Lightweight Vehicle Detection Based on Improved Yolox-nano

XiuPing Jing, Ying Tian

Abstract—**Vehicle detection based on deep learning plays a vital role in various fields, such as autopilot and intelligent transportation. Moreover, it presents a major development direction to computer vision in recent years. Lightweight vehicle detection aims to reduce the burden of computer storage and computing, including exploration of network structure and computing efficiency. But it is difficult to cope with complex and changeable traffic scenes. The urgent problem is to enhance lightweight network performance while maintaining inference speed. In this context, this study proposes an improved Yolox-nano to alleviate the above-mentioned problems by splitting the vanilla convolution unit into two parts and utilizing fewer convolution kernels to generate several feature maps. Subsequently, several linear transformations are further applied to generate cheap feature maps efficiently. The cheap convolutional unit Ghost Module can control feature map ratio of vanilla convolution and linear transformation. The activation function in Ghost Module is improved by SiLU, which has a more powerful nonlinear ability. In addition, the convolution and the batch normalization in Ghost Module can be fused in the pre-processing stage to speed up inference time. Experimental results show that the improved network provides 4.9% and 3.1% mAP on Pascal VOC and MS COCO vehicle datasets. The improved network enhances the performance and maintains an efficient inference speed, and detection results become further competitive using Yolox-nano.**

Index Terms—**Vehicle Detection, Yolox-nano, Lightweight Network, Ghost Module.**

I. Introduction

With the rapid economic growth, the number of vehicles has increased year by year. People pay more attention to safety issues, and vehicle detection [1] has become one of the research hotspot of computer vision. Vehicle detection has been widely used in autopilot, vehicle monitoring, and so on.

Traffic problems such as running red lights, illegal parking, and illegal vehicles on the road are very frequently. These problems pose a enormous threat to traffic safety. In the past, it mainly relied on the traffic police to maintain order or stare at videotapes of illegal vehicles, which makes the police very tired and cost a lot of time.

Traditionally, vehicle detection relied on the HOG [2] or SVM [3] algorithm, which lacks robustness, generalization, and the ability to cope with complex traffic scenes.

Nowadays, object detection algorithm based on deep learning has a more powerful feature extraction capability and robustness, and vehicle detection algorithm method has made great progress.

At the beginning of deep learning, there are many two-stage object detection algorithms. These algorithms usually have high detection accuracy, but need enormous computational complexity. There are some deficiencies when detecting fast moving vehicles. The emergence of lightweight network can solve these problems better.

Lightweight vehicle detection [4] aims to reduce the burden of computer storage and computing, including the exploration of network structure and computing efficiency, which promotes the application of computer vision and other technologies in many devices and is widely used in many fields such as intelligent transportation. Due to the limitation of hardware performance and cost, lightweight networks can not only perceive the surrounding situation accurately but also interact with the external environment quickly. Lightweight networks have a better application scenario.

But it is difficult for lightweight networks to cope with complex and changeable traffic scenes. The urgent problem is to enhance the lightweight network performance while maintaining inference speed. In this context, this study proposes an improved Yolox-nano to alleviate the above-mentioned. This study proposes to reconstruct the residual bottleneck and part of the convolution unit by using Ghost Module. The activation function in Ghost Module is improved by SiLU, which nonlinear ability is more powerful. In addition, the convolution and the batch normalization in Ghost Module can be fused in the pre-processing stage to speed up inference time. These operations enhance the ability to extract features without reducing inference speed substantially.

II. Related Algorithms

Yolox [5] is a one-stage anchor-free object detector, and Yolox-nano is the lightest network in the Yolox series, which is a new-generation object detector that integrates various training strategies. Yolox-nano gathered experience from previous Yolo [6]-[8] series, offering data augmentation, decoupled head, and SimOTA algorithm to the program. It is an end-to-end high-performance object detector.

Training pictures after various data enhancements first pass through the Focus module. The Focus module divide three RGB channel pictures into a group every four pixels. Then the pixels at the corresponding positions in each group

Manuscript received April 29, 2022; revised January 04, 2023.

XiuPing Jing is a postgraduate student majoring in software engineering at School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Liaoning 114051, China. (e-mail: jxp_snowman@163.com)

Ying Tian, the corresponding author, is a professor of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Liaoning 114051, China. (phone: +8613898015263; e-mail: astianying@126.com)

can be spliced together, and the pixels at the corresponding positions can be concatenated together according to the channel dimension. At this time, the channel dimension is four times larger than before.

The backbone of Yolox-nano is CSPDarknet [9]. To reduce the model size, a part of the convolution in Yolox-nano is a Depthwise Separable Convolution [10]. Same to EfficientNets [11], [12] series, it reduces the model size by controlling the model's depth and width.

Most networks used only one branch to predict the classes, confidence, and offsets. But in recent years, decoupled heads are used in many advanced single-stage or two-stage networks. Researchers found that classification tasks differ from regression tasks in object detection experiments. There are some differences between classification tasks and regression tasks in the network. Classification tasks and regression tasks focus on different feature information. Classification tasks pay more attention to the texture information of the target, while regression tasks pay more attention to the edge information of the target. Hence, single-branch prediction is inadequate, and the head of the detector needs to be decoupled. Decoupling features before prediction improves network performance. Furthermore, two parallel branches are needed to finish classification and regression tasks, respectively, such as Fcos [13], and VFNet [14]. Yolox-nano also adopts the same strategy, and prediction layers don't share one decoupled head.

Most object detectors are anchor-based, and these detectors are lacking generalization. When dealing with different issues, it is necessary to use the K-means algorithm to analyze datasets. Moreover, anchor-based networks have excessive hyperparameters, which renders debugging codes more complex. However, Yolox-nano is an anchor-free network that avoids aforementioned problems.

III. Improvements

A. Ghost Module

Due to the development of computer vision, researchers pay more attention to practical value. To augment the use of models by various devices, neural networks need a lightweight structural design. In this context, multiple new ideas in network structure design have emerged in recent years. Take for example OctConv [15] reduces feature redundancy by reducing high-frequency features. SPConv [16] proposes to get more feature maps by pointwise convolutions. SilmConv [17] gets feature maps increase by inverting SE [18] attention weight, while HSBlock [19] divides feature maps into multi-level divisions. These operations could reuse features. These modules are designed ingeniously, but most of them are difficult to implement, which prevents convolution layer and batch normalization layer from being fused in the pre-processing stage, thus slowing down inference speed.

GhostNet [20] is a new efficient neural network architecture proposed by Huawei Noah Lab. The Ghost module aims at generating more feature maps with cheap operations. On this basis, the feature map generated by vanilla convolution contains partially similar redundant feature maps, it's necessary to reduce redundancy feature maps by using less convolution kernels.

Given the input feature maps x, where $c1$ is the number of input channels. The operation of the convolutional layer for generating $c2$ feature maps can be calculated as:

$$Y = W \cdot x + b \tag{1}$$

where $\cdot$ represents the matrix broadcast multiplication, b is the bias, Y is the output feature map with $c2$ channels, and W is the convolution filters. The number of filters and the channel number is a considerable constant when dealing with high dimension feature maps.

It can be known from data visualization that the output results of vanilla convolutional layers include some similar results, so it is unnecessary to consume such huge computing resources to generate these redundant feature maps. First, $\frac{c2}{s}$ feature maps Y' are generated using a vanilla convolution(bias is omitted):

$$Y' = W' \cdot x \tag{2}$$

where W' is the utilized filters, and $s \geq 2$. Second, to further generate the desired $c2$ feature maps, cheap linear transformations on each feature maps are applied in Y' to generate $c2 - \frac{c2}{s}$ ghost features, according to the function: $y = \Phi(Y')$. And Φ are linear transformations, and y represents the cheap features which generated by a series of linear transformations. Finally, it is necessary to concatenate the two parts of the feature map to get the final output. Fig. 1 is the structure of Ghost module when $s = 2$.

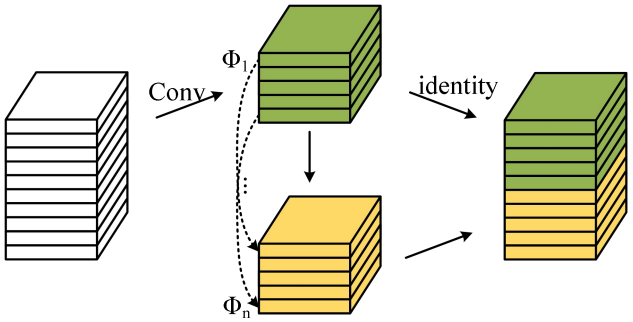


Fig. 1. The structure of Ghost Module when $s = 2$.

When building Ghost Module, affine transformation is also considered as a cheap linear transformation. However, under the acceleration of current computing device and software algorithms, convolution is also recognized as an efficient linear transformation. And well supported by various computing platforms. Thus it covers several widely used linear operations. Hence, this study chooses DepthWise and GroupWise convolutions to achieve linear transformation in experiments.

Taking linear transformation and vanilla convolution of the same size in one Ghost module for efficient execution, the kernel size d of the linear transformation is the same as the kernel size k of identity mapping, and $s \ll c$.

B. Activation Functions

A proper nonlinear mapping plays a critical role in neural networks. On the other hand, multiple popular activation functions exist in convolutional networks, such as ReLU,

SiLU, Metacon, etc. Different activation functions greatly affect the training stability and the performance of object detector.

The activation of Ghost Module is different from GhostNet in improved networks. The original activation function of GhostNet is ReLU, which performance is unsatisfactory in Yolox-nano. Hence, this study improved the activation function in Ghost Module as SiLU to get a more powerful nonlinear ability.

As for SiLU, it is a special case of Swish [21]. Swish is a widely used activation proposed by google brain, using automatic search techniques. Swish is defined as $x \cdot sigmoid(\beta x)$, and β is either a predefined number or a trainable tensor. Various activation functions are present in Fig. 2, such as ReLU, SiLU, and different values of β for Swish. When $\beta = 0$, Swish becomes the scaled linear function $f(x) = 0.5x$. When $\beta \to \infty$, Swish becomes positive proportional function. Subsequently, it is almost coincident with ReLU when $\beta = 0.1$. Swish can be regard as a smooth function which the range of function image is between the positive proportional function and the ReLU function. And SiLU is a special case of Swish when $\beta = 1.0$.

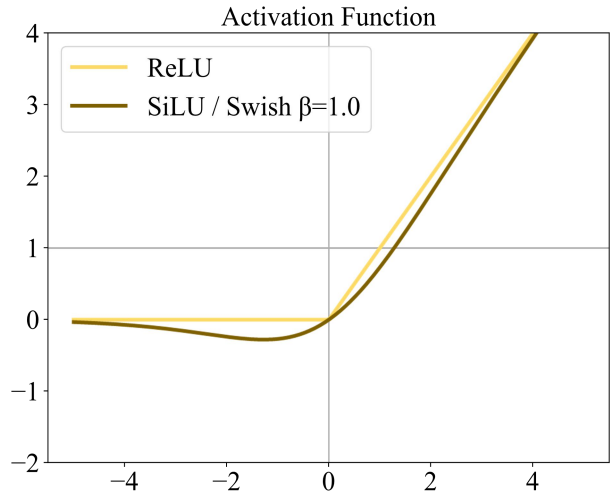


Fig. 2. Different kinds of activation functions.

Similar to Swish, SiLU is unbounded above and bounded below. Furthermore, it is smooth, nonmonotonic, and derivative everywhere. It is non-monotonic when x<0, and that is the starkest difference from ReLU. When extracting features, SiLU is more diverse than ReLU.

C. Convolution and Batch Normalization fusion

Batch normalization is essential in convolutional neural networks, which speeds up its convergence speed and enhances model performance. On the other hand, it increases the calculation cost of the network and slows down the inference speed. The convolution layer and the batch normalization layer can be fused in Ghost Module in the pre-processing stage. The calculation process of a normal convolution layer can be expressed as:

$$Y = W \cdot x + b \tag{3}$$

the same as (1). Moreover, the calculation process of a batch normalization layer can be expressed as:

$$Y_{bn} = \gamma \cdot \frac{(x - \mu)}{\sqrt{\delta^2 + \varepsilon}} + \beta \tag{4}$$

where γ and β are trainable parameters, μ is mean value, δ^2 is the variance, and ε is a non-zero decimal. Bringing (3), into (4), we can get:

$$\begin{aligned} Y_{bn} &= \gamma \cdot \frac{(W \cdot x + b - \mu)}{\sqrt{\delta^2 + \varepsilon}} + \beta \\ &= \frac{\gamma \cdot W}{\sqrt{\delta^2 + \varepsilon}} \cdot x + \frac{\gamma \cdot (b - \mu)}{\sqrt{\delta^2 + \varepsilon}} + \beta \end{aligned} \tag{5}$$

$$W_{fuse} = \frac{\gamma \cdot W}{\sqrt{\delta^2 + \varepsilon}} \qquad \beta_{fuse} = \frac{\gamma \cdot (b - \mu)}{\sqrt{\delta^2 + \varepsilon}} + \beta \tag{6}$$

Consequently, an arbitrary convolutional layer fused by a batch normalization layer can be formulated as:

$$Y_{fuse} = W_{fuse} \cdot x + \beta_{fuse} \tag{7}$$

Using this method, a convolution layer and a batch normalization layer are fused into one layer in the pre-processing stage. These operations do not harm to precision but speed up inference time.

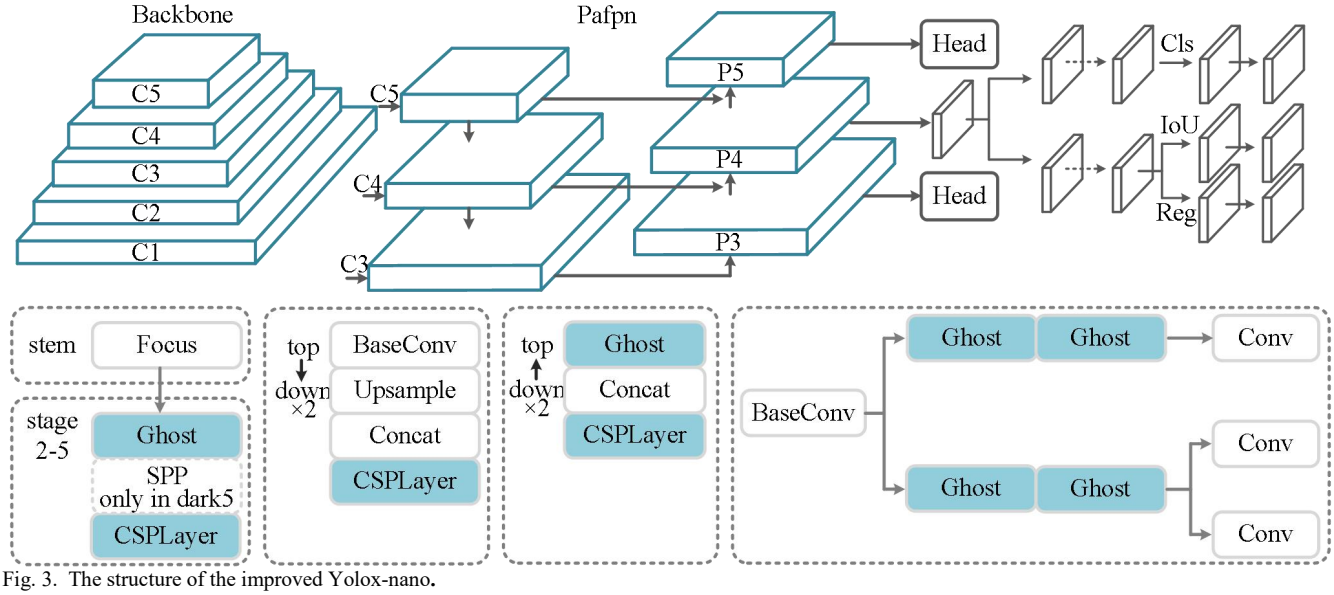


Fig. 3. The structure of the improved Yolox-nano.

D. Module Detials

As shown in Fig. 3, downsample, decoupled head, and some of the CSPLayer convolutions are replaced with Ghost Module. The network size can be controlled by variable *s*. The network performance is explored when generating a different number of cheap feature maps. Hence, cheap feature maps generated by Ghost module can be controlled by *s*=2 or 4. Although *s* could be an odd number (such as 3 or 5), the feature maps generated by linear transformations are truncated during concatenating. This operation wastes calculating resources. Hence, to ensure that the generated feature maps are not wasted, *s* is only equal to 2 or 4 in the experiments.

The SPP module consists of three different kinds of kernel size maxpool and a 1×1 convolution unit, mainly used to get different kinds of feature maps of the receptive field. The SPP module is applied exclusively in stage 5, meaning it does not exist in other stages. Both Pafpn [22] and decoupled head, the first BaseConv units are 1×1 reduction convolutions. Three vanilla convolutions at the end of the decoupled head are used to predict the class, confidence and offset, respectively, without activation function and batch normalization. The improved modules are drawn in dark color.

C5 means that backbone downsampling is 32 times at most. The convolution in the Focus module is a 3×3 convolution unit used for downsampling. These modules are unmodified in experiments. The improved Yolox-nano remains in three decoupled heads. Since the three convolutions at the end of the decoupled head are vanilla convolutions, the convolution at this position is not replaced by Ghost Module.

The left top-to-bottom part of the Pafpn, which operations are similar to the right bottom-up, is executed twice. The whole network structure and the number of residuals are kept. During feature extraction, the number of residuals of CSPLayer in each stage is 1, 3, 3, and 1.

The improved network didn't attempt to add more Ghost Module because this operation makes the network too large and slows down the inference speed. And Ghost Module doesn't replace 1×1 convolution unit, because this operation makes the number of network layers explosive increase. The network may have difficulty converging. The experiment only made one attempt, replacing all convolutions with Ghost Module. Although the computational complexity of the network can be greatly reduced, the training process became extremely unstable. Obviously, this is not the expected experimental result.

Fig. 4 is the structure of the improved CSPLayer. The latter consists of left and right branches. The left branch is made up of a 1×1 BaseConv unit, mainly used to reduce channels, similarly to the first right BaseConv unit. The Bottleneck is mainly used to extract features. The n represents the number of times when the Bottleneck is added repeatedly. BaseConv unit includes a vanilla convolution, a batch normalization, and an activation function. The improved part has been marked in dark color.

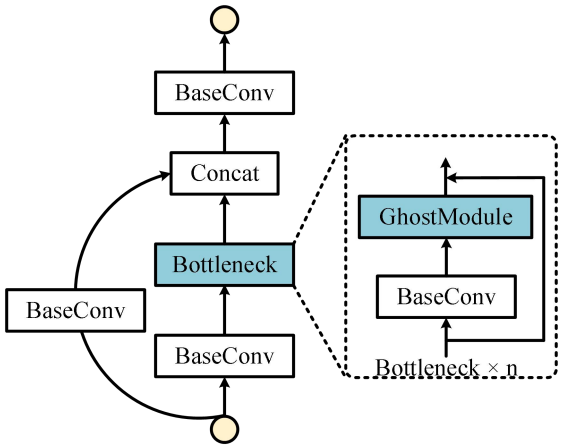


Fig. 4. The structure of the improved CSPLayer.

IV. Experiments

A. Experiment Environments

All experiments ran on the DELL server. In the experiment, the commonly used Centos7 was selected as the training platform and the inference time is also estimated on this device. GTX1080ti × 2 are used as a computing device to save training time. And more comparative experiments are executed to explore the performance of the improved network and verify the superiority of the improved model. Due to the limitation of Cudnn version, Pytorch and Cuda version are 1.7.1 and 10.1 in the experiment.

B. Datasetsets and Settings

Fig. 5. The Pascal VOC vehicle datasetset.

Fig. 6. The MS COCO vehicle datasetset.

Object detection experiments were conducted on two common-used datasets, including Pascal VOC and MS COCO.

The Pascal VOC 2012 dataset presents 20 categories for about 17k images. All vehicle classes are extracted, including car, bus, train, bicycle, and motorbike, for a total of 3073 pictures, 80% of which are randomly split as the training set and 20% as validation set. Furthermore, the same number of images is extracted as the validation set from Pascal VOC 2007 test set for testing. Subsequently, this study chose 2459 images for training and 614 images for validation and testing.

The MS COCO 2017 dataset presents 80 categories for about 120k images. All vehicle classes are extracted, including bicycles, car, motorbike, bus, train and truck, and selecting randomly a total of 7000 pictures in total, 80% of which are randomly split as the training set and 20% as the validation set and test set, respectively. In this setting, 5600 images were chosen for training and 700 for validation and testing in total. Fig. 5 and Fig.6 are the two vehicle datasets, respectively.

C. Object Detection

This study compares the improved Yolox-nano to Yolov4-tiny and Yolov3-tiny, which is the same lightweight network as Yolox-nano. Moreover, the one-stage object detection network RetinaNet [23] is chosen, which is a typical one-stage object detector widely used in the benchmarks of detection tasks.

The experimental code is based on the official Yolox repository. Models are trained from scratch on two GTX 1080ti GPUs by using 640×640 pixels. Models are trained for 700 epochs and 350 epochs on Pascal VOC and MS COCO vehicle datasets. This study employed batch size of 32 to train models, and following the cosine [23] decay learning rate schedule. The original data augmentation principles of Yolox-nano were followed, closing mosaic augmentation for the last 15 epochs. The standard of evaluation was IoU0.5 on test set. Training precision defaults to FP32.

The experimental code for RetinaNet is based on mmDetection repository, the version is 2.15. The optimizer is the same as Yolox-nano. Due to the limitation of GPU memory capacity, and the learning rate of the network is calculated by 8-GPUs in offical repository. This study choose learning rate with an initial lr= 0.005 and batch size of 4 for training. During the training phase, the ResNet-50 [25] is initialized with the weights pre-trained on ImageNet, while the other layer weights are obey the Xavier distribution. The model defaults to freeze in stage 1. All the latency in this report is measured with FP32-precision on a single GTX 1080ti.

For Yolov3-tiny and Yolov4-tiny, a relatively new and high performance implementation process is adopted in the experiment. For better comparison with lightweight networks. Therefore, the experimental results of these two programs are higher than original program.

During the experiments, to calculate the time consumed by network more accurately, the time consumed by Non-Maximum Suppression is taken into the consideration. Object detectors might generate multiple bounding boxes during validation or inference. However, few high-confidence bounding boxes need to be preserved. Non-Maximum Suppression is able to suppress most of the overlapping low-confidence boxes according to the threshold. Finally, the high-confidence bounding boxes are retained.

We named the network when Ghost module $s = 2$ as Yolox-nano-Ghost-s2. In parallel, when Ghost module $s = 4$ as Yolox-nano-Ghost-s4. All experimental details are shown in Table II and Table III. "#Params(M)" represents the parameters of the network. "GFLOPs" is estimated by the input size of 640×640. "Inference time(ms)" refers to the time that network takes to detect per image.

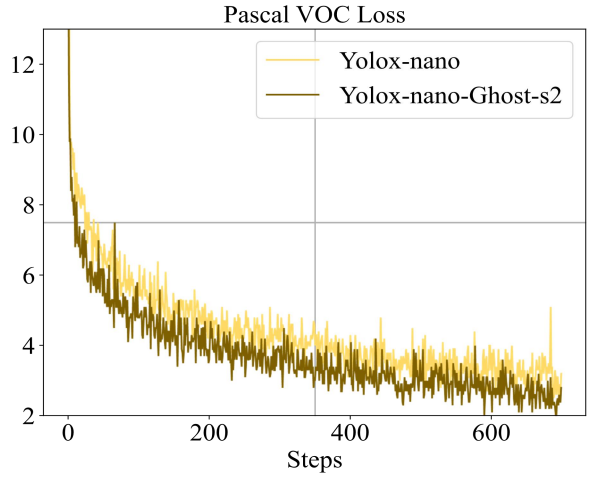
As it's shown in Table I, the mAP of Yolox-nano-Ghost-s2 is 4.9 points better than Yolox-nano, while the mAP of Yolox-nano-Ghost-s4 is 3.2 points better than Yolox-nano on Pascal VOC vehicle dataset. Although the number of parameters and GFLOPs of improved networks increased slightly, it is still less than Yolov3-tiny and Yolov4-tiny. The inference time is still maintained around the original level(the lower the better). The mAP of Yolox-nano-Ghost-s2 is 4.4 points lower than RetinaNet, but the number of parameters and GFLOPs of RetinaNet is as high as 36.2M and 122.2GFLOPs. The calculation scale of Yolox-nano-Ghost-s2 and Yolox-nano-Ghost-s4 are still more than 20 times smaller than RetinaNet. Moreover, the inference time per image is also about 14 times faster than RetinaNet.

The mAP of Yolox-nano-Ghost-s2 is 3.1% better than Yolox-nano, and the mAP of Yolox-nano-Ghost-s4 is 2.4% better than Yolox-nano on MS COCO vehicle dataset. The improved network has improved on both datasets. Thus, the improved network has strong generalization.

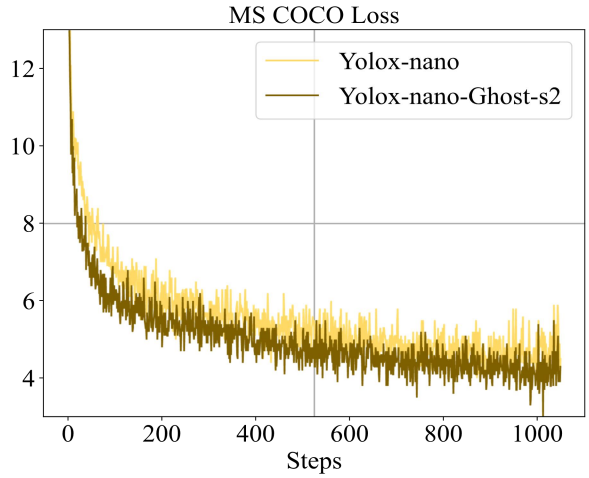
TABLE I
THE PERFORMANCE OF EACH NETWORK ON TWO VEHICLE DATASET

Module name	Img Size	#Params(M)	GFLOPs	mAP(%) Pascal VOC	mAP(%) MS COCO	Inference time(ms)
Yolox-nano	640×640	0.9	2.5	76.8	53.7	4.3
Yolox-nano-Ghost-s2	640×640	1.5	4.4	81.7	56.8	4.6
Yolox-nano-Ghost-s4	640×640	1.1	3.2	80.0	56.1	4.5
Yolov4-tiny	640×640	6.4	17.4	70.7	54.3	3.9
Yolov3-tiny	640×640	8.9	13.3	70.3	51.5	3.7
RetinaNet	1000×600	36.2	122.2	84.4	61.2	52.8

Fig. 7 exhibits the total loss of training set on two vehicle dataset(the lower the better). The light color is Yolox-nano, the dark color is Yolox-nano-Ghost-s2. The loss value of two datasets is close to 25 at most. To make the figure more clear, the maximum value of y-axis is 13, the minimum value of y-axis is 2. The line of Yolox-nano is significantly underneath Yolox-nano-Ghost-s2.



(a) Pascal VOC loss



(b) MS COCO loss

Fig. 7. (a) And (b) are the total loss of training set on two vehicle dataset, respectively.

D. Ablation Experiment

This section conducts ablation experiments on the Pascal VOC vehicle dataset. To verify the feasibility of each improvements. In GhostNet, the activation function in Ghost Module is ReLU. In the improved works, the activation functions of Ghost Module are replaced by SiLU to get powerful nonlinear ability. Table II shows the experimental results.

After the activation function in Ghost Module is changed from SiLU back to ReLU, the performance of Yolox-nano-Ghost-s4 and Yolox-nano-Ghost-s2 reduces about 0.7% and 0.9% mAP, respectively. Two improved models reduce about 0.2 ms inference time.

TABLE II
THE INFLUENCE OF ACTIVATION FUNCTION ON PASCAL VOC VEHICLE DATASET

Module name	ReLU	SiLU	mAP(%)	Inference time(ms)
Yolox-nano-Ghost-s4	√	×	79.3	4.4
	×	√	80.0	4.6
Yolox-nano-Ghost-s2	√	×	80.8	4.4
	×	√	81.7	4.6

The following experiments explore the impact of convolution layer and batch normalization layer fusion on network speed. The experiment uses Yolox-nano-Ghost-s2 as a benchmark.

Table III shows the impact of convolution and batch normalization fusion on network speed. It can be seen that after removing convolution and batch normalization fusion in Ghost Module, the network consumes an additional 0.2 ms inference time.

TABLE III
THE INFLUENCE OF CONVOLUTION AND BATCH NORMALIZATION FUSION ON PASCAL VOC VEHICLE DATASET

Module name	fuse	without fuse	Inference time(ms)
Yolox-nano-Ghost-s2	√	×	4.6
	×	√	4.8

E. Inference Result

Fig. 8 and Fig. 9 are the inference results on two vehicle test set. The inference results are generated by Yolox-nano, Yolox-nano-Ghost-s4, and Yolox-nano-Ghost-s2, respectively.

It is clear from Fig. 8 that Yolox-nano-Ghost-s2 provides higher confidence in detecting four overlapping bicycles, while Yolox-nano misses one of them. Furthermore, Yolox-nano misidentifies the crowd and the fence on the right as motorbikes and cars. Moreover, in Fig. 9, Yolox-nano-Ghost-s2 detects bicycles covered with flowers very precisely, unlike Yolox-nano, which is unable to detect them. When detecting the half-exposed car on the left side and the motorbike half-covered by people on the right side, the improved networks have higher precision. This study obtains the same conclusion for both datasets.

Fig. 8. The inference result on Pascal VOC vehicle dataset.

Fig. 9. The inference result on MS COCO vehicle dataset.

V. Conclusion

In short, this study proposes to improve Yolox-nano's performance by using Ghost module, which separates the vanilla convolution unit into two parts and use less convolution kernels to obtain less feature maps. The improved networks performed exquisitely in various aspects. The activation function in Ghost Module is replaced by SiLU. In addition, the convolution layer and the batch normalization layer in Ghost Module can be fused in the pre-processing stage to accelerate the inference time. Experiments show that the improved networks can get higher performance and significant robustness on two common used vehicle datasets. Furthermore, the improved networks are highly competitive in visual tasks, compared to other lightweight models. The improved network can meet the actual needs.

References

[1] Binbin Sun, Wentao Li, Huibin Liu, Jinghao Yan, Song Gao, and Penghang Feng, "Obstacle Detection of Intelligent Vehicle Based on Fusion of Lidar and Machine Vision." *Engineering Letters*, vol. 29, no.2, pp722-730, 2021

[2] Sulistyaningrum, D. R., et al. "Vehicle detection using histogram of oriented gradients and real adaboost." *Journal of Physics: Conference Series*. Vol. 1490. No. 1. IOP Publishing, 2020.

[3] Zhihua Ding."SVM Vehicle Identification based on LSD Method for Triangulation and Mesh Vector Feature." *International Journal of Computational and Engineering 1*.2(2016).

[4] D. L. Yuan, and Y. Xu, "Lightweight Vehicle Detection Algorithm Based on Improved YOLOv4," *Engineering Letters*, vol. 29, no.4, pp1544-1551, 2021.

[5] Ge, Zheng, et al. "Yolox: Exceeding yolo series in 2021." *arXiv preprint arXiv:2107.08430* (2021).

[6] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 779-788.

[7] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7263-7271.

[8] Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767* (2018).

[9] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934* (2020).

[10] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv: 1704. 04861* (2017).

[11] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International Conference on Machine Learning*. PMLR, 2019, pp. 6105-6114.

[12] Tan, Mingxing, and Quoc Le. "Efficientnetv2: Smaller models and faster training." *International Conference on Machine Learning*. PMLR, 2021, pp. 10096-10106.

[13] Tian, Zhi, et al. "Fcos: Fully convolutional one-stage object detection." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9627-9636.

[14] Zhang, Haoyang, et al. "Varifocalnet: An iou-aware dense object detector." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 8514-8523.

[15] Chen, Yunpeng, et al. "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3435-3444.

[16] Zhang, Qiulin, et al. "Split to be slim: An overlooked redundancy in vanilla convolution," *arXiv preprint arXiv:2006.12085* (2020).

[17] Qiu, Jiaxiong, et al. "Slimconv: Reducing channel redundancy in convolutional neural networks by features recombining." *IEEE Transactions on Image Processing* 30 (2021): 6434-6445.

[18] Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7132-7141.

[19] Yuan, Pengcheng, et al. "HS-ResNet: hierarchical-split block on convolutional neural network." *arXiv preprint arXiv:2010.07621* (2020).

[20] Han, Kai, et al. "Ghostnet: More features from cheap operations." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1580-1589.

[21] Ramachandran, Prajit, Barret Zoph, and Quoc V. Le. "Searching for activation functions." *arXiv preprint arXiv:1710.05941* (2017).

[22] Liu, Shu, et al. "Path aggregation network for instance segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8759-8768.

[23] Lin, Tsung-Yi, et al. "Focal loss for dense object detection" *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2980-2988.

[24] Loshchilov, Ilya, and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983* (2016).

[25] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770-778.