YOLO Vehicle Detector Based on BN Scaling Factor Pruning

Guangmin Sun[1,a], Zihao Zhang[1,b], Zixi Zhu[1,c], Shikui Guan[1,d], Yu Weng[1,e] and Chong Shi[1,f]

[1]Faculty of Information Technology, Beijing University of Technology, Beijing, China
a. gmsun@bjut.edu.cn, b. zihaozhang9@foxmail.com, c. 1577831099@qq.com
d. xiyangde1993@163.com, e. xxsybws@163.com, f. 953871440@qq.com

Keywords: Network pruning, vehicle detection, BN scaling factor, deep learning, YOLO.

Abstract: Deep learning makes object detection easy, but the network structure is complex and there is a lot of redundancy, which results in a very slow operation. This paper proposes a pruning method based on the BN scaling factor to prune the redundant structure of the network and apply it to vehicle detection tasks. First, train the network for channel sparsity to make it easier to screen out important channels. While training network weights, the L1-norm is applied to the scale factor and bias of BN. Structures with smaller BN layer parameter values are penalized. Secondly, the network is automatically pruned to obtain the minimum structure. The BN scaling factors in the network structure are sorted, and the structures with smaller scaling factors are pruned according to the proportion. Then, the bias of the pruned structure is transferred to subsequent layers to maintain accuracy. In this paper, the pruning strategy considers all convolutional layers, so a high pruning rate is obtained, and the model after pruning can still maintain high accuracy without any training. For vehicle data, after training and testing, the accuracy of the model after pruning only decreased from 50.4Map to 47.27Map, but the network structure reached 76% compression. GFLOPS decreased from 33.12 to 8.16, and the number of million parameters decreased from 61.95 to 6.95.

1. Introduction

With the development of urbanization and the increase in the number of motor vehicles, social problems such as traffic congestion and difficulty in parking have arisen. The construction of intelligent parking lots, and intelligent transportation systems can facilitate vehicle management. Vehicle detection is the first step in an intelligent transportation system, and it is necessary to quickly sense the presence of the vehicle and the location of the vehicle.

Traditional vehicle detection technology is divided into vehicle detection based on motion model[1] and vehicle detection methods based on feature[2]. The former segment moving targets from a continuous video sequence, such as using: the frame difference method[3], the Background Subtraction[4], the Gaussian mixture model[5], or the optical flow method[6]. The latter, in a single picture, first extracts the relevant features for the vehicle, then uses the machine learning classifier to classify the vehicle features, and uses sliding window technology to locate the vehicle position.

Low-level vehicle features include color, shading, edges, and more. High-level feature descriptors include SITF, SUFT, HOG[7], LBP[8], and Harr-like[9].

In recent years, deep learning has developed rapidly, and there are a large number of CNN-based object detection networks, which are roughly divided into two stages and single stages according to the detection steps. In the former, such as Fast-RCNN[10]、Faster-RCNN[11], are usually complex in design and require too much calculation. The latter, such as YOLOV3[12], SSD[13], they have good speed and high accuracy. However, there are still problems of too many redundant structures and more calculation operations, which makes it difficult to be practically applied to vehicle detection.

Parameter pruning[14] selects and removes unimportant structures in the network through certain rules to obtain a more compact model. OBD[15] reduces the number of connections by Hessian matrix. At present, methods for evaluating the importance of the network structure include: L1 norm of the convolution weights[16], BN layer scaling factor[17], geometric center[18] and so on. In these methods, the precision disappears after pruning, and it needs to be retrained to restore accuracy. But a lot of training consumes resources and wastes time.

This paper proposes a pruning method based on the BN scaling factor to compress the network structure. This method not only can effectively pruning the network, but also has almost no loss in accuracy after pruning. First, train the network for channel sparsity to make it easier to screen out important channels. While training network weights, the L1-norm is applied to the scale factor and bias of BN. Secondly, the network is automatically pruned to obtain the minimum structure of the network. The BN scaling factors in the network structure are sorted, and the structures with smaller scaling factors are pruned according to the proportion. Then, the bias of the pruned structure is adjusted to the affected layer to maintain accuracy.

In this paper, the pruning strategy for the YOLO network structure considers all convolutional layers, so a high pruning rate is obtained. For vehicle data, after training and testing, the accuracy of the model after pruning only decreased from 50.4Map to 47.27Map, but the network structure reached 76% compression. Gigabit floating point operations per second (GFLOPS) decreased from 33.12 to 8.16, and the number of million parameters decreased from 61.95 to 6.95.

2. Channel Sparsity Training

Both the backbone network and the detection head in the YOLOV3 detector contain a large number of convolution operations, among which there are many redundant structures. In this paper, an evaluation index based on the BN scaling factor is proposed for pruning the YOLOV3 network structure. Batch normalization is a widely used standard method for improving the fast convergence and generalization performance of CNNs. Assume that x^l is the input of layer l, x^{l+1}represents the output, B represents the current batch, and the operation of the BN layer is shown in Equation 1:

$$x^{l+1} = \gamma^l \cdot \frac{x^l - \mu_B^l}{\sqrt{{\sigma^l}_B^2 + \epsilon}} + \beta^l \tag{1}$$

where, μ_B^l and ${\sigma^l}_B^2$ are the mean and variance of the batch activation value. γ^l and β^l are the scale and displacement of the affine transformation parameters that can be trained. The scaling factor γ of the BN layer can show the importance of the convolution channel of the previous layer. If the value of γ is small, the output of the previous layer of the convolution channel is multiplied by γ to obtain a smaller value, which makes $\gamma^l \cdot \frac{x^l - \mu_B^l}{\sqrt{{\sigma^l}_B^2 + \epsilon}} \approx 0$. Therefore, in the training process of this paper, the regular term of γ is introduced, and γ is artificially constrained, so that the output of the unimportant convolution channel is close to zero. At the same time, use the same method to limit β, so that the BN layer outputs a smaller value to reduce the impact on the subsequent convolution layers. The loss function in this paper is shown in Equation 2.

$$L = \sum_{(x,y)} l(f(x,W),y) + \lambda \sum_{\gamma\in\Gamma} l_n(\gamma) + \lambda \sum_{\beta\in\Gamma} l_n(\beta) \quad (2)$$

Where x is the training input, y is the target, and W is the weight of the convolution kernel. In the equation, the first part is the normal loss of the weight of the trained network, $l_n(\cdot)$ is the penalty function for γ and β, and λ is the weight that keeps the two terms in balance. In practice, the $l_n(\cdot)$ function is L1 norm, as shown in Equation 3.

$$g(x) = |x| \quad (3)$$

In the training process of this paper, sparse regularization is added to the loss function of normal training convolution weights. The joint training network weights, BN layer scaling factors, and BN layer biases cause the convolution channel after training sparse and the weights shift to zero. In the pruning process, the channel with close to zero output is cut out, and it has no effect on the accuracy.

3. Network Structure Pruning

3.1.YOLOV3 Structure Pruning

After channel sparsity training is performed on YOLOV3, the scaling factors of the BN layers behind it are extracted for all convolutional layers to be cropped. The absolute values of the scale factor values of all channels are sorted, and the global threshold is selected according to the pruning ratio. In practice, the pruning ratio p is 0.8. As shown in Equation 4.

$$\gamma(i,j) = \begin{cases} \gamma(i,j) & \text{if } \gamma(i,j) > \text{thresh} \\ \text{remove} & \text{otherwise} \end{cases} \quad (4)$$

Where $\gamma(i,j)$ represents the scaling factor of the j-th channel of the i-th layer. In the YOLOV3 network structure, for general convolution channels, channels above the threshold are retained, and channels below the threshold are removed. In YOLOV3, except for the output layer, all convolutional layers are connected to the BN layer behind it, so the solution in this paper can prune all network structures. At the same time, there are a large number of deep and shallow layer fusion mechanisms in the YOLOV3 network structure. Careful design of the pruned convolution layer is required to ensure that the network structure after pruning is correctly connected.

The operation of the shortcut layer is to add the channel output results of the two convolutional layers, so the two convolutional layers after pruning should also have the same number of channels. And the position j of the convolution channel reserved by the two convolution layers should be the same. As shown in Figure 1, for the convolution layers involving the shortcut layers, a pruning mask of a channel is first obtained according to Equation 4, where 0 represents being pruned and 1 represents being reserved. Take or operate on multiple masks, and merge the new Mask to apply to the convolutional layers involving the shortcut layers.

The operation of the route layer is to concatenate the two convolution layers together, so the channels after pruning must also be connected together. As shown in Figure 2, for the convolutional layers involving the route layers, the pruning mask is obtained according to Equation 4, and then multiple masks are connected.

For the output layers in YOLOV3, the number of output channels cannot be reduced, and no operation is performed in this paper. Therefore, this paper considers the pruning operation of all convolutional layers of YOLOV3, so a high pruning rate can be obtained.

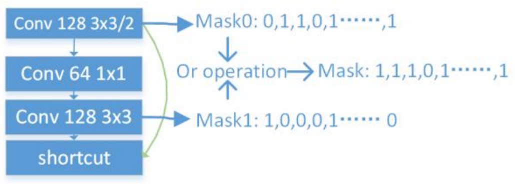

Figure 1: Pruning shortcut layer.

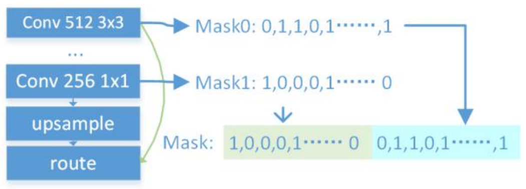

Figure 2: Pruning route layer.

3.2.Transfer Bias

In the YOLOV3 network, most convolutional layers are followed by a BN layer and an activation function, and these convolutional layers do not use bias. For convenience, these operations are combined into one equation.

$$x^{l+1} = Act^{l}\left(\gamma^{l} \cdot BN_{\mu^{l},\sigma^{l},}\left(w^{l} * x^{l}\right) + \beta^{l}\right) \qquad (5)$$

where Act^{l} represents the activation function of the layer l,which is generally the ReLU function in YOLOV3. When pruning the l layer channel, it is equivalent to zeroing γ of some channels, and $\gamma^{l} \cdot BN_{\mu^{l},\sigma^{l},}\left(w^{l} * x^{l}\right)$ is zero. But the β of the layer l will still affect the subsequent convolutional layers. Although we constrained the value of β in the channel sparsity training, even small changes in the value of the detection network will affect the accuracy[19]. Pruning the β of layer l will change the mean value μ^{l+1} of layer l + 1. This paper uses Equation 6 to update this change.

$$\mu^{l+1'} = \mu^{l+1} - I\,(\gamma = 0) \cdot Act^{l}\,(\beta)^{T}\ sum\left(w^{l+1}_{:,:,:,:}\right) \qquad (6)$$

Among them, $I\,(\gamma = 0)$ indicates that the cropped channel is selected, and $sum\left(w^{l+1}_{:,:,:,:}\right)$ indicates the sum of the convolution kernels of the layer l + 1. Update the output of the layer l + 1 to:

$$x^{l+2} \approx \gamma^{l} \cdot BN_{\mu^{l+1'},\sigma^{l+1},}\left(w^{l+1} * x^{l+1}\right) + \beta^{l+1} \qquad (7)$$

For the output layer of YOLOV3, no BN layer is used, but the convolution layer uses a bias. Equation 8 is used to represent this operation.

$$x^{l+2} = w^{l+1} * x^{l+1} + b^{l+1} \tag{8}$$

When layer l is pruned, β of layer l is updated to the bias of layer l + 1.

$$b^{l+1'} = b^{l+1} - I(\gamma = 0) \cdot Act^{l}(\beta)^{T} \ sum\left(w^{l+1}_{:,:,:,\cdot}\right) \tag{9}$$

So the output of the l + 1 layer is :

$$x^{l+2} \approx w^{l+1} * x^{l+1} + b^{l+1'} \tag{10}$$

In this paper, the impact of the loss of all parameters on subsequent convolutional layers is considered when pruning, so the accuracy is maintained after pruning.

4. Experiments

4.1.Data

In this paper, the vehicle data is selected from the UA-DETRAC[20] public data set. The UA-DETRAC dataset is very complex, including weather changes, occlusion, posture, angle, and size changes. The overview of the dataset is shown in Figure 3.

Training the entire dataset requires a lot of resources. This paper selects some samples to express the dataset. Select from the original training data, MVI_20011, MVI_20012, MVI_20032, MVI_20033, MVI_20034 video sequences, a total of 3621 pictures, as the training dataset. Video sequence: MVI_20051, MVI_20052, MVI_20061, MVI_20062, MVI_20063, a total of 4000 pictures, as a test dataset.

Figure 3: Sample UA-DETRAC dataset.

4.2.Training Skills

The computer hardware environment of the training experiment includes 1080TI graphics card, 8G memory and I5 CPU. The training software environment includes: CUDA10.0, cudnn7.6.5, pytorch1.1.0. Our experiment trains 100 generations, using the SGD optimizer, and setting the momentum to 0.97. In the first 6 generations, we used warmup[21] to adjust the learning rate, and then set the learning rate to 0.002324. The learning rate is attenuated when the 70th and 90th generations, and the decay rate is 0.1.

Use data augmentation on the original data during training to prevent overfitting of the network and improve training accuracy, including: 0.5 probability mirror flip, random rotation of plus or minus 5 degrees, random translation of about 5%, plus or minus 5% scale Transform. At the same time, this paper randomly performs rectangular mosaic stitching, randomly selects 4 pictures from the training data, randomly selects different positions of images from each picture, and then stitches them into one image. The result is shown in Figure 4.

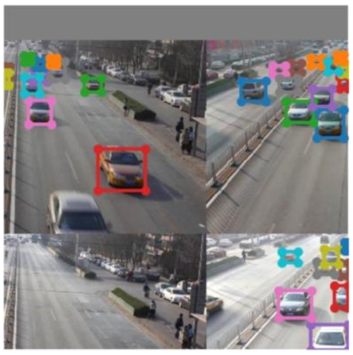

Figure 4: Example of data enhancement results.

4.3.Experimental Results

In this paper, two sets of controlled trials, normal YOLOV3 and pruned YOLOV3, were established. For pruning YOLOV3, channel sparsity training is required first, followed by structural pruning. At the same time, the model after pruning was fine-tuned for comparison.

The evaluation results of the pruning of the YOLOV3 detector are shown in Table 1. Compared with the normal training results, the channel sparsity training resulted in a 5.89% accuracy decrease, and the pruning resulted in a 6.21% accuracy decrease. After fine-tuning the accuracy increased to the original 95.93%. After pruning, the amount of network calculation and parameters has dropped significantly, about 24.78% and 11.15% of YOLOV3 structure. After comparison, channel sparsity training caused a decrease in accuracy, pruning has almost no effect on accuracy, and the fine-tuned network can recover part of the accuracy.

In Table 2, CBN represents the convolution layer and BN layer, K1 represents a kernel size of 1 and S2 represents a stride of 2. It can be analyzed from this: 1) The proportion of pruning off the front and back layers in the network is almost the same. 2) The structure of 1x1 convolution kernel has more pruning than 3x3 convolution kernel. 3) The ratio of pruning is higher for layers with more convolution kernels. These conclusions after automatic pruning may help to manually design the structure in the future.

Table 1: YOLOV3 detector pruning evaluation results.

Network structure	mAP	Flops(G)	Params(M)
Normal training	50.4	33.1	61.9
Channel parsity training	47.43 (94.11%)	-	-
Model after pruning	47.27 (93.79%)	8.2 (24.78%)	6.9 (11.15%)
Fine-tune pruning model	48.35 (95.93%)	-	-

Table 2: YOLOV3 Structure comparison table before and after pruning.

YOLOV3	YOLOV3_pruning
CBN 32 K3 CBN 64 K3 S2	CBN 25 K3 CBN 41 K3 S2
1x{CBN 32 K1 CBN 64 K3 Shortcut} CBN 128 K3 S2 2x{CBN 64 K1 CBN 128 K3 Shortcut} Conv 256 K3 S2	1x{CBN 16 K1 CBN 41 K3 Shortcut} CBN 114 K3 S2 1x{CBN 14 K1 CBN 114 K3 Shortcut} 1x{CBN 34 K1 CBN 114 K3 Shortcut} CBN 250 K3 S2
…	
Yolo out3	
3x{CBN 128 K1 CBN 256 K3}	CBN 84 K1 CBN 111 K3 CBN 55 K1 CBN 86 K3 CBN 49 K1 CBN 108 K3
Yolo out1	

5. Conclusions

1) Aiming at the problem of a large amount of redundancy in the structure of the deep learning object detector, this paper proposes a pruning method based on the BN scaling factor to reduce the model structure. First, channel sparsity training is applied to the network. The training process simultaneously trains the network weights and BN layer scaling factor and bias. Sort the numerical values of the scaling factors in the network structure, and determine the pruning threshold according to the pruning ratio. Pruning the network structure according to the threshold.

2) Aiming at the phenomenon that the precision of the original pruning strategy disappears after pruning, this paper uses the method of parameter transfer to retain the parameters that are pruned. When there is a BN layer behind the pruned convolution layer, the pruned BN layer is biased and superimposed on the mean value of the subsequent BN layer. When there is no BN layer behind the pruned convolution layer, the bias of the pruned BN layer is superimposed on the bias of the subsequent convolution layer. According to the experiments, the pruning strategy in this paper causes almost no loss of accuracy after pruning.

3) Aiming at the complex network structure of the YOLOV3 network, the pruning strategy in this paper considers the pruning of all convolutional layers. Use the pruning mask to merge the shortcut layers. Use the pruning mask connection the route layer. By experimenting with the pruning method in this paper, a high pruning rate can be obtained. The pruning rate of the YOLOV3 network structure reached 76%, which greatly reduced the parameters and calculations, but the accuracy remained almost the same as the original. Smaller models can be applied in practice.

Acknowledgments

This work was supported by the National Key R&D Program of China (2018YFF01012300) and National Natural Science Foundation of China (11527801, 41706201).

References

[1] Alomari Saleh, Refai Mohammed, Abu Karaki, Hussam. (2019). A Comprehensive Survey of the Vehicle Motion Detection and Tracking Methods for Aerial Surveillance Videos.. 19. 93-106.

[2] Alex, David, S, et al. Vision-based Vehicle Detection Survey[J]. International Journal of Recent Contributions from Engineering, 2016.

[3] Lin Y, Fang M, Shihong D. An Object Reconstruction Algorithm for Moving Vehicle Detection Based on Three-Frame Differencing[C] Ubiquitous Intelligence & Computing & IEEE Intl Conf on Autonomic & Trusted Computing & IEEE Intl Conf on Scalable Computing & Communications & Its Associated Workshops. IEEE, 2016.

[4] Elgammal A, Harwood D, Davis L. Non-parametric Model for Background Subtraction[C] European Conference on Computer Vision. Springer, Berlin, Heidelberg, 2000.

[5] Stauffer C, Grimson W E L. Adaptive Background Mixture Models for Real-Time Tracking[C] cvpr. IEEE Computer Society, 1999.

[6] Bhaskar P K, Yong S P, Jung L T. Enhanced and effective parallel optical flow method for vehicle detection and tracking[C] 2015 International Symposium on Mathematical Sciences and Computing Research (iSMSC). IEEE, 2016.

[7] Dalal N, Triggs B. Histograms of Oriented Gradients for Human Detection[C]. IEEE Computer Society, 2005.

[8] Liao S, Zhu X, Lei Z, et al. Learning Multi-scale Block Local Binary Patterns for Face Recognition[M] Advances in Biometrics. DBLP, 2007.

[9] Viola P, Jones M. Rapid Object Detection using a Boosted Cascade of Simple Features[C]. IEEE Computer Society, 2001.

[10] Girshick R. Fast R-CNN[J]. Computer Science, 2015.

[11] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015, 39(6):1137-1149.

[12] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J]. 2018.

[13] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[J]. 2015.

[14] Cheng Y, Wang D, Zhou P, et al. A Survey of Model Compression and Acceleration for Deep Neural Networks[J]. 2017.

[15] Yann Le Cun, John S. Denker, Sara A. Solla. Optimal Brain Damage[J]. Advances in neural information processing systems, 2000, 2.

[16] Li, Hao, Kadav, Asim, Durdanovic, Igor.. Pruning Filters for Efficient ConvNets[J].

[17] Liu Z, Li J, Shen Z, et al. Learning Efficient Convolutional Networks through Network Slimming[J]. 2017:2755-2763.

[18] He Y, Liu P, Wang Z, et al. Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration[J]. 2018.

[19] Ye J, Lu X, Lin Z, et al. Rethinking the Smaller-Norm-Less-Informative Assumption in Channel Pruning of Convolution Layers[J]. 2018.

[20] Lyu S, Chang M C, Carcagni P, et al. UA-DETRAC 2017: Report of AVSS2017 & IWT4S Challenge on Advanced Traffic Monitoring[C] 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). IEEE Computer Society, 2017.

[21] Goyal P, Dollár, Piotr, Girshick R, et al. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour[J]. 2017.