Based on improved YOLOv8 and Bot SORT surveillance video traffic statistics

Yiqun Yang
Hubei Normal University
Daneng Pi
pidaneng@163.com
Hubei Normal University
Lingyan Wang
Hubei Normal University
Mingliang Bao
Hubei Normal University
Jianfu Ge
Hubei Normal University
Tingchen Yuan
Hubei Normal University
Houshi Yu
Hubei Normal University
Qi Zhou
Hubei Normal University

Research Article

Keywords: YOLOv8, Bot SORT, SPD-Conv, CoTAttention, Traffic flow statistics

Posted Date: March 28th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-4161504/v1

Additional Declarations: No competing interests reported.

Based on improved YOLOv8 and Bot SORT surveillance video traffic statistics

Yiqun Yang, Daneng Pi* , Lingyan Wang, Mingliang Bao, Jianfu Ge, Tingchen Yuan, Houshi Yu, and Qi Zhou

School of Electrical Engineering and Automation, Hubei Normal University, Huangshi 435002, P. R. China

***E-mail**: yangyiqun2025@163.com, pidaneng@163.com.*

Keywords: YOLOv8, Bot SORT, SPD-Conv, CoTAttention, Traffic flow statistics

Abstract:Aiming at the problems of leakage detection and low detection accuracy of existing deep learning based surveillance video traffic flow detection algorithms, a traffic flow counting system combining improved YOLOv8 detection and Bot SORT tracking is proposed. First, the backbone network is used to incorporate the SPD-Conv convolutional layer to improve the network's ability to detect small targets. Then, the attention mechanism CoTAttention is introduced into the neck network to further improve the model generalization ability. Finally, the improved YOLOv8 model and the Bot SORT algorithm are combined to design and implement a traffic counting system capable of monitoring video traffic in real time, and trained and tested on the open-source UA-DETRAC vehicle detection dataset. The experimental results show that the improved YOLOv8 algorithm improves F1, P, mAP50, and mAP50-95 by 0.36, 2.2, 1.8, and 2.1 percentage points, respectively, compared with the original algorithm. Combined with the Bot SORT tracking, it achieves more accurate and reliable results in the task of traffic counting, which provides a strong support for the vehicle detection and counting in the monitoring system.

1. Introduction

With the continuous progress of urbanization and the popularity of transportation, traffic flow management on urban roads has become increasingly complex and important. In this context, the surveillance video traffic counting system becomes one of the important tools for urban traffic management. By using computer vision technology, especially target detection algorithms, the traffic flow counting system can accurately analyze the traffic flow in the surveillance video in real time and provide powerful data support for urban traffic management[1].

Surveillance video-based traffic counting usually contains two parts: vehicle detection[2] and vehicle tracking[3]. Traditional target detection algorithms include frame difference method[4], optical flow method[5], and background difference method[6]. In this case, the frame difference method is a pixel level based method that detects a target by comparing the differences between consecutive frames. When a target undergoes motion, it causes a change in pixel values between neighboring frames. However, the frame difference method is less effective in dealing with lighting variations and large scene dynamics, and is susceptible to noise. The optical flow method captures the motion information of a target by analyzing the temporal displacement of pixels in an image, and in the presence of occlusion and illumination variations, the optical flow method is prone to erroneous motion estimation, leading to inaccurate target detection. The background difference method detects a moving target by modeling a stationary background and comparing the current frame with the background. In the presence of lighting changes, camera shaking, or dynamic backgrounds in the scene, the background difference method is prone to false alarms or missed alarms, which reduces the accuracy and stability of detection. Although these traditional methods are still useful in some simple application scenarios, with the development of computer vision and deep learning, in order to overcome the shortcomings of traditional vehicle detection methods, target detection algorithms based on deep learning, have made significant progress to better cope with complex scenarios and diverse target detection tasks[7].

2. Related work

Deep learning based target detection algorithms are mainly categorized into two types: the One-Stage[8] and Two-Stage[9] structures. Among them, the Two-Stage algorithm is to first generate candidate regions, and then to classify and precisely locate the targets in these candidate regions. Such as R-CNN[10], Fast RCNN[11], Faster R-CNN[12], Mask R-CNN[13], etc.; One-Stage algorithms directly predict the target bounding box and categories in the image with a feed-forward neural network structure, and usually use regression to accomplish the target localization and classification tasks at the same time, such as the SSD[14], RetinaNet[15], and YOLO[16] series.

Currently, the YOLO model has evolved to the latest YOLOv8[17], released by Ultralytics, YOLOv8 has higher detection accuracy and faster detection speed compared to the previous version. It employs a deep neural network architecture based on the Darknet framework by dividing the input image into multiple grid cells, each of which is responsible for detecting the targets in it. This grid-based design allows YOLOv8 to detect multiple targets at the same time and has an advantage in processing speed.

For vehicle tracking, SORT[18] is a lightweight multi-target tracking algorithm based on the Hungarian algorithm. It achieves real-time target tracking by first predicting the target using Kalman filtering and then matching the association between detection and prediction using the Hungarian algorithm[19]. However, when the target is partially or completely occluded, Kalman filtering[20] may not be able to accurately estimate the target's position and the tracking effect is poor.The SORT algorithm is susceptible to the problem of target ID confusion in multi-target tracking which means that different targets are incorrectly assigned the same ID. To solve such problems, Bot SORT[21] is an online and real-time target tracking algorithm. It is based on an improved version of the SORT algorithm, which can better deal with the problem of target re-identification by introducing appearance features.Even if the target is not visible for a period of time, it is still able to be matched and tracked by

the appearance features,and target matching and ID assignment can be performed more accurately.

To summarize, this paper adopts an improved YOLOv8n-based target detection model as a detector, and then combines it with the Bot SORT target tracking algorithm for traffic counting.YOLOv8n is able to efficiently carry out target detection, while Bot SORT has an improvement in multi-target tracking, which improves the accuracy of target re-recognition through the introduction of appearance features, and thus better adapts to the complex traffic scenarios. This end-to-end solution will effectively realize accurate traffic counting and tracking, providing strong support for traffic management and decision-making.

3. Introduction to the YOLOv8n algorithm

The network structure of YOLOv8n is divided into four main parts: the Input input, the Backbone backbone feature extraction network, the Neck network, and the Head output, as shown in Figure 1.

The Input input is mainly composed of Mosaic data enhancement, automatic image cropping and splicing, and adaptive anchor frames[21-23].Mosaic data enhancement is used to increase the diversity of the training data by splicing multiple images together to form a single large input image.In this way, the model can be made to learn the features of the target better at different locations and scales. Automatic image cropping stitching is used to generate a larger input image by automatically cropping and stitching images. This increases the diversity of the training data and can handle targets at different scales. Adaptive anchor frame is to automatically adjust the size and scale of the anchor frame according to the distribution of the training data to improve the accuracy of target detection.

The Backbone backbone feature extraction network adopts the CSPDarknet network architecture, which mainly consists of CBS (standard convolutional layer), C2f module, and SPPF[24-26]. The CBS is composed of a series of convolutional, batch normalization, and activation function layers for extracting features of the input image.

The C2f module replaces the original C3 module, which is mainly referenced to the C3 module and the ELAN module, and obtains richer gradient information by branching more gradient streams in parallel, thus obtaining richer feature information while ensuring lightweight.The SPPF pyramid pooling fusion module, which is used to perform pooling operations on feature maps at different scales, and fuses pooled feature maps to be able to capture image information at different scales and improve the network's ability to adapt to scale changes.

The Neck network is located between Backbone, the backbone network of YOLOv8n, and Head, the output, and plays the role of feature fusion and feature enhancement. A PAN-FPN[27] structure is used, which fuses feature maps from different layers through bottom-up and top-down paths and generates a series of feature maps with different resolutions.

The Head outputs include components such as prediction of bounding boxes, prediction of categories, loss function layers, processing of feature maps, and non-extreme value suppression. These outputs can help to determine the location and category of the target in the image and generate the final target detection result.

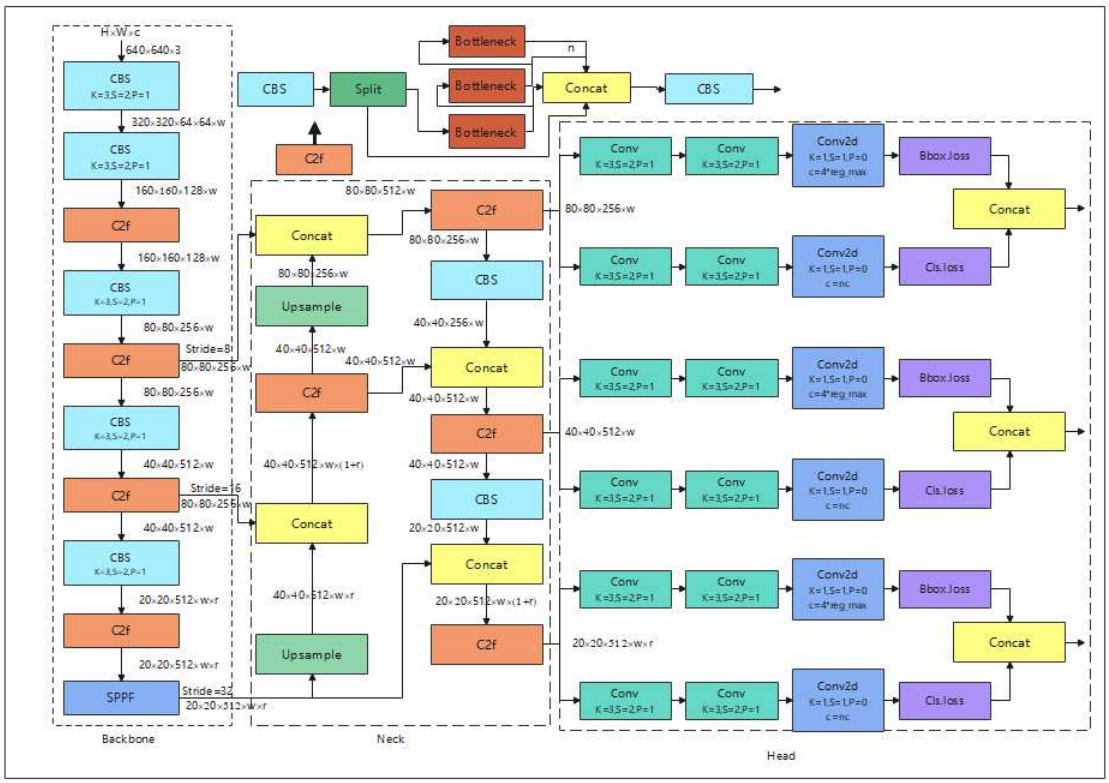

Figure 1 YOLOv8n network architecture diagram

4. Improvement of YOLOv8n algorithm

4.1 Introduction of the SPD-Conv module

SPD-Conv is composed of a space-to-depth (SPD) layer and a non-stepwise convolution (Conv) layer[28].SPD-Conv first performs a space_to_depth operation on the input feature maps to convert the spatial dimension to the depth dimension. Then, ordinary convolution operation is performed on the converted feature map. The output feature map is then subjected to depth_to_space operation to convert the depth dimension back to the spatial dimension. Finally, by combining the space_to_depth operation with the ordinary convolution operation, SPD-Conv can increase the sensory field of the network and extract richer features while keeping the shape of the input feature map unchanged. This is because the space_to_depth operation increases the receptive field of each neuron, while the ordinary convolutional operation extracts localized features.SPD-Conv is used in the convolutional operation to improve the network's ability to detect small targets.

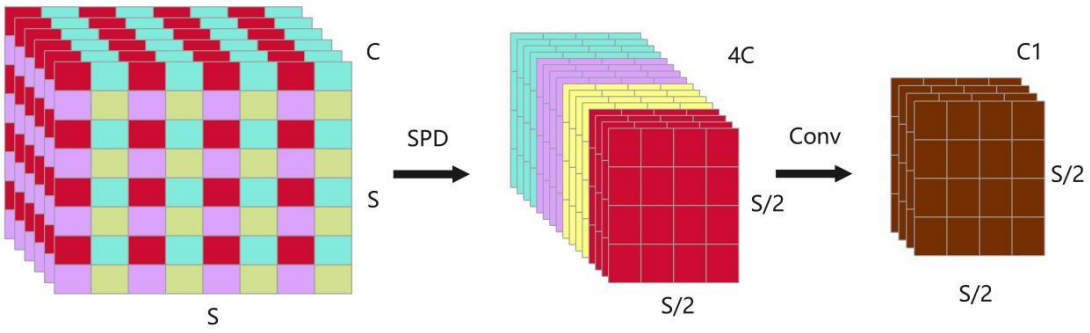

Figure 2 Structure of SPD-Conv module

Consider an intermediate feature mapping X of arbitrary size S×S×C, as shown in Figure 2. The SPD-Conv component introduces the input feature map transformation technique into the backbone network of YOLOv8n. The feature mapping X is partitioned into a series of sub-feature mappings, which are split as follows:

$$f_{0,0} = X[0:S:scale, 0:S:scale], f_{0,1} = X[1:S:scale, 0:S:scale], ...,$$

$$f_{scale-1,0} = X[scale-1:S:scale, 0:S:scale];$$

$$f_{0,1} = X[0:S:scale,1:S:scale], f_{1,1} = X[1:S:scale,1:S:scale],...,$$

$$f_{scale-1,1} = X[scale-1:S:scale,1:S:scale];$$

$$f_{0,scale-1} = X[0:S:scale,scale-1:S:scale], f_{1,scale-1} = X[1:S:scale,scale-1:S:scale],...,$$

$$f_{scale-1,scale-1} = X[scale-1:S:scale,scale-1:S:scale],...,$$

For the usual case, for any (original) feature graph X, the $f_{x,y}$ subgraph consists of all entries $X(i,j)$ that satisfy $i+x$ and $i+y$ that are divisible by scale. Thus, each subgraph downsamples X by a factor of scale. Sub-feature map formation: for the sub-feature maps obtained from the slicing operation, when scale=2, they are denoted by indexing as $f_{0,0}, f_{0,1}, f_{1,0}, f_{1,1}$, and each sub-feature map has the shape $(\frac{S}{2}, \frac{S}{2}, C)$. All sub-feature maps are connected along the channel dimension to form a new feature map X. This new feature map is reduced in spatial dimension by a factor of scale scale and increased in channel dimension by a factor of 2 of scale scale. The size of the resulting feature map X becomes $(\frac{S}{scale}, \frac{S}{scale}, scale^2 C)$. This indicates that the SPD method realizes the downsampling of the original feature map by slicing, joining and scale transforming the feature map.

4.2 Introducing the CoTAttention Attention Mechanism

Traditional Transformer models use Self-Attention mechanisms in the encoder and decoder to model the contextual relationships of input sequences. However, the Self-Attention mechanism only takes into account the associations within the input sequences and not the relationships with other input sequences, which leads to poor performance of the model when dealing with complex associations.

CoTAttention[29] solves this problem by introducing a contextual attention mechanism, as shown in Figure 3. It introduces an additional contextual attention matrix on top of the self-attention mechanism to capture the associations between input sequences. Assume that there is the same input 2D feature graph $X \in R^{H \times W \times C}$.

The keys, queries and values are defined as $K = X$, $Q = X$ and $V = XWv$. Unlike the typical self-attention mechanism where 1×1 convolution is performed for each key, the CoT block first contextualizes each key representation by spatially contextualizing all neighboring keys within the $k \times k$ grid through $k \times k$ group convolution. The contextual culture key $K1 \in R^{H \times W \times C}$ naturally reflects the static contextual information between local neighboring keys, and we consider K1 as the static contextual representation of input X.

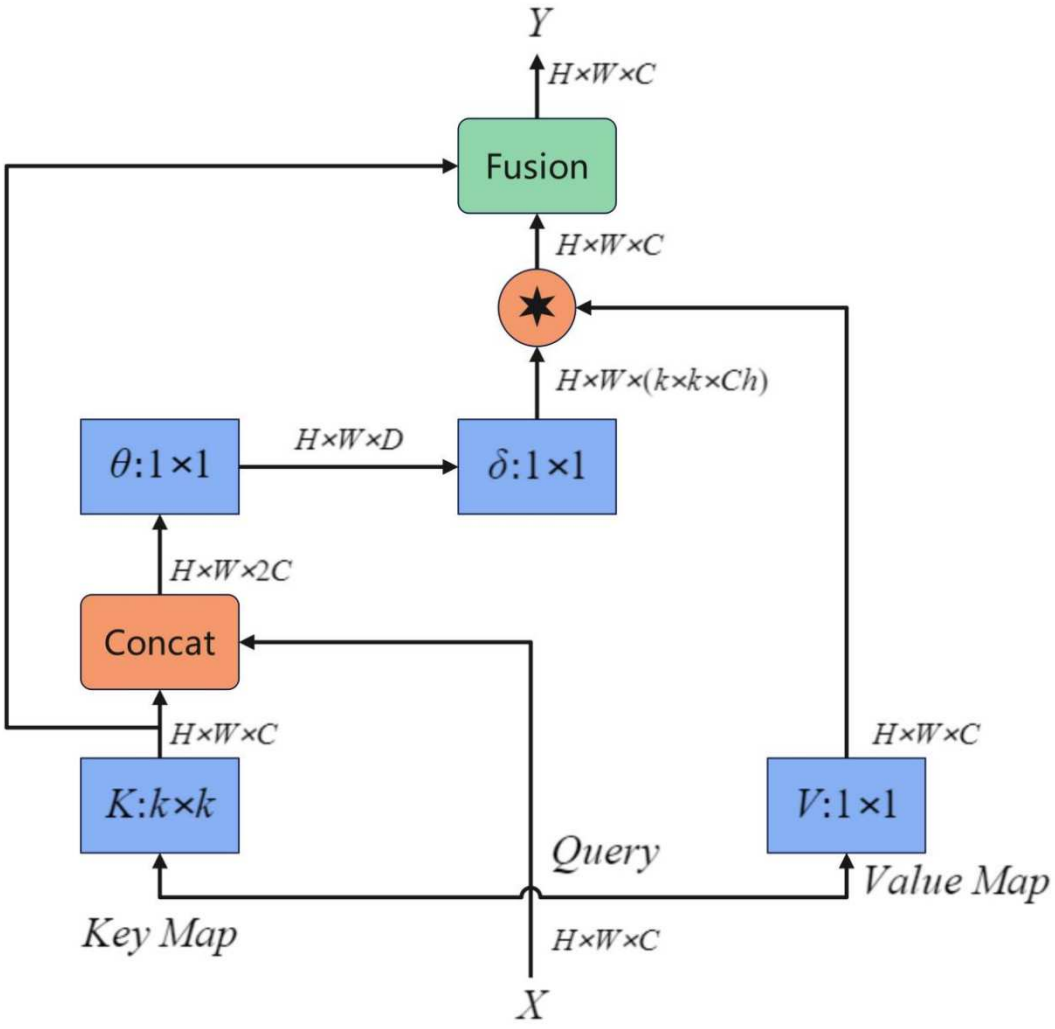

Figure 3 Context converter

Specifically, CoTAttention adjusts the attention distribution in the self-attention mechanism by calculating the similarity matrix between input sequences and using it as the weight of the contextual attention matrix. By introducing the contextual attention mechanism, CoTAttention is able to better capture the semantic associations

between input sequences, which improves the performance of the model in semantic understanding and generation tasks.

4.3 YOLOv8n_S_C algorithm

The SPD-Conv module is embedded in specific locations (indexes 2, 5, 8, and 11) of the YOLOv8n backbone network to perform a spatial depth transformation, as shown in Figure 4. It organizes blocks of pixels in a specific region into deeper channels, and this operation serves several purposes:

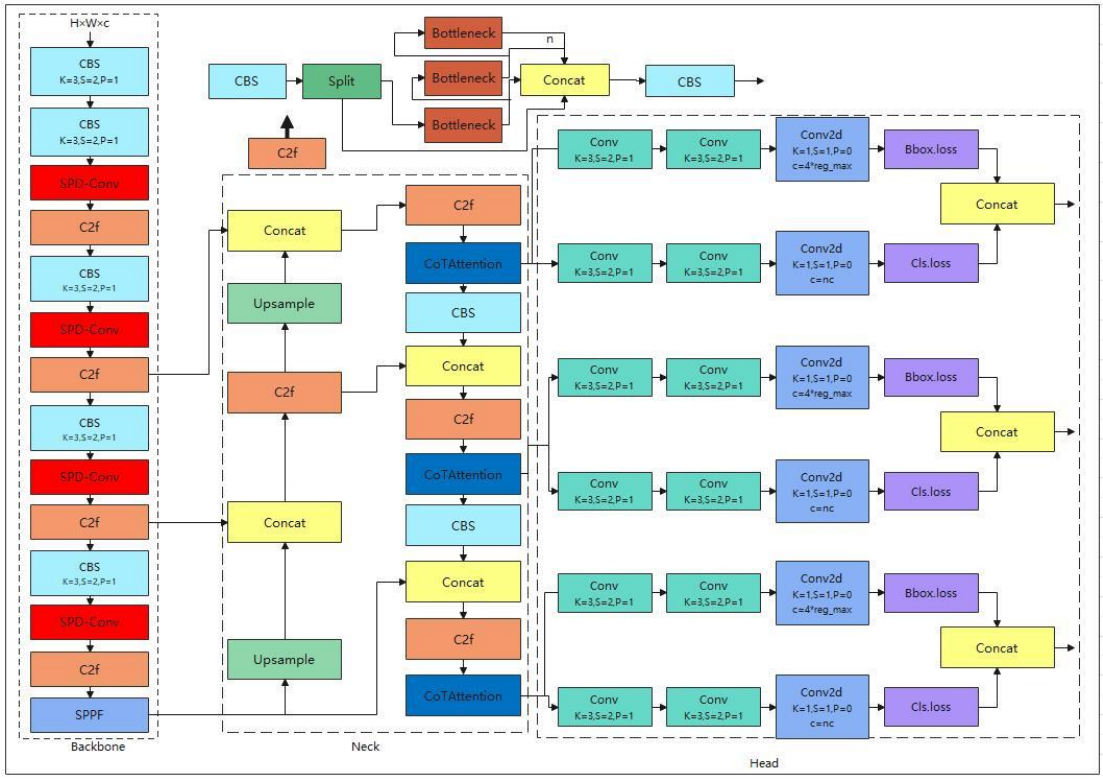

Figure 4 YOLOv8n_S_C network structure

Reduction of spatial resolution: the spatial depth transformation results in a reduction of the spatial resolution of the feature map. By organizing adjacent pixels into deeper channels, the model can capture information over a larger contextual range without significantly increasing the number of parameters.

Increased receptive field: since pixels in a specific region are organized together, the response of each channel can cover a larger receptive field. This is useful for detecting contextual information about an object, especially when the object size is relatively small.

Introduction of non-linear relationships: The spatial depth transform introduces non-linear relationships between channels. This helps the model to better learn complex features at specific locations.

In addition to this, the CoTAttention layer, which is a context transforming attention mechanism, is embedded in the configuration file at specific locations indexed 19, 22, 25 in the neck, and CoTAttention works as follows:

Contextual Attention: the CoTAttention enables the model to dynamically attend to different parts of the input image instead of processing the whole image uniformly. This is very helpful in improving the model's understanding of the relationships between objects and the context.

Enhanced Perception: by introducing the attention mechanism, the model can focus more on processing task-relevant information, thus enhancing its perceptual capabilities.

5. BoT-SORT tracking algorithms

BoT-SORT is an advanced tracking algorithm that combines motion and appearance information, camera motion compensation, and more accurate Kalman filtered state vectors, and integrates these improvements into ByteTrack to enhance the robustness of target tracking.Key features of BoT-SORT include:

Combining motion and appearance information: BoT-SORT not only utilizes the target's appearance information for tracking, but also considers the target's motion information. The accuracy of tracking is improved by joint modeling of motion trajectories and appearance features.

Camera Motion Compensation: in scenes where camera motion exists, the motion of the camera can interfere with target tracking.BoT-SORT reduces the effect of camera motion on target tracking by compensating for the camera motion[30].

More accurate Kalman-filtered state vectors: BoT-SORT uses more accurate Kalman-filtered state vectors to predict and update the target position. This helps to reduce the tracking error and improve the target localization accuracy.

Multi-target tracking: BoT-SORT can track multiple targets at the same time, taking into account the inter-relationships and occlusions between the targets. This enables accurate target tracking even in complex scenes.

Strong Robustness: Since BoT-SORT utilizes a combination of information for tracking, it has strong robustness. Accurate target tracking can be achieved even when the target's appearance changes, occlusion or camera movement occurs.

6. Experimental results and analysis

6.1 Experimental environment

In this experiment, the programming language was Python 3.9 and deep learning training was performed using the PyTorch framework without the use of pre-trained models. The parameter configurations were standardized and the experiments were conducted on a computer with Windows 10 operating system. The computer hardware configuration included a 12th Gen Intel Core i5-12500H processor, 32 GB of RAM, and an NVIDIA GeForce GTX 3070 Ti graphics card (8 GB of video memory). In order to fully utilize the computational power of the graphics card, CUDA 11.8 was chosen as the development environment for accelerated computation on the GPU. During deep learning training, a configuration of 16 samples per batch and a setting with an initial learning rate of 0.01 were used, and performance was continuously optimized over 500 training rounds.

6.2 Data set

Vehicle Detection Dataset: The UA-DETRAC dataset[31] was harvested from 24 different locations in Beijing and Tianjin, China. The dataset includes more than 140,000 frames with a total duration of 10 hours, capturing vehicle behavior in urban traffic scenes. Manually annotated with 8250 vehicles and containing 1.21 million labeled object bounding boxes, the dataset contains different traffic scenarios and environmental contexts, such as highways, intersections, and zigzags, daytime, nighttime, cloudy, and rainy. This makes the trained model more adaptable to various

complex traffic situations and environmental conditions. In this experiment, the purpose of this paper is to perform traffic counts without detailed detection of vehicle types. In order to simplify the task and increase efficiency, the choice was made to unify car, bus, van and other in the original dataset as car. such manipulation does not change the basic meaning of the information, but simply treats all vehicles as the same category when counting the traffic flow in order to process the data in a more concise manner. This simplification helps to reduce the complexity of the model and improve the accuracy and efficiency of the traffic count. A partial dataset was used for the experiments and randomly divided in the ratio of 8:1:1 to form the training, validation and test sets for a comprehensive experimental evaluation.

6.3 Evaluation indicators

In this study, the improved YOLOv8n algorithm is evaluated experimentally using several performance evaluation metrics, including precision P (Precision), recall R (Recall), average precision (AP), mean average precision (mAP), F1 score (F1 Score), and frame rate (Frames per Second, FPS).

(1) Precision (Precision) P: describes how many of all the samples predicted as positive cases by the model are truly positive cases. The calculation formula is as follows:

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \times 100\%$$

(2) Recall (Recall) R: A measure of how many of all true positive examples of the model are successfully predicted as positive. The formula is as follows:

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \times 100\%$$

(3) Average Precision (AP) AP: measures the detection performance of the model on different categories. Average precision is obtained by calculating the area under the precision-recall curve for each category. The calculation formula is as follows:

$$\mathrm{AP} = \int_0^1 \mathrm{P(R)dR}$$

(4) Mean Average Precision (mAP): indicates the average detection precision of all categories. The calculation formula is as follows:

$$\mathrm{mAP} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i}}^{\mathrm{n}} \mathrm{AP(i)} \times 100\%$$

(5) F1 score (F1 score): combines the precision rate and the recall rate, and is the reconciled average of the precision rate and the recall rate. The formula is as follows:

$$\mathrm{F1} = \frac{2 \times \mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}} \times 100\%$$

(6) Frames per Second (FPS): Indicates the performance of the algorithm in terms of processing the number of frames per second.

6.4 Ablation experiment

In order to analyze the impact of different improvement strategies on the model detection performance, this paper takes YOLOv8n as the baseline, and introduces the SPD-Conv module in the backbone network, which helps to improve the accuracy of object detection; and introduces the CoTAttention attention mechanism in the neck, which helps to reduce the model's sensitivity to the redundant information, and further improves the model's generalization ability. Three sets of experiments were designed and training was completed while ensuring the same dataset and training parameters. The experimental results are shown in Table 1.

Table 1 Results of ablation experiments

SPD-Conv	CoTAttention	mAP50/%	mAP50-95/%	Parameters/M	GFLOPs	FPS
—	—	78.6	56.8	3.01	8.2	303
√	—	79.9	58.3	3.27	11.7	**384**
√	√	**80.4**	**58.9**	**4.17**	**13.2**	322

Through the ablation experiments in Table 1, the first row reflects the benchmark performance of the original YOLOv8n on the dataset. After introducing the SPD-Conv module and CoTAttention attention mechanism respectively, it is observed that the SPD-Conv module improves the detection results more significantly,

including the mAP50, mAP50-95 and FPS. Through analysis, it is concluded that the SPD-Conv module introduces spatial depth convolution to better capture the context and features of the object, which is especially suitable for scenes that need to deal with objects of different scales. In contrast, the relatively weak performance of the CoTAttention attention mechanism is due to the fact that the contextual information has less impact on the detection accuracy, and therefore, CoTAttention has a small improvement on the detection performance. Meanwhile, the introduction of SPD-Conv and CoTAttention achieves the best results for the detection network, with mAP50 and mAP50-95 improving by 1.8, 2.1 percentage points. This shows that the combined effect of the two modules achieved a synergistic effect in improving the detection performance.

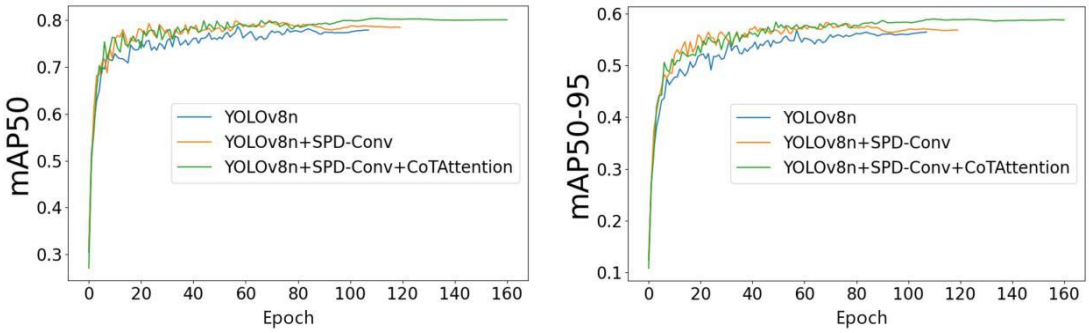

Figure. 5 Comparison of mAP during training for ablation experiments

According to the observation in Figure 5,the blue curve in both graphs performs the worst in terms of mAP, while the green curve performs the best. When the original YOLOv8n model is introduced into the SPD-Conv module and trained for 15 rounds, the mAP metrics of the orange curve begin to significantly exceed those of the blue curve, showing significant performance improvement. Subsequently, when the CoTAttention attention mechanism is introduced and training continues to about 90 rounds, the mAP of the blue curve starts to exceed the orange curve, showing further performance improvement.

(a) YOLOv8n (b) YOLOv8n+SPD-Conv (c) YOLOv8n+SPD-Conv+CoTAttention

Figure 6 Comparison of visualization results of ablation experiments

In order to analyze the effect of SPD-Conv and CoTAttention on the detection results, the effect of vehicles detected by the model is tested by ablation experiments in Figure 6. (a) demonstrates the detection results of the original YOLOv8n, which shows high detection accuracy for vehicles but poor performance for long-range vehicles. In (b), some of the detection results with the introduction of the SPD-Conv module are shown in Fig. (b) compared to Fig. (a), which successfully detects a vehicle at a long distance in the same image with a probability of being considered as a vehicle of 0.4. Further, in (c), the detection results with the introduction of CoTAttention are shown. Compared to Fig. (b), the probability of detecting a vehicle at a long distance in Fig. (c) improves to 0.7, which is an improvement of 0.3 in comparison to Fig. (b).Meanwhile, the model maintains a high detection accuracy. This indicates that the introduction of CoTAttention further optimizes the detection performance, especially in the detection of long-distance targets, which achieves significant improvement.

6.5 Experimental results of different models

In order to make a side-by-side comparison, the three networks YOLOv5n, YOLOv6n, and YOLOv8n are selected to be trained and tested on the same dataset respectively in this paper, and the experimental results are shown in Table 2.

Table 2 Comparison of results for different detection networks

Model	F1/%	P/%	R/%	mAP50/%	mAP50-95/%	Parameters/M	GFLOPs	FPS
YOLOv5n	73.84	78.0	70.1	78.4	55.8	2.51	7.2	**400**
YOLOv6n	73.95	82.2	67.2	78.1	56.4	4.23	11.9	303
YOLOv8n	75.23	80.5	**70.6**	78.6	56.8	3.01	8.2	303
YOLOv8n_S_C	**75.59**	**82.7**	69.6	**80.4**	**58.9**	**4.17**	**13.2**	322

As can be observed in Table 2, the improved YOLOv8n_S_C model performed the best in terms of accuracy at 82.7%. The model achieved the highest score of 80.4% for mAP50 and also the highest level of 58.9% for mAP50-95. This indicates that the YOLOv8n_S_C model has excellent comprehensive performance in the target detection task. Its F1 score is 75.59%, which is the highest level among the models.

7. Traffic statistics

7.1 Traffic flow system interface

The user interface of the monitoring video traffic system was developed using the Python language in conjunction with the PySide6 module. At the top of the interface is the title of the system, and directly below the title are four display boxes showing the total number of categories, the total number of targets, the average frame rate, and the usage model. Between the title and the display boxes, the current traffic flow is displayed. Below the hidden buttons on the left side of the system interface are some function buttons, including Local File, Call Camera, Call RTSP Monitor, Traffic Line Chart, Single Target Tracking, and Enable Web Side.

Figure 7 Monitoring video traffic flow system

There are some setting options under the right side settings of the system interface, including model, interaction ratio, confidence level, delay time, and alarm threshold. The specific settings are as follows: the model is improved YOLOv8n_S_C

in this paper, the interaction ratio is set to 0.36, the confidence level is set to 0.51, the delay time is set to 0ms, and the alarm threshold is set to 7. The whole interface is designed so that the user can monitor and manage the traffic flow system easily. Figure 7 above shows the scenario where the model is detecting the monitoring video traffic flow.

7.2 Multiple target tracking

Multi-target tracking has a wide range of applications in the fields of video surveillance, autonomous driving, and UAV tracking[31]. Its goal is to track multiple targets in a video sequence, usually by finding the position of the target in each frame. In surveillance video traffic counting, multi-target tracking serves to track the movement trajectory of each vehicle so that the behavior of the vehicle can be analyzed and counted. Multi-target tracking is shown in Figure 8.

Figure 8 Single target tracking

In a scenario based on improved YOLOv8 and Bot SORT for traffic counting, target detection is performed using improved YOLOv8n_S_C, i.e., vehicles in the image are identified in each frame. The detected vehicle information is passed to Bot SORT for single-target tracking.Bot SORT assigns a unique identifier to each vehicle and tracks their motion in the video sequence. The motion of each vehicle is recorded, which can be achieved by correlating the target position in each frame with the target

position of the same identifier in the previous frame. By tracking individual vehicles, the system can more accurately understand the dynamics of the entire traffic flow, providing useful information for traffic management, security monitoring, etc.

7.3 Line graph of traffic flow

A real-time traffic flow line chart is a chart that shows the changes in traffic flow through time. In the system, generating a line graph of traffic flow helps to visualize the traffic situation, analyze the traffic flow at the current moment, and take appropriate measures when needed. As shown in Figure. 9, the horizontal axis represents the time and the vertical axis represents the number of vehicles. Through the change of traffic flow in the line graph, it can be observed that the current traffic flow in the surveillance video is mainly concentrated between 5 and 6, showing a relatively moderate state.

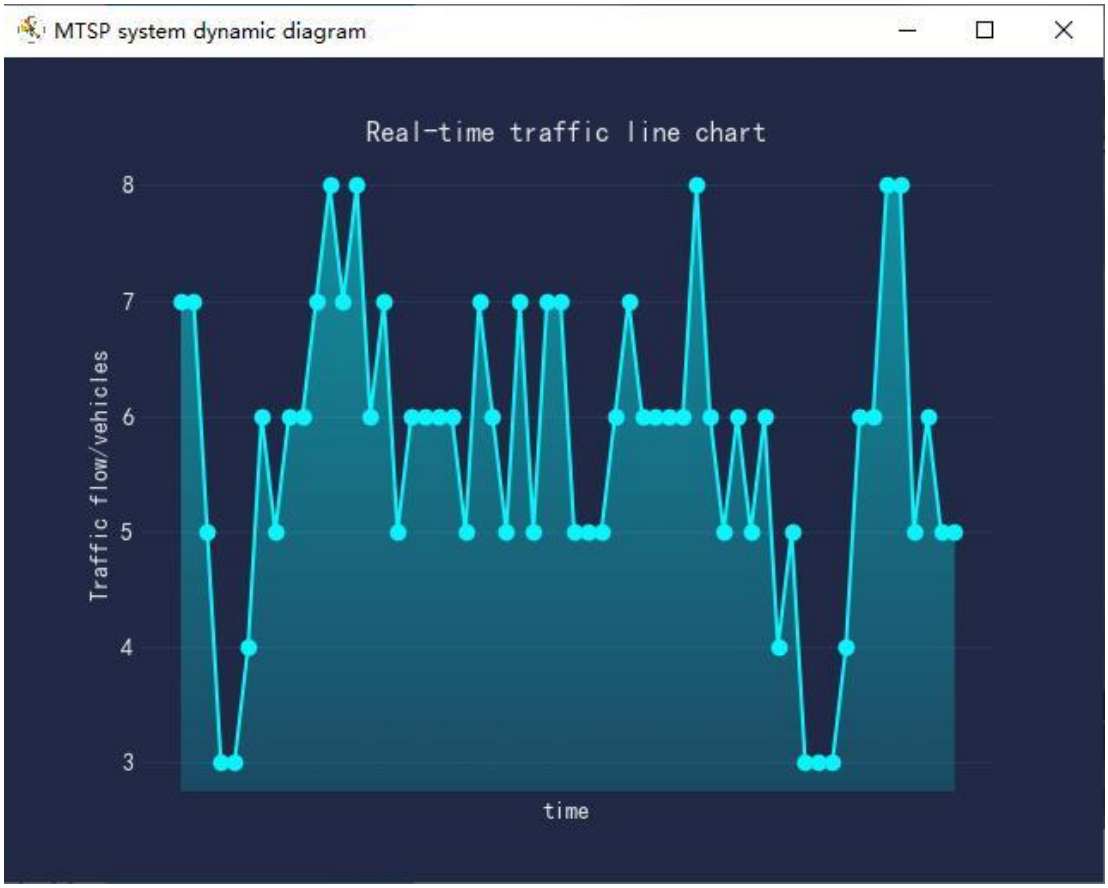

Figure 9 Line graph of traffic flow

Throughout the monitoring time period, there were several moments when the traffic volume exceeded the alarm threshold of 7 vehicles. Specifically, there were 5 moments when the number of vehicles reached 8, the current traffic volume subtitle would turn red and the system would generate an alarm alert, indicating a high volume of traffic; at the same time, there were 5 moments when the number of vehicles was only 3, indicating a low volume of traffic. The change of the line over time reflects the fluctuation of the traffic flow. Real-time updating of the traffic flow line graph enables continuous analysis of changes in traffic flow, allowing decision makers to stay informed of the traffic situation and take immediate action when needed. This helps to optimize traffic management and improve the real-time responsiveness of the system.

8. Concluding remarks

In a system based on improved YOLOv8 with Bot SORT surveillance video traffic counting, this paper combines a target detection technique (YOLOv8n_S_C) and a single-target tracking algorithm (Bot SORT) to achieve accurate detection, tracking, and counting of vehicles in surveillance video. This system provides a powerful tool for traffic management and surveillance, capable of accurately capturing the location, movement trajectory, and changes in traffic flow of vehicles in real-time video streams. Through the simple operation of the system, it not only generates line graphs of traffic flow to visualize the trend of traffic flow over time, but also analyzes in-depth the impact of traffic on traffic flow. This provides traffic managers with more comprehensive information, enabling them to develop more targeted traffic optimization strategies to improve the overall efficiency of the road network. It brings innovation and convenience to urban traffic management and lays a solid foundation for building a smarter city.

9. References

[1] J. E. Park, W. Byun, Y. Kim, H. Ahn, D. K. Shin, Journal of Advanced Transportation, **2021**, 1.

[2] Z. Wang, J. Zhan, C. Duan, X. Guan, P. Lu, K. Yang. IEEE Transactions on Neural Networks and Learning Systems.**2022**.

[3] Y. Fang, C. Wang, W. Yao, X. Zhao, H. Zhao, H. Zha.. On-road vehicle tracking using part-based particle filter. IEEE transactions on intelligent transportation systems, **2019**, 20, 4538.

[4] J. Ju, J. **ng. Multimedia tools and applications, **2019** ,78, 29937.

[5] M. P. Dessauer, S. Dua. In Ground/air multi-sensor interoperability, integration, and networking for persistent ISR. **2010** 7694, 366.

[6] B. Hardjono, H. Tjahyadi, M. G. Rhizma, A. E. Widjaja, R. Kondorura, A. M. Halim. IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference. **2018.** 556.

[7] A. Anoop, G. Harikrishnan, K. Nair, B. Sangeetha, V. Praseedalekshmi. International Conference on Innovations in Science and Technology for Sustainable Development. **2022**. 258.

[8] M. Carranza-García, J. Torres-Mateo, P. Lara-Benítez, J. García-Gutiérrez. Remote Sensing, **2020,** 13, 89.

[9] H. Wang, Y. Yu, Y. Cai, X. Chen, L. Chen, Y. Li. IEEE Transactions on Intelligent Vehicles, **2020**, 6, 100.

[10] K. Lenc, A. Vedaldi, . ar**v preprint ar**v.**2015**. 1506 . 06981.

[11] R. Girshick. In Proceedings of the IEEE international conference on computer vision. **2015**, 1440.

[12] H. Jiang, E. Learned-Miller. IEEE international conference on automatic face gesture recognition. **2017**, 650.

[13] K. He, G. Gkioxari, P. Dollár, R. Girshick. In Proceedings of the IEEE international conference on computer vision. **2017**, 2961.

[14] J. Jeong, H. Park, N. Kwak. ar**v preprint ar**v. **2017**, 1705, 09587.

[15] H. Zhang, H. Chang, B. Ma, S. Shan, X. Chen. ar**v preprint ar**v. **2019**, 1907, 06881.

[16] M. Hussain. Machines. **2023**, 11, 677.

[17] P. Kranz, U. Ali, A. Mueller, M. Hornauer, M. Loeser, F. Sukkar, T. Kaupp. **2021.**

[18] X. Hou, Y. Wang, L. P. Chau. IEEE International Conference on Advanced Video and Signal Based Surveillance. **2019**, 1.

[19] E. Hamuda, B. Mc Ginley, M. Glavin, E. Jones. Computers and electronics in agriculture, **2018**, 148, 37.

[20] C. K. Chui, G. Chen. Berlin, Germany: Springer International Publishing. **2017**, 19.

[21] S. Yan, Y. Fu, W. Zhang, W. Yang, R. Yu, F Zhang. International Conference on Electronic Engineering and Informatics. **2023**. 506.

[22] Z. Tan, B. Chen, L. Sun, H. Xu, K. Zhang, F Chen. Information Technology and Control. **2023**, 52, 878.

[23] M. T. Ibrahim, R. Hafiz, M. M. Khan, Y. Cho. Multimedia Systems,**2016**, 22, 379-392.

[24] Z. Zhang, X. Lu, G. Cao, Y. Yang, L. Jiao, F Liu. In Proceedings of the IEEE/CVF international conference on computer vision.**2021**, 2799.

[25] G. Yu, X. Zhou. Mathematics. **2023**,11, 2377.

[26] Y. Fan, G. Tohti, M. Geni, G. Zhang, J. Yang. A marigold corolla detection model based on the improved YOLOv7 lightweight. **2023**

[27] G Wang, Y. Chen, P. An, H. Hong, J. Hu, T. Huang. Sensors, **2023**, 23, 16, 7190.

[28] Z. Yang, Q. Wu, F. Zhang, X. Zhang, X. Chen, Y. Gao. Symmetry,**2023**, 15, 1037.

[29] Q. Geng, H. Liu, T. Gao, R. Liu, C. Chen, Q. Zhu, M. Shu. In Healthcare.**2023**.

[30] J. Lee, K. C. Lee, S. Jeong, Y. J. Lee, S. H. Sim. Mechanical Systems and Signal Processing. **2020**, 140, 106651.

[31] L. Wen, D. Du, Z. Cai, Z. Lei, M. C. Chang, H. Qi, S. Lyu. UA-DETRAC: Computer Vision and Image Understanding, **2020**, 193, 102907.

[32] P. Dendorfer, A. Osep, A. Milan, K. Schindler, D. Cremers, I. Reid, L. Leal-Taixé. Motchallenge: International Journal of Computer Vision. **2021**, 129, 845.