An Edge Computing-Enabled Track Obstacle Detection Method Based on YOLOv5

1st Ziqi Zhang
School of Electronic and Information Engineering
Beijing Jiaotong University
Beijing, China
21120276@bjtu.edu.cn

2nd Yifei Cai
Locomotive & Car Research Institute
China Academy of Railway Sciences Corporation Limited
Beijing, China
Beijing Zongheng Electro-Mechanical Technology
Beijing, China
caiyf2010@sina.cn

3rd Henjun Lu
School of Electronic and Information Engineering
Beijing Jiaotong University
Beijing, China
20211105@bjtu.edu.cn

4th Tao Wen*
School of Electronic and Information Engineering
Beijing Jiaotong University
Beijing, China
wentao@bjtu.edu.cn

5th Baigen Cai
School of Electronic and Information Engineering
Beijing Jiaotong University
Beijing, China
bgcai@bjtu.edu.cn

2023 International Conference on Electromagnetics in Advanced Applications (ICEAA) | 979-8-3503-2058-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEAA57318.2023.10297912

***Abstract*—Rail transit is developing towards intelligence, which requires a lot of computing resources to carry out deep learning tasks. However, the limited computing power of on-board equipment hinders the operation of complex detection networks for the execution of complex deep learning tasks. And the existing railway inspection methods are mostly based on fixed equipment, their detection range is limited. In this article, we propose a mobile platform railway inspection method based on edge computing. Edge computing can reduce stress by offloading the workload to edge nodes. The mobile platform vehicle is responsible for acquiring real-time images. Edge computing servers are used to help execute object detection algorithms off-orbit and carry most of the computing power. The use of edge computing architecture for railway inspection can effectively reduce the pressure of the on-board server and reduce the network dependence. Aiming at the complex scene of track detection, we propose an improved YOLOv5-based target detection algorithm. By adjusting the network model and deploying it on the edge server, the method reduces the computing burden and realizes the collaborative reasoning with the whole edge computing architecture. We implement the collaborative reasoning scheme in practical experiments and find that the proposed improved target detection method based on edge computing can complete the target detection task with few computing resources, and meet the demand for precision and real-time performance.**

***Index Terms*—edge computing, Jetson Nano, YOLOv5, DeepStream, object detection**

I. Introduction

Currently, China's railway transportation is in a period of rapid development. Railway transportation has made significant contributions to China's economic construction and development through freight and human transportation. In addition, with the development of technology, railway transportation plays an increasingly important role in passenger and freight transportation. Therefore, ensuring the safety of railway traffic operation is particularly important. Especially, the monitoring of track conditions and the health services of track components directly affect the safety of the track [1].

For many years, the safety of railway transportation has been highly valued, but failures in railway infrastructure, such as incorrect signal lights, physical environmental changes caused by bad weather, and railway obstacles, pose a threat to the safety of railway transportation [2].

However, most existing machine vision based detection methods are based on fixed devices [3]. Their detection range is limited and is greatly affected by external signals such as heavy fog, rain and snow, which leads to a certain monitoring window period and blind spots in traditional equipment inspections. At the same time, most machine learning foreign object detection algorithms currently have problems such as large model parameter space, high cost, poor real-time performance, and high cost in railway safety monitoring deployment.

In order to solve the problem of low detection accuracy in complex environments, this paper proposes an improved YOLOv5s obstacle detection algorithm. The algorithm architecture is improved based on the YOLOv5s network, integrating the attention mechanism of GAM [8]and the adaptive feature fusion mechanism of ASFF [9]to improve the performance of the network. The improved network has better accuracy in obstacle detection in complex environments.

In a centralized computing architecture, all collected video source data will be transmitted to the computing center for further analysis, and the results will be sent back to the client. However, due to the fact that all video data must be transmitted to the computing center, and the analysis process is all carried out in the cloud, this structure can lead to high latency and increase network bandwidth consumption, and

*Corresponding author

requires significant computing power performance from the computing center.

Edge computing is an emerging computing architecture that moves computing power from a centralized computing center to the edge closer to users and mobile devices. At present, edge computing has many applications in urban transportation and other fields, such as [5] for video analysis and [7] for urban transportation. However, the current development of edge computing is still limited by the deployment of devices. They are mostly deployed on fixed equipment or trains.

In order to reduce the computational burden of on-board computing devices and meet certain performance requirements, this paper proposes a computing architecture for rail transit obstacle detection based on edge computing. This method uses mobile platforms to collect images, and then upload computing tasks to the edge computing server. Performing partial or complete calculations at the edge can reduce latency, provide real-time response, reduce network bandwidth load, mitigate privacy leakage risks, improve data security, and alleviate performance bottlenecks in centralized computing centers [4].

Combining YOLOv5 model and edge computing, the contributions of this paper are summarized as follows:

- We propose an obstacle detection algorithm that adjusts the network and parameters to achieve a balance between accuracy and performance among different computing nodes.
- We propose a edge computing architecture for distributed collaborative reasoning. In this structure, mobile platform vehicle terminals and edge servers can collaborate for object detection and inference.
- We evaluated the performance of the obstacle detection algorithm and edge computing architecture. The evaluation results indicate that the proposed collaborative reasoning scheme has a certain improvement in the average performance of program execution.

The second section describes the details of our analytical model. The third section outlines the layered edge computing architecture for obstacle detection. The fourth section evaluates our proposed design plan. The fifth part summarizes the entire text.

II. Model Construction

A. The Basic YOLOv5 Structure

In order to meet the real-time requirements of train obstacle detection, we have chosen a fast and accurate real-time object detection algorithm YOLO (you only look once). YOLO is also one of the best single stage object detection models. YOLOv5s not only ensures high accuracy, but also has very fast detection speed and a small number of model parameters, making it easy to deploy on embedded terminals.

The YOLOv5s model is mainly composed of Backbone and Head, and its network structure is shown in Figure 1. Backbone is a feature extraction module in the YOLOv5s network, mainly including modules such as Focus, C3, and SPP. The Head section mainly includes neck Neck and Detect modules for extracting fusion features.

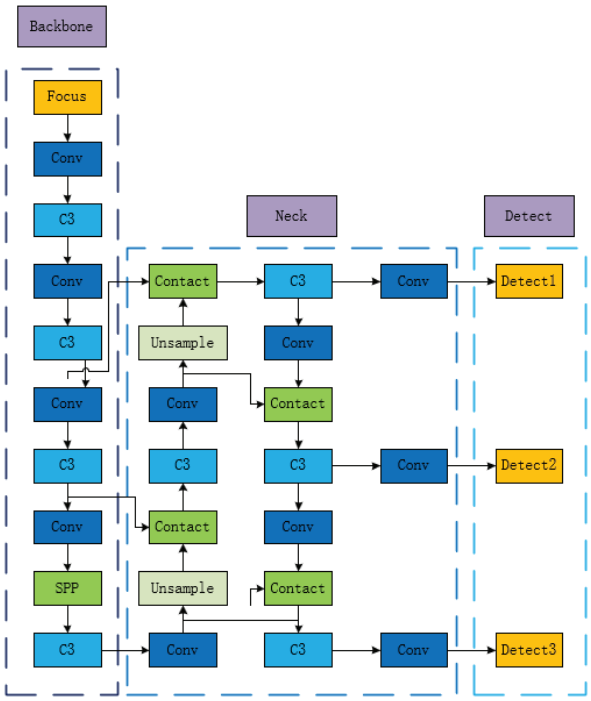

Fig. 1. YOLOv5 network architecture.

B. ASFF

YOLOv5s use FPN structure in the feature fusion network, which enriches the Semantic information of the feature map. However, it only adjusts different feature layers to a unified size before accumulation. Inconsistency between different feature scales will lead to greater noise in the fusion feature map, which will make the effect worse.

In order to improve feature extraction capabilities and effectively suppress the interference of invalid features in complex backgrounds, it is necessary to strengthen the fusion and utilization of multi-scale features. The ASFF [9] module enables the network to adaptively learn the weights of each position on each feature layer, making important information features dominant during fusion. For each feature layer to be fused, first convert the other feature layers to the same resolution, and then train to find the optimal weight for fusion. The structural design diagram of ASFF is shown in Figure 2.

In the figure, X1, X2, and X3 are the last three feature layers of the backbone network. $X^{1\to3}$ and $X^{2\to3}$ are the feature layers generated by X1 and X2 after passing through the feature scaling layer, which are the same size as X3. In order to achieve consistent size, it is necessary to modify the up sampling and Downsampling strategies of feature layers with different scales. For upsampling of the feature layer, apply 1 $\times$ 1 convolution to compress the number of channels in the feature layer Shrink to the same size as the previous level,

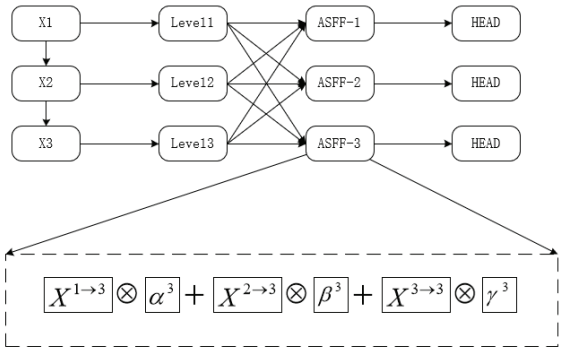

Fig. 2. ASFF network structure.

and then use interpolation upsampling to improve resolution. For one Downsampling of the feature layer, 3 × 3 convolution with a step of 2 is used to simultaneously expand the number of channels and reduce the resolution. For the second Downsampling of the feature layer, first pool the feature layer in two steps to halve the resolution of the feature layer, and then modify the number of channels and resolution simultaneously through 3 × 3 convolution in two steps.

After feature scaling, it is necessary to adaptively fuse the feature layers. α_{ij}^l, β_{ij}^l and γ_{ij}^l refer to the importance weights of the features mapped from three different feature layers at positions (i, j) to the features in layer l. The calculation formula for α_{ij}^l is as follows:

$$\alpha_{ij}^l = \frac{e^{\lambda_{\alpha_{ij}^l}^l}}{e^{\lambda_{\alpha_{ij}^l}^l} + e^{\lambda_{\beta_{ij}^l}^l} + e^{\lambda_{\gamma_{ij}^l}^l}} \tag{1}$$

The above equations $\lambda_{\alpha_{ij}^l}^l$, $\lambda_{\beta_{ij}^l}^l$, and $\lambda_{\gamma_{ij}^l}^l$ are calculated using 1 × 1 convolution for the three feature layers after feature scaling. Simultaneously limit $\alpha_{ij}^l + \beta_{ij}^l + \gamma_{ij}^l = 1$ and $\alpha_{ij}^l, \beta_{ij}^l, \gamma_{ij}^l \in [0, 1]$. Calculate a using the above equation, and similarly obtain b and c. These three importance weight parameters are trained through standard backpropagation to obtain the final results. $x_{ij}^{n\to l}$ represents the mapping from the features in layer n of positions (i, j) to the features in layer l, and the fusion formula is as follows:

$$y_{ij}^l = \alpha_{ij}^l * x_{ij}^{1\to l} + \beta_{ij}^l * x_{ij}^{2\to l} + \gamma_{ij}^l * x_{ij}^{3\to l} \tag{2}$$

The above equation y_{ij}^l indicates that the features at (i, j) after fusion are weighted fusion of the features at (i, j) of the first three feature maps. After back-propagation learning, the Semantic information that is conducive to detection on feature maps at all levels is enhanced to achieve the best fusion effect.

In obstacle detection, different scale feature layers contain different Semantic information. Due to the small proportion of obstacles, more Semantic information is concentrated in the shallow feature layer, which is easy to be annihilated by invalid information in the process of feature fusion. This article uses the ASFF module to replace the original FPN module for feature fusion to enhance the target features. Compared with the original structure, the ASFF structure can effectively fuse features of different scales through backpropagation.

C. GAM

Due to the fact that this article is focused on railways, the surrounding environment is complex and variable, and obstacle targets only occupy a small part of the image. In order to improve the model's feature expression ability for obstacle parts, this article adds a GAM attention module at the end of the main network of YOLOv5s and the last three convolutional networks of different sizes.

The GAM attention module [8] is a global attention mechanism proposed in 2021 to improve the performance of neural networks by reducing information diffusion and amplifying global interaction representations. The GAM global attention module can improve the network model's attention to obstacles, reduce interference from complex backgrounds, and improve model detection accuracy. The GAM global attention module uses sequential channels in the CBAM [10] (Convolutional Block Attention Module). Firstly, the channel attention feature map CA (Channel Attention) corrects the original feature map, and then the spatial attention feature map SA (Spatial Attention) corrects the correction results to obtain the final feature map. The GAM module structure is shown in Figure 3.

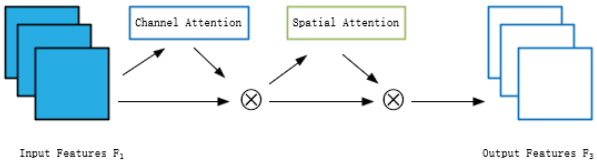

Fig. 3. GAM module.

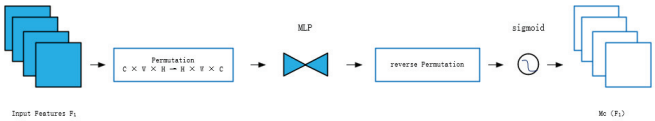

Fig. 4. Channel attention submodule.

Figure 3 shows the structure of channel attention submodule. First, set the size of the input feature map F1 to C × W × H. Among them, H and W represent the length and width of the input features, C represents the number of channels, and 3D arrangement is used to preserve the 3D information of the input image. The output is then amplified through a 2-layer MLP (Multi Layer Perception) to enlarge the cross dimensional channel, and reverse 3D arrangement is used to activate the results through the sigmoid function. Finally, a new feature map $Mc(F_1)$ is obtained.

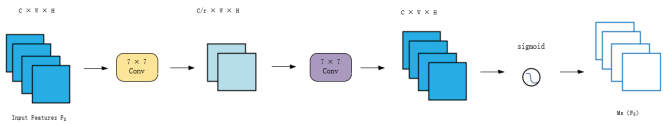

Fig. 5. Spatial attention submodule.

Figure 4 shows the structure of spatial attention submodule, assuming that the size of the input feature map F2 is C × H × W. Firstly, the input features are spatially fused through two 7x7 sized convolutional layers, and then the output is activated through the sigmoid function to obtain a new feature map $Mc(F_2)$.

III. Edge Computing Architecture

Edge computing architecture is a distributed hierarchical structure composed of cloud nodes, edge nodes and Edge devices [11].Using the edge computing architecture can maintain the privacy of data. Secondly, edge computing can unload tasks in the regional network closer to the data source. Data generated by Edge device can be directly processed on edge nodes without uploading to cloud nodes, thus reducing network latency and communication costs.

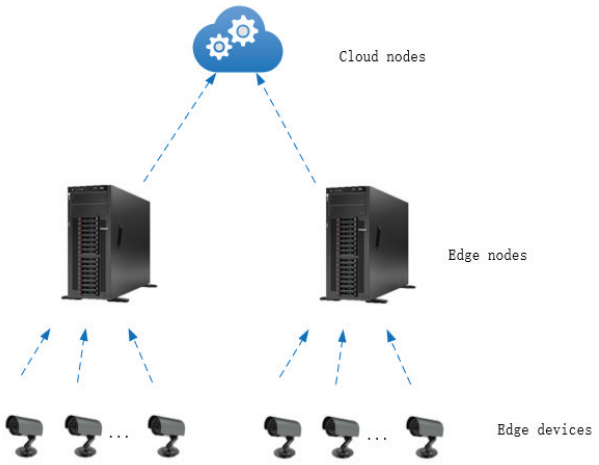

Fig. 6. Edge computing architecture.

However, the development of traditional edge computing is still limited by the deployment of devices, which are mostly deployed on fixed devices or trains, with poor versatility and high cost. In order to achieve convenience, lightweight, real-time, and reduce the cost of railway inspection, based on the goal of minimizing delay, energy consumption, and maximizing benefits, this paper proposes a computing architecture for rail transit obstacle detection based on edge computing. This method uses a mobile platform to collect images, and then upload computing tasks to the edge computing server.The edge computing architecture designed for object detection is shown in Figure 7.

The overall architecture of the system is divided into two layers, the hardware layer uses the Jetson Nano embedded development board to build an intelligent robot mobile platform

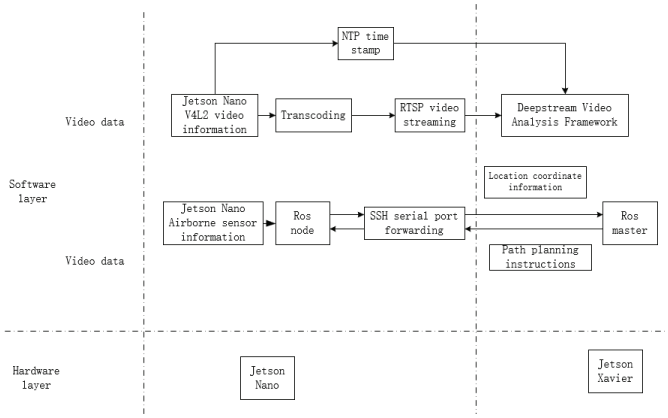

Fig. 7. Edge computing architecture designed for object detection.

as a Edge device, and the Jetson Xavier embedded development board as an edge node. The software services in the software layer are also distributed across different hardware devices. The Ros system is deployed on the Edge device to control the robot and move conveniently along the rail. Due to the poor performance of the Jeton Nano, it is unable to carry high precision algorithm operation. The terminal does not perform algorithm reasoning, but only collects video information, and pushes it to the upper edge server through rtsp Flow network for obstacle recognition reasoning. The system adopts Jetson Xavier with good performance as the edge server part, controls the terminal robot to move along the rail through SSH serial port, and pulls and processes the video information of the terminal. In order to maximize resource utilization on edge servers, they need to perform inference tasks on multiple terminal nodes.

Fig. 8. Mobile robot platform as Edge device.

The program for video stream processing on Jetson Xavier is based on a deepstream pipeline architecture, as shown in Figure 8. DeepStream is a streaming analysis software package developed based on GStreamer, which is open-source and has the characteristics of modular structure, comprehensive software and hardware support, and fast stream file processing.

The pipeline consists of multiple parts:

- Stream reading module: RTSP/RTMP network real-time stream, camera video stream reading, supporting H.264, H.265, and MPEG video encoding formats.

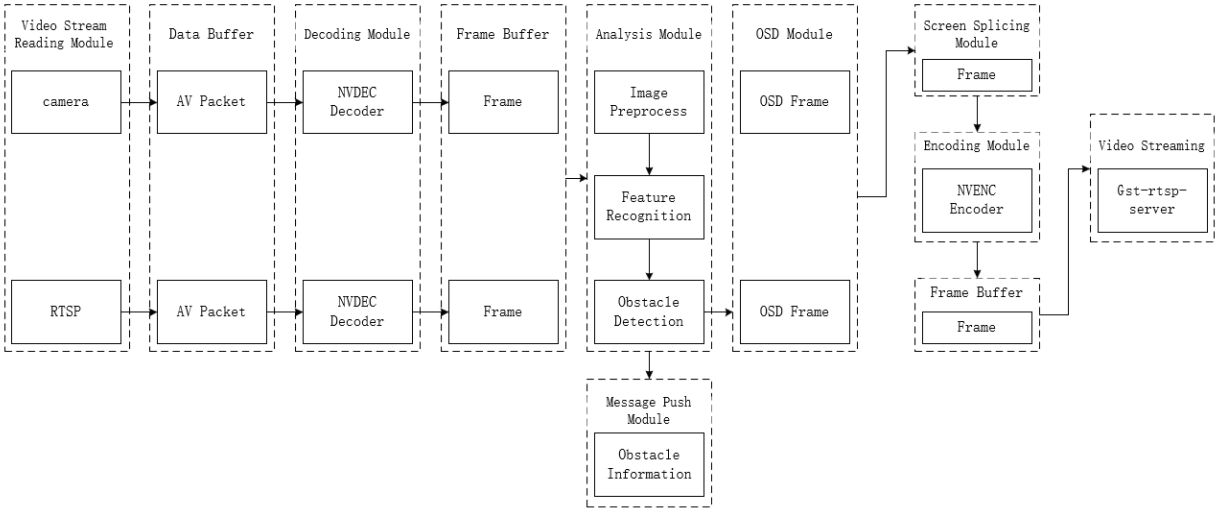

Fig. 9. DeepStream pipeline architecture.

- Encoding and decoding module: After reading the video stream, the decoding module will extract the image from the compressed H.264, H.265, and MPEG video streams. After the video processing is completed, the encoding module compresses the processed image into H.264 and H.265 video streams.
- Image intelligent analysis module: a module for obstacle detection.
- Image content overlay module: After the recognition of the image is completed, it is necessary to overlay the content on the screen, so it is necessary to support the image content overlay function.
- Screen splicing module: The DeepStream pipeline can support multiple video streams.
- Serialization and push module: Push includes structured string push and video stream push. Structured string push serializes intelligent analysis results in the form of strings and pushes them to remote message queues. Video streaming pushes processed videos through RTSP/RTMP real-time network video streaming for streaming.

IV. Results and Analysis

A. Performance test of Obstacle detection algorithm

We conducted model training experiments on NVIDIA T4 and used GPU acceleration. The basic programming environment includes Pytorch-3.7, cuda-11.0, and cudnn 8.1. The focus of our experiment is on the impact of the improvement and whether it is possible to improve accuracy while reducing computational resource usage. Considering the scenario targeted by the algorithm, we chose the BDD100k dataset [12], which is the largest open traffic scenario dataset and contains 10 categories of traffic scenario targets.

It can be seen from Table 1 that different blocks of the backbone network have different FLOPS and parameters. With this data, idle resources can be reasonably matched with corresponding parameters to maximize the efficiency of obstacle detection. Both GAM module and ASFF module have a positive impact on improving network detection accuracy. Finally, the GAM attention mechanism and ASFF adaptive feature fusion mechanism were simultaneously integrated into the network, effectively improving the detection accuracy of the model. Compared with the original YOLOv5s network, mAP0.5 improved by 2.5% and the accuracy improved by 1.5%.

Fig. 10. Object detection results.

Finally, we tested some trackside scenes using the improved algorithm. The results show that the algorithm can effectively detect obstacles and targets on the trackside.

B. Performance test of edge computing architecture

We transplanted the model after training into the edge computing architecture, and tested it with multi-channel 720p camera real-time video stream and local video.

From the figure, it can be seen that as the number of video channels connected to the system doubles, the average frame rate of each video channel is approximately halved. The

TABLE I
DETECTION RESULTS ON BDD100K

ID	ASFF	GAM	Parameters	FLOPs	Precious(%)	Recall(%)	mAP0.5(%)
1	×	×	7.05M	15.9G	66.6	43.7	47.1
2	✓	×	11.63M	22.4G	67.8	45.5	49.0
3	×	✓	12.48M	24.3G	68.3	46.9	50.2
4	✓	✓	16.50M	29.9G	68.1	46.2	49.4

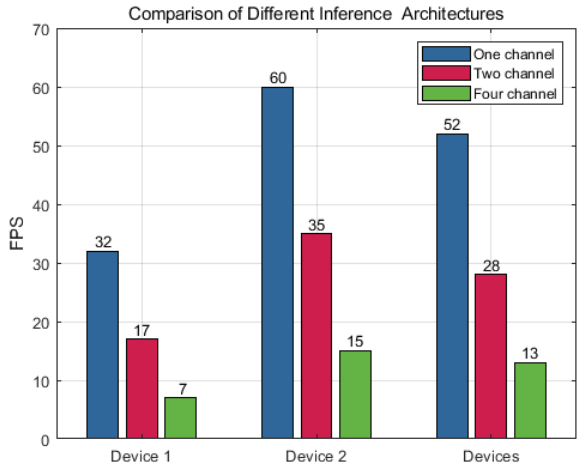

Fig. 11. Edge computing architecture.

performance of edge nodes is significantly better than that of Edge device. Combining them using edge architecture can effectively reduce the computing burden of Edge device as mobile platform devices. However, the simultaneous use of multiple devices can also increase the computational burden, and the system needs further optimization to further explore its computational limits, in order to support intelligent analysis of obstacle detection on a larger scale.

V. Conclusion

Experimental results show that the algorithm can accurately identify obstacles on Railway track, and has good real-time and accuracy. In addition, the algorithm combines the computing architecture of Ros robot operating system and edge computing, and can effectively use edge computing equipment, thus meeting the real-time, efficient and accurate requirements of Railway track foreign matter detection.

The possible future work is to combine the rail safety maintenance method based on slam point cloud mapping with the obstacle detection method in this paper, and propose a edge computing architecture based on multi-source information fusion and collaboration. While maximizing the utilization of edge server resources, it also improves detection efficiency and accuracy.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1313301; in part by the National Natural Science Foundation of China under Grant 62120106011 and Grant 52172323.

References

[1] Karaköse, Mehmet et al. “A New Approach for Condition Monitoring and Detection of Rail Components and Rail Track in Railway.” Int. J. Comput. Intell. Syst. 11 (2018): 830-845.
[2] D. He, Z. Zou, Y. Chen, B. Liu, X. Yao, and S. Shan, “Obstacle detection of rail transit based on deep learning,” Measurement,vol. 176, article 109241, 2021.
[3] Song Li, Hongli Zhao, Jinmin Ma, and Chi-Hua Chen. 2021. An Edge Computing-Enabled Train Obstacle Detection Method Based on YOLOv3. Wirel. Commun. Mob. Comput. 2021 (2021). https://doi.org/10.1155/2021/7670724
[4] S. Wan, S. Ding, and C. Chen, “Edge computing enabled video segmentation for real-time traffic monitoring in Internet of vehicles,” Pattern Recognition, vol. 121, p. 108146, 2021.
[5] X. Ran, H. Chen, X. Zhu, Z. Liu, and J. Chen, “Deepdecision: a mobile deep learning framework for edge video analytics,” in IEEE INFOCOM 2018- IEEE Conference on Computer Communications, pp. 1421–1429, Honolulu, HI, USA, 2018.
[6] J. Ren, Y. Guo, D. Zhang, Q. Liu and Y. Zhang, ”Distributed and Efficient Object Detection in Edge Computing: Challenges and Solutions,” in IEEE Network, vol. 32, no. 6, pp. 137-143, November/December 2018, doi: 10.1109/MNET.2018.1700415.
[7] C. Chen, B. Liu, S. Wan, P. Qiao, and Q. Pei, “An edge traffic flow detection scheme based on deep learning in an intelligent transportation system,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 3, pp. 1840–1852, 2021.
[8] Liu Y , Shao Z , Hoffmann N . Global Attention Mechanism: Retain Information to Enhance Channel-Spatial Interactions[J]. 2021.
[9] Liu S , Huang D , Wang Y . Learning Spatial Fusion for Single-Shot Object Detection[J]. 2019.
[10] Woo S , Park J , Lee J Y ,et al.CBAM: Convolutional Block Attention Module[J].Springer, Cham, 2018: 3-19,arXiv:1807.06521.
[11] L. Tong, Y. Li, and W. Gao, “A hierarchical edge cloud architecture for mobile computing,” in IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9, San Francisco, CA, USA, 2016.
[12] Yu F, Chen H, Wang X, et al. BDD100K: A diverse driving dataset for heterogeneous multitask learning[C]. IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2020: 2633-2642.
[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: unified, real-time object detection,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recog- nition (CVPR), Las Vegas, June 2016.
[14] S. Wang, S. Yang, and C. Zhao, “Surveiledge: real-time video query based on collaborative cloud-edge deep learning,” in IEEE INFOCOM 2020- IEEE Conference on Computer Communications, pp. 2519–2528, Toronto, ON, Canada, 2020, https://arxiv.org/abs/2001.01043.
[15] A. Ceselli, M. Premoli and S. Secci, ”Mobile Edge Cloud Network Design Optimization,” in IEEE/ACM Transactions on Networking, vol. 25, no. 3, pp. 1818-1831, June 2017, doi: 10.1109/TNET.2017.2652850.