International Journal of Wireless and Mobile Computing

ISSN online: 1741-1092 - ISSN print: 1741-1084
https://www.inderscience.com/ijwmc

Lightweight object detection algorithm for automatic driving scenarios

Zhou Tu, Ming Chen

DOI: 10.1504/IJWMC.2023.10061411

Article History:

Received:	23 December 2022
Last revised:	20 May 2023
Accepted:	27 June 2023
Published online:	07 February 2024

Lightweight object detection algorithm for automatic driving scenarios

Zhou Tu and Ming Chen*

College of Computer and Information Technology,
China Three Gorges University,
Hubei, Yichang, China
and
Hubei Key Laboratory of Intelligent Vision-Based
Monitoring for Hydroelectric Engineering,
Hubei, Yichang, China
Email: tz2062750487@163.com
Email: chenming131@163.com
*Corresponding author

Abstract: Aiming to solve the problems of large model size, unbalanced detection speed and detection accuracy of traditional target-detection algorithms in autonomous driving scenarios, a lightweight YOLO algorithm is proposed based on YOLOv4. First, the lightweight network Mobilenetv1 was used to replace the original YOLOv4 feature-extraction network and a Depth-Wise Cross-Stage Part module (DW-CSP) was proposed to improve the detection speed. Then, a new Lish activation function was designed and the *K*-means++ clustering algorithm was used to regenerate prior frames of different scales. Finally, the FocalLoss loss function was introduced instead of the cross-entropy loss function. Experiments show that compared with the YOLOv4 algorithm, the improved YOLO algorithm improves the detection accuracy by 1.72% and the detection speed by 53%, and the model size is reduced by about four times. The algorithm is more in line with the target-detection requirements of autonomous driving scenarios.

Keywords: automatic driving; object detection; lightweight network; YOLOv4; activation function.

Reference to this paper should be made as follows: Tu, Z. and Chen, M. (2024) 'Lightweight object detection algorithm for automatic driving scenarios', *Int. J. Wireless and Mobile Computing*, Vol. 26, No. 1, pp.74–82.

Biographical notes: Zhou Tu is a graduate student of the School of Computer and Information Technology of China Three Gorges University. His research interests include deep learning and target detection.

Ming Chen received his PhD degree in Information and Telecommunications Engineering of Telecommunication Department from Huazhong University of Science and Technology in 2010. His research interests include SOPC technology, wireless broadband communication technology, embedded AI and applications.

1 Introduction

Target detection is an important part of automatic driving. With the continuous research into target-detection algorithms, target-detection technology for automatic driving has developed rapidly (Wang et al., 2021; Tian et al., 2021; Xu et al., 2021). This is a research hotspot in target detection in automatic driving to improve the detection speed while ensuring the present levels of accuracy (Yuan et al., 2020). At present, mainstream target detection algorithms are divided into two types, namely two-stage target-detection algorithms and single-stage target-detection algorithms. The two-stage target-detection algorithms are mainly divided into two steps: extracting the region of interest generated by the network and CNN classification regression. The mainstream two-stage target-detection algorithms include R-CNN (Girshick et al., 2014), Fast-RCNN (Girshick, 2015) and Faster RCNN (Ren et al., 2015). Single-stage target detection algorithms usually obtain the classification box information by directly regressing and predicting the feature map. Such algorithms mainly include YOLO (Redmon et al., 2016), SSD (Liu et al., 2016) and YOLOv2 (Redmon and Farhadi, 2017). With the development of deep learning, target detection methods based on CNN are gradually being applied to target detection in automatic driving. Luo et al. (2021) proposed an image-adaptive algorithm to improve the Faster CNN algorithm and improve its detection capability for multi-scale vehicle targets. Yang and Tong (2022) proposed a YOLOv3 algorithm with a multi-scale attention module, which accomplished real-time detection of traffic signs. Ding and Zhao (2022) proposed a head-aware

pedestrian detection network, which effectively improved the detection ability of pedestrians blocking the road.

This paper has made many improvements based on the YOLOv4 (Bochkovskiy et al., 2020) algorithm. Firstly, for the application scenario of automatic driving target detection, the *K*-means++ (Xiong and Tang, 2018) clustering algorithm is used to cluster the data sets to generate an a priori frame of different sizes; In terms of network structure, the lightweight network Mobilenetv1 (Howard et al., 2017) was first used to replace the original backbone network CSPdarknet53 as a new feature extraction network, which improved the model operation speed; Secondly, the depth and width of the original YOLOv4 feature fusion network are redesigned, and a depth-wise cross-stage part module (DW-CSP) is proposed, which greatly reduces the number of model parameters. At the same time, a new Lish activation function is designed according to the characteristics of LeakyRelu (Maas et al., 2013) and Mish (2019) activation functions to effectively improve the saturation and generalisation of the model. Finally, the FocalLoss (Lin et al., 2017) loss function is introduced to replace CrossEntropy loss function (Boer et al., 2005) for calculating confidence, which alleviates the imbalance of positive and negative sample proportion during training and further improves the detection capability of the model for multi-scale targets.

2 Related work

2.1 YOLOv4 algorithm

The YOLOv4 algorithm first uses the *k*-means clustering algorithm to cluster the prior boxes of various target samples of the data set, and generates 9 prior boxes of different sizes. In terms of network structure, as shown in Figure 1, for the input image of 416×416 size, the YOLOv4 network first uses the Darknet backbone network combined with cross-stage local module to extract feature information in the feature extraction network, and after multiple downsampling, the feature map is divided into three different sizes: 13×13, 26×26, 52×52, and then the feature map of 13×13 is pooled for feature pyramid pooling (He et al., 2015) in order to expand the receptive field of the feature map and input it into the feature fusion network. In the feature fusion network, the YOLO algorithm fuses the features of the multi-scale feature map to enhance the feature expression ability of the feature map, firstly, the feature map of 13×13 is fused with the multi-scale feature map of 26×26, 52×52 through the path aggregation network PANet (Liu et al., 2018), and then the feature enhancement from top to bottom is carried out, and three feature maps of different sizes are output respectively for detecting targets of different sizes finally.

2.2 Mobilenetv1 algorithm

Mobilenetv1 is a lightweight convolutional neural network. The core uses depth-wise separable convolution instead of standard convolution, which reduces the number of parameters of the model. Assuming that the input feature map size is $W \times H \times Cin$, if *padding* is 0 and *stride* is 1, the calculation graph of standard convolution is shown in Figure 2. The size of the convolution kernel is $Kw \times Kh \times Cout$, and the theoretical calculation amount *Flops1* at this time is shown in Formula (1).

$$Flops1 = Kw \cdot Kh \cdot Cin \cdot Cout \cdot W \cdot H \tag{1}$$

The depth-wise separable convolution is shown in Figure 3. Assuming that the number of groups is the number of channels of the input feature map, first, each channel of the feature map is calculated separately through the grouped convolution, and *Cin* unrelated grouped feature maps are obtained. The calculation amount *Flops2* of the grouped convolution is shown in Formula (2).

$$Flops2 = Kw \cdot Kh \cdot Cin \cdot W \cdot H \tag{2}$$

Figure 1 YOLOv4 structure diagram

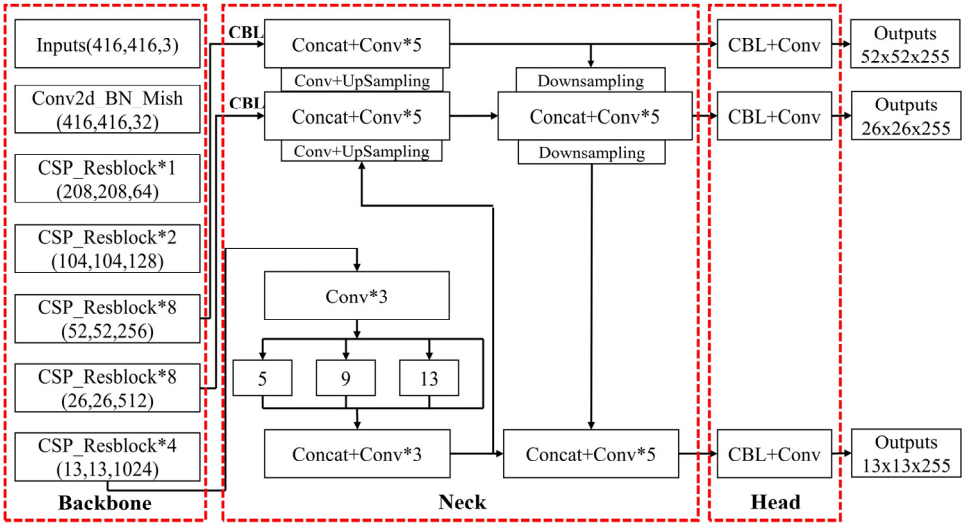


Figure 2 Standard convolutional structure

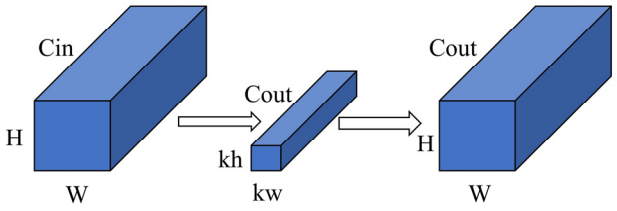
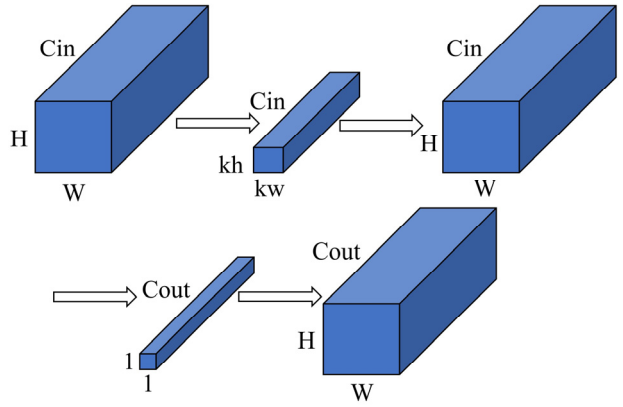


Figure 3 Depth-wise separable convolutional structure



Then, 1×1 point convolution is used to concatenate the information of these unrelated grouped feature maps. The calculation amount, *Flops3*, of this operation is shown in Formula (3).

$$Flops3 = 1 \cdot 1 \cdot Cin \cdot Cout \cdot W \cdot H \tag{3}$$

As shown in Formula (4), combining the computations of the grouped convolution and the point convolution is the computation of the depth-wise separable convolution *Flops4*.

$$\begin{aligned} Flops4 &= Flops2 + Flops3 \\ &= Kw \cdot Kh \cdot Cin \cdot W \cdot H + 1 \cdot 1 \cdot Cin \cdot Cout \cdot W \cdot H \\ &= (Kw \cdot Kh + Cout) * Cin * W * H \end{aligned} \tag{4}$$

Formula (5) can be derived by comparing the computational effort of standard convolution with depth-wise separable convolution.

$$\frac{Flops4}{Flops1} = \frac{1}{Cout} + \frac{1}{Kw \cdot Kh} \tag{5}$$

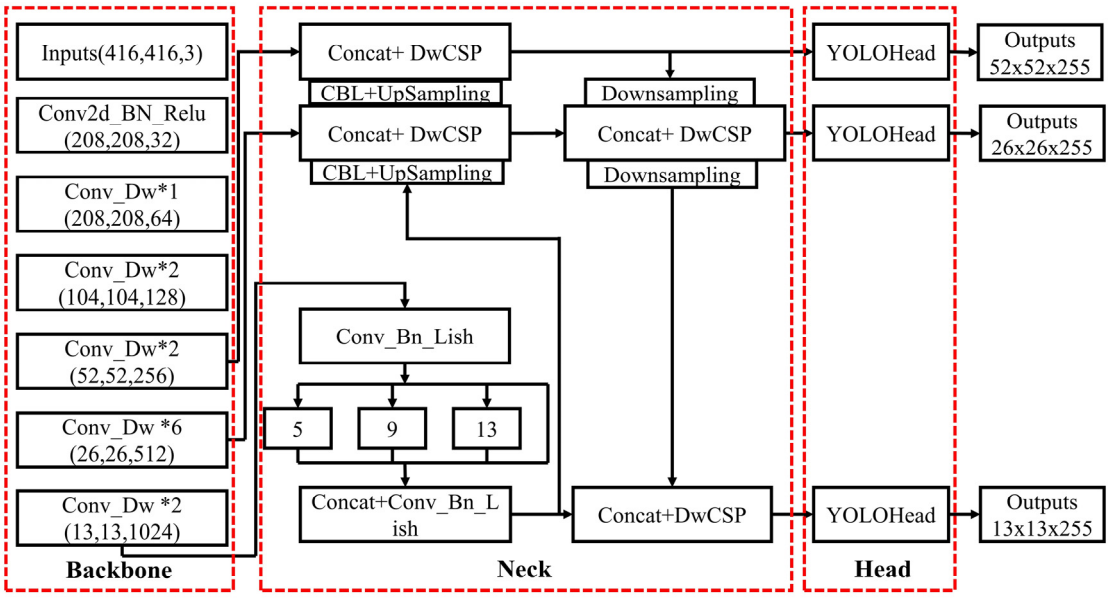
It can be seen that when the size of the convolution kernel is 3×3, if *Cout* >> 3, the calculation amount of the depth-wise separable convolution is much smaller than that of the standard convolution. Using the Mobilenetv1 feature extraction network to replace the original CSPDarknet53 not only ensures the feature extraction capability, but also can effectively improve the model's detection speed.

3 Network optimisation methods

3.1 Improved network structure

Compared with the original YOLOv4 network structure, the network in this paper improves the detection speed by using the Mobiletv1 lightweight network as a new feature-extraction network. In the feature fusion network, the feature fusion part of the original PANet structure and FPN structure is maintained, the DW-CSP module is proposed to replace the linear convolution module after each feature map stitching. At the same time, a new Lish activation function is designed to be applied to the feature fusion network. The improved YOLOv4 network structure is shown in Figure 4.

Figure 4 Improved YOLO algorithm network framework



3.2 DW-CSP structure

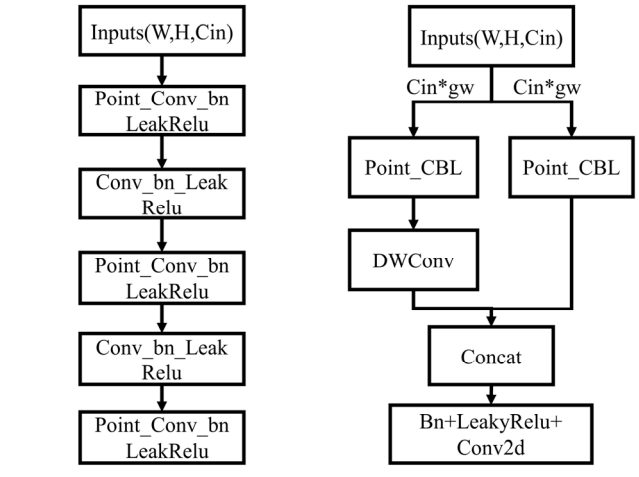
In the feature fusion network, the high-scale feature map generated by the PANet network is downsampled by YOLOv4 to generate a feature pyramid FPN (Lin et al., 2017) structure. And five consecutive convolution blocks are used to enhance the expressiveness of feature information. The expression calculated by the convolution module is as in formula (6):

$$y = x_k = F_k\left(x_{k-1}, F_{k-1}\left(x_{k-2}\right), ..., F_1\left(x_0\right), x_0\right) \tag{6}$$

where y represents the linear or non-linear output of the convolution module, and F_k represents a series of operations consisting of a set of convolutions and an activation function. Although the input of each convolutional module can receive the output from all previous layers, which can minimise the path of gradient propagation, it also causes the gradient data of the k-th layer to be passed to all previous convolutional layers, and updating the weights in this way leads to the continuous learning of redundant information.

This paper proposes the DW-CSP structure based on the Cross-Stage Partial Network (Wang et al., 2020) to replace the continuous convolution module in YOLOv4. By separating the gradient flow, the feature information is propagated in different paths, which reduces the model calculation amount and allows richer gradient-fusion information to be obtained. Figure 5 shows the structure comparison between the continuous convolution module and the DW-CSP module. The DW-CSP module uses a separate gradient flow strategy. First, the input feature map is divided into two parts, so that the feature information is propagated in the convolution layer and the transition layer, respectively. The parameter *gw* is used in the input layer to divide the input feature map's size, in which the transition layer uses 1×1 point convolution for feature information mapping and then performs connection calculation with the feature information from the convolutional layer. The convolutional layer uses a depth-wise separable convolution, which effectively reduces the amount of the 3×3 convolution calculation.

Figure 5 Comparison of the continuous convolution module and the DW-CSP module



(a) Continuous convolution module (b) DW-CSP module

The DW-CSP module doubles the gradient propagation path of the feature information through the segmentation and merging strategy. Its feedforward transfer calculation expression is shown in Formula (7), where * represents the convolution operation, and w_i and y_i represent the network weights and outputs of the i-th layer. In the input layer, the feature map is divided into two parts, namely x_0' and $x0''$; pass through the convolution layer and the transition layer, respectively, of the DW-CSP module, and the output y_c of the final convolution layer and the output y_t of the transition layer are spliced to perform the mapping calculation to generate the output y_u.

$$\begin{aligned} y_c &= w_c * x_0' \\ y_t &= w_t * x_0'' \\ y_U &= w_U * [y_c, y_t] \end{aligned} \tag{7}$$

For example, Formula (8) is the back-propagation operation of DW-CSP, where f_i represents the weight update function of the i-th layer, and g_i is the gradient of back-propagation to the i-th layer. It can be seen that the gradient information from the convolutional layer and that from the transition layer are integrated separately, and the two propagation paths do not contain redundant gradient information from the other side. At the same time, by truncating the gradient flow, the disadvantages of direct splicing with explicit feature map replication can be alleviated, and improve the reusability of feature information.

$$\begin{aligned} w_c' &= f_c\left(w_c, g_c\right) \\ w_t' &= f_t\left(w_t, g_t\right) \\ w_U' &= f_U\left(w_U, g_c, g_t\right) \end{aligned} \tag{8}$$

3.3 Lish activation function

The activation functions used in the YOLOv4 algorithm include the Leaky Relu function and the Mish function. The graph of the Leaky Relu function is shown in Figure 6(a), and the calculation expression is shown in Formula (9). In Formula (9), α is the slope control factor, which is usually set to 0.01.

$$F(x)_{leaky-relu} = \begin{cases} x, x > 0 \\ \alpha x, x <= 0 \end{cases} \tag{9}$$

The Leaky Relu function is an improvement of the Relu activation function. It can be seen from Figure 6(a) that it is not smooth for the input near the 0 interval, and the input in the negative interval is a linear function, resulting in limited overall non-linear expression ability. The graph of another Mish activation function is shown in Figure 6(b), and its calculation expression is shown in Formula (10). Compared with the Leaky Relu activation function, the non-monotonicity of Mish in the negative interval can make the model obtain better accuracy and generalisation, but when its x value is at a negative value far from 0, the activation output is also 0, resulting in the weight and gradient of the corresponding neuron tending to 0 during the backpropagation process, so there may be a 'Dead Neuron' phenomenon that occurs.

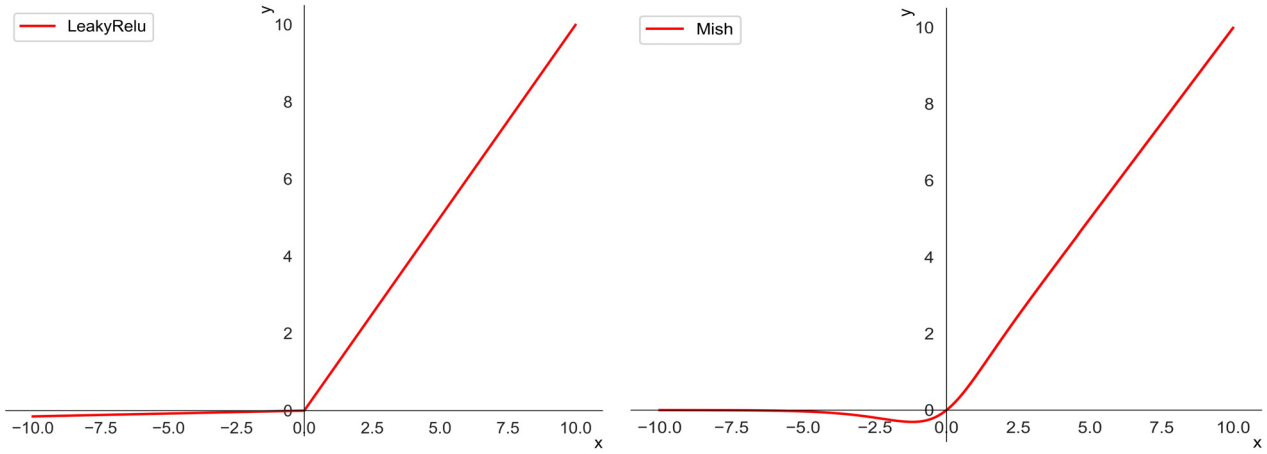
$$F(x)_{mish} = x * \tanh\left(\ln\left(1+e^{x}\right)\right) \tag{10}$$

This paper combines the characteristics of the two activation functions of the Leaky Relu function and the Mish function and proposes a new Lish activation function, which is applied to the feature fusion network. Its curve diagram is shown in Figure 6(c), and the calculation expression is shown in Formula (11), where λ is the bias value of the negative interval of the Lish function, which can add a non-linear component to the input from the negative interval of the Lish function, and its value is determined as 1.01 according to the experiment.

$$Lish(x) = \begin{cases} x, x > 0 \\ x\ln\left(\lambda + e^{x}\right), x \le 0 \end{cases} \tag{11}$$

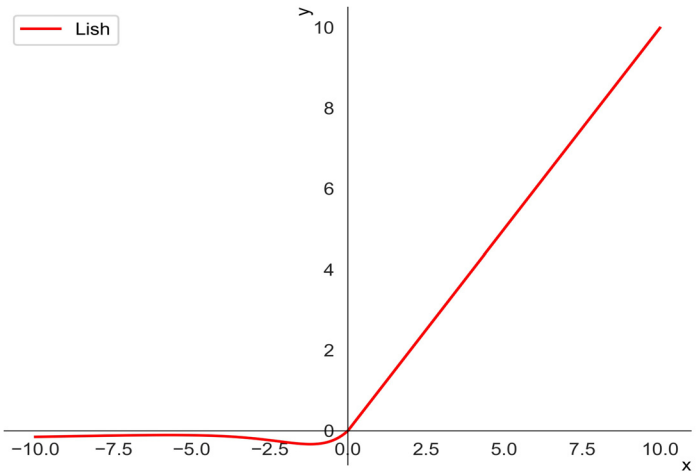
It can be seen from Figure 6 that the Lish activation function has the characteristics of whether there is a lower bound and not an upper bound, which can avoid gradient saturation which cause a sharp drop in the training speed, and is conducive to the realisation of strong regularisation effect, so that the model can be better fitted. The Lish activation function has smooth and non-monotonic characteristics similar to the Mish function in the vicinity of the 0 interval, which can improve the expression ability of network context information. At the same time, the Lish function has the same slope as the Leaky Relu function for the negative value input in the range far from 0. By inputting a non-linear component to the negative interval, the negative value input that is too small is never 0, which effectively prevents the 'Dead Neuron' phenomenon.

Figure 6 Activation function curve graph



(a) Leaky Relu

(b) Mish



(c) Lish

4 Other improvement strategies

4.1 K-means++ clustering algorithm

In order to improve the accuracy of the prior frame of the algorithm, in this paper, the *k*-means++ clustering algorithm is used to recluster the data collected in the automatic driving scene, and nine a priori boxes with different sizes are generated. The *k*-means++ clustering algorithm is an improvement of the *k*-means (Chen et al., 1998) clustering algorithm. The main difference is that the *k*-means algorithm generates *k* initial clustering centre points by random selection, while the *k*-means++ algorithm first randomly selects a sample point in the data set as the first initial clustering centre point. Secondly, the distance between each sample point in the data set and the initialised cluster centre point is calculated. Then, the maximum distance point is selected as the new cluster centre point until all *k* cluster centres are generated. After determining all the initial cluster centres, the *k*-means algorithm is used to calculate the final cluster centre. By selecting the clustering a priori frame, finally, the initial size of the prior frame of the YOLO algorithm in this paper is [13 26, 26 90, 36 42, 49 155, 68 81, 91 253, 134 134, 200 249, 316 307].

4.2 FocalLoss function

In order to adjust the serious imbalance of the proportion of positive and negative samples of YOLO, the FocalLoss loss function is introduced to replace the cross-entropy loss function of YOLO in order to calculate the confidence. The FocalLoss loss function is an improvement based on the cross-entropy loss function. The expression of the cross-entropy loss function is shown in Formula (12), where y' is the probability predicted through the sigmoid activation function, with a value between 0 and 1, and y is 1 or 0, representing positive and negative samples, respectively.

$$\begin{aligned} L &= -y\log y' - (1-y)\log(1-y') \\ &= \begin{cases} -\log y', y=1 \\ -\log(1-y'), y=0 \end{cases} \end{aligned} \tag{12}$$

The FocalLoss loss function adds a modulation coefficient $(1-y')^{\gamma}$ on the basis of the cross-entropy function, which reduces the weight of the simple negative samples in the training to improve the model optimisation ability. Its expression is shown in Formula (13). When $\gamma > 0$, compared with the ordinary cross-entropy function, for easy-to-classify samples, the predicted probability y' tends to be 1 and the modulation coefficient is close to 0, which reduces the loss of easy-to-classify samples. For difficult-to-classify samples, the predicted probability is small, the modulation coefficient is close to 1, and the loss value is basically unchanged, which increases the weight of the hard-to-classify samples in the overall loss and improves the model's attention to the detection of hard-to-classify samples. At the same time, by introducing a balance factor α into the FocalLoss loss function, the problem of an unbalanced proportion of positive and negative samples in the training process is alleviated. Since the modulation coefficient may lead to an excessive loss of hard-to-classify samples, the balance factor α is usually set to 0.25 to balance the loss value between positive and negative samples.

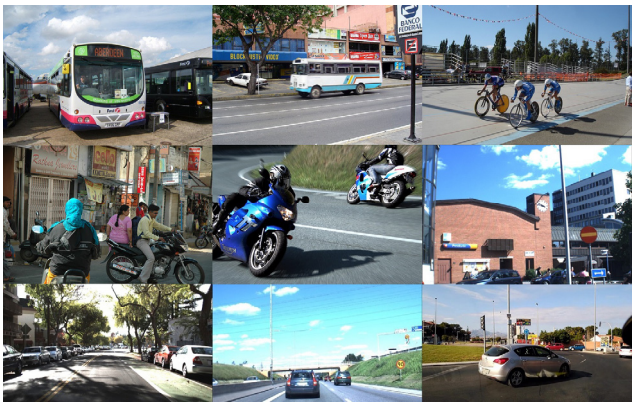
$$L_{focal} = \begin{cases} -\alpha(1-y')^{\gamma}\log y', y=1 \\ -(1-\alpha)(y')^{\gamma}\log(1-y'), y=0 \end{cases} \tag{13}$$

5 Related experiments and analysis

5.1 Data set and experimental environment

In view of the research background faced in this paper, the Urban Object Detection automatic driving data set released by The Robotics and Tridimensional Vision Group (RoViT) is used for experiments. Some of the data from the Urban Object Detection data set were collected by in-vehicle high-definition cameras in various scenarios. The detection targets include seven categories, bicycles, buses, motorcycles, cars, pedestrians, traffic signs and traffic lights, which basically summarise the main detection targets in an automatic driving scenario. The actual scene of various detection targets is shown in Figure 7. By cleaning and screening the data set, 21,385 images in the training set, 4277 images in the validation set and 5343 images in the test set were finally obtained.

Figure 7 Target actual scenario diagram



5.2 Experimental setting

When the model in this paper was trained, the input image size of the data set was 416×416. It was found in the experiment that the Mosaic data-enhancement method in the original YOLOv4 algorithm could significantly reduce the convergence speed of the model and could not bring effective performance improvement to the model, so the original Mosaic data enhancement was turned off. Standard data-enhancement methods such as image flip, translation and colour gamut distortion were used. The Adam gradient descent method is used for training backpropagation for optimisation, the initial learning rate is 0.001, the cosine annealing learning rate adjustment strategy is used; the training period is 300, and the batch size is set to 64.

In this experiment, precision, recall, *F*1 score and mean precision (*map*) (Everingham et al., 2015) were used as quantitative evaluation criteria for model detection performance. Additionally, the expression of the map is shown in Formula (14), where *C* is the number of classes in the data set, *P*(*R*) represents a two-dimensional curve with precision and recall as the abscissa and vertical coordinates, and the area under the *P*-*R* curve is the average accuracy (*AP*) of a single class target. The expressions for precision and recall are shown in Formula (15) and Formula (16). And the expression of the F1 score is shown in Formula (17), and the F1 score is a weighted fusion of accuracy and recall, which can more effectively evaluate model performance. In addition, the size of the model and the inference speed (FPS) are also used as a lightweight evaluation standard for the model, the FPS value is an important factor in whether the algorithm can achieve real-time detection of automatic driving scenarios, and the size of the model determines whether the algorithm can run on edge devices with limited storage resources.

$$map = \frac{1}{C}\sum_{i=1}^{C}\int_0^1 P(R)d(R) \tag{14}$$

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

5.3 Experimental result and analysis

In order to test the optimisation effect of each improved module of the YOLO algorithm in this paper, this paper conducts ablation experiments, adding various improved modules on the Urban Object Detection automatic driving data set. The results of the ablation experiments with various improved modules are shown in Table 1. In this experiment, the average precision map, FPS, and model size of the algorithm after the combination of each module are tested. Among them, A to F are the model names used after continuously superimposing the improved modules on the original YOLOv4.

It can be seen from Table 1 that when the original YOLOv4's CSPdarknet53 backbone network is replaced with the lightweight network Mobilenetv1, although the model map has been reduced to a certain extent, the model size has been reduced by nearly 100MB, and the detection speed has been greatly improved, verifying the effectiveness of the depth of separable volumes. Then, according to the application background of automatic driving, the *k*-Means++ clustering algorithm is used to generate more accurate a priori boxes, so the model detection results are more accurate. On this basis, the DW-CSP module is introduced into the feature fusion network; by dividing the gradient flow, the reusability of the feature information is improved and the amount of model parameters is simplified. The results show that although the recall of the improved YOLO algorithm is slightly reduced after the introduction of the DW-CSP module, the size of the model is reduced by more than 100Mb and the model map is maintained, and the model's inference time for a single image is improved by about 18%, indicating that the DW-CSP module can significantly improve the model's detection performance. Using this model, the cross-entropy loss function for calculating confidence in the original algorithm is replaced by the FocalLoss loss function, and the model map and F1 score are again increased by 1 percentage point. By reducing the loss of untargeted negative samples, the detection ability of positive samples is improved. Finally, the Lish activation function is proposed in the feature fusion network, and the model map is again increased by nearly 1 percentage point, at the same time, the precision of the model reaches 87.5%, significantly improving the model detection performance. By taking into account the advantages of the Leaky Relu function and the Mish function, the detection accuracy of the algorithm is improved.

Table 1 Improved YOLOv4 ablation experiment results

Model	*YOLOv4 and improvements*	*Precision (%)*	*Recall (%)*	*F1 (%)*	*Map (%)*	*Fps*	*Size (MB)*
A	YOLOv4	85.38	69.61	76.69	77.79	34	256.4
B	A+Mobilenetv1	72.78	70.26	71.49	76.89	63	166.7
C	B+K-means++	74.17	71.53	72.82	77.48	63	166.7
D	C+DW-CSP	78.03	68.24	72.80	77.44	76	58.0
E	D+FocalLoss	80.45	68.97	74.26	78.60	76	58.0
F	E+Lish	87.50	69.19	77.27	79.51	74	58.0

In order to further illustrate the effectiveness of the improved algorithm, this paper conducted performance comparison experiments with other mainstream target detection algorithms on the Urban Object Detection automatic driving data set. In the experiment, the experimental environment was kept unchanged, and the model's training period was 300 epochs. The pre-trained model initialisation weight parameters were used, and the model detection targets included seven types of traffic targets in the Urban Object Detection data set. The main comparison models include Retinanet50, Retinanet101, Faster-RCNN, YOLOv3, YOLOv4 and YOLOv5m, in addition, a number of mainstream lightweight YOLOv4 algorithms are used to compare with the algorithms in this paper, such as Mobilenetv2-YOLOv4 and Ghostnet-YOLOv4 using lightweight backbone networks, and the experimental results are shown in Table 2.

It can be seen from Table 2 that compared with the benchmark algorithm YOLOv4, the algorithm used in this paper has a 1.72% improvement in accuracy, the model size is reduced by about 4 times and the detection speed is increased by about 53% compared with the faster detection speed of Retinanet50, YOLOv3, etc. Compared with the YOLOv5m algorithm, which is more balanced in all aspects of the conventional algorithm, it is still lower than the improved algorithm in each detection result. The improved YOLO algorithm not only has a faster inference speed but

also outperforms other algorithms on the map. Compared with other target-detection algorithms, the algorithm in this paper improves the detection accuracy by up to 23.55% and the detection speed by up to about 61%, and the model size is reduced by up to about 4.4 times. Compared with other lightweight YOLOv4 algorithms, although the size of the proposed model is slightly increased, it maintains a faster inference speed, and the F1 score and map of the model are higher. At the same time, its detection speed reaches 74 frames per second, which can meet the real-time requirements well compared with other object-detection algorithms. Based on the detection results of various aspects, the improved YOLO algorithm is shown to have good detection performance.

Table 2 Comparison results of object detection algorithm

Algorithm	*Precision (%)*	*Recall (%)*	*F1 (%)*	*Map (%)*	*Fps*	*Size (MB)*
Retinanet50	42.73	73.38	54.00	63.70	44	146
Retinanet101	49.32	73.04	58.88	68.96	30	222.4
Faster-RCNN	40.06	73.96	51.97	62.14	29	113.7
YOLOv3	63.22	71.74	67.21	73.83	52	246.6
YOLOv4	85.38	69.61	76.69	77.79	34	256.4
YOLOv5m	67.38	69.42	68.4	74.32	51	84.1
Mobilenetv2-YOLOv4	70.90	71.10	70.99	75.88	49	49.2
Mobilenetv3-YOLOv4	72.02	70.79	71.39	76.57	44	56.8
Ghostnet-YOLOv4	66.19	71.25	68.62	74.73	37	45.0
Improved YOLO	87.50	69.19	77.27	79.51	74	58.0

Figure 8 shows the comparison of the detection results between the improved YOLO algorithm and the benchmark algorithm YOLOv4 in four scenarios. The top-down observation of the four scenarios in Figure 8 shows that the improved YOLO algorithm has better detection capabilities compared to the YOLOv4 algorithm. For example, in the detection of the bus target in the first scenario, the prediction frame of the YOLOv4 algorithm did not accurately detect the target position, but the prediction frame of the improved YOLO algorithm could completely frame the bus target, indicating that the improved YOLO algorithm is more accurate for the prediction of targets' position. In the detection for other scenarios, the YOLOv4 algorithm has different degrees of missed selection. For example, in the second and fourth scenarios, the YOLOv4 algorithm missed the detection of bicycles and pedestrians at the edge of the image, but the improved YOLO algorithm could effectively detect them. In the third scenario, the improved YOLO algorithm also detected farther traffic signs than the YOLOv4 algorithm.

Figure 8 Comparison of target detection results

(a) YOLOv4 algorithm (b) Improved algorithm

6 Conclusions

This paper redesigns the YOLOv4 network, which reduced the number of parameters of the model. At the same time, the Lish activation function was designed in the feature fusion network to reduce the loss of contextual information during model inference. Subsequently the *k*-means++ clustering algorithm was used to regenerate a priori boxes of different scales to improve the accuracy of the algorithm's prediction boxes. Finally, the FocalLoss loss function was used to replace the cross-entropy loss function for calculating the confidence, which improves the model optimisation ability. Through the analysis of ablation experiments and comparative experiments, the effectiveness of the improved module in this paper was verified. However, due to the inclusion of many occlusion small targets in the autonomous driving scene, the experiment in this paper lacks the verification of occlusion small target detection, and there are still risks in the detection of small targets in complex scenes, and future work should study the detection research of small targets in autonomous driving scenarios.

References

Bochkovskiy, A., Wang, C. and Liao, H. (2020) 'YOLOv4: optimal speed and accuracy of object detection [EB/OL]', *arXiv: 2004.10934*.

Boer, P.T.D. and Kroese, D.P. and Mannor, S. et al. (2005) 'A tutorial on the cross-entropy method', *Annals of Operations Research*, Vol. 134, No. 1, pp.19–67.

Chen, C.W., Luo, J.B. and Parker K.J. (1998) 'Image segmentation via adaptive k-means clustering and knowledge-based morphological operations with biomedical application', *IEEE Transactions on Image Processing*, Vol. 7, No. 12, pp.1673–1683.

Ding, J.L., Liu, T. and Zhao, Y. et al. (2022) 'HAPNet: a head-aware pedestrian detection network associated with the affinity field', *Science China (Information Sciences)*, Vol. 65, No. 6, pp.17–32.

Everingham, M., Eslami, S.M.A. and Van Gool, L. et al. (2015) 'The pascal visual object classes challenge: a retrospective', *International Journal of Computer Vision*, Vol. 111, No. 1, pp.98–136.

Girshick, R. (2015) 'Fast R-CNN', *IEEE International Conference on Computer Vision*, IEEE, Santiago, pp.1440–1448.

Girshick, R., Donahue, J. and Darrell, T. et al. (2014) 'Rich feature hierar chies for accurate object detection and semantic segmentation', *IEEE Conference on Computer Vision and Pattern Recognition*, pp.580–587.

He, K., Zhang, X. and Ren, S. et al. (2015) 'Spatial pyramid pooling in deep convolutional networks for visual recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pp.1904–1916.

Howard, A.G., Zhu, M. and Chen, B. et al. (2017) 'Mobilenets: effificient convolutional neural networks for mobile vision applications [EB/OL]', arXiv: https://arxiv.org/pdf/1704.04861.pdf.

Lin, T., Dollar, P. and Girshick, R. et al. (2017) 'Feature pyramid net works for object detection', *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, No. 2, pp.2117–2125.

Lin, T.Y., Goyal, P. and Girshick, R. et al. (2017) 'Focal loss for dense object detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 99, pp.2999–3007.

Liu, S., Qi, L. and Qin, H. et al. (2018) 'Path aggregation network for instance segmentation', *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.8759–8768.

Liu, W., Angelov, D. and Erhan, D. et al. (2016) 'SSD: single shot multi-box detector', *European Conference on Computer Vision*, pp.21–37.

Luo, J.Q., Fang, H.S. and Shao, F.M. et al. (2021) 'Multi-scale traffic vehicle detection based on faster R-CNN with NAS optimization and feature enrichment', *Defence Technology*, Vol. 17, No. 4, pp.1542–1554.

Maas, A.L., Hannun, A.Y. and Ng, A.Y. (2013) 'Rectifier nonlinearities improve neural network acoustic models', *Proceedings of International Conference on Machine Learning (ICML),* Vol. 30.

Misra, D. (2019) 'Mish: a self regularized non-monotonic neural activation function [EB/OL]. 2019-10-2: https://arxiv.org/abs/1908.08681.

Redmon, J. and Farhadi, A. (2017) 'YOLO9000: better, faster, stronger', *Computer Vision and Pattern Recognition*, pp.6517–6525.

Redmon, J., Divvala, S. and Girshick, R. et al. (2016) 'You only look once: Unifified, real-time object detection', *IEEE Conference on Computer Vision and Pattern Recognition*, pp.779–788.

Ren, S.Q., He, K.M. and Girshick, R. et al. (2015) 'Faster R-CNN: towards real-time object detection with region proposal networks', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp.1137–1149.

Tian, Z.H., Sun, Y.Y. and Wei, H.T. (2021) 'Traffic sign detection algorithm based on SSD model', *Computer Applications and Software*, Vol. 38, No. 12, pp.201–206.

Wang, C., Mark, L.H. and Wu, Y. et al. (2020) 'CSPNet:a new backbone that can enhance learning capability of CNN', *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Seattle, WA, USA, pp.1571–1580.

Wang, L., Zhang, S. and Wang P. et al. (2021) 'A real-time vehicle target detection algorithm based on MS-YOLOv3', *Computer Applications and Software*, Vol. 38, No. 10, pp.189–195.

Xiong, L. and Tang, W.M. (2018) 'Research on multi-classifier selection classification based on k-means++', *Journal of Chongqing Normal University (Natural Science Edition)*, Vol. 35, No. 6, pp.88–96.

Xu, X.K., Ma, Y. and Qian X. et al. (2021) 'Scale-aware real-time pedestrian detection in autonomous driving scenes', *Chinese Journal of Image Graphics*, Vol. 26, No. 1, pp.93–100.

Yang, T.T. and Tong, C. (2022) 'Real-time detection network for tiny traffic sign using multi-scale attention module', *Science China (Technological Sciences)*, Vol. 65, No. 2, pp.396–406.

Yuan, Z.H., Sun, Q. and Li, G.X. et al. (2020) 'Target detection for autonomous driving based on Yolov3', *Journal of Chongqing University of Technology (Natural Science)*, Vol. 34, No. 9, pp.56–61.