Digital Transportation
and Safety

ARTICLE

Open Access

https://doi.org/10.48130/dts-0024-0009
Digital Transportation and Safety **2024**, 3(3): 82–91

A YOLOv8-CE-based real-time traffic sign detection and identification method for autonomous vehicles

Yuechen Luo[1], Yusheng Ci[1*], Hexin Zhang[2] and Lina Wu[3]

[1] *School of Transportation Science and Engineering, Harbin Institute of Technology, Harbin, China*
[2] *School of Future Technology, Harbin Institute of Technology, Harbin, China*
[3] *School of Automobile and Traffic Engineering, Heilongjiang Institute of Technology, Harbin, China*
* Corresponding author, E-mail: ciyusheng1999@126.com

Abstract

Traffic sign detection in real scenarios is challenging due to their complexity and small size, often preventing existing deep learning models from achieving both high accuracy and real-time performance. An improved YOLOv8 model for traffic sign detection is proposed. Firstly, by adding Coordinate Attention (CA) to the Backbone, the model gains location information, improving detection accuracy. Secondly, we also introduce EIoU to the localization function to address the ambiguity in aspect ratio descriptions by calculating the width-height difference based on CIoU. Additionally, Focal Loss is incorporated to balance sample difficulty, enhancing regression accuracy. Finally, the model, YOLOv8-CE (YOLOv8-Coordinate Attention-EIoU), is tested on the Jetson Nano, achieving real-time street scene detection and outperforming the Raspberry Pi 4B. Experimental results show that YOLOv8-CE excels in various complex scenarios, improving mAP by 2.8% over the original YOLOv8. The model size and computational effort remain similar, with the Jetson Nano achieving an inference time of 96 ms, significantly faster than the Raspberry Pi 4B.

Keywords: YOLOv8-CE-based; Real-time; Traffic; Signs; Detection

Citation: Luo Y, Ci Y, Zhang H, Wu L. 2024. A YOLOv8-CE-based real-time traffic sign detection and identification method for autonomous vehicles. *Digital Transportation and Safety* 3(3): 82–91 https://doi.org/10.48130/dts-0024-0009

Introduction

Although cars have undoubtedly improved people's lives, they have also introduced higher risks of traffic accidents and fatalities due to factors like fatigue, drowsiness, and road conditions. From the past to the present, various solutions have been developed to establish the infrastructure for assisted driving (ADAS) and autonomous driving. ADAS aims to assist drivers and vehicles in identifying potentially hazardous situations and taking emergency measures to enhance the safety and comfort of the driving experience.

Deep learning-based methods have demonstrated their excellence in assisted driving tasks, including traffic sign detection[1], lane detection[2], pedestrian detection[3], and other tasks. And traffic sign detection is a crucial component of it. Traffic sign detection systems are crucial components of both intelligent transportation systems and autonomous driving systems. The accuracy and real-time performance of traffic sign detection technology are critical in enabling these systems to make informed decisions. As such, achieving a balance between real-time performance and accuracy is of utmost importance in ensuring the effectiveness of these technologies[1]. Therefore, we want to assist drivers in driving by deploying traffic sign detection on edge devices to achieve real-time detection.

The key components of traffic sign detection and recognition include feature extractor, classification, and localization of traffic signs, among which localization is focused on by researchers as a part specific to object detection. With the rapid development of deep learning technology, object detection algorithms, such as Faster RCNN, YOLO, SSD, etc. have been widely used in traffic sign detection.

In research on traffic sign detection, Wang et al.[4] proposed a deep model for traffic sign detection and recognition in complex road conditions, incorporating innovations like Coordinate Attention (CA), angle loss, SimOTA for label assignment, and a Hierarchical-Path Feature Fusion Network (HPFANet). Their model significantly improves precision, recall, and mAP over YOLOv5s, demonstrating superior performance and robustness across various datasets. Lai et al.[5] introduced STC-YOLO for traffic sign detection, enhancing YOLOv5 with advanced data augmentation, modified architecture for small object detection, and a novel feature extraction module with multi-head attention, yielding substantial accuracy gains compared to traditional approaches. Chu et al.[6] developed a model with a global feature extraction module using self-attention and a lightweight parallel detection head to enhance small traffic sign detection accuracy, supported by extensive data augmentation for improved robustness.

While deep learning methods have made some progress in traffic sign detection tasks, they still face limitations when dealing with complex natural environments and real-time edge detection. For example, when deployed to edge platforms, the inference speed is low, and the accuracy is low. To address the above problems, this paper presents several enhancement strategies based on the lightweight version of the YOLOv8 algorithm, namely the YOLOv8-CE algorithm.

(1) Coordinate Attention is a feature extraction mechanism that enables the model to better pinpoint and recognize the target area, while also capturing inter-channel connections.

(2) Changing the localization loss function to EIoU helps the model converge quickly and makes the regression process more stable, which improves the regression accuracy of the prediction box.

Related works

Traffic sign detection stands as a pivotal area of research in autonomous driving. Many researchers have introduced diverse algorithms aimed at classifying and identifying road traffic signs. Broadly speaking, these algorithms fall into two categories: the first utilizes traditional detection techniques such as color-based, shape-based features, and feature fusion. The second category is rooted in deep learning methodologies.

Traffic sign detection-based traditional techniques

Since the 1970s, a group of researchers has been working on traffic signs. Firstly, the traditional traffic sign detection, where researchers extracted features manually, such as color features, shape features, and fusion features, and performed the corresponding detection: de la Escalera et al.[7] segmented by color to extract the region of interest (ROI), and then got shape features for analysis to detect traffic signs. Fleyeh[8] first converted RGB images into IHLS color space and then used segmentation algorithms to extract the color features of traffic signs for detection. However, these algorithms are not able to obtain the expected results consistently, they face various challenges such as internal and external conditions of the traffic sign environment, the external conditions are some environmental factors such as weather conditions, lighting conditions, occlusion degradation of traffic signs, these conditions are not changeable, so some researchers try to solve these problems, but often can not take care of all the cases so the detection effect is limited. On the other hand, internal conditions are variables that can be controlled by algorithms such as response time and detection accuracy, and these series of challenges to improve accuracy and increase detection speed contributed to the development of traditional traffic sign detection. After this, with the development of machine learning, support vector machine SVMs were also applied to sign recognition, and Maldonado-Bascón et al.[9] used support vector machines (SVMs) for shape classification and content recognition.

Traffic sign detection-based deep learning

However, in actual driving, for high-speed vehicles, the requirements for traffic sign speed are very strict, and this method does not meet the demand for real-time detection, researchers have consequently turned their attention to convolutional neural networks. Cireşan et al.[10] were able to attain a classification accuracy of 99.15% on the GTSRB dataset through the utilization of a convolutional neural network (CNN). The arrival of convolutional neural networks (CNN) opened a new era in image processing. Since the introduction of AlexNet, CNN has been continuously optimized with increased depth and more complex structures. Its results in the field of computer vision have been extremely good. Deep learning methods have allowed traffic sign detection accuracy to be greatly improved. In comparison with traditional sign recognition methods such as color and shape and machine learning methods, better recognition and detection can be obtained, faster and can be adapted to more complex scenes.

The current mainstream detection algorithms can be classified as two-stage and one-stage. The two-stage mainly includes R-CNN[11], Fast R-CNN[12], Faster R-CNN[13], etc. The R-CNN series of algorithms first obtains the region containing the object, and then uses the classifier for classification and regression. The one stage mainly includes YOLO[14–17] as the main representative, the YOLO series algorithms directly use CNN for feature extraction and consider the task as regression to directly complete object classification and location localization.

Huang et al.[1] introduced asymptotic feature pyramid network (AFPN) into YOLOv8 with the goal of highlighting the influence of key layer features after feature fusion and solving the direct interaction of non-adjacent layers. Chen & Fan[18] introduced Multi-Scale Group Convolution to replace the C2f module and integrated Deformable Attention into the model to improve the detection efficiency and performance of complex targets. While the parameters are reduced by 59.6%, the accuracy remains at a high level. Zhang et al.[19] proposed a multi-scale traffic sign detection model, CR-YOLOv8 based on YOLOv8. By incorporating an attention module in the feature extraction stage and an RFB module in the feature fusion stage, the model enhances key features and improves multi-scale object detection with minimal computational overhead.

The advancement of convolutional neural networks has been accompanied by significant growth in the field of cloud platforms[20], but at the same time, numerous issues have arisen, making it challenging for centralized cloud services to fulfill the real-time demands of most intelligent transportation applications amidst the current deluge of big data. Therefore, transferring computing resources from cloud centers to network edge devices close to users has become an inevitable requirement for IoT technology development, real-time computing, and achieving network edge intelligence. In response to the existing situation, many researchers have started to focus on lightweight neural networks and use some lightweight methods to deploy detection algorithms to inexpensive embedded devices to achieve edge detection and share the computational pressure of the central computer. Luo et al.[21] opted to use Ghostnet as the feature extraction network. This lightweight network reduced the number of parameters and computations. The author's test results on the edge device Raspberry Pi was 790 ms. Additionally, Artamonov & Yakiomov[22] utilized the processing power of NVIDIA mobile platforms, such as Jetson TX1 and Jetson TX2, to deploy the YOLO algorithm for continuous video traffic sign detection with GPUs.

Methodology

Data processing

To better train the model and make it more generalizable, processing of the data is required. The techniques utilized in this section for handling data involve Mosaic, MixUp, adaptive image scaling, and adaptive anchor box calculation. Firstly, the data augmentation method is used in YOLOv4 to stitch four images with random scaling respectively. Then combining them into one image, which improves the detection of small

objects more effectively. Then, adaptive image scaling was used to obtain a standard image of 640 × 640 for training. In addition, if the difference between the anchor box and the object size is large, the K-means algorithm was employed to find the most suitable anchor box size and use it for training. The image after data processing is shown in Fig. 1.

YOLOv8

YOLOv8 is the most mainstream single-stage object detection algorithm. According to the depth and height of the network, it can be divided into five models YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. As it is to be deployed to embedded devices, YOLOv8n, which has the smallest volume, is chosen as the basic network model in the present paper.

Structure of the network

The network architecture of YOLOv8 is illustrated in Fig. 2. YOLOv8 consists of four components: input, backbone, neck, and head.

Firstly, the input layer is enriched with the Mosaic data augmentation method to enrich the dataset with low hardware device requirements and the final input is 640 × 640 standard-size images.

In backbone, the core of the network consists of Conv, C2f, and SPPF modules, which are responsible for extracting features from images. The Conv module comprises a combination of Conv2d, BN, and the Swish activation function. The C2f module is an improvement over the C3 module, with adjustments to the number of channels for different scale models, representing a refined tuning of the model structure that significantly enhances performance. The SPPF module is an improvement over SPP[23], using multiple small-sized pooling kernels in series instead of a single large-sized pooling kernel in the SPP module. This modification retains the original functionality of fusing feature maps with different receptive fields, thereby enriching the feature map's expressiveness while further improving running speed.

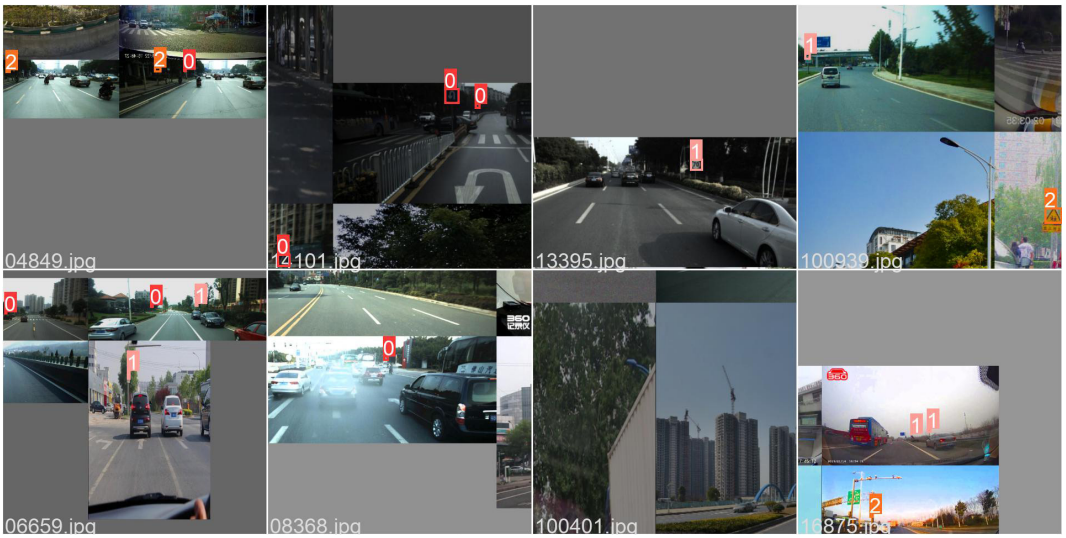

Fig. 1 Results of data processing.

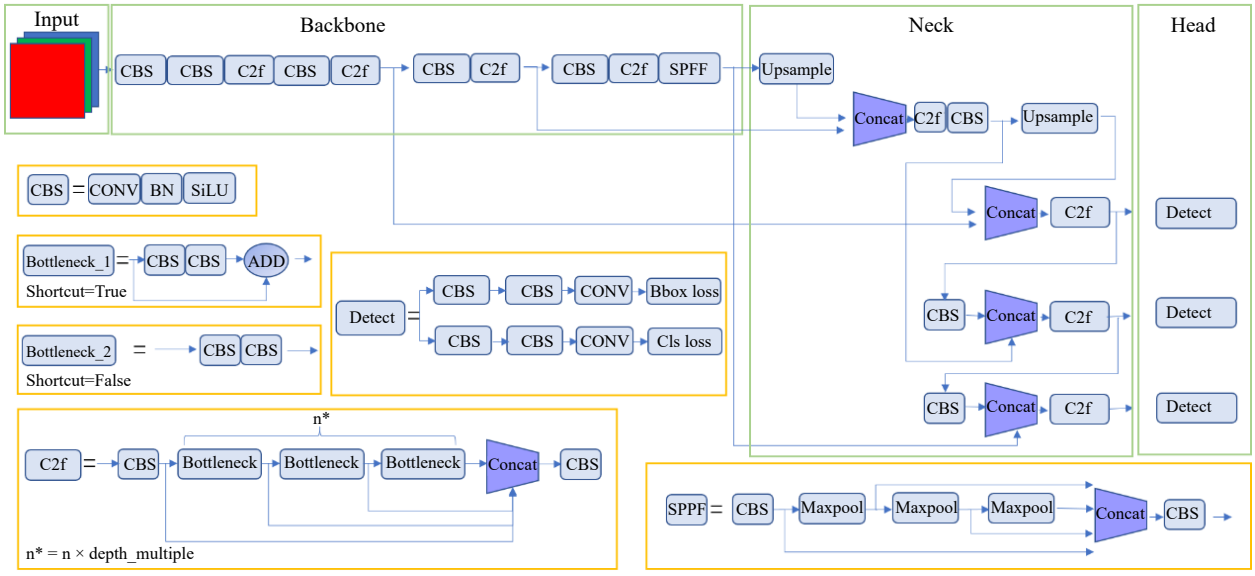

Fig. 2 Structure of YOLOv8.

In addition, the neck part is mainly composed of FPN[24], and PANet[25] for fusing feature information at different scales, and on the basis of FPN, PANet introduces a bottom-up path, which can make the bottom-up feature fusion after the top-down feature fusion, so that such bottom-up location information can also be transferred to the deeper layers, thus enhancing the localization capability at multiple scales.

Finally, the head section has been replaced with the current mainstream decoupled head structure, separating the classification and detection heads. Additionally, it has shifted from an Anchor-Based to an Anchor-Free approach.

Loss function

The loss function of YOLOv8 consists of two components: classification loss and regression loss:

- Classification loss, calculate whether the anchor matches the correct category.
- Regression loss, indicates the error in the position between the predicted box and the Ground Truth. It includes CIoU Loss and Distribution Focal Loss.

YOLOv8 uses BCE-With-Logits-Loss to calculate the classification loss (L_{cls}), which is determined by the following formula:

$$Loss = -\frac{1}{n}\sum_{i}^{n}\left[y_i \cdot log\left(\sigma(x_i)\right) + (1-y_i)\cdot log\left(1-\sigma(x_i)\right)\right] \quad (1)$$

$$\sigma(a) = \frac{1}{1+exp(-a)} \quad (2)$$

The metric often used to calculate the localization loss in YOLOv8 is IoU, which represents the overlap ratio between true box and predicted box, and that is:

$$IoU = \frac{\left|b \cap b^{gt}\right|}{\left|b \cup b^{gt}\right|} \quad (3)$$

In the original YOLOv8, the regression loss function is CIoU[26], which incorporates a penalty term αv to DIoU[26], as an influencing factor. This factor considers the difference in the aspect ratio between the predicted box and the true box. In other words, the penalty term of CIoU is expressed as:

$$R_{CIoU} = \frac{\rho^2(b,b^{gt})}{c^2} + \alpha v \quad (4)$$

Therefore, the loss calculation formula for CIoU is:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b,b^{gt})}{c^2} + \alpha v \quad (5)$$

where,

$$\alpha = \frac{v}{(1-IoU)+v} \quad (6)$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \quad (7)$$

where, b and b^{gt} are the prediction box and label box respectively. w^{gt} and h^{gt} represent the width and height of the labeled box. w and h are the width and height of the prediction box respectively. ρ represents the distance between the center points of the two boxes, and c is the maximum distance between the boundaries of the two boxes.

Improved algorithm model YOLOv8-CE (YOLOv8-Coordinate Attention-EIoU)

YOLOv8 based on the Coordinate Attention module

The visual attention mechanism is a specialized signal-processing mechanism in the human brain for processing visual information. Humans encounter some obstacles in processing information. Therefore, they will focus on some of the information and ignore some of the less useful information[27]. Similar to the selective visual attention mechanism in humans, the attention mechanism in neural networks are designed to extract the information relevant to the current task for processing. The important information is enhanced by introducing an attention mechanism that assigns different weights to each input part. The aim is to focus attention on the more important information and reduce the attention on the remaining minor information, thus reducing the computational burden and improving the model performance[28]. In the present paper, Coordinate Attention[29] is added to the backbone network.

Hou et al. proposed an attention mechanism – Coordinate Attention (CA) in 2021, which not only captures cross-channel information but also incorporates direction-aware and position-sensitive information, precise object region detection and finer localization of traffic signs in small objects. This not only enhances model accuracy, but also requires minimal computational overheads[29].

CA encodes the channel relationship and long-term dependency by precise location information. Firstly, we embed Coordinate information and then generate Coordinate Attention. The structure diagram is shown in Fig. 3.

The first one is Coordinate information embedding. When global encoding of spatial information of channel attention, the global pooling method, compresses the global spatial information and lacks the location information. To be able to obtain more accurate location information and capture remote spatial interactions, this paper decomposes the global pooling and converts it into a bunch of one-dimensional feature encoding operations with the following equation:

$$z_c = \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W} x_c(i,j) \quad (8)$$

The input is X. The pooling kernel of size (1, W) or (H, 1) is applied to encode each channel in horizontal and vertical coordinates respectively. Thus, the output of the c-th channel with height h can be expressed as follows:

$$z_c^h(h) = \frac{1}{W}\sum_{0\le j<W} x_c(h,i) \quad (9)$$

In the same way, the output of channel c with width w can be expressed as:

$$z_c^w(w) = \frac{1}{H}\sum_{0\le j<H} x_c(j,w) \quad (10)$$

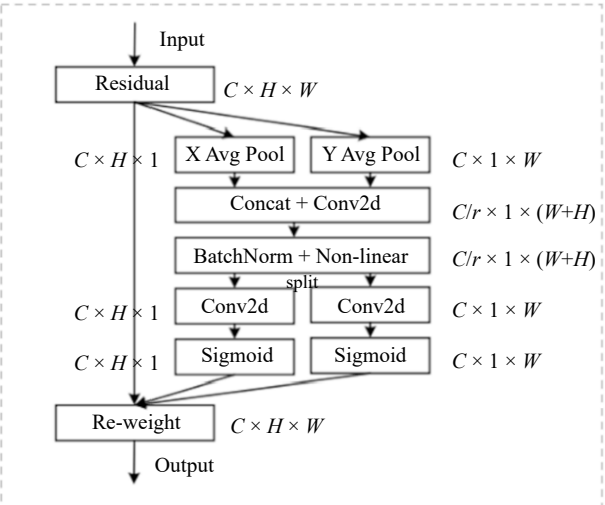

Fig. 3 Structure of coordinate attention.

Then the Coordinate Attention is generated. After passing the transformations in the information embedding, this part concatenates the result of the embedding and then transforms it using the convolutional transform function.

$$f = \delta\left(F_1\left(\left[z^h, z^w\right]\right)\right) \tag{11}$$

$$g^h = \sigma\left(F_h\left(\left[f^h\right]\right)\right) \tag{12}$$

$$g^w = \sigma\left(F_w\left(\left[f^w\right]\right)\right) \tag{13}$$

After the above module, the final output y is obtained and can be expressed as:

$$y_c(i,j) = x_c(i,j) \times g_c^h(i) \times g_c^w(j) \tag{14}$$

Improvement of loss function

Although the usage of CIoU in the original algorithm accelerates the regression of the predicted box to some extent, there are still certain issues. During the regression of the predicted box, if the aspect ratio of the predicted box matches the true box's aspect ratio, the predicted box's width and height cannot increase or decrease simultaneously, and the regression optimization process cannot proceed. Therefore, the CIoU function is replaced by the EIoU function in this paper[30].

The EIoU is calculated as Eqn (15), where w and h are the width and height of the minimum external box that covers the real box of the prediction box. It takes into account the overlap area, the distance between centroids, and the real distance between centroids, as well as the actual differences in width and height. Moreover, Focal Loss is introduced to address the problems of other localization loss functions, which helps the model converge quickly, makes the regression process more stable, and enhances the precision of the predicted bounding box regression.

$$\begin{aligned} L_{EIoU} &= L_{IOU} + L_{dis} + L_{asp} \\ &= 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \frac{\rho^2(w, w^{gt})}{C_w^2} + \frac{\rho^2(h, h^{gt})}{C_h^2} \end{aligned} \tag{15}$$

where, C_w and C_h represent the width and height of the minimum external box covering both boxes.

In summary, the improved YOLOv8 model architecture is illustrated in Fig. 4.

Experiments

Datasets and evaluation metrics

The traffic sign dataset in this paper is derived from the CCTSDB dataset[31], which was produced by the team of Zhang from Changsha University of Science and Technology (Changsha, China), in which the images are Chinese street scenes taken under the driving recorder. The dataset covers traffic sign images under various traffic environments, which is more in line with real traffic scenes.

There are 17,856 images in the CCTSDB dataset, which includes three categories of traffic signs: prohibition, indication, and warning, as shown in Fig. 5, and their locations are calibrated. The distribution of the number of each category in the training set is illustrated inFig. 6. Based on the 1,500 original test sets, the papers categorized them into six categories of multi-weather test sets, including cloud, foggy, night, rain, snow, and sunny.

To better evaluate the model, Inference Time and mAP were chosen to evaluate the detection speed and accuracy of the model, where Inference Time is the inference time, indicating the speed of model inference. mAP is the average of the region enclosed by the P-R curve, reflecting the recognition accuracy of the model.

Experimental environment

The experiments in the paper are trained on a server equipped with a GPU, model NVIDIA GEFORCE RTX 2080Ti, the server's operating system is Ubuntu system with version 18.04 and with 11 GB of video memory, Python language development, Pytorch-based deep learning framework, and GPU acceleration tool CUDA11.1.

In the inference stage, to confirm the real-time effectiveness of the model, this study opts for the NVIDIA Jetson Nano embedded platform, which has a core processing unit using CPU + GPU heterogeneous computing mode, which establishes that the platform can run neural network applications for image classification, object detection, etc. Meanwhile, the deep learning inference acceleration engine TensorRT is available to

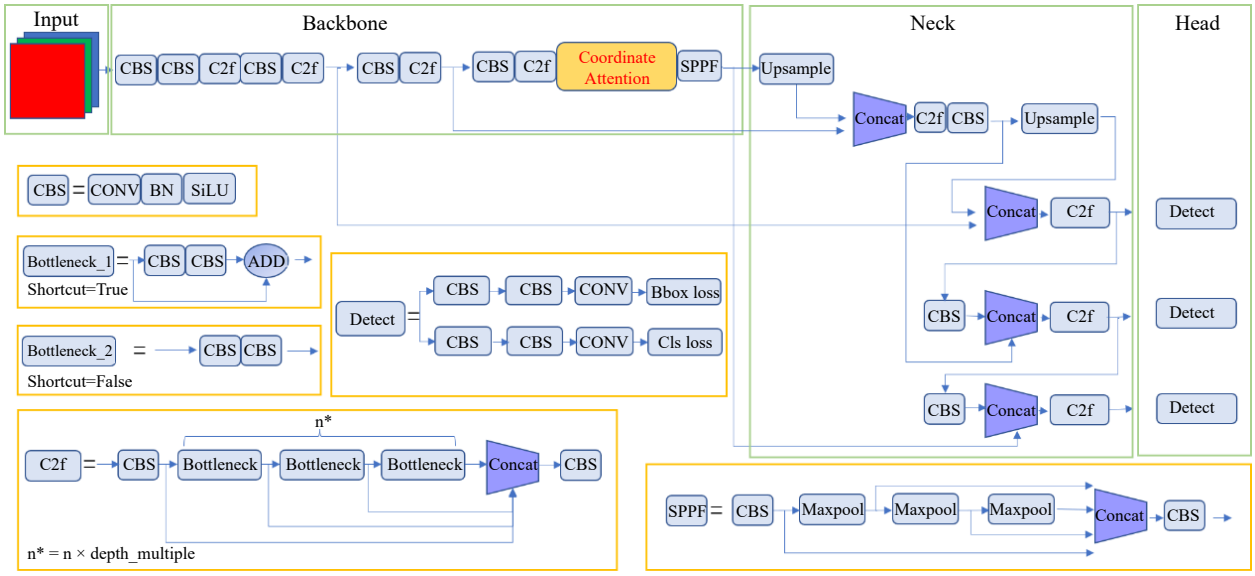

Fig. 4 Structure of the improved YOLOv8.

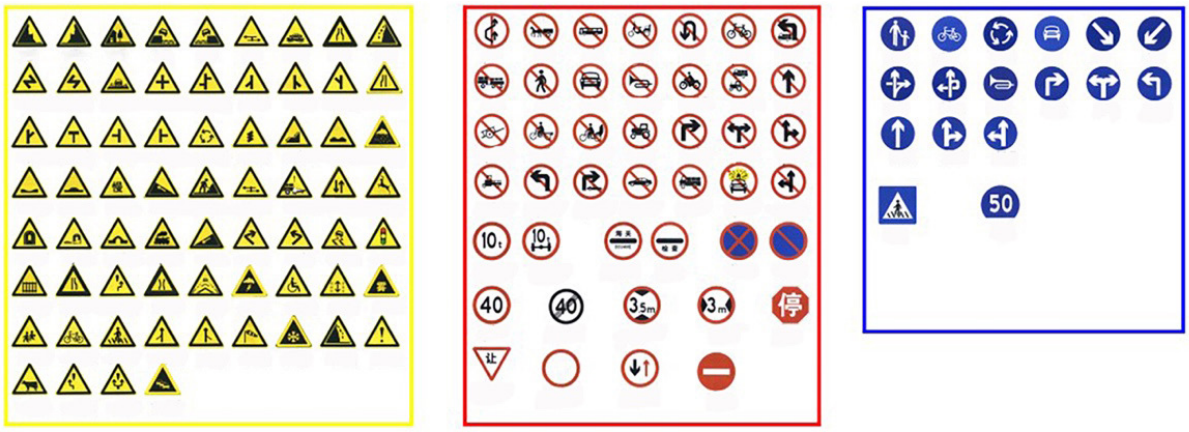

Fig. 5 Traffic sign categories.

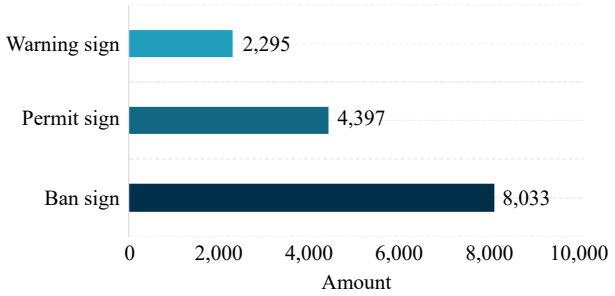

Fig. 6 Number of traffic signs for each category.

accelerate the Inference time of artificial intelligence projects. In the inference stage, to confirm the real-time effectiveness of the model, this study opts for the NVIDIA Jetson Nano embedded platform, which has a core processing unit using CPU + GPU heterogeneous computing mode, which establishes that the platform can run neural network applications for image classification, object detection, etc. Meanwhile, the deep learning inference acceleration engine TensorRT is available to accelerate the Inference time of artificial intelligence projects. Figure 7 illustrates the hardware platform, while Table 1 specifies the configuration used in the study.

During training, the model takes 640 × 640 pixel input images and employs the Adam optimizer with an initial learning rate of 0.01, runs for 300 epochs with a batch size of 16, and a weight decay of 0.0005.

Experiment results and analysis

Ablation study

To confirm the validity of each part of the improvement methods, ablation studies are conducted on the CCTSDB dataset, based on YOLOv8n, combining Coordinate Attention, and EIoU to verify the effect of different improvement methods for the network. Precision, Recall, and mAP @0.5 are used as evaluation metrics. The resolution of 640 × 384 is selected, and the complex integrated image test set is chosen, and the outcomes of the ablation study are presented in Table 2.

Comparing YOLOv8 with the models CA and EIoU, it is evident from the results that both CA and EIoU enhance the models' performance, and improve by 1.6% and 1.2% on mAP, respectively. After YOLOv8 combines all the improvements, the model performs well in terms of Precision, Recall, and mAP. Compared with the baseline model, the improvement in mAP was 2.8%. According to the findings from the ablation experiments, it is known that the CA and EIoU enhancements are effective.

Comparison with other algorithms

To further validate the comprehensive performance effect of the YOLOv8-CE algorithm in terms of detection accuracy and inference speed on different types of test sets, five object

Fig. 7 Structure of Jetson Nano.

Table 1. Hardware parameters of Jetson Nano.

Parameter	Technical specifications
AI performance	472 GFLOPs
GPU	128-core Maxwell
Memory	4 GB 64-bit LPDDR4 25.6 GB/s
CPU	Quad-core ARM A57 @ 1.43 GHz
CPU max frequency	1.43 GHz
Storage	microSD (not included)
Connectivity	Gigabit Ethernet, M.2 Key E
USB	4x USB 3.0, USB 2.0 Micro-B
Power	5 W - 1 0W

Table 2. Results of the ablation study.

Model	Precision	Recall	mAP@0.5
YOLOv8n	88.7%	73.6%	83.3%
CA	89.6%	75.4%	84.9%
EIoU	87.9%	75.1%	84.5%
CA + EIoU	90.2%	78.1%	86.1%

detection algorithms were selected, including YOLOv8n, YOLOv8-ghost, YOLOv8-ghostv2, YOLOv8-shufflenetv2, and YOLOv8-mobilenetv3, to compare with this algorithm. The comparison experiments were performed on the seven test sets delineated in this paper, and 640 × 384 resolution was selected. Four metrics, Precision, Recall, mAP, and Inference time on Jetson Nano, were selected to evaluate each algorithm. Table 3 illustrates the results of the comparison of accuracy and inference speed among different algorithms. Table 4 illustrates the results of accuracy comparison between different algorithms under different weather test sets. Figure 8 presents the detection results of YOLOv8-CE under different weather test sets.

It can be seen that the mAP at the highest in all three test sets is YOLOv8-CE, whose mAPs are 86.1% on the original test set, and the inference time of this model on Jetson Nano is only 96 ms. YOLOv8n exhibits the shortest inference time on Jetson Nano, but it has lower accuracy compared to the algorithm used in this study on all test sets. In comparison to YOLOv8n, YOLOv8-CE achieved an accuracy of 2.8% on the original test set, while detecting only 4 ms slower. Meanwhile, YOLOv8-CE showed the best overall performance on all data sets. The algorithm outperforms YOLOv8n by an average of 6.1% in terms of accuracy. For the other algorithms in the experiment, the algorithm shows better performance. The algorithm ensures real-time detection of embedded devices while also improving detection accuracy to a certain extent, and performs well in extreme weather conditions and other conditions.

Table 3. Comparison of traffic sign detection models.

Model	Precision	Recall	mAP@50	mAP@50-95	Inference time (ms)
YOLOv8-CE(ours)	90.2	78.1	86.1	57.2	96
YOLOv8n	88.7	73.6	83.3	53.7	92
YOLOv8-ghost	89.2	72.5	82.1	52.1	243
YOLOv8-ghostv2	89.1	71.6	82.2	52.5	230
YOLOv8-shufflenetv2	81.1	53.6	61.8	61.8	110
YOLOv8-mobilenetv3	73.4	54.6	61.4	35.7	65

Results of field tests

To confirm the real-time performance of the model on embedded devices, the trained model in this paper is deployed into the NVIDIA Jetson Nano embedded system, and the input resolution is set to 640 × 384 to complete the traffic sign detection in the live video of Harbin, China. The hardware system of

Table 4. Comparison of traffic sign detection models under different weather test sets.

Model	Orinal	Cloud	Foggy	Night	Rain	Snow	Sunny
YOLOv8-CE(ours)	86.1	92.5	81.6	76.5	43.1	86.5	94.8
YOLOv8n	83.3	89.5	68.3	73.8	32.7	77.8	93.2
YOLOv8-ghost	82.1	88.8	77.9	75.5	29.8	82.5	91.3
YOLOv8-ghostv2	81.9	89.2	63.9	73.0	39.9	70.0	91.3
YOLOv8-shufflenetv2	61.4	74.9	53.7	40.6	10.8	45.7	80.0
YOLOv8-mobilenetv3	61.8	75.9	55.2	40.6	16.9	58.6	77.6

Fig. 8 Detection results of YOLOv8-CE under different weather test sets.

Jetson Nano is shown in Fig. 9 and the detection results are shown in Fig. 10.

In addition, this paper compares the Jetson Nano with the Raspberry 4B Pi and deploys the same algorithm on both devices for testing and comparing the inference time, as shown in Table 5. It can be seen that the computational volume and Weight Size of the algorithm in this paper are similar to YOLOv8n, and the inference time is around 96 ms. The inference time on Jetson Nano is 1/7 of that on Raspberry Pi 4B. Although the inference time of YOLOv8-CE is not the fastest, its overall performance is the best and the inference time is within 100 ms, which is far enough to meet the requirements of real-time detection. In addition, it can also be seen that the inference time of all models on Jetson Nano is within 250 ms, which achieves real-time detection.

Discussion and conclusions

In this paper, an improved detection algorithm of YOLOv8 combined with the embedded system Jetson Nano was proposed to realize real-time detection of traffic signs on self-driving or assisted-driving vehicles. The main findings are as follows:

Fig. 9 Hardware system of Jetson Nano.

(1) Through ablation study, the effectiveness of the improved network combining Coordinate Attention, EIoU is considered as a function in the paper is verified. The improved YOLOv8n model achieves 86.1% mAP @0.5 in the original test set. Compared to the original YOLOv8, Precision, Recall, and mAP @0.5 are improved by 1.5%, 3.5%, and 2.8%, respectively, and the accuracy is also improved in other test sets, with better results in all kinds of scenarios and stronger generalization ability. In addition, the inference time on Jetson Nano is increased by 4 ms, the model memory is increased by 0.03 MB, and the FLOPs are approximately equal. Furthermore, the improved method adopted in the paper increases the detection accuracy substantially while slightly reducing the detection speed. The experimental results demonstrate the superiority of the YOLOv8-CE method adopted in this paper.

(2) The superiority of the YOLOv8-CE model is further confirmed by comparing it with classical lightweight models such as YOLOv8-mobilenetv3, and YOLOv8-ghost. The improved model outperforms the other three models in terms of accuracy and detection speed in general, showing the best performance. The experimental results further demonstrate that the improved approach adopted in the paper has good detection performance.

(3) By conducting live video tests on different embedded devices, it can be seen that the Jetson Nano far outperforms the Raspberry Pi in the detection of traffic signs, and the detection speed of YOLOv8-CE enters within 100 ms on the Jetson Nano, reaching 96 ms, achieving the performance of real-time

Table 5. Experiments on different devices.

Model	FLOPs (G)	Weight size (MB)	Inference time (ms)	
			Jetson Nano	Raspberry pi 4B
YOLOv8-CE(ours)	8.1	5.99	96	690
YOLOv8n	8.1	5.96	92	678
YOLOv8-ghost	6.8	5.97	243	810
YOLOv8-ghostv2	6.8	5.17	230	791
YOLOv8-shufflenetv2	5.0	3.48	110	580
YOLOv8-mobilenetv3	2.8	2.52	65	313

Fig. 10 Field test on Jetson Nano in Harbin, China.

vehicle traffic sign detection. The experimental results demonstrate the feasibility of this paper's model for edge computing platforms.

To summarize, the YOLOv8-CE model can detect the type and location of traffic signs more accurately and quickly, which serves as a foundation for future advancements in real-time traffic sign detection and provides a basis for the implementation of autonomous driving. In addition, future studies will concentrate on the following fields:

(1) Some lightweight methods, such as model pruning[32,33], model quantization[34,35], and knowledge distillation[36,37], will be used to be less computationally intensive and consume less model, while combining lightweight network models to create a lightweight network with better performance that is compatible with detection in small mobile devices to further enhance the speed of traffic sign detection.

(2) Collecting data sets from more situations for model training and processing the data to make the model more generalizable and able to adapt to traffic sign recognition in various situation scenarios.

(3) The network model is going to be further improved and self-attention will be added to fuse features more effectively, which in turn will better detection performance of the model.

Author contributions

The authors confirm contribution to the paper as follows: study conception and design: Luo Y, Ci Y; data collection: Luo Y, Zhang H, Wu L; analysis and interpretation of results: Luo Y, Ci Y, Zhang H, Wu L; draft manuscript preparation: Luo Y, Ci Y. All authors reviewed the results and approved the final version of the manuscript.

Data availability

The data that support the findings of this study are available in the github repository. These data were derived from the following resources available in the public domain: https://github.com/csust7zhangjm/CCTSDB.

Acknowledgments

This work was financially supported by Heilongjiang Provincial Natural Science Foundation of China (LH2023E055), and the National Key R & D Program of China (2021YFB2600502).

Conflict of interest

The authors declare that they have no conflict of interest.

Dates

Received 8 July 2024; Accepted 23 July 2024; Published online 30 September 2024

References

1. Huang Z, Li L, Krizek GC, Sun L. 2023. Research on traffic sign detection based on improved YOLOv8. *Journal of Computer and Communications* 11:226–32
2. Zheng T, Huang Y, Liu Y, Tang W, Yang Z, et al. 2022. CLRNet: cross layer refinement network for lane detection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, LA, USA, 18-24 June 2022*. USA: IEEE. pp. 888–97. doi: 10.1109/cvpr52688.2022.00097
3. Qie K, Wang J, Li Z, Wang Z, Luo W. 2024. Recognition of occluded pedestrians from the driver's perspective for extending sight distance and ensuring driving safety at signal-free intersections. *Digital Transportation and Safety* 3:65–74
4. Wang Q, Li X, Lu M. 2023. An improved traffic sign detection and recognition deep model based on YOLOv5. *IEEE Access* 11:54679–91
5. Lai H, Chen L, Liu W, Yan Z, Ye S. 2023. STC-YOLO: small object detection network for traffic signs in complex environments. *Sensors* 23:5307
6. Chu J, Zhang C, Yan M, Zhang H, Ge T. 2023. TRD-YOLO: a real-time, high-performance small traffic sign detection algorithm. *Sensors* 23:3871
7. de la Escalera A, Moreno LE, Salichs MA, Armingol JM. 1997. Road traffic sign detection and classification. *IEEE Transactions on Industrial Electronics* 44:848–59
8. Fleyeh H. 2004. Color detection and segmentation for road and traffic signs. *IEEE Conference on Cybernetics and Intelligent Systems, Singapore, 1–3 December 2004*. USA: IEEE. pp. 809–14. doi: 10.1109/iccis.2004.1460692
9. Maldonado-Bascón S, Lafuente-Arroyo S, Gil-Jimenez P, Gómez-Moreno H, López-Ferreras F. 2007. Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems* 8:264–78
10. Cireşan D, Meier U, Masci J, Schmidhuber J. 2011. A committee of neural networks for traffic sign classification. *The 2011 International Joint Conference on Neural Networks. San Jose, CA, USA, 31 July – 5 August 2011*. USA: IEEE. pp. 1918–21. 10.1109/ijcnn.2011.6033458
11. Girshick R, Donahue J, Darrell T, Malik J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA, 23–28 June 2014*. USA: IEEE. pp. 580–87. doi: 10.1109/cvpr.2014.81
12. Girshick R. 2015. Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV). Santiago, Chile, 7–13 December 2015*. USA: IEEE. pp. 1440–48. DOI: 10.1109/iccv.2015.169
13. Ren S, He K, Girshick R, Sun J. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39:1137–49
14. Redmon J, Divvala S, Girshick R, Farhadi A. 2016. You only look once: unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27-30 June 2016*. USA: IEEE. pp. 779–88. doi: 10.1109/cvpr.2016.91
15. Redmon J, Farhadi A. 2017. YOLO9000: better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21-26 July 2017*. USA: IEEE. pp. 6517–25. doi: 10.1109/cvpr.2017.690
16. Redmon J, Farhadi A. 2018. YOLOv3: an incremental improvement. *arXiv* Preprint:1804.02767
17. Bochkovskiy A, Wang CY, Liao HY M. 2020. YOLOv4: optimal speed and accuracy of object detection. *arXiv* Preprint:2004.10934
18. Chen B, Fan X. 2024. MSGC-YOLO: an improved lightweight traffic sign detection model under snow conditions. *Mathematics* 12:1539
19. Zhang LJ, Fang JJ, Liu YX, Hai FL, Rao ZQ, et al. 2024. CR-YOLOv8: multiscale object detection in traffic sign images. *IEEE Access* 12:219–28
20. Kim W. 2009. Cloud computing: today and tomorrow. *The Journal of Object Technology* 8:65–72
21. Luo Y, Ci Y, Jiang S, Wei X. 2024. A novel lightweight real-time traffic sign detection method based on an embedded device and YOLOv8. *Journal of Real-Time Image Processing* 21:24
22. Artamonov NS, Yakimov PY. 2018. Towards real-time traffic sign recognition via YOLO on a mobile GPU. *Journal of Physics: Conference Series* 1096:012086

23. He K, Zhang X, Ren S, Sun J. 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37:1904–16
24. Lin TY, Dollár P, Girshick R, He K, Hariharan B, et al. 2017. Feature pyramid networks for object detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017*. USA: IEEE. pp. 936–44. doi: 10.1109/cvpr.2017.106
25. Li H, Xiong P, An J, Wang L. 2018. Pyramid attention network for semantic segmentation. *arXiv* Preprint:1805.10180
26. Zheng Z, Wang P, Liu W, Li J, Ye R, et al. 2020. Distance-IoU loss: faster and better learning for bounding box regression. *Proceedings of the AAAI Conference on Artificial Intelligence* 34:12993–3000
27. Soydaner D. 2022. Attention mechanism in neural networks: where it comes and where it goes. *Neural Computing and Applications* 34:13371–85
28. Sun Z, Yang H, Zhang Z, Liu J, Zhang X. 2022. An improved YOLOv5-based tapping trajectory detection method for natural rubber trees. *Agriculture* 12:1309
29. Hou Q, Zhou D, Feng J. 2021. Coordinate attention for efficient mobile network design. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021*. USA: IEEE. pp. 13708–17. doi: 10.1109/cvpr46437.2021.01350
30. Zhang YF, Ren W, Zhang Z, Jia Z, Wang L, et al. 2022. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* 506:146–57
31. Zhang J, Zou X, Kuang LD, Wang J, Sherratt RS, et al. 2022. CCTSDB 2021: a more comprehensive traffic sign detection benchmark. *Human-centric Computing and Information Sciences* 12:23
32. Molchanov P, Tyree S, Karras T, Aila T, Kautz J. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv* Preprint:1611.06440
33. Han S, Mao H, Dally WJ. 2015. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* Preprint:1510.00149
34. Rastegari M, Ordonez V, Redmon J, Farhadi A. 2016. XNOR-net: ImageNet classification using binary convolutional neural networks. In *Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science*, eds. Leibe B, Matas J, Sebe N, Welling M. vol. 9908. Cham: Springer. pp. 525–42. doi: 10.1007/978-3-319-46493-0_32
35. Li Z, Ni B, Zhang W, Yang X, Gao W. 2017. Performance guaranteed network acceleration via high-order residual quantization. *2017 IEEE InternationalConference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017*. USA: IEEE. pp. 2584–92. doi: 10.1109/iccv.2017.282
36. Romero A, Ballas N, Kahou SE, Chassang A, Gatta C, et al. 2014. FitNets: hints for thin deep nets. *arXiv* Preprint:1412.6550
37. Kim J, Park S, Kwak N. 2018. Paraphrasing complex network: network compression via factor transfer. *arXiv* Preprint:1802.04977