need to employ model compression techniques to ameliorate model size, expedite model inference speed, and achieve superior detection performance at the edge.

Traditional model pruning algorithms make significant progress in accelerating model detection speed while minimally compromising detection accuracy. They achieve this by removing redundant convolutional kernels, eliminating parameters with minimal impact, and fine-tuning training processes.

Traditional pruning methodologies[17-20] predicate their operations on the L1 norm of each convolutional kernel, employing a preset pruning threshold. Convolutional kernels that exceed the L1 norm threshold are retained, while those below it are considered insignificant due to their minimal impact on feature computation. However, determining a suitable threshold range beforehand remains challenging, necessitating iterative testing with incremental adjustments. This involves comparing the model's parameter count and detection accuracy before and after pruning to identify an effective pruning threshold. Following the establishment of model pruning accuracy, fine-tuning training becomes imperative after each pruning iteration, thereby engendering considerable computational complexity and protracted experimental cycles. Thus, in a bid to enhance the efficiency of model pruning, this study introduces a method grounded in sparse training to ascertain the optimal threshold. Upon the completion of sparse training, the maximum threshold can be discerned, and to forestall precipitous declines in detection accuracy, the pruning rate is set to its maximum value, with only parameters in the batch normalization (BN) layer being pruned.

The calculation formula for the BN layer is given by

$$\hat{z}=\frac{z_{\text{in}}-\mu_{\text{B}}}{\sqrt{\sigma_{\text{B}}^{2}+\varepsilon_{0}}},\ z_{\text{out}}=\gamma\hat{z}+\beta \tag{5}$$

where $\hat{z}$ is the normalized output result, z_{in} the output matrix of the convolutional layer, which serves as the input matrix for the BN layer, z_{out} the corresponding output of the BN layer in that channel, σ_{B} the variance parameter of the BN layer, μ_{B} the mean coefficient of that layer, ε_0 a very small number to avoid calculation errors caused by a denominator being zero, γ the scaling sparse for normalization, and β the bias repair parameter for normalization operations.

Eq.(5) demonstrates that z_{out} exhibits a positive correlation with the coefficient γ. When γ approaches zero, the resultant activation value becomes exceedingly diminutive. Consequently, the sensitivity of the output of the BN layer to z_{in} diminishes, rendering its contribution to model computation negligible. Thus, it can be deemed as superfluous and removable.

Grounded on the aforementioned principle of model simplification, the BN layer pruning procedure outlined in this study is depicted in Fig.5. Initially, the CBAM-DIOU_NMS-YOLOV5 model is employed as input for the pruning network, with appropriate sparsity rate parameters being configured. Subsequently, L1 sparse training is conducted. Upon acquiring the sparse model, channels with scaling factors below the threshold are eliminated to obtain the preliminary pruned model. Secondary accuracy assessment is then conducted to ascertain whether a notable loss in accuracy ensues. If such a loss is observed, accuracy callback training is initiated; otherwise, the final pruned model is directly outputted.

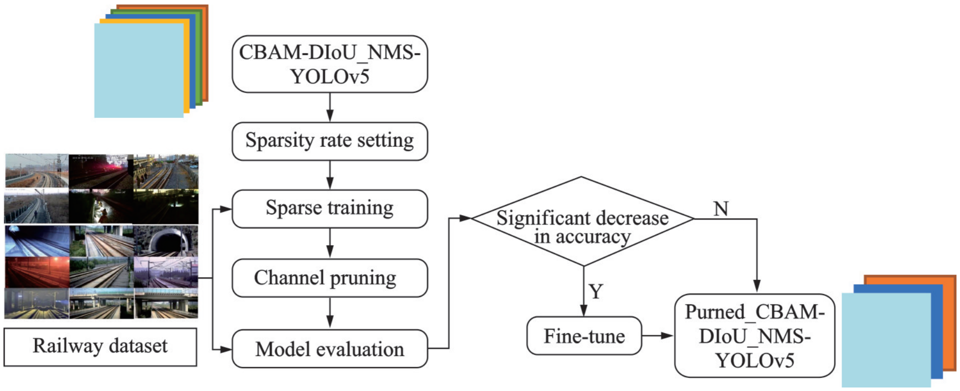


Fig.5 Flow chart of BN layer pruning based operation

1.4 Pseudocode of the proposed algorithm

We provide the pseudocode for the proposed CBAM-DIoU_NMS-YOLOv5 model, as shown in Algorithm 1. This pseudocode outlines the overall construction process of the model.

Algorithm 1 Pruned CBAM-DIoU_NMS-YOLOv5

```
Input: Training set
Output: Model weights and evaluation metrics
Begin:
(1) Initialize. Initialize model parameters, including weights, epochs, batchsize, img_size, etc.
(2) Data augmentation. Nine images are randomly combined with Mosaic-9.
(3) While epoch < epochs
(4) CBAM attention mechanism. CSPdarknet53 is used as the backbone network for training, and CBAM attention mechanism is added.
(5) DIoU_NMS strategy. DIoU_NMS is employed to constrain multiple potential detections of the same target.
(6) Loss Function. CIoU loss function is used to optimize the regression of boundary box.
(7) End while
(8) Evaluation. Calculate evaluation metrics, including the mean average precision (mAP).
```

```
(9) Sparsity rate $\lambda \leftarrow 0.001$
(10) Sparse training. Begin sparse training by adjusting the distribution of BN layer scaling factors.
(11) Prune Rate $\leftarrow 30\%$
(12) Pruning. Start removing channels with scaling factors below the threshold according to the specified pruning rate.
(13) Evaluation. Re-calculate evaluation metrics.
(14) If mAP(pruned) $\ll$ mAP(original), then
(15) Fine-tune training
(16) End if
Return: Model weights and evaluation metrics
```

2 Edge Detection System Solution Based on Edge Computing

To mitigate the challenges of data loss and latency attributed to backend processing and to enhance the real-time detection of foreign objects in railway environments, this study implements a combination of hardware and algorithms directly at railway sites via edge computing. This approach enables foreign object detection tasks to be completed at the frontend, eliminating the necessity of transmitting data to remote servers for processing.

2.1 Overall scheme design

Starting with the operational scenario of the detection system and considering the requirements for both detection accuracy and real-time performance, the devised scheme is delineated in Fig.6. The system architecture is bifurcated into two primary components: The edge and the backend.

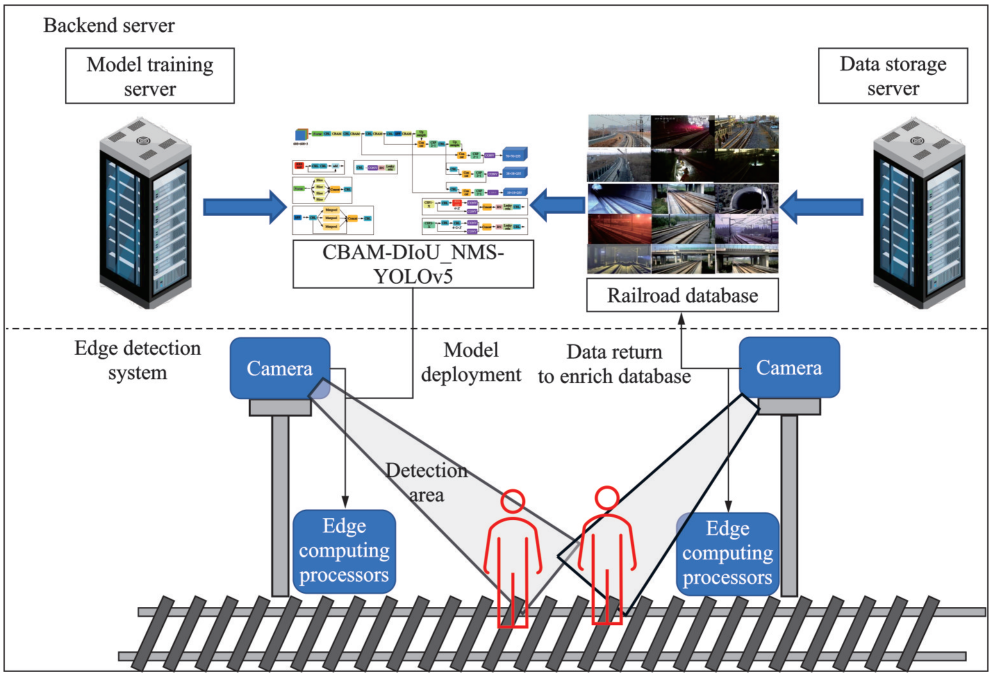


Fig.6 Overall system architecture diagram

In accordance with the operational scenario, the comprehensive system can be segregated into the backend server segment and the edge detection system segment. The backend, furnished with high-performance data servers, is primarily allocated for training high-accuracy detection models. Conversely, the edge component is deployed within the pertinent sections of interest. It encompasses cameras and processors aimed at facilitating real-time image acquisition of railway scenes and the detection of foreign object intrusion.

2.2 Acceleration optimization scheme based on TensorRT

Owing to constraints in processor computing power, the detection speed of algorithms deployed at the edge is anticipated to diminish. Hence, to fulfill the real-time imperatives of railway security, it becomes imperative to optimize the inference of detection leveraging an inference library at the edge, thereby enhancing the model detection speed.

Following the training of the initial model on the model training server, TensorRT[21-22] is harnessed to generate serialized files, thereby constructing optimized engines tailored for inference. TensorRT undertakes the reconstruction of deep learning networks through the fusion of layers or tensors, amalgamates operation modules[23-24], and accelerates network computation efficiency. Additionally, it economizes network computational and memory space during inference through the substitution of low-precision data types.

3 Experimental Comparison and Analysis

3.1 Railway dataset introduction

Due to significant differences between real railway scene images and those in general datasets, ensuring accurate detection algorithms in railway scenarios is crucial. To achieve this, we annotate real images captured from monitoring cameras along actual railway lines such as Baolan and Guangzhou-Shenzhen lines. Additionally, to mitigate overfitting or underfitting issues caused by high scene repetitiveness or insufficient specific samples, we incorporate some general scene images. Examples of representative images are shown in Fig.7.

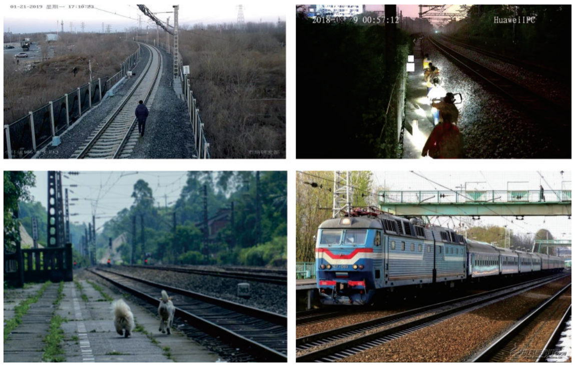


Fig.7 Representative images from the railway scene dataset

We conduct a detailed analysis of the target distribution within the dataset, categorizing specific objects found in railway scenes. The dataset comprises 29 330 instances of pedestrian targets, 422 instances of animal targets, and 4 761 instances of train targets.

In addition to object type distribution, we also analyze the scale distribution of these targets based on the COCO dataset standards. According to the COCO dataset, small targets are defined as those with pixel areas less than 32×32, medium targets have pixel areas between 32×32 and 96×96, and large targets have pixel areas greater than 96×96. According to our analysis, the distribution of target scales in the railway dataset is as follows: Small targets make up 58.9% of the dataset, medium targets constitute 18.8%, and large targets account for 22.3%.

3.2 Algorithm testing experiments based on the backend server

The algorithm testing experiments are conducted on the Ubuntu 16.04 system using the PyTorch framework. A single NVIDIA 1060 GPU facilitates the training process. The training dataset comprises 20 000 images sourced from a self-constructed railway sample database, categorized into four classes: Scenarios of pedestrian intrusion, animal intrusion, train operation images, and empty scenes. The dataset is partitioned into training and validation sets at an 8∶1 ratio. The training regimen extends over 300 epochs, commencing with an initial learning rate set to 1×10^{-4}, with each batch comprising eight images.

Firstly, we compare the baseline model of our method, YOLOv5, with other advanced models from the same era, specifically PPYOLO[25] and YOLOX[26], using the COCO public dataset. This comparison highlights the potential of our baseline in the object detection task.

PPYOLO, developed by Baidu, is an advanced object detector built upon YOLOv3. It enhances the network's feature extraction capabilities by introducing deformable convolutions on top of the ResNet-50 architecture. PPYOLO also employs several augmentation strategies to improve detection performance, including increasing the batch size, using exponential decay as a loss function, and incorporating DropBlock regularization.

Similarly, YOLOX, proposed by Megvii Technology, is another object detection model built on YOLOv3. It enhances the base model by integrating various strategies, such as exponential moving average (EMA), binary cross entropy loss for separate training of the classification and localization branches, multiple data augmentation techniques, and a decoupled head design to further boost model performance.

We report the evaluation results of the three models on the COCO dataset in Table 1. The performance metrics clearly illustrate the detection potential of our baseline model, YOLOv5.

Table 1 Comparison of three different mainstream detections on COCO dataset

Model	mAP@0.5	Detection speed/(frame•s^{-1})
PPYOLO	65.2	72.9
YOLOXx	51.2	51.2
YOLOV5x	66.6	66.6

Next, we conduct ablation experiments on the railway dataset using YOLOv5 as the baseline model. These experiments aim to explore the impact of the CBAM module and DIoU_NMS on the model's performance. Additionally, we introduce the SE module[27] as a comparative counterpart to the CBAM module. The comparative outcomes of the acquired performance metrics are presented in Table 2. It encapsulates mAP across diverse network architectures, mAP for distinct categories, model size, and detection speed. Examination of the experimental findings unveil that the incorporation of CBAM modules results in an augmentation of model parameters, consequently impeding the model's inference speed. Nonetheless, the enhancement in detection accuracy is noteworthy: mAP values for pedestrians, trains, and animals attain 0.944, 0.875, and 0.76, respectively. Furthermore, advancements in post-processing methodologies facilitate an uptick in detection accuracy for pedestrians, without adversely impacting the model's detection speed and size.

Table 2 Comparison of detection effects of different models

Network architecture	mAP					Model size/MB	Detection speed/(frame•s^{-1})
	Across diverse network architectures	Excluding animals	People	Train	Animal		
YOLOv5	0.823	0.895 0	0.920	0.870	0.68	170.2	20.0
SE-YOLOv5	0.835	0.897 5	0.920	0.875	0.71	170.2	19.6
CBAM-YOLOv5	0.860	0.910 0	0.944	0.875	0.76	175.6	18.9
CBAM-DIoU_NMS-YOLOv5	0.862	0.913 0	0.950	0.875	0.76	175.6	18.9

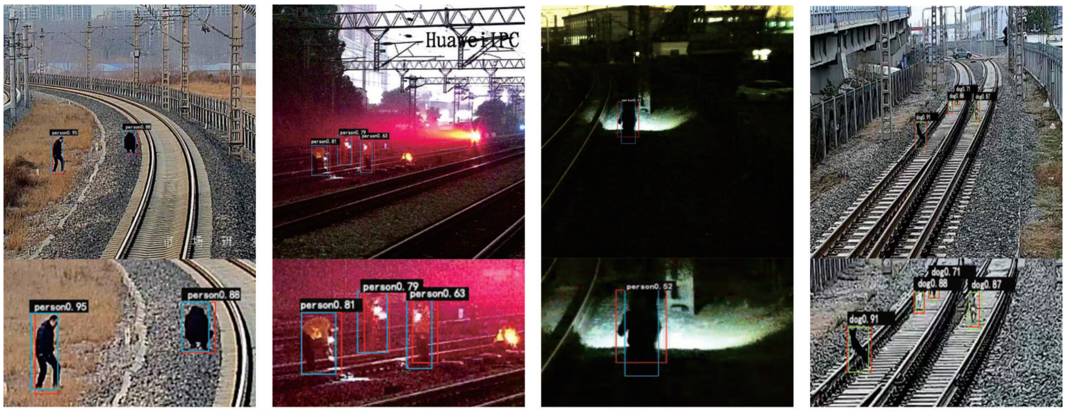
In the real-world operational environment of trains, the detection model faces various complex scenarios. To validate the enhanced detection model's generalization capability across diverse scenarios, CBAM-DIOU_NMS-YOLOV5 is scrutinized across four railway scenes, with the actual detection results showcased in Fig. 8. The red squares in the test outcomes denote the ground truth of various targets in the images, while the blue and green boxes represent the model's inference results for pedestrian and animal targets, respectively.

(1) The image portrays different pedestrian postures on the track under favorable daylight conditions. The network assigns elevated classification scores to pedestrians and adeptly predicts their positions.

(2) This image captures scenes under conditions of blur and spotlighting. Due to intense illumination, pedestrians exhibit fewer discernible features, resulting in a decrease in classification confidence.

(3) This image depicts a dimly lit nighttime scene with limited lighting. The pedestrian facing away from the camera and holding light sources shows fewer distinctive features, thus resulting in poorer output information in the image. Confidence in the detection results of pedestrians decreases to very low levels.

(4) Within this image lie multiple small animal targets. Despite their muddy fur colors, which con-



(a) Normal light (b) Low resolution (c) Dark scene (d) Animals

Fig.8 Detection results in different scenarios

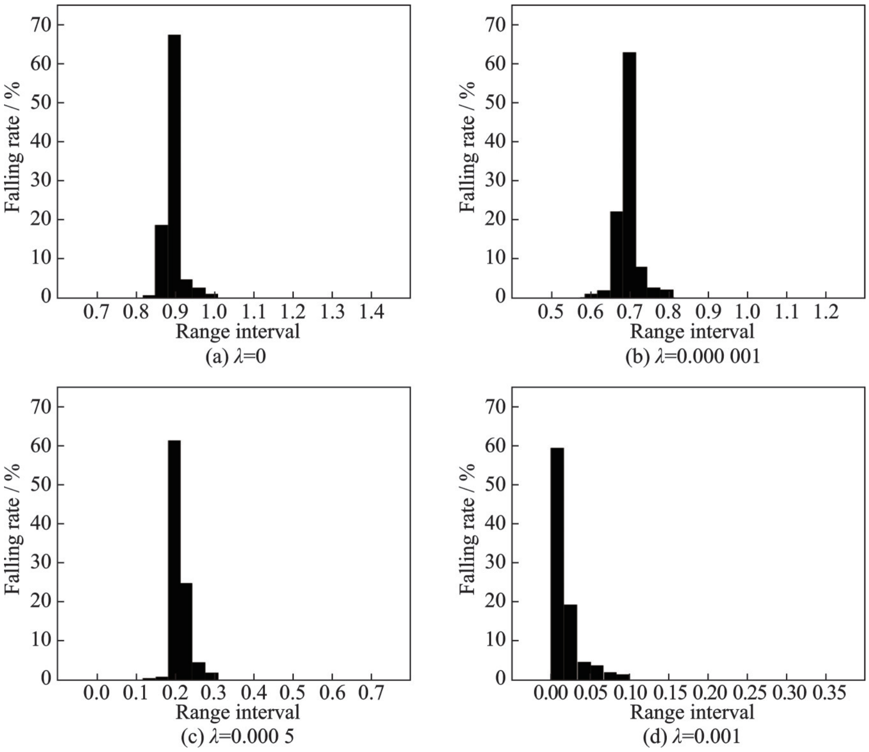
tribute to reduced contrast with the surrounding environment, the model aptly detects all four animals accurately.

3. 3 Model pruning experiment

While CBAM-DIoU_NMS-YOLOv5 shows excellent detection accuracy, its large size impedes satisfactory detection speed. Therefore, experiments are conducted to prune the model based on sparse training of BN layers.

The setting of the sparsity rate λ directly affects the outcome of the model's sparse training. To balance effective pruning with the minimization of resource and time consumption in the model compression experiment, it is crucial to conduct a statistical analysis of the weight factor distribution corresponding to different sparsity rates. Fig.9 illustrates the distribution of weight factors when λ is set to 0.001, 0.000 5, 0.000 001, and 0. The horizontal axis of the graph represents the range intervals of the weight factors, while the vertical axis denotes the proportion of factors falling within each interval relative to the total number of factors.

From Fig.9, it can be observed that in the initial state, where the sparsity rate λ is set to 0, the



(a) λ=0 (b) λ=0.000 001 (c) λ=0.000 5 (d) λ=0.001

Fig.9 Distribution of scaling factors corresponding to different sparsity rates

distribution of BN layer scaling factors approximates a normal distribution. As the sparsity rate increases from 0 to 0.001, the distribution of scaling factors progressively shifts towards 0. When λ reaches 0.001, the optimal sparsity training effect is achieved, with a significant portion of the scaling factors approaching zero. This indicates the emergence of many weights in the network that can potentially be pruned.

Under the premise of setting the sparsity rate $\lambda=0.001$, experiments are conducted to determine the optimal pruning rate. Pruning rates of 10%, 20%, 30%, 40%, and 50% are respectively used, and the resulting model pruning outcomes are shown in Table 3.

Table 3 Metrics of the model at different pruning rates

Pruning rate/%	Model size/MB	mAP	Accuracy loss/%
0	175.6	0.913	0
10	158.0	0.913	0
20	140.5	0.901	1.3
30	123.0	0.897	1.7
40	105.4	0.865	4.9
50	Fail	Fail	Fail

From the analysis of the data in Table 3, it is observed that when the pruning rate exceeds 40%, there are no channels left in the model that can be removed, resulting in failed pruning operations. Setting the pruning rate at 30% represents a critical point where the model achieves a relatively balanced trade-off between volume reduction and accuracy loss. Specifically, the model achieves a 30% reduction in volume at the cost of a 1.7% decrease in detection accuracy.

Following fine-tuning training, the model's detection accuracy rebounds to 0.905, registering a marginal decrease of only 0.008 compared to the original model. Notably, the pruned model achieves a detection speed of 24.3 frame/s , marking a nearly 30% enhancement relative to the model before pruning. Detailed performance metrics are delineated in Table 4.

Table 4 Changes of detection index of the model before and after cropping

Model	Model size/ MB	mAP	Detection speed/ (frame•s^{-1})
Original model	175.6	0.913	18.9
Pruned model	123.0	0.905	24.3

The obtained results underscore the effectiveness of incorporating the CBAM module and DIoU_NMS mechanism to enhance model detection accuracy, alongside executing model pruning operations to bolster algorithm processing speed at the edge. Moreover, the detection outcomes of the model across diverse scenarios for varied targets, particularly in environments characterized by strong light interference and low-light conditions, underscore the enhanced network's robustness. This attests to the improved network's capability to effectively fulfill the requisites of practical applications across disparate environments.

3.4 Experiments in real scene

To validate the practical effectiveness of the railway foreign object intrusion detection system proposed in this study in real railway environments, field testing experiments are conducted, as illustrated in Fig.10. Cameras are strategically installed at the entrance of the Shihekou Tunnel and the channel door area, with image data being transmitted to the edge processor Jetson Xavier. Renowned for its compact size, facile on-site deployment, and robust image processing capabilities, the Jetson Xavier facilitates intrusion target detection leveraging the deployed CBAM-DIoU_NMS-YOLOv5 model.

To verify the detection performance of the system in practical operational scenarios, we conduct statistics on the operational and alert conditions of the deployed system. Using the method of averaging across multiple measurements, we evaluate the CBAM-DIoU_NMS-YOLOv5 model proposed in this study for its GPU memory usage and power consumption during the inference process.

According to the statistics, during the inference of a single image, the CBAM-DIoU_NMS-YOLOv5 model requires 1.4 GB of GPU memory,

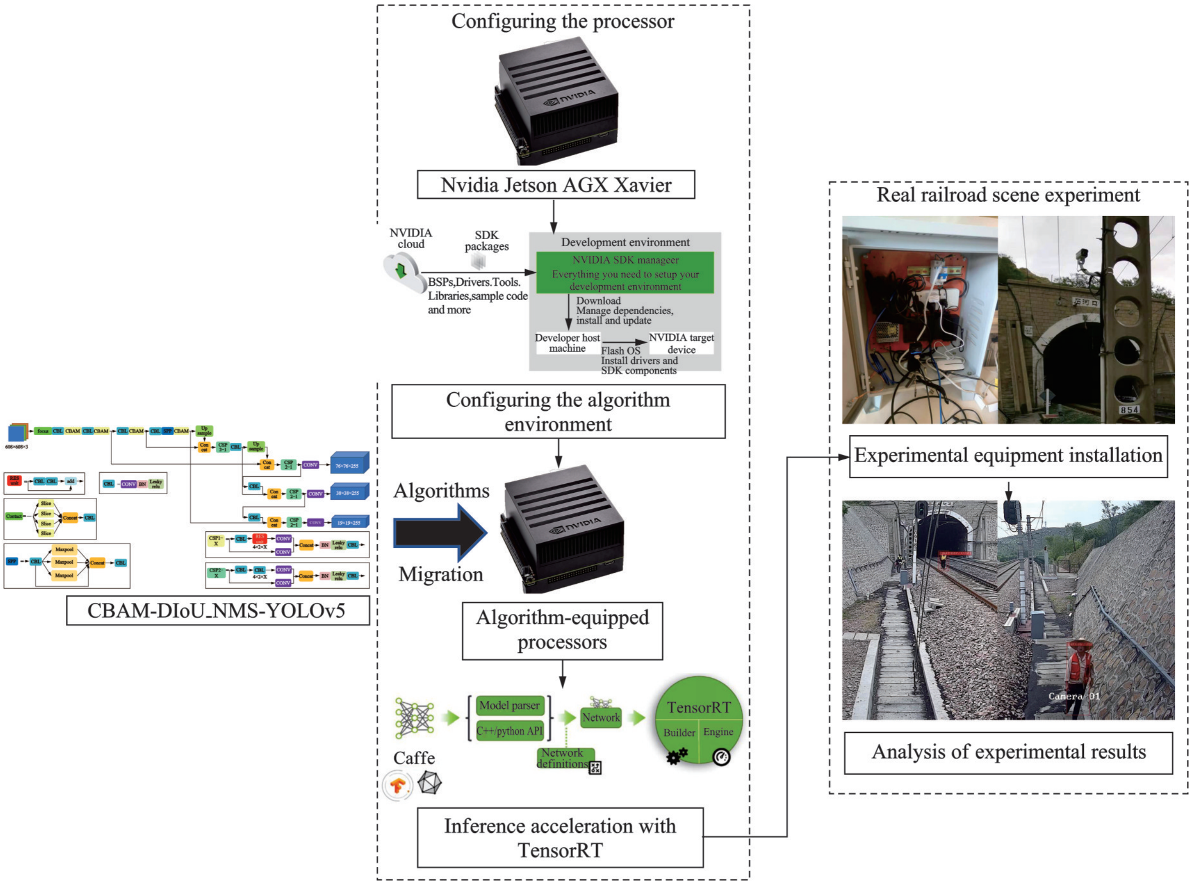

Configuring the processor
Nvidia Jetson AGX Xavier
NVIDIA cloud
SDK packages
BSPs,Drivers.Tools. Libraries,sample code and more
Development environment
NVIDIA SDK manageer
Everything you need to setup your development environment
Download
Manage dependencies, install and update
Developer host machine
Flash OS
Install drivers and SDK components
NVIDIA target device
Configuring the algorithm environment
Algorithms
Migration
CBAM-DIoU_NMS-YOLOv5
Algorithm-equipped processors
Caffe
Model parser
C++/python API
Network
Network definitions
TensorRT
Builder
Engine
Inference acceleration with TensorRT
Real railroad scene experiment
Experimental equipment installation
Analysis of experimental results
Camera 01

Fig.10 System test experiment based on Jetson Xavier

with an inference time of 0.067 s and an average power consumption of 22.172 W. We record a total of 160 alert messages, consisting of 148 accurate alerts and 12 false alarms, resulting in an alert accuracy of 92.5%. The real-time detection speed reaches 15 frame/s. Several alarm images are delineated in Fig.11.

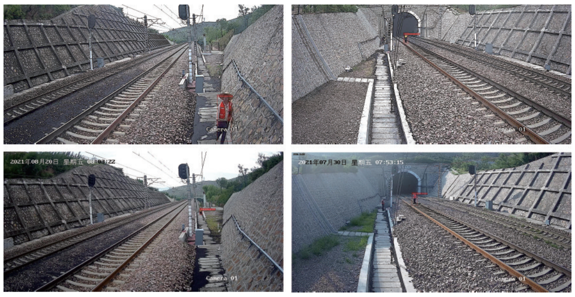


Fig.11 Several alarm images

4 Conclusions

This paper tackles the challenges of delayed detection and lack of sensitivity to railway scene targets prevalent in existing railway intrusion detection methods, and presents a design and implementation of a railway foreign object detection system leveraging deep learning and edge computing.

Firstly, the CBAM module is integrated to mitigate irrelevant noise interference, thereby enhancing the model's capability to perceive intrusion targets in complex operational scenarios. Concurrently, the DIoU_NMS strategy is adopted in post-processing to effectively suppress redundant bounding boxes for the same target.

Secondly, through BN layer pruning and fine-tuning operations, the model achieves a notable 30% increase in detection speed with minimal decrease in detection accuracy. Subsequently, the detection model is deployed on edge processors, with TensorRT facilitating inference acceleration.

Evaluation on a self-constructed railway foreign object intrusion dataset demonstrates that the CBAM-DIoU_NMS-YOLOv5 model achieves a detection accuracy of 86.2%, with particularly high accuracies of 95% for intruding pedestrians, marking improvements of 2.9% and 3% compared to YO-

LOv5, respectively.

Field experiments conducted in real railway scenarios reveal that the detection system attains an alarm accuracy rate of 92.5% and a commendable detection speed of 15 frame/s, effectively meeting the requisites of railway foreign object detection.

Yet, Due to experimental constraints, the railway sample database constructed for this study contains too few animal images, resulting in suboptimal performance of the trained model in detecting animals. If conditions permit in the future, we can increase the number of animal samples in railway scenarios by placing animal models on the test tracks. Additionally, the collected data includes a limited number of railway scene images under rainy and snowy weather conditions. Therefore, further data collection is needed to enhance the dataset with images of railway scenes in adverse weather conditions.

References

[1] TAO H J, LU X B. Automatic smoky vehicle detection from traffic surveillance video based on vehicle rear detection and multi-feature fusion[J]. IET Intelligent Transport Systems, 2019, 13(2): 252-259.

[2] ZHANG Han, YAN Huaiping, ZHANG Zhan. Shadow detection with multi-feature fusion and MMSE optimization[J]. Dianzi Jishu Yingyong, 2018, 44(10): 153-157. (in Chinese)

[3] CUI B B, CREPUT J C. A systematic algorithm for moving object detection with application in real-time surveillance[J]. SN Computer Science, 2020, 1(2): 106.

[4] WANG J X. Research and implementation of intrusion detection algorithm in video surveillance[C]//Proceedings of 2016 International Conference on Audio, Language and Image Processing (ICALIP). Shanghai, China: IEEE, 2017.

[5] LYU Zonglei, CHEN Liyun. SA-FRCNN: An improved object detection method for airport apron scenes[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2021, 38(4): 571-586.

[6] LYU Zonglei, PAN Fuxi, XU Xianhong. A novel deep neural network compression model for airport object detection[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2020, 37(4): 562-573.

[7] WANG Yifan, WEI Jiatian, ZUO Chenglin, et al. An improved CenterNet method for wing icing detection[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2023, 40(6): 703-713.

[8] WANG J, CHEN Y, DONG Z, et al. SABV-Depth: A biologically inspired deep learning network for monocular depth estimation[J]. Knowledge-Based Systems, 2023, 263: 110301.

[9] WANG J, CHEN Y, DONG Z, et al. Improved YOLOv5 network for real-time multi-scale traffic sign detection[J]. Neural Computing and Applications, 2023, 35(10): 7853-7865.

[10] YANG Yu. A high-speed railroad foreign object intrusion detection method based on feature fusion enhancement network[D]. Beijing: Beijing Jiaotong University, 2021. (in Chinese)

[11] SHI Jiafeng. Research on the application of FCN and CNN in railroad intrusion target recognition[D]. Beijing: Beijing Jiaotong University, 2020. (in Chinese)

[12] ZHAN Pengji. Traffic target detection and recognition based on edge computing[D]. Harbin: Harbin Institute of Technology, 2020. (in Chinese)

[13] HU Jialing, SHI Yiping, XIE Siya, et al. An improved MobileNet face recognition system based on Jetson nano[J]. Sensors and Microsystems, 2021, 40(3): 102-105. (in Chinese)

[14] WANG L, HE M T, XU S, et al. Waste classification and detection based on YOLOv5s network[J]. Packaging Engineering, 2021, 42(8): 50-56.

[15] WOO S, PARK J, LEE J, et al. CBAM: Convolutional block attention module[C]//Proceedings of European Conference on Computer Vision. Munich, Germany: ECVA, 2018.

[16] ZHANG Yu. Research on cross and ratio loss function in target detection[D]. Hefei: Anhui University, 2021. (in Chinese)

[17] HU H, PENG R, TAI Y W, et al. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures[EB/OL]. (2016-07-12). http://arxiv.org/abs/1607.03250v1.

[18] HE Y, ZHANG X, SUN J. Channel pruning for accelerating very deep neural networks[C]//Proceedings of the IEEE International Conference on Computer Vision. Venice, Italy: IEEE, 2017.

[19] LUO J H, ZHANG H, ZHOU H Y, et al. ThiNet: Pruning CNN filters for a thinner net[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018, 41(10): 2525-2538.

[20] LIU Z, LI J, SHEN Z, et al. Learning efficient convolutional networks through network slimming[C]//Proceedings of the IEEE International Conference on Computer Vision. Venice, Italy: IEEE, 2017.

[21] GU Deying, LUO Yulun, LI Wenchao. Complex scene traffic target detection based on improved YOLOv5 algorithm[J]. Journal of Northeastern University (Natural Science Edition), 2022, 43(8): 1073-1079. (in Chinese)

[22] ZHOU Lijun, LIU Yu, BAI Lu, et al. Deep learning inference using TensorRT[J]. Applied Optics, 2020, 41(2): 337-341. (in Chinese)

[23] LONG Shike, JIANG Qihang, BAO Younan, et al. Design of Jetson Nano-based vision application platform[J]. Sensors and Microsystems, 2022, 41(9): 99-101,108. (in Chinese)

[24] SHI Yifei. An exploration of the efficiency of accelerated AI deep learning inference using TensorRT[J]. Science and Technology Perspectives, 2017(31): 26-27. (in Chinese)

[25] LONG X, DENG K P, WANG G Z, et al. PP-YOLO: An effective and efficient implementation of object detector[EB/OL]. (2020-08-03) [2021-10-12]. https://arxiv.org/abs/2007.12099v3.

[26] GE Z, LIU S, WANG F, et al. YOLOX: Exceeding YOLO series in 2021[J]. ArXiv pre-print arXiv, 2021: 2107.08430.

[27] HU J, SHEN L, SUN G, et al. Squeeze-and-excitation networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Los Alamitos: IEEE Computer Society Press, 2018: 7132-7141.

Acknowledgements This work was supported in part by the Science and Technology Innovation Project of CHN Energy ShuoHuang Railway Development Company Ltd (No. SHTL-22-28), the Beijing Natural Science Foundation-Fengtai Urban Rail Transit Frontier Research Joint Fund (No.L231002), and the Major Project of China State Railway Group Co., Ltd.(No.K2023T003).

Authors Mr. **SHI Jiang** serves as a senior engineer at CHN Energy ShuoHuang Railway Development Company Ltd. His research primarily focuses on railway transportation.

Prof. **GUO Baoqing** is currently a professor with the State Key Laboratory of Advanced Rail Autonomous Operation, Beijing Jiaotong University, Beijing, China. His research interests include intelligent control and railway safety.

Author contributions Mr. **SHI Jiang** devised the on-site plan and led the experiments. Mr. **BAI Dingyuan** constructed the model, conducted the model training and experiments, and wrote and revised the manuscript. Prof. **GUO Baoqing** analyzed the experimental data. Prof. **WANG Yao** and Dr. **RUAN Tao** contributed to the discussion and background of the research. All authors commented on the manuscript draft and approved the submission.

Competing interests The authors declare no competing interests.

(Production Editor: ZHANG Huangqun)

基于注意力机制的铁路入侵目标识别方法

石江[1]，白丁元[2]，郭保青[2]，王尧[2]，阮涛[2]

(1.国能朔黄铁路发展有限责任公司,北京 100080,中国;
2.北京交通大学机械与电子控制工程学院,北京 100044,中国)

摘要:异物入侵检测对于保障铁路运营安全十分重要,针对传统铁路综合视频监控效率低、检测精度差以及现有智能检测算法检测速度慢等问题,结合注意力机制和目标检测模型在边端进行入侵目标检测。在提高检测精度方面,将包括空间注意力模块和通道注意力模块的卷积注意力模块(Convolutional block attention module, CBAM)模块融合到YOLOv5模型当中,构建了CBAM-YOLOv5模型,并采用距离交并比非极大值抑制(Distance intersection-over-union_non-maximum suppression, DIoU_NMS)算法代替加权非极大值抑制算法,从而改善模型对入侵目标的检测效果;在提升检测速度方面,基于批量归一化(Bath normalization, BN)层对模型网络裁剪并对TensorRT推理加速,最终将算法移植到边缘设备。CBAM-YOLOv5模型在自建的铁路数据集上的检测精度提升了2.1%,平均精度均值(mean Average precision, mAP)达到了95.0%,在边缘设备上的推理速度达到了15帧/s。

关键词:异物检测;铁路防护;边缘计算;空间注意力模块;通道注意力模块