


RESEARCH

Open Access



An algorithm for highway vehicle detection based on convolutional neural network

Linkai Chen^{1,2}, Feiyue Ye², Yaduan Ruan^{1*} , Honghui Fan² and Qimei Chen¹

Abstract

In this paper, we present an efficient and effective framework for vehicle detection and classification from traffic surveillance cameras. First, we cluster the vehicle scales and aspect ratio in the vehicle datasets. Then, we use convolution neural network (CNN) to detect a vehicle. We utilize feature fusion techniques to concatenate high-level features and low-level features and detect different sizes of vehicles on different features. In order to improve speed, we naturally adopt fully convolution architecture instead of fully connection (FC) layers. Furthermore, recent complementary advances such as batch-norm, hard example mining, and inception have been adopted. Extensive experiments on JiangSuHighway Dataset (JSHD) demonstrate the competitive performance of our method. Our framework obtains a significant improvement over the Faster R-CNN by 6.5% mean average precision (mAP). With 1.5G GPU memory at test phase, the speed of the network is 15 FPS, three times faster than the Faster R-CNN.

Keywords: Vehicle detection, Convolution neural network, *k*-means, Feature concatenate

1 Introduction

Vehicle detection is a very important component in traffic surveillance and automatic driving [1]. The traditional vehicle detection algorithms such as Gaussian mixed model (GMM) [2] has achieved promising achievements. But it is not ideal due to illumination changes, background clutter, occlusion, etc. Vehicle detection is still an important challenge in computer vision.

With the revival of DNN [3], object detection has achieved significant advances in recent years. Current top deep-network-based object detection frameworks can be divided into two categories: the two-stage approach, including [4–8], and one-stage approach, including [9–11]. In the two-stage approach, a sparse set candidate object boxes is first generated by selective search or region proposal network, and then, they are classified and regressed. In the one-stage approach, the network straightforward generated dense samples over locations, scales, and aspect ratios; at the same time, these samples will be classified and regressed. The main advantage of one-stage is real time; however, its detection accuracy is usually behind the two-stage, and one of the main reasons is class imbalance problem [12].

In the two-stage, Region-based Convolutional Network method (R-CNN) is the pioneer of deep-network-based object detection. R-CNN utilizes selective search to generate 2000 candidate boxes; each candidate box is to be warped into fixed size and as an input image of CNN, so 2000 candidate boxes will be computer 2000 times. It has too low efficiency. In order to reduce computer, Fast R-CNN [5] generates candidate boxes on the last layer feature map and adopts Rol pooling.

Under Fast R-CNN pipeline, Faster R-CNN [4] shares full-image convolutional feature with the detection network to enable nearly cost-free region proposals. The aforementioned approaches adopt fully connection layers to classify object. It is time-consuming and space-consuming both in training and inference time. R-FCN [8] uses fully convolution and adding position-sensitive score maps. Nevertheless, R-FCN still needs region proposals generated from region proposal network.

The aforementioned methods are general object detection methods. However, vehicle detection is special detection. If we straightforwardly use general object detection algorithms to detect vehicles, the effect is not the best. The main reasons are the following three aspects: (1) Faster R-CNN and Single Shot MultiBox Detector (SSD) using aspect ratios are [0.5, 1, 2], but the aspect ratio range of vehicles is not so big. (2) In Faster R-CNN and

* Correspondence: e13685231980@126.com

¹School of Electronic Science and Engineering, Nanjing University, Xianlin Road No.163, Nanjing 210023, China

Full list of author information is available at the end of the article

SSD extract candidate regions on high-level feature map, the high-level feature has more semantic information, but cannot locate well. (3) Vehicle detection requires high real-time, but Faster R-CNN adopts FC layers. It takes about 0.2 s per image for VGG16 [13] network.

Aimed to the general object detection methods, there exist problems. We present an efficient and effective framework for vehicle detection and classification from traffic surveillance cameras. This method fuses the advantages of two-stage approach and one-stage approach. Meanwhile, we use some tricks such as hard example mining [14], data augmentation, and inception [15]. The main contributions of our work are summarized as follows:

- 1) We use k -means algorithm to cluster the vehicle scales and aspect ratios in the vehicle datasets. This process can improve 1.6% mean average precision (mAP).
- 2) We detect vehicles on different feature map according to different size vehicles.
- 3) We fuse the low-level and high-level feature map, so the low-level feature map has more semantic information.

Our detector is time and resource efficient. We evaluate our framework on JiangSuHighway Dataset (JSHD) (Fig. 1) and obtain a significant improvement over the state-of-the-art Faster R-CNN by 6.5% mAP. Furthermore, our framework achieves 15 FPS on a NVIDIA TITAN XP, three times faster than the seminal Faster R-CNN.

2 Related work

In this section, we give a brief introduction of vehicle detection in traffic surveillance cameras. Vision-based vehicle detection algorithms can be divided into three categories: motion-based approaches, hand-crafted feature-based approaches, and CNN-based approaches.

Motion-based approaches include frame subtraction, optical flow, and background subtraction. Frame subtraction computes the differences of two or three consecutive

frames sequences to detect the motion object. Frame subtraction is characterized by simple calculation and adapting dynamic background, but it is not ideal for motion that is too fast or too slow. Optical flow [16] calculates the motion vector of each pixel and tracks these pixels, but this approach is complex and time-consuming. Background subtraction such as GMM are widely used in vehicle detection by modeling the distribution of the background and foreground [2]. However, these approaches cannot classify and detect still vehicles.

Hand-crafted feature-based approaches include Histogram of Oriented

Gradients (HOG) [17], SIFT [18], and Harr-like. Before the success of CNN-based approaches, hand-crafted feature approaches such as deformable part-based model (DPM) [19] have achieved the state-of-art performance. DPM explores improved HOG feature to describe each part of vehicle and followed by classifiers like SVM and Adaboost. However, hand-crafted feature approaches have low feature representation.

CNN-based approaches have shown rich representation power and achieved promising results [4–6, 9, 11]. R-CNN uses object proposal generated by selective search [20] to train CNN for detection tasks. Under the R-CNN framework, SPP-Net [7] and Fast R-CNN [5] speed up through generating region proposal on feature map; these approaches only need computer once. Faster R-CNN [4] uses region proposal network instead of selective search, then it can train end to end and the speed and accuracy also improve. R-FCN [8] tries to reduce the computation time with position-sensitive score maps. Considering the high efficiency, the one-stage approach attracts much more attention recently. YOLO [9] uses a single feed-forward convolutional network to directly predict object classes and locations, which is extremely fast. SSD [11] extracts anchors of different aspect ratios and scales on multiple feature maps. It can obtain competitive detection results and higher speed. For example, the speed of SSD is 58FPS on a NVIDIA TITAN X for 300×300 input, nine times faster than Faster R-CNN.

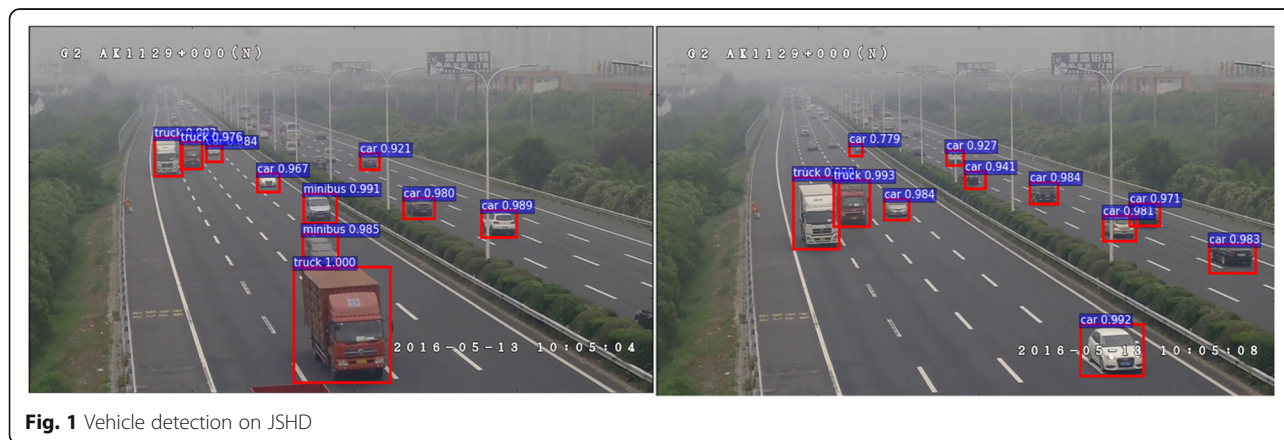


Fig. 1 Vehicle detection on JSHD

3 Methods

This section describes our object detection framework (Fig. 2). We first introduce k -means algorithm to prepare data in Section 3.1. Then, in Section 3.2, we present feature concatenate to fuse high-level and low-level feature map. Next, we explain how to generate candidate anchor boxes on different feature map in Section 3.3. In Section 3.4, we discuss how to detect different size vehicles on different feature map. Finally, we introduce batch-norm, hard example, and inception; these tricks can improve the result.

3.1 Bounding box clustering

The traditional object detection algorithms use sliding window to generate candidate proposal, but these methods are time-consuming. In CNN-based detectors such as Faster R-CNN and SSD use aspect ratios [0.5, 1, 2], so the candidate proposals are less than sliding window. But there are two issues in this way. The first issue is that the aspect ratios are hand-picked. If we pick better priors of dataset, it will be easier for the network to predict good detections. The second issue is that the aspect ratios are designed for general object detection such as PASCAL VOC [21] and COCO [22] dataset. It is not very suitable for vehicle detection. In order to solve these issues, we run k -means clustering on our dataset instead of choosing aspect ratios by hand. The cluster centroids are significantly different than hand-picked anchor boxes. They are suitable for vehicle detection. The k -means algorithm can be formulated as:

$$E = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|_2^2 \tag{1}$$

where x is the sample, μ_i is the average vector of C_i , and k is the center of clustering. We run k -means by various k on vehicle sizes and aspect ratios (see Fig. 2). We choose $k = 5$ for vehicle weight and height and $k = 3$ for aspect ratios as a good trade-off between accuracy

and speed. It can improve 1.6% mAP on our vehicle dataset.

3.2 Baseline network

We use VGG-16 as the baseline network, which is pre-trained with ImageNet [23] dataset. It has 13 convolutional layer and three fully connected layers. In order to improve detection speed and reduce the parameters, we use convolutional layer instead of the last three fully connected layers. It has been proved to be effective in paper [8].

3.3 Feature concatenate

Previous work on Faster R-CNN only uses the last feature map to general candidate proposal, and it is not good enough for vehicle detection, because the vehicle scale change is larger. It is beneficial to concatenate the high-level and low-level feature [24]. The high-level feature layers have more semantic information for object classification but lack insight to precise object localization. However, the low-level feature layers have a better scope to localizing objects as they are closer to raw image. In [24], objects detect on a single concatenate feature map, and it is not accurate enough for multi-scale. In order to detect on multi-layers, we adopt feature pyramid in our network, as shown in Fig. 3. Feature pyramid can enrich the feature presentation and detect objects on different feature layers. This way is suitable for multi-scale; we can detect different size vehicles on different feature layers. As shown in Fig. 3, Firstly, a deconvolutional layer is applied to the last feature map (conv7), and a convolutional layer is grafted on backbone layer of conv6 to guarantee that the inputs have the same dimension. Then, the two corresponding feature maps are merged by element-wise addition. In our network, the last layer feature map is 5×5 , after deconvolution is 10×10 . The size is the same as conv6 feature map. We use four convolutional layers (conv4–7) to generate four different size feature pyramid layers. So we can detect different

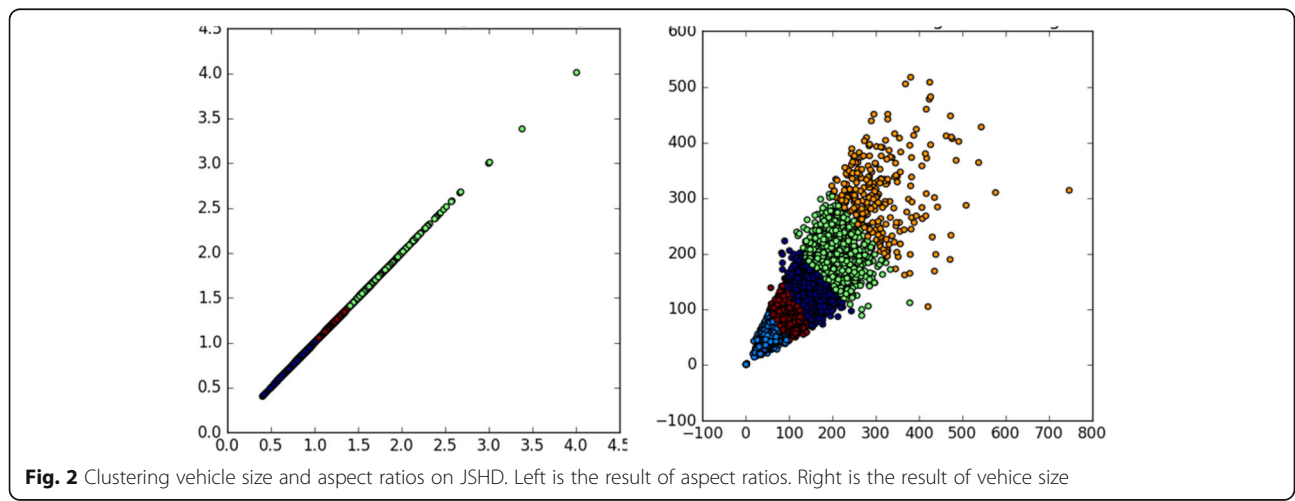


Fig. 2 Clustering vehicle size and aspect ratios on JSHD. Left is the result of aspect ratios. Right is the result of vehicle size

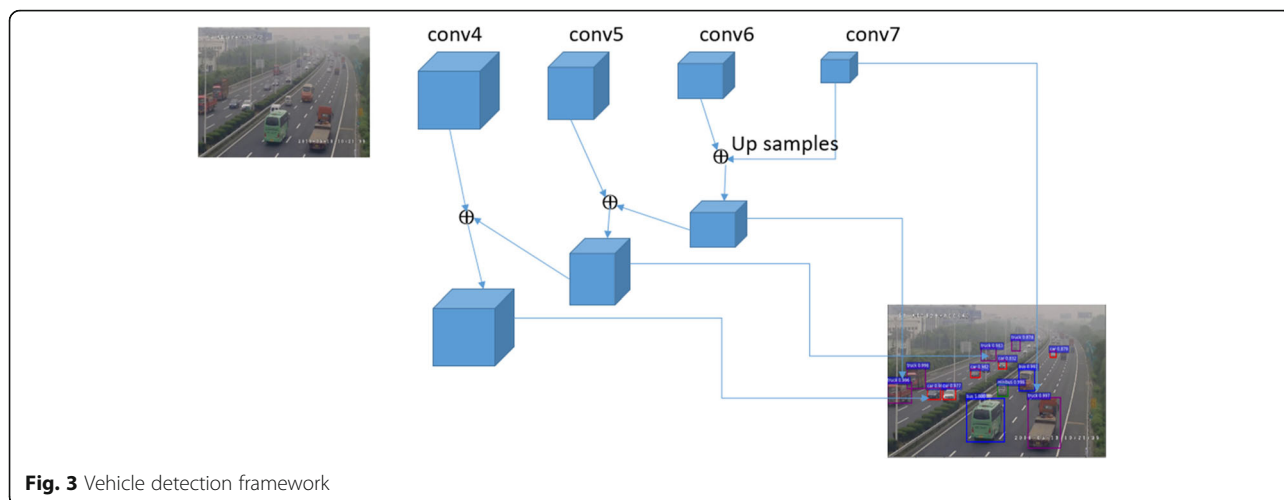


Fig. 3 Vehicle detection framework

size vehicles on different size feature pyramid layers. And this way can improve detection accuracy.

3.4 Reference boxes

In order to detect different size vehicles on different feature map, we generate candidate proposals on different feature map. As we all know, different feature maps have different receptive field sizes. Low-level feature map has smaller receptive field sizes compared with high-level feature map. We run *k*-means on JSHD to get five vehicle sizes and three aspect ratios. The width and height of each box are computed with respect to the aspect ratio. In total, there are two scales and three aspect ratios at each feature map location. Finally, we combine them together and use non-maximum suppression (NMS) to refine the results.

3.5 Inception

We use some new technology to improve detection results including inception and batch normalization. We employ the inception module on our network. In this paper, we just use the most simple structure inceptionV1, as shown in Fig. 4. The inception module can improve the feature presentation and detection accuracy. Batch normalization leads to significant improvements in convergence while training. By adding batch normalization on all feature layers in our network, we obtain more than 1.9% improvement in mAP. Batch normalization also helps regularize the model.

4 Training and testing

In this section, we introduce the details of network training and testing, including data augmentation, hard example mining loss function, and parameter selection.

4.1 Data augmentation

We are lack of labeled data, because labeling data is expensive. In order to avoid overfitting while training network,

we adopt data augmentation. We mainly use two data augmentation methods to construct a robust model to adapt a variation vehicle. One is using flipping input image. The second is randomly sampling a patch whose edge length is {0.5, 0.6, 0.7, 0.8, 0.9} of the original image and at least one vehicle’s center is within this patch. Please refer to [11] for more details.

4.2 Hard negative mining

After the matching step, most of the default boxes are negatives; similar to [11], we use hard negative mining to mitigate the extreme class imbalance. At each mini-batch, we make the ration between negative and positive below 1:3, instead of using all negative anchors in training.

4.3 Loss function

We use a multi-task loss function to jointly train our network end to end. The loss function consists of two parts, the loss of classification and bounding box regression. We denote the loss of bounding box regression

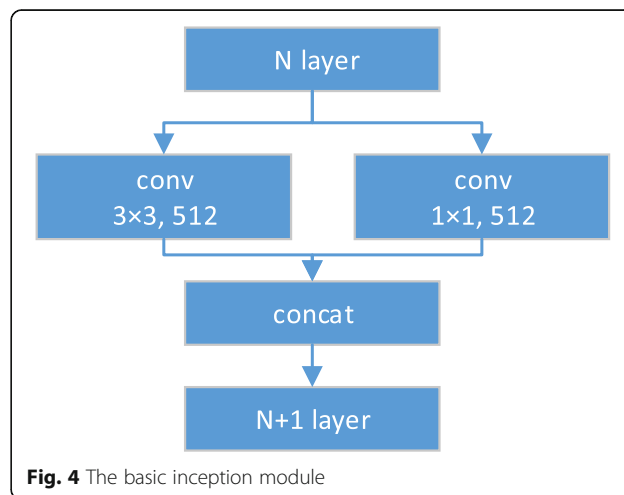


Fig. 4 The basic inception module

with L_{loc} . It optimizes the smoothed L_1 loss between the predicted bounding box locations $t = (t_x, t_y, t_w, t_h)$ and target offsets $t^* = (t_x^*, t_y^*, t_w^*, t_h^*)$. We adopt the parameterizations of the four coordinates as follows (2):

$$\begin{aligned} t_x &= (x-x_a)/w_a, t_y = (y-y_a)/h_a, \\ t_w &= \log(w/w_a), t_h = \log(h/h_a), \\ t_x^* &= (x^*-x_a)/w_a, t_y^* = (y^*-y_a)/h_a, \\ t_w^* &= \log(w^*/w_a), t_h^* = \log(h^*/h_a), \end{aligned} \tag{2}$$

where $x, y, w,$ and h denote the box's center coordinates and its width and height. Variables $x, x_a,$ and x^* are for the predicted box, anchor box, and ground-truth box respectively (likewise for y, w, h).

The loss of classification is denoted as L_{cls} , which is computed by a Softmax over $K + 1$ outputs for each anchors.

Therefore, our loss function is defined as follows (3):

$$L = \frac{1}{N_{cls}}L_{cls} + \lambda \frac{1}{N_{loc}}L_{loc} \tag{3}$$

where λ balances the two parts. We consider the two parts equally import, and $\lambda = 1$ is set to computer the multi-stage loss.

5 Experiments, results, and discussion

In order to evaluate the effectiveness of our network. We conduct experiments on our vehicle dataset (JSHD). The experiments are implemented on Ubuntu 16.04 with a GPUs (NVIDIA TITAN XP) and i7 7700 CPU.

5.1 Dataset

There are large number of various types of vehicles in highway surveillance video. And it is suitable for traffic video analysis because of the large and long view of the road. So we construct a new dataset from 25 videos of JiangSu highway, Jiangsu province, which we called JSHD, as shown in Figs. 1 and 5. The dataset contains 5000 frames which are captured from the videos. The

Table 1 Results on the JSHD

Models	Overall (mAP)	Car	Bus	Minibus	Truck
Fast R-CNN	67.2	53.6	83.2	62.5	69.5
Faster R-CNN	69.2	55.2	85.4	64.7	71.3
YOLO	58.9	46.6	75.2	53.4	60.5
SSD300	68.8	54.4	85.1	64.3	71.5
SSD512	71.2	57.4	87.2	66.8	73.4
RON320 [25]	73.6	60.2	89.5	69.1	75.5
Ours	75.7	62.4	91.8	71.3	77.3

vehicle is classified into four categories (bus, minibus, car, truck). We do not care for the vehicles that are too small. Specifically, the vehicle whose height is less than 20 pixels will be ignored. We use random 3000 frames to train our network and 2000 frames to test.

5.2 Optimization

During training phase, stochastic gradient descent (SGD) is used to optimize our network. We initialize the parameters for all the newly added layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. We set the learning rate as 0.001 for the first 60k iterations and 0.0001 for the next 60k iterations. The batch size is 16 for a 320×320 model. We use a momentum of 0.9 and a weight decay of 0.005.

5.3 Results

To evaluate the effectiveness of our proposed network, we compare our proposed network to the state-of-the-art detectors on JSHD. Table 1 shows the results of our experiment. It is obvious that our network outperforms the other algorithms. We achieve a significant overall improvement of 6.5% mAP over the state-of-the-art Faster R-CNN and 6.9% over SSD300. It is clear that the precision of the car is lower than that of the bus and truck because the deep learning detection

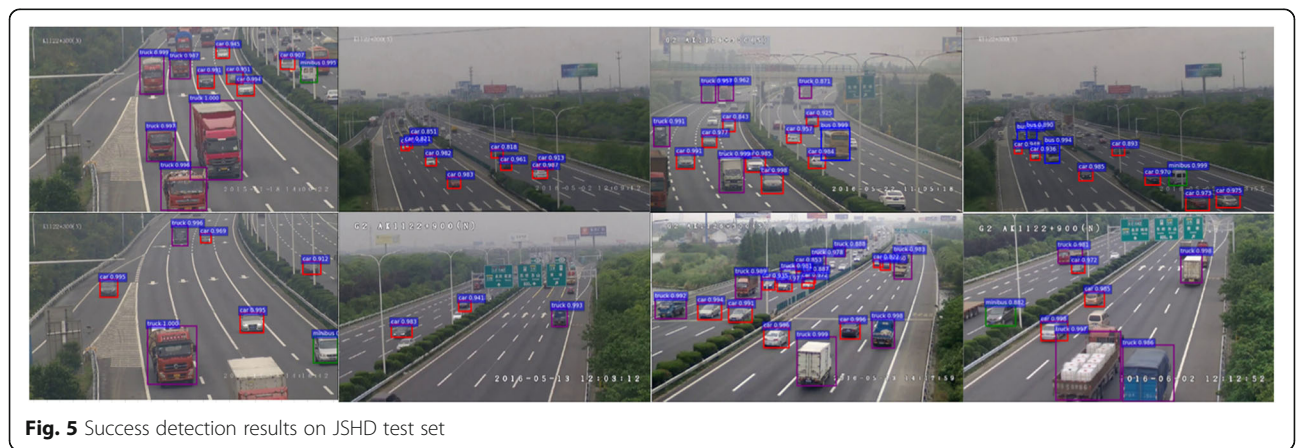


Fig. 5 Success detection results on JSHD test set

Table 2 Module ablation analysis

	Faster R-CNN			Ours	
<i>k</i> -means?	✓	✓	✓	✓	✓
Feature concatenate?		✓	✓	✓	✓
Detecting on different feature?			✓	✓	
Batch normalization?					✓
JSHD (mAP)	69.2	70.8	72.8	73.8	75.7

algorithms is not friendly to small object. Our network shows a strong ability on detecting vehicles with a large variance of scales, especially for small vehicles. In testing, the speed of our network is 15 FPS, three times faster than the Faster R-CNN. It can be applied to real-time intelligent transportation systems. In summary, our network achieves the best trade-off between accuracy and speed.

5.4 Discussion

Our network baseline is Faster R-CNN and SSD. We improve the baseline with *k*-means, feature concatenate, and detecting on different features to enhance detection. The analysis results have been shown in Table 2. We can see that the feature concatenate module is important for detection.

Figure 5 demonstrates qualitative evaluations for our approach on the test set. We succeed in detecting most of the vehicles in different appearances, different scales, and heavy occlusion.

6 Conclusions

In this paper, we present an improved convolutional network for fast and accurate highway vehicle detection. We use *k*-means to cluster dataset and learn prior information. We use feature concatenate to extract more rich features. In order to detect different sizes of vehicles, we detect on different features. With these technology application, our framework obtains a significant improvement over Faster R-CNN and SSD, especially small vehicles. Furthermore, we will do challenging research in urban with occlusion and complex scene.

Abbreviations

CNN: Convolution neural network; FC: Fully connection; GMM: Gaussian mixed model; JSHD: JiangSuHighway Dataset; NMS: Non-maximum suppression; SGD: Stochastic gradient descent

Acknowledgements

Thanks for the editor and reviewers.

Funding

This research was supported by the National Natural Science Foundation of China (Grant Nos. 61472166 and 61502226), Natural Science Foundation of Changzhou (CE20175026), Qing Lan Project of Jiangsu Province.

Availability of data and materials

Data will not be shared; the reason for not sharing the data and materials is that the work submitted for review is not completed. The research is still

ongoing, and those data and materials are still required by the author and co-authors for further investigations.

Authors' contributions

LC and YR designed the research. FY, HF, and QC analyzed the data. LC wrote and edited the manuscript. All authors read and approved the final manuscript.

Ethics approval and consent to participate

We approved.

Consent for publication

We agreed.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details

¹School of Electronic Science and Engineering, Nanjing University, Xianlin Road No.163, Nanjing 210023, China. ²School of Computer and Engineering, Jiangsu University of Technology, Zhongwu Road No.1801, Changzhou 213001, China.

Received: 28 July 2018 Accepted: 28 September 2018

Published online: 24 October 2018

References

- X. Hu et al., "SINet: A Scale-insensitive Convolutional Neural Network for Fast Vehicle Detection," *arXiv*, vol 1 (2018), pp. 1–10
- C. Stauffer and W. E. L. Grimson, Adaptive background mixture models for real-time tracking, in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on*, Fort Collins, 1999, p. 252 Vol. 2
- G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets. *Neural Comput.* **18**(7), 1527–1554 (2006)
- S. Ren, K. He, R. Girshick, J. Sun, *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks* (Nips, Montréal, 2015)
- R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision, 2015*, vol. 2015. 1440–1448 Inter, pp
- R. Girshick, J. Donahue, T. Darrell, U.C. Berkeley, J. Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation* (2014), pp. 2–9
- K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(9), 1904–1916 (2015)
- J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection Via Region-Based Fully Convolutional Networks," 2016
- J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2015
- J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2016
- W. Liu, D. Anguelov, D. Erhan, and C. Szegedy, "SSD: Single Shot MultiBox Detector," no. 1 (2016)
- T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," 2017
- K. Simonyan, A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition* (2014), pp. 1–14
- A. Shrivastava, *Training Region-based Object Detectors with Online Hard Example Mining* (Cvpr, Las Vegas, 2016)
- C. Szegedy et al., *Going Deeper with Convolutions* (2014), pp. 1–9
- Z. Sun, G. Bebis, R. Miller, in *The International IEEE Conference on Intelligent Transportation Systems, 2004. Proceedings*. On-road vehicle detection using optical sensors: a review (2004), pp. 585–590
- N. Dalal, B. Triggs, in *IEEE Computer Society Conference on Computer Vision & Pattern Recognition*. Histograms of oriented gradients for human detection (2005), pp. 886–893
- D.G. Lowe, *Distinctive Image Features from Scale-Invariant Keypoints* *International Journal of Computer Vision*, **60**(2), 91–110 (2014)

19. P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan, Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **47**(2), 6–7 (2014)
20. J.R.R. Uijlings, K.E.A. Van De Sande, T. Gevers, A.W.M. Smeulders, Selective search for object recognition. *Int. J. Comput. Vis.* **104**(2) (2013)
21. L. Wen et al., in *arXiv*. DETRAC: A new benchmark and protocol for multi-object tracking (2015)
22. T.-Y. Lin et al., in *Computer Vision -- ECCV*. Microsoft COCO: common objects in context, vol 2014 (2014), pp. 740–755
23. O. Russakovsky et al., ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015)
24. T. Kong, A. Yao, Y. Chen, and F. Sun, “HyperNet: Towards Accurate Region Proposal Generation and Joint Object Detection,” 2016
25. T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, “RON: Reverse Connection with Objectness Prior Networks for Object Detection,” p. 2017, 2017

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)
