sensors

MDPI

*Article*

# Real-Time and Efficient Multi-Scale Traffic Sign Detection Method for Driverless Cars

**Xuan Wang [1], Jian Guo [1], Jinglei Yi [1], Yongchao Song [1], Jindong Xu [1], Weiqing Yan [1] and Xin Fu [2,3,*]**

[1] School of Computer and Control Engineering, Yantai University, Yantai 264005, China
[2] College of Transportation Engineering, Chang'an University, Xi'an 710064, China
[3] Engineering Research Center of Highway Infrastructure Digitalization, Ministry of Education, Xi'an 710064, China
* Correspondence: fuxin@chd.edu.cn

**Abstract:** Traffic signs detection and recognition is an essential and challenging task for driverless cars. However, the detection of traffic signs in most scenarios belongs to small target detection, and most existing object detection methods show poor performance in these cases, which increases the difficulty of detection. To further improve the accuracy of small object detection for traffic signs, this paper proposed an optimization strategy based on the YOLOv4 network. Firstly, an improved triplet attention mechanism was added to the backbone network. It was combined with optimized weights to make the network focus more on the acquisition of channel and spatial features. Secondly, a bidirectional feature pyramid network (BiFPN) was used in the neck network to enhance feature fusion, which can effectively improve the feature perception field of small objects. The improved model and some state-of-the-art (SOTA) methods were compared on the joint dataset TT100K-COCO. Experimental results show that the enhanced network can achieve 60.4% mAP(Mean Average Precision), surpassing the YOLOv4 by 8% with the same input size. With a larger input size, it can achieve a best performance capability of 66.4% mAP. This work provides a reference for research on obtaining higher accuracy for traffic sign detection in autonomous driving.

**Keywords:** traffic sign detection; attention mechanism; feature pyramid network



**Citation:** Wang, X.; Guo, J.; Yi, J.; Song, Y.; Xu, J.; Yan, W.; Fu, X. Real-Time and Efficient Multi-Scale Traffic Sign Detection Method for Driverless Cars. *Sensors* **2022**, *22*, 6930. https://doi.org/10.3390/s22186930

Academic Editor: Gwanggil Jeon

Received: 29 August 2022
Accepted: 10 September 2022
Published: 13 September 2022



## 1. Introduction

With the continuous development of deep learning, computer vision has achieved remarkable achievement and can be used in real-life scenarios. Object detection is one of the current subtasks of computer vision. It is widely used in the field of intelligent transportation, such as motor/non-motor vehicle recognition, pedestrian detection, autonomous driving vehicles, etc. This paper focuses on the study of traffic sign detection, which is a significant part of high-precision map construction for driverless cars. Actual traffic scenes are very complex: traffic signs are densely distributed, and weather factors such as fog, rain, and snow can also affect the accuracy of traffic signs detection. In this case, it is undoubtedly fatal for pedestrians and drivers. Therefore, it is particularly vital to improve the performance of traffic sign detection in various complex environments.

In recent years, computer vision has seen rapid development thanks to the contributions of researchers to convolutional neural networks. Earlier neural networks could only handle classification tasks in small, low-resolution datasets, such as CIFAR [1]. Then, Alex et al. [2] proposed AlexNet, which consists of convolutional and fully connected layers, to achieve the classification task in large datasets such as ImageNet [3]. After that, researchers proposed networks with more layers, such as VGG [4] and GoogLeNet [5], which improved the accuracy of networks to some extent. However, the gradient will drop or disappear when reaching a certain depth. He et al. [6] proposed ResNet, which uses cross-layer connections to fuse the input with the output of the residual blocks, ensuring that deeper network layers can obtain no fewer features than shallower network layers,

effectively alleviating the phenomenon of insignificance or even disappearance of deeper features. Most of the previous networks improve the network performance by adjusting the depth, width, and input image resolution of the network model. Tan et al. [7] proposed EfficientNet based on the search for the best scaling factor to unify the three factors, which first established an optimal benchmark and then adjusted the benchmark network based on different scaling factors.

With strong detection capabilities and real-time performance, YOLOv4 has been applied to many scenarios, but it often fails to obtain satisfactory results in traffic sign datasets containing many small objects. Our paper improves the model for the following two problems. Due to the nature of convolutional networks, global features are lost in the feature extraction process, and feature links between dimensions are ignored. Additionally, it is challenging to capture localization information and feature information of small objects because there is not enough feature fusion across scales.

The contributions of this paper are as follows:

- An improved Triplet attention mechanism module is employed in the residual block of the backbone network to determine the importance of spatial and channel dimensions. This module improves the detection accuracy of the neural network, without a significant increase in computational effort.
- We improved the fusion mode of feature pyramids in the neck network by replacing PANet with BiFPN based on YOLOv4 to obtain richer localization and feature information.
- Compared to the original YOLOv4 model, the mean average precision (mAP) of this paper was improved by 8% for traffic sign detection on the TT100K-COCO dataset when the same input was used.

The rest of this paper is organized as follows. Section 2 introduces related works about object detection methods, attention mechanisms, and small object detection. Section 3 describes the proposed method in detail. The experimental results are presented in Section 4. Finally, Section 5 offers the conclusion.

## 2. Related Work

### 2.1. Object Detection Methods

Object detection, as one of the essential sub-tasks of computer vision, requires the correct identification of predefined classes in an image and the precise localization of their coordinates. Object detection is divided into one-stage [8–12] and two-stage [13–16] detection methods. The difference between the two is the presence or absence of the interest extraction step. The former can obtain faster inference, and the latter's inference is based on the region of interest to get higher accuracy. The most typical one-stage detection methods are anchor-based YOLOv3 [9] and SSD [12], which can be predicted directly by the feature map. The principle is to predefine multiple anchor boxes for each cell of the feature map to detect the object, and then calculate the Intersection over Union (IOU) between anchor boxes and ground-truth boxes to delete the invalid and low-scoring anchor boxes. RCNN [13] uses a candidate region selection algorithm to obtain the input image containing the object, which is then used as the input to the convolutional network. However, feature region extraction can also be performed on the feature map. FasterRCNN [15] improves the inference speed of the network by completing the extraction of interest regions during the training process according to the Region Proposal Network (RPN) network.

Anchor-free one-stage networks [17–19] have been developed in recent years. Tian et al. [17] analyzed the disadvantages of predefined anchor boxes: (1) different hyperparameters need to be trained to adjust the deflation ratio when processing different datasets, and (2) there is a greater computational effort, which increases the burden on the hardware. To solve the above problems, the FCOS network was proposed, aiming to replace the anchor box with the center of the object and introduce centeredness to suppress the bounding box far from the center of the object to achieve a detection method that is not weaker than the anchor-based performance. The analysis of Zhang et al. [18] found that the differences between anchor-based and anchor-free performance stemmed from the selection of positive

and negative samples, which were not significantly different under the same conditions, and proposed the Adaptive Training Sample Selection (ATSS) to classify positive and negative samples.

### *2.2. Attention Mechanism*

The attention mechanism is often introduced as a module in backbone networks, such as [20–22]. SENet [20] is a channel attention mechanism that compensates for the shortcomings of convolutional networks focusing on local information by using the squeeze module to compress the feature map to obtain the global information, then using the excitation module to excite the channels containing important information, and finally applying the obtained excitation weights to the original feature map. CBAM [21] takes into account the spatial information while obtaining channel attention. First, it performs maximum global pooling and mean pooling on the input by channel to obtain spatial attention weights and apply them to the input feature layer. Then, it performs global pooling and mean pooling on the input by spatial to obtain spatial attention weights. The Triplet attention mechanism [23] is divided into three branches. The first two branches allow the channel dimension to respectively interact with the width and height dimensions, and the third branch allows the width and height dimensions to interact.

### *2.3. Small Object Detection*

The detection of traffic signs plays a crucial role in driverless technology, guiding cars to follow traffic guidelines and ensuring driving safety. However, traffic signs occupy only a tiny portion of the area relative to the large and complex background, which places higher demands on the recognition and detection capabilities of network models. Most of the initial work focused on multi-scale image pyramids, such as SNIP [24] and SNIPER [25]. SNIP is a scale normalization method, which aims to solve the problem of small objects that are difficult to identify in the training sample by setting a range in advance. When things satisfy the range condition in images with different scales of the image pyramid, they are put into training, and those objects that are too small scale are eliminated to improve the accuracy of the network model. Since SNIP trains on all pixel points of the whole image, it leads to slow speed and requires more computational resources. To address the above shortcomings, SNIPER crops the image according to the regions in which the objects are located, but it leads to insufficient learning of background information, which increases the error rate of prediction results. Based on this, the false detection rate can be reduced by cropping the negative sample region, which does not contain the positive sample region containing the objects.

Since these two methods are based on image pyramids, which consume a lot of computational resources, feature pyramid structure networks [26–29] subsequently emerged. These networks often act as neck components. They are combined with the backbone network for feature fusion at different scales to obtain better prediction results. FPN [26] incorporates the low semantic features of the high-resolution feature map, and also combines the high semantic features of the low-resolution feature map, combining the more accurate localization information of the former and the greater amount of feature information of the latter, which greatly improves the detection accuracy. PANet [27] is improved for FPN networks by designing a top-down module to obtain richer features. BiFPN [28], as an improvement of the PANet network, optimizes the route of feature fusion by eliminating some unnecessary connections. Meanwhile, it constructs cross-layer connections and repeats the module as the basic unit of feature fusion several times. TridentNet [29] uses three branches to assign different dilation to the convolutions that need to be replaced, obtaining different ranges of receptive fields. Similar to SNIP, it uses the Regions of Interest (ROI) to perform pretraining. Those that meet the scope of candidate anchor boxes are considered positive samples and put into training; negative samples are ignored without back-propagation of the gradient.

## 3. Method

The YOLOv4 network structure has obtained good results in the field of object detection, and our proposed network is further optimized based on YOLOv4, mainly by improving the backbone network and the neck network. The overall structure of the improved network is shown in Figure 1.

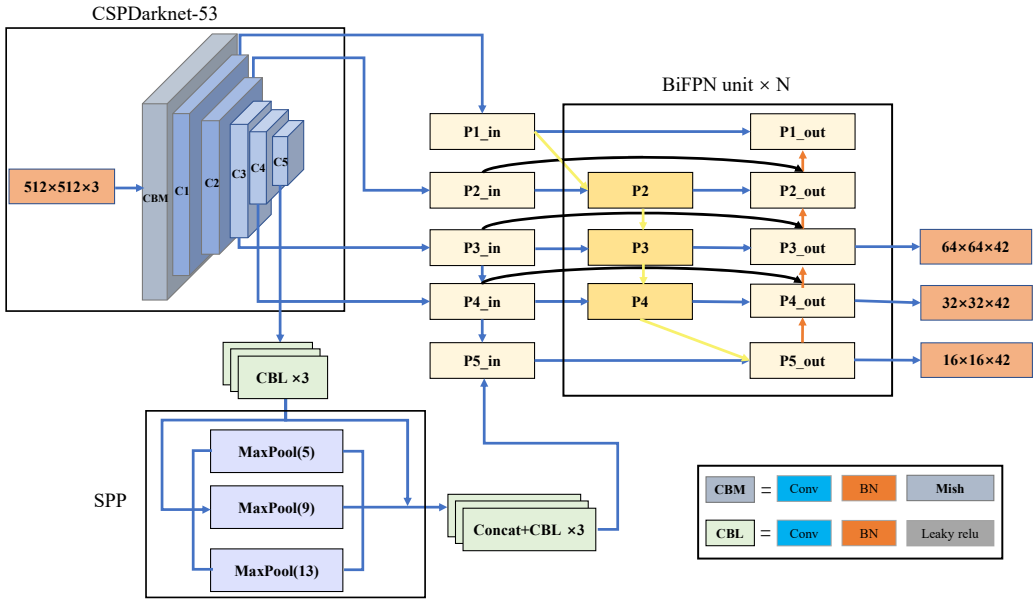


**Figure 1.** Architecture of improved YOLOv4.

### *3.1. Backbone*

CSPDarknet53 is chosen as the backbone network. The nature of convolution is to make the network pay more attention to the local information, but the global and spatial information are ignored. Therefore, we combined the Triplet attention mechanism in the backbone network to pay more attention to global information. The specific location is shown in Figure 2.

The Triplet attention mechanism is a spatial and channel attention mechanism which mainly consists of three branches and focuses more on cross-dimensional connections. Each branch input is subjected to a *Z-pool* operation, which stacks the results after maximum global pooling and average pooling, as shown in Figure 3.

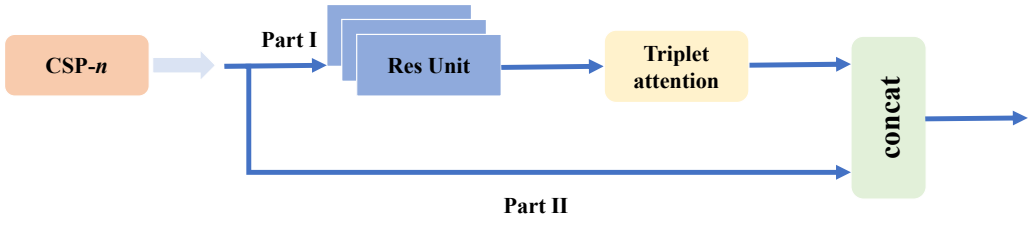


**Figure 2.** Triplet attention position.

In the first branch, the input dimension is first adjusted to advance the width dimension, and the feature layer after *Z-pool* is convolved using a $7 \times 7$ convolution kernel, and then the weights are obtained using the sigmoid activation function. Finally, the weights are applied to the original input feature layer so that the feature layer focuses more on the dependence between height and channel dimension. If given a tensor, $x \in R^{C \times H \times W}$, where $H$ and $W$ are the spatial dimensions and $C$ is the channel dimension. After that, the tensor $x$ dimensionally adjusted tensor is denoted as $x_1$, which has the shape $(W \times C \times H)$. After the *Z-pool* operation on $x_1$, the shape is $(2 \times C \times H)$, and after

performing the $7 \times 7$ convolution, the tensor of shape $(1 \times C \times H)$ is obtained denoted as $\hat{x_1}$. In the next step, batch normalization is performed, and then the sigmoid activation function is used to obtain weights, which are then applied to $x_1$. Finally, the dimension is recovered as $(C \times H \times W)$.

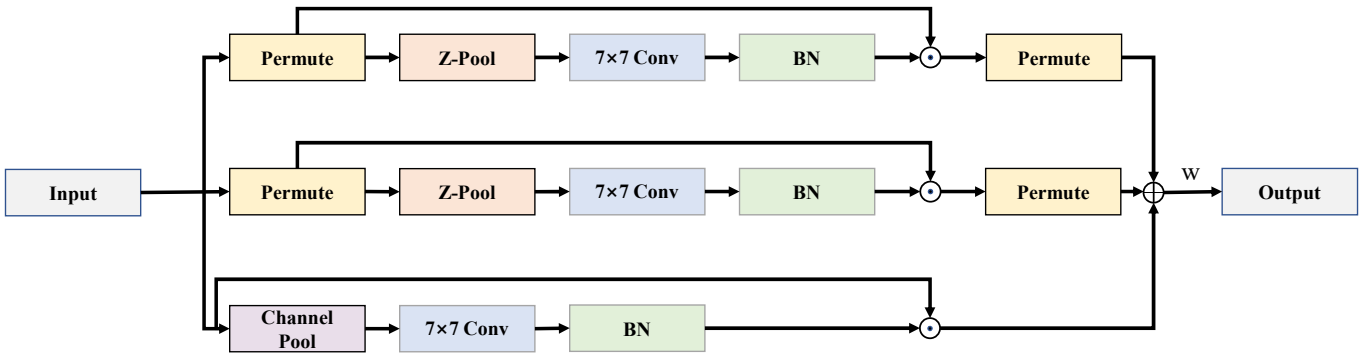


**Figure 3.** Architecture of improved triplet attention.

The second branch is similar to the first branch. The difference is that the height dimension is advanced and the shape is $(H \times C \times W)$ noted as $x_2$. Finally, the weights of the channel and width are obtained, and applied to $x_2$. The dimension is recovered to $(C \times H \times W)$.

In the last branch, the dimension is not adjusted, and is kept as it is, i.e., $x \in R^{C \times H \times W}$. After the *Z-pool* operation, we use convolution, batch normalization, and finally obtain the weights using the sigmoid activation function to learn the attention weights of height and width for application on $x$. The three branches are merged after completing their respective tasks, as shown in Equation (1):

$$O = \omega(x_1\sigma(\omega_1(\hat{x_1})) + x_2\sigma(\omega_2(\hat{x_2})) + x\sigma(\omega_3(\hat{x}))) \quad (1)$$

where $x_1$, $x_2$, $x$ represent the inputs of three different branches, respectively, and these tensors are represented as $\hat{x_1}$, $\hat{x_2}$, $\hat{x}$ by a $7 \times 7$ convolution after *Z-pooling*. Meanwhile, $\sigma$ represents sigmoid activate function and $\omega$ represents a learnable weight to optimize the fusion of the three branches.

### *3.2. Neck*

For the model neck, we use BiFPN to replace PANet. BiFPN, as an improvement of PANet, can fuse more features across scales, as shown in Figure 4. First, the first change is that the input of the feature-enhanced network is changed from three to five, which means that the information of five different scale feature layers can be acquired. To facilitate the fusion, the number of channels of all inputs is uniformly adjusted to 256 before entering the BiFPN. Second, the connection routes of PANet feature layers are optimized. The $p_1$ feature layer should have existed between $p_1^{in}$ and $p_1^{out}$. $p_1$ has only one input compared to $p_2$, $p_3$, and $p_4$ which have two inputs and contain less feature information, so this is omitted to save computational resources. The $p_5$ feature layer consists of two inputs, $p_4$ and $p_5^{in}$, but $p_5^{out}$ is also composed of the same input, resulting in functional duplication, so $p_5$ is also omitted. In addition, three cross-layer connections are also established with the purpose of preventing the deeper feature layers from being feature agnostic, so that the deeper features are not weaker than the shallow ones. Most of the previous neck networks used to fuse information of different dimensions treat the contribution of each feature layer as the same, but each feature layer does not fill the same role. To make each component occupy the appropriate weight, the weights are therefore optimized, so that critical information occupies a greater weight and feature layers with less contribution occupy a lower weight.

This module applies weights to each input and makes feature fusion more efficient via weighting. The formula is as follows:

$$Y = \sum_k \frac{w_k}{\varepsilon + \sum_i w_i} \cdot X_k \tag{2}$$

where $X_k$ represents input feature k, and $w_k$ represents a learnable weight for k. $w_i$ denotes all the weights of input tensors and $Y$ denotes the output after feature fusion. A small value of $\varepsilon$ is introduced to maintain stability, generally set to 0.0001.

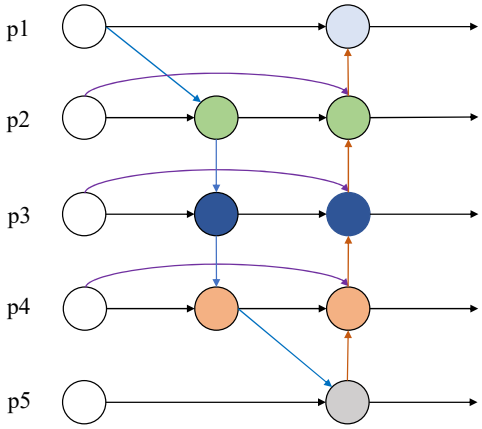


**Figure 4.** BiFPN Network.

The idea of this scheme is derived from soft-max, where the probability between 0 and 1 is obtained by dividing the total weight by the weight of each feature layer to limit the range of the weight change, which makes the gradient change more stable during the training process. Still, the exponential operation increases the amount of operation and slows down the GPU operation. To solve this problem, the exponential operation is therefore abandoned, and the total weights are divided by the weights directly, which can still obtain the probabilities in the 0-1 interval and improve the computational efficiency. It is worth noting that the weights are not constantly greater than 0, so the weights here must be limited to positive numbers.

Taking $p_4$ and $p_4^{out}$ as examples, we can see that the $p_4$ feature layer has two inputs, $p_4^{in}$ and $(p_3)'$, after $p_3$ downsampling, while $p_4^{out}$ consists of three inputs: $(p_5)'$ upsampled from $p_5^{out}$, $p_4$, and $p_4^{in}$. The formula is as follows:

$$p_4 = \psi\left(\frac{w'_{p_3}(p_3)' + w_{p_4^{in}} p_4^{in}}{\varepsilon + w'_{p_3} + w_{p_4^{in}}}\right) \tag{3}$$

$$p_4^{out} = \psi\left(\frac{w'_{p_4} p_4 + w'_{p_4^{in}} p_4^{in} + w'_{(p_5)}(p_5)'}{\varepsilon + w'_{p_4} + w'_{p_4^{in}} + w'_{(p_5)}}\right) \tag{4}$$

where $\psi$ donates depthwise separable convolution operation. In the top-down path, $p_4$ is the intermediate feature at layer 4, while in the bottom-up path, $p_4^{out}$ is the output feature at layer 4. $w'_{p_3}$, $w_{p_4^{in}}$, $w'_{p_4^{in}}$, $w'_{(p_5)}$ represent respective weights.

## 4. Experiment Results

### *4.1. Dataset and Experimental Details*

The experiment is run on a Linux server with Ubuntu 18.04, Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50 GHz, Tesla V100 GPU with 32 GB memory, based on the Pytorch deep learning framework, Pytorch version 1.11.0, CUDA version 11.6.

TT100K [30] consists of 100,000 images with 2048 × 2048 resolution, covering most traffic signs, which can provide support for both classification and recognition tasks of traffic signs in real-world scenes. The TT100K dataset contains objects of different sizes, divided into three sizes: small, medium, and large. The resolution of small objects is less than 32 × 32, the resolution of medium objects is between 32 × 32 and 96 × 96, and the rest are large objects. The 45 classes and the traffic lights and stop signs in the COCO [31] dataset are used as our dataset, called TT100K-COCO. TT100K-COCO contains a total of 9865 images, of which 8878 are used for the training process. The training set includes 7990 images, the validation set contains 888 images, and 987 images are used as tests. Due to the random division of images, the number of instances of classes varies greatly, which tests the detection ability of the model even more. The composition of the data set was shown in Table 1 and Figure 5.

**Table 1.** The composition of TT100K-COCO Dataset.

| Classname | Instances | Classname | Instances | Classname | Instances | Classname | Instances |
|---|---|---|---|---|---|---|---|
| traffic light | 1226 | pl60 | 800 | p11 | 1466 | pm55 | 136 |
| stop sign | 338 | w57 | 385 | pl5 | 472 | il80 | 293 |
| pl40 | 1319 | pl120 | 295 | wo | 111 | w32 | 104 |
| p26 | 756 | pl100 | 665 | io | 846 | il100 | 131 |
| p27 | 131 | il60 | 478 | po | 1124 | p19 | 120 |
| pne | 2039 | p10 | 331 | i4 | 707 | pr40 | 199 |
| i5 | 1549 | w55 | 169 | pl70 | 147 | ph4 | 120 |
| p5 | 376 | ph4.5 | 182 | pl80 | 852 | p23 | 266 |
| ip | 318 | w13 | 121 | pl50 | 1000 | w59 | 180 |
| pl30 | 578 | pl20 | 154 | i2 | 439 | pm20 | 156 |
| pn | 2851 | p12 | 172 | pg | 147 | ph5 | 113 |
| p3 | 139 | p6 | 108 | pm30 | 107 | - | - |

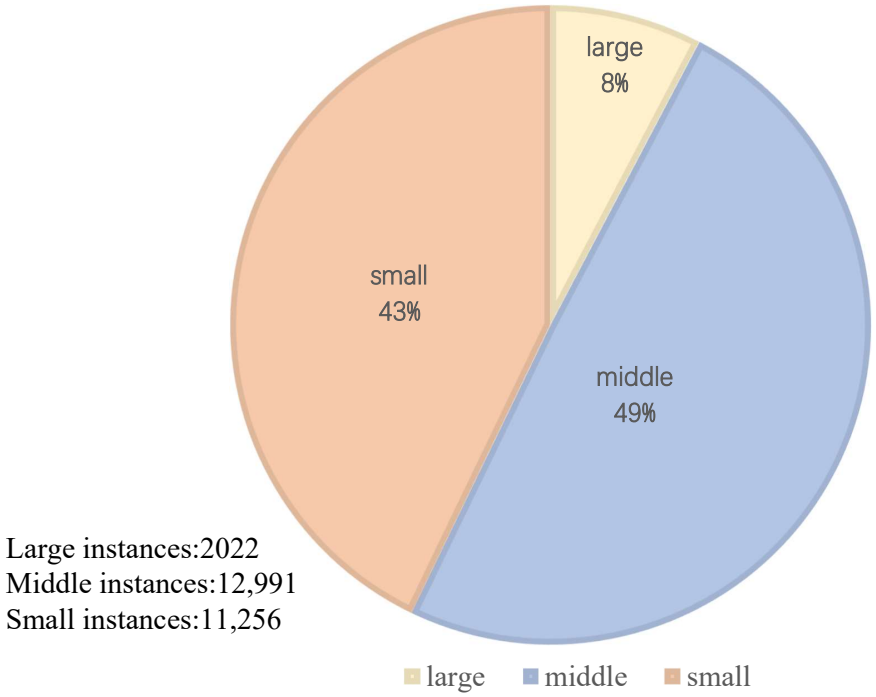


**Figure 5.** Instance size distribution.

### *4.2. Analysis of Results*

To demonstrate the performance of the convolutional neural network proposed in this paper, some evaluation metrics are used to compare the SOTA models. Precision, which is the percentage of the number of correctly classified samples in the total sample, is

shown as (5). Recall evaluates the ability of the model to find all positive samples, shown as (6). Therefore, we use mAP, which can reflect both precision and recall performance comprehensively, as an evaluation metric, shown in Equation (7). mAP@0.5 is the mAP value when the Intersection over Union (IoU) threshold is 0.5.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

where TP (True Positive) means that the model predicts positive classes as positive, FP (False Positive) means that the model predicts negative classes as positive, and FN (False Negative) denotes that the model predicts positive classes as negative.

$$\text{AP} = \int_0^1 p(r)dr \tag{7}$$

$$\text{mAP} = \frac{1}{\text{N}} \sum_{\text{i}}^{\text{N}} \text{AP}_\text{i} \tag{8}$$

where AP is average precision and $p$ and $r$ are the values on the precision–recall curve, respectively. mAP means average AP for all classes.

As shown in Table 2, the mean average precision value of the original YOLOv4 model (mAP@0.5) is 52.6% on the test set, and the mAP of our improved model reaches 60.6%, which is an 8% improvement and achieves good results. In order to more accurately evaluate the detection ability of the model for different sizes of objects, the same evaluation metrics of the coco benchmark are used to calculate the mAP values for different sizes. The improved model achieves 14.8%, 39.4%, and 44.1% in small, middle, and large sizes, respectively, which is 2.3% higher, 2.2% higher, and 3.1% lower than the original model. Based on these results, we can conclude that our model has better accuracy than the original model in realistic scene detection filled with small and middle-sized targets, although it degrades in large targets. Compared to YOLOv3, the improvement is 2.4%, 10.3%, and 15.2%, respectively. Our model is further enhanced in small object detection when the input is 512 × 512, with a 3.3% improvement compared to 416 × 416.

**Table 2.** mAP (%) obtained by several state-of-the-art methods.

| Methods | Size | mAP@0.5 | $\text{AP}_S$ | $\text{AP}_M$ | $\text{AP}_L$ |
|---|---|---|---|---|---|
| SSD [12] | 300 × 300 | 34.2 | 2.9 | 19.1 | 58.1 |
| YOLOv3 [9] | 416 × 416 | 50.3 | 12.4 | 29.1 | 28.9 |
| YOLOv4 [10] | 416 × 416 | 52.6 | 12.5 | 37.2 | 48.3 |
| Ours | 416 × 416 | 60.6 | 14.8 | 39.4 | 44.1 |
| Ours | 512 × 512 | 66.4 | 18.1 | 42.6 | 45.2 |

Table 3 demonstrates the effect of the BiFPN number on FLOPs (Floating Point Operations), Parameters, and mAP. As the number of BiFPN module increases, the mAP improves on the TT100K-COCO dataset, even though the number of parameters and FLOPs also increases. That is, more computing resources are exchanged for the improvement in accuracy. Due to this, we can choose the number of BiFPNs to meet our needs according to the actual hardware situation.

**Table 3.** Results on different numbers of BiFPN modules.

| Neck | FLOPs(G) | Parameters(M) | mAP@0.5 |
|---|---|---|---|
| BiFPN × 1 | 58.56 | 56.676 | 55.88 |
| BiFPN × 2 | 77.89 | 58.827 | 57.42 |
| BiFPN × 3 | 97.23 | 60.977 | 60.6 |

Figure 6 shows the Miss Rate (MR) of each model at 16 different traffic signs. MR is used to indicate the rate of missed detection in the test results, as shown in Equation (9):

$$\mathrm{MR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}} \tag{9}$$

According to Equation (9), MR is actually the proportion of samples that were not correctly detected to the total number of samples. From Figure 6, it can be found that in general, our methods have a low MR, although some categories are higher.

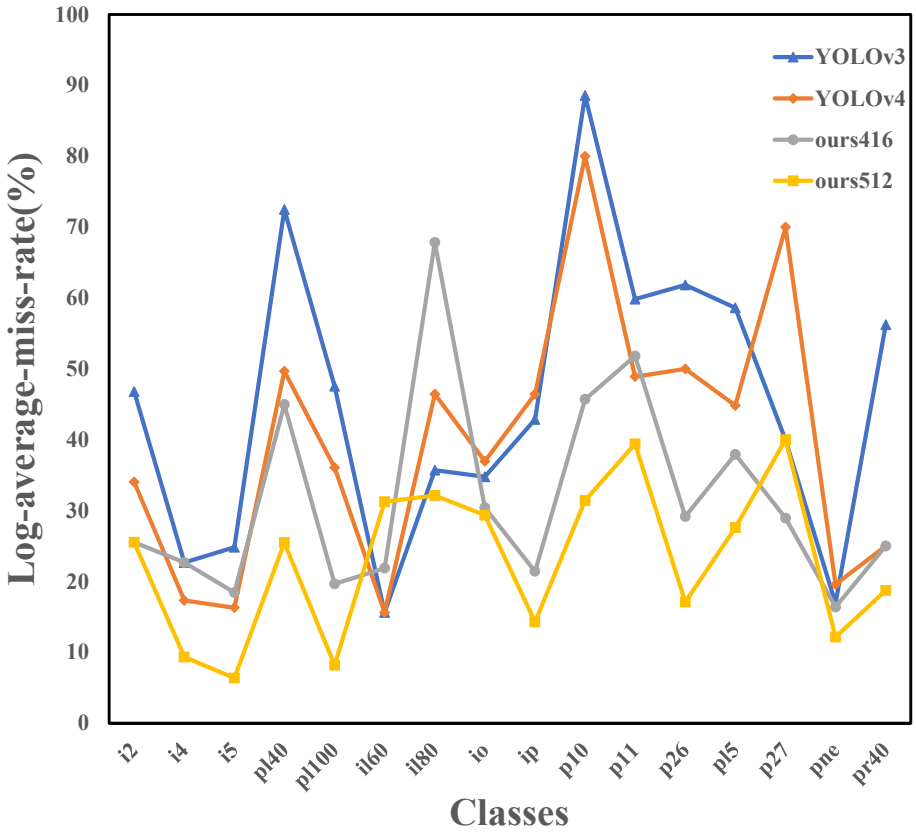


**Figure 6.** Comparison of miss rate.

Figure 7 shows the comparison of traffic sign detection results in different scenes. Figure 7a shows the original images, Figure 7b–d show the result images of YOLOv4, our method (416 × 416), and our method (512 × 512) detection, respectively. The detection of the first image shows that our model improves the detection of "ip" object by 5% and 13%, respectively, compared to the original YOLOv4. The detection of the second image is similar to the first one, with an improvement in accuracy. Then, we find that sample three has a smaller, darker, and more blurred object than the first two clearly visible samples, which directly leads to the missed detection of YOLOv4. At the same time, our method with input image 512 × 512 can detect the objects that are difficult to identify better than the model with input 416 × 416. In sample four, all three models detected the larger sign at the front, but the "pn" sign at the back was detected only by our model. Therefore, the method

proposed by this paper is more adaptable and has significant performance for small objects, and especially has significant performance for small objects.

(**a**) (**b**) (**c**) (**d**)

**Figure 7.** Comparison of traffic sign detection results.

## 5. Conclusions

In this paper, we proposed an improved version of the algorithm based on YOLOv4 that focuses more on small object detection to overcome the challenges faced by traffic sign detection tasks in realistic scenarios. The following two practical and feasible improvements are proposed: an improved Triplet attention mechanism module used to make the fusion of the three branches more reasonable, and the use of a more nodal and structurally complex BiFPN instead of PANet to enhance cross-scale feature fusion. While our proposed algorithm achieves good results in the detection of small objects, there are still areas for improvement. In the future, the feature fusion network can be further optimized and streamlined to reduce the number of parameters, or the attention mechanism module could be added to this network. In the future, the feature fusion network may be further optimized and streamlined to reduce the number of parameters, or the attention mechanism module may be considered for addition to this network to improve the performance.

**Author Contributions:** Conceptualization, X.W. and Y.S.; methodology, J.G.; software, J.G. and J.Y.; investigation, W.Y., J.X. and X.F.; formal analysis, X.W.; writing—original draft preparation, J.G., X.W. and J.Y.; writing—review and editing, X.W. and Y.S.; supervision, W.Y., J.X. and X.F.; funding acquisition, X.W. and J.X. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Natural Science Foundation of Shandong Province (ZR2020QF108, ZR2020MF148), and supported by the Fundamental Research Funds for the Central Universities, CHD (300102342511) and the National Natural Science Foundation of China (62072391, 62066013, 62172351), and the Youth Innovation Science and Technology Support Program of Shandong Province under Grant 2021KJ080.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The TT100K-COCO dataset and codes are available on Github at https://github.com/waterdou/improved-yolov4.git, accessed on 8 August 2022.

**Acknowledgments:** We would like to thank the anonymous reviewers for their supportive comments that improved our manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Krizhevsky, A.; Hinton, G. Convolutional deep belief networks on cifar-10. *Unpubl. Manuscr.* **2010**, *40*, 1–9.
2. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. Commun. ACM **2017**, *60*, 84–90 . [CrossRef]
3. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
4. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
5. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
6. Ren, S.; Sun, J.; He, K.; Zhang, X. Deep residual learning for image recognition. In Proceedings of the CVPR, Las Vegas, NV, USA, 27–30 June 2016; Volume 2, p. 4.
7. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
8. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
9. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
10. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
11. Long, X.; Deng, K.; Wang, G.; Zhang, Y.; Dang, Q.; Gao, Y.; Shen, H.; Ren, J.; Han, S.; Ding, E.; et al. PP-YOLO: An effective and efficient implementation of object detector. *arXiv* **2020**, arXiv:2007.12099.
12. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
13. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
14. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
15. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*. [CrossRef] [PubMed]
16. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
17. Tian, Z.; Shen, C.; Chen, H.; He, T. Fcos: Fully convolutional one-stage object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9627–9636.
18. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S.Z. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9759–9768.
19. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as points. *arXiv* **2019**, arXiv:1904.07850.

20. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
21. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
22. Lee, H.; Kim, H.E.; Nam, H. Srm: A style-based recalibration module for convolutional neural networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 1854–1862.
23. Misra, D.; Nalamada, T.; Arasanipalai, A.U.; Hou, Q. Rotate to attend: Convolutional triplet attention module. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2021; pp. 3139–3148.
24. Singh, B.; Davis, L.S. An analysis of scale invariance in object detection snip. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3578–3587.
25. Singh, B.; Najibi, M.; Davis, L.S. Sniper: Efficient multi-scale training. *Adv. Neural Inf. Process. Syst.* **2018**, *31*. [CrossRef]
26. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
27. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
28. Tan, M.; Pang, R.; Le, Q.V. Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10781–10790.
29. Li, Y.; Chen, Y.; Wang, N.; Zhang, Z. Scale-aware trident networks for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 6054–6063.
30. Zhu, Z.; Liang, D.; Zhang, S.; Huang, X.; Li, B.; Hu, S. Traffic-sign detection and classification in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2110–2118.
31. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.