A method based on Multi-Convolution layers Joint and Generative Adversarial Networks for Vehicle Detection

Guang Han, Jinpeng Su and Chengwei Zhang
Engineering Research Center of Wideband Wireless Communication Technique, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[e-mail: hanguang8848@njupt.edu.cn]
*Corresponding author: Guang han

Received July 2, 2018; revised September 21, 2018; accepted October 28, 2018; published April 30, 2019

Abstract

In order to achieve rapid and accurate detection of vehicle objects in complex traffic conditions, we propose a novel vehicle detection method. Firstly, more contextual and small-object vehicle information can be obtained by our Joint Feature Network (JFN). Secondly, our Evolved Region Proposal Network (EPRN) generates initial anchor boxes by adding an improved version of the region proposal network in this network, and at the same time filters out a large number of false vehicle boxes by soft-Non Maximum Suppression (NMS). Then, our Mask Network (MaskN) generates an example that includes the vehicle occlusion, the generator and discriminator can learn from each other in order to further improve the vehicle object detection capability. Finally, these candidate vehicle detection boxes are optimized to obtain the final vehicle detection boxes by the Fine-Tuning Network(FTN). Through the evaluation experiment on the DETRAC benchmark dataset, we find that in terms of mAP, our method exceeds Faster-RCNN by 11.15%, YOLO by 11.88%, and EB by 1.64%. Besides, our algorithm also has achieved top2 comaring with MS-CNN, YOLO-v3, RefineNet, RetinaNet, Faster-rcnn, DSSD and YOLO-v2 of vehicle category in KITTI dataset.

Keywords: Vehicle detection, non-maximum suppression, generative adversarial networks, joint feature map, mask occlusion

1. Introduction

Vehicle detection based on video is especially important in a variety of computer vision applications, especially in traffic monitoring [1]. Traditional computer vision methods have made great contributions to the practical application of vehicle detection. However, because of vehicles' own diversity including the occlusion, deformation and the impact of extreme weather changing, how to detect road vehicle with high accuracy and real-time, it is still a difficult problem.

Traditional object detection methods can be divided into frame subtraction [2], adaptive background modeling [3] and optical flow method [4]. Among them, frame subtraction is a time difference between two consecutive pixels or three adjacent boxes in a vehicle video, and the calculation amount is small, but it is easy to generate a hollow phenomenon in the interior of a moving vehicle. Adaptive background modeling has a relatively small amount of computation, the background updating technology is used to accomplish adaptive updating of the background, and object can be segmented more accurately, but it is strongly affected by the weather outside. Optical flow method needs a large amount of computation and has poor anti-noise capability. Besides, because this method needs special hardware equipment, its application is limited. In order to further improve detection precision, deformation-based model DPM (Deformable Parts Model) [5] uses a star structure system, including the root and component filters and related deformation models for object detection, even if the object is partially occluded, it can also be successfully detected by DPM. However, it divides the pictures by moving the sliding window, and carries out HOG (Histogram of Oriented Gradient) feature extraction and SVM (Support Vector Machines) classification for each divided region. This process consumes a lot of computation.

Since 2012, the deep convolutional neural networks [6, 7] have shown their rich representative power in a variety of computer vision applications, including object segmentation, object detection and semantic recognition. In object detection, the RCNN [8] based region has received great attention from scholars, it combines with object proposals, features learned by CNN and SVM classier for improving the detection precision. In order to further inprove the detection speed and accuracy, Fast RCNN [9] uses a region of interest (ROI) pooling layer and the multi-task loss to predict bounding-box positions. More recently, Faster-RCNN [10] achieves the state-of-the-art performance by using the RPN with shared convolution features to generate potential object locations and using cascade detection strategies to reduce candidate boxes. At last, it runs with a speed of 5 FPS on a single GPU. Recently, Fully Convolution Network (FCN) [11] shows impressive performance on semantic segmentation, the author combines the high layer information which is coarse with the low layer information which is fine for semantic segmentation. But it's slow. In order to further improve the real-time detection performance, end-to-end object detection algorithms such as YOLO [12] and SSD [13] remove the region proposal network, which improves the detection speed and achieving real-time results. YOLO uses a global feature map and a fully connected layer to predict the detection boxes in a fixed set of regions. SSD uses two different convolution kernels to predict the category scores and offsets of a series of default boxes in multiple convolutional layers, which has greatly improved the accuracy and speed of detection. In [14], the author presents a comprehensive study of the effect of spatial resolution and color on the vehicle classification process in terms of accuracy and performance. In [15], it introduces the novel idea of using human social norms and human emotions to improve the

collision avoidance of autonomous vehicles. In [16], it presents a new approach to the detection, localization, and recognition of vehicles in infrared imagery using a deep convolutional neural network that completely avoids the need for manually-labelled training data by using synthetic imagery and a transfer learning strategy. However, in actual scenes, vehicles in the traffic surveillance are easily occluded, most vehicles are small and the weather changing is extreme. In this case, the above methods can't solve the problems well.

The generative adversarial network is a new idea of object detection. Its main idea is to adopt a generator model and a discriminator model. The generator model captures the distribution of the sample data and generates Gaussian or distributed noise to generate a sample similar to the real training data. The discriminator model is a binary classifier that estimates the probability of a sample from training data. In the process of training, both counteract learning and eventually reach a steady state. By using the structural association between objects of different scales and introducing low-level and fine-grained features to improve the representation of small objects, perceptual generative adversarial network [17] improves the detection rate of small objects. The method includes two sub-networks, the generator and sensor network. The generator network is a deep residual feature generation model that converts the initial poor features into high-resolution features. Perceptually resolved networks to distinguish small-object generate high-resolution features from real large-object features. However, this method also can't solve the vehicle occlusion problem well. Besides, the speed of detection in perceptual generative adversarial network is relatively slow.

Inspired by the above research, we have come up with a vehicle object detection algorithm which achieves high precision for detecting vehicle objects especially under conditions of occlusion and intense lighting changes based on jointing multiple convolutional layers and the generative adversarial networks. Firstly, by combining deep and coarse information with shallow and fine information, the convolution layers are fused to make the features more abundant. Secondly, proposed boxes are evolved by EPRN network. Then, the generative adversarial network makes full use of the characteristics of the vehicle data itself in its own unique way. However, our goal is to generate examples where detector is difficult to detect/classify, rather than trying to find hard examples through vehicle data. Meanwhile, we only add occlusions to the current existing examples by limiting the space of new positive generation, and the adversarial networks will predict occlusion between vehicles and learn its characteristics, as such occlusion could cause misclassification. Lastly, the boxes will be optimized to obtain the final vehicle detection boxes. The detection speed on the GPU Titan X can reach 8-10 FPS.

2. Framework

Fig. 1 shows the framework of our detection algorithm. The framework include Joint Feature Network (JFN), Evolved Region Proposal Network (EPRN), Mask Network (MaskN) and Fine-Tuning Network(FTN). JFN is in charge of extracting rich features by fusing different convolution layers from an image containing vehicles. EPRN uses soft-NMS algorithm to filter out a big number of candidate boxes that are unlikely to contain vehicles. In order to further improve the vehicle target detection capability, MaskN generates an example that includes the vehicle occlusion. It uses the features extracted by the ROI pooling layer as input image patches. FTN fine-tunes the position of the candidate boxes of the previous stage in order to obtain more accurate detection boxes.

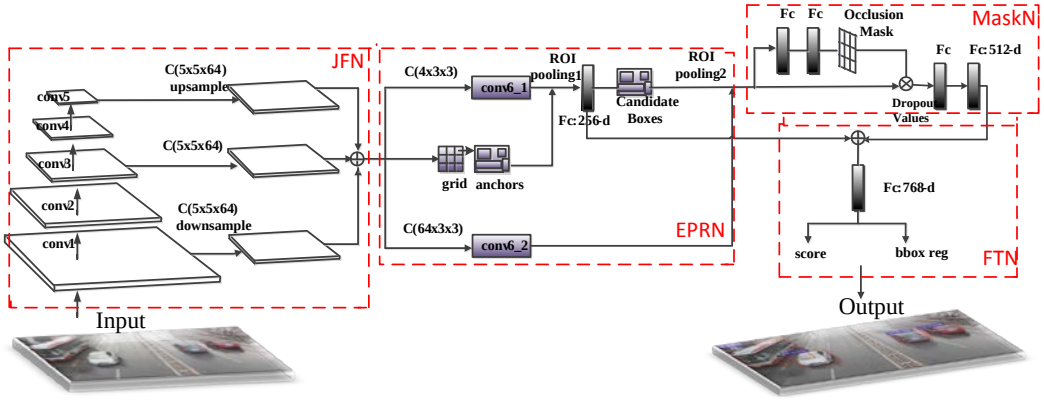


Fig. 1. The framework of our detection algorithm

(⊕ : feature concatenation from different networks. ⊗ : dropout values)

2.1 Joint Feature Network

For the input image, we use the convolution and max-pooling layers in the VGG-16 [18] network and load the pre-trained weights on the Image-Net dataset [19]. It is a common strategy in some excellent deep learning networks in order to ensure that the detection can benefit from the large dataset. The high convolutional layers have strong semantic information, which is good for classification, but the resolution is low, which is not conducive to localization. In contrary, low convolutional layers have a better range to locate objects because they have higher resolution. Prior studies [20] show that jointing feature maps from different convolutional layers can improve detection accuracy to some extent. Therefore, we combine feature maps from conv1, conv3, and conv5. In order to make the multi-layer feature map have the same resolution, we down-sample the low-level conv1, and up-samples the high-level conv5 to align with the middle-level conv3 feature map. Each sampled result will be sent to the convolutional layer (Conv). The Conv operation not only can extract more semantic features, but also can compress them into a unified resolution. Lastly, these feature maps are normalized using Batch Normalization [21] to form a joint feature map with 192 channels. The joint feature map has the following two characteristics: (a) multi-level abstraction; (b) better resolution.

2.2 Evolved Region Proposal Network

In real road traffic， the background objects of the vehicle are often diverse. Thus, filtering out bad candidate boxes is crucial for the vehicle detection algor ithm. In order to achieve this goal, we adopt an evolved region proposal network. Behind the joint feature map, two convolution kernels of size 4×3×3 and 64×3×3 are applied respectively. Conv6_1 aims to decrease the number of channels in the JFN fusion feature map from 192 to 4, which can speed up the generation of candidate regions. The whole image and the output feature map are all split into fixed-size grids. Each of grid generates a big number of candidate boxes with fixed sizes of 4×4, 8×8, 16×16, 32×32, 64×64, 128×128 and different aspect ratios of 0.5, 1 and 1.5. The ROI pooling 1 crops out a 7×7×4 feature vector for each candidate box, and then returns the initial boxes' coordinates (x, y, w, h) and detection score s through a 256-d full-connected layer, soft-NMS is used to filter out a big number of false candidate boxes. The dimension is reduced to 64 by Conv6-2, and a 7×7×64 feature vector for each candidate box after the initial screening is generated by ROI pooling 2.

NMS is a commonly used algorithm in deep learning object detection. It can generate detection boxes according to the vehicle detection score. The other boxes that are significantly overlapped with the box *M* corresponding to the highest detection score are suppressed. If the small object's detection score is within the preset overlap threshold, the small object which is suppressed directly won't able to be detected, thereby the detector performance will be degraded. Therefore, a linear weighted soft-NMS algorithm [22] is used in our paper. Instead of completely removing the detection score of the non-maximum detection boxes, The soft-NMS algorithm is to continuously attenuate it.

The soft-NMS will attenuate the detection score of the adjacent detection box that has an overlapping area with the detection box *M*. The more highly overlapping the detection boxes with *M*, the more likely there will be false positive results, and their fractional attenuation should be more severe. When the *IOU* between the adjacent detection boxes and *M* exceeds the the presset threshold, the detection score of the detection box will be linearly degenerated. In this case, the detection boxes close to *M* are attenuated to a large extent, while the detection boxes far from *M* are not affected. The linearly weighted soft-NMS score reset function is as follows:

$$S_i = \begin{cases} S_i, IOU(M, B_i) < N_t \\ S_i(1 - IOU(M, B_i)), IOU(M, B_i) \geq N_t \end{cases} \tag{1}$$

where $B = \{B_1, B_2, ..., B_N\}$, $S = \{S_1, S_2, ..., S_N\}$, $IOU(M, B_i) = \frac{size(M \cap B_i)}{size(M \cup B_i)}$. B represents the initial detection box set, S represents the score set corresponding to the initial detection boxes, N_t represents the NMS threshold, M represents the detection box with the highest score, and $IOU(M, B_i)$ represents the overlap between the detection box of B and M.

2.3 Mask Network

We are committed to generating a vehicle detector which is robust to occlusion. We try to automatically generate data that are hard examples for the vehicle detector to learn by means of adding the Mask Network. We are committed to generating occlusion in a finite space rather than generating data directly across the entire pixel space.

Mathematically, it is assumed that $F(x)$ represents the original object detector network where x is the region of interest. We assume that x is labeled with a ground-truth class μ and a ground-truth bounding-box regression object ν. So detector provides two outputs including $F_\mu(x)$ that represents object class output and $F_\nu(x)$ that represents predicted box location. Where $0 \leq F_\mu(x) \leq 1$. The initial detector loss function is expressed as follows:

$$Loss = L_{cls}(F_\mu(x), \mu) + \lambda[\mu \geq 1] L_{loc}(F_\nu(x), \nu) \tag{2}$$

There is a hyper-parameter λ control the loss balance between classification and boxes location. The iverson bracket indicator function $[\mu \geq 1]$ evaluates to 1 when $\mu \geq 1$ and 0 otherwise. By convention, the catch-all background class is labeled μ=0. There is no concept of a ground-truth bounding box, when the ROI is the background, L_{loc} is is equal to 0.

The first item on the right side of the equation is about the softmax loss of the category, $L_{cls}(F_\mu(x), \mu)$=-log($F_\mu(x)$) is log loss function for true class μ. The second item is location loss function between ground truth box location and predicted bounding box location (Applies only for foreground classes).

It is assumed that $A(x)$ represents the generation adversarial network which is given a feature x computed on the image I, and generates new adversarial sample x_A. Since the batches contain few original images and samples of adversarial, the loss function of the detector remains the same. We use the following generation adversarial network loss function:

$$L_A = 1 - L_{cls}(F_\mu(A(x)), \mu) \quad (3)$$

When characteristics generated by the adversarial network are easily classified by the object detector, the loss value of L_A will be very high. On the contrary, if the features generated by the adversarial network are difficult to identify, we will get high loss of detector and low loss of adversarial network.

We use the mask network to generate vehicle occlusions on the depth features of foreground objects in the vehicle image. We treat the region features obtained for each foreground object after ROI pooling 2 as the input to the mask network. Once the features of an object are given, the mask network will generate a mask, which can indicate that the specific part of the feature is to be discarded so that the detector cannot identify the object. The pooling feature maps generate masks through two fully connected layers.

Mask is a binary mask (a value of 0 or 1) that is the same size as the input. We extract the feature X on the vehicle image of size $d \times d \times c$, where d represents the spatial dimension and c represents the number of channels. Given this feature, the mask network will predict a mask m which also is $d \times d$. We denote m_{ij} as the value for the i_{th} row and j_{th} column of the mask. Similarly, the value in channel k at location (i, j) of the feature map is represented by X_{ijk}. If m_{ij} =1, the values of all channels in the space corresponding to the feature map are removed, i.e. $X_{ijk} = 0, \forall k$.

2.4 Fine-Tuning Network

The function of the FTN network is mainly to further adjust the remaining candidate boxes. The 768-d full connectivity layer is obtained by adding the 256-d full connectivity layer in the EPRN network to the 512-d full connectivity layer channels in the FTN network. The output of this fully-connected layer is a 5-dimensional vector that includes fine-tuned box position (x', y', w', h') and detection scores s'. Unlike YOLO and Faster-RCNN, which use direct regression to the object box, we use the structure of channel summation, it is very helpful for improving object detection.

2.5 Training

(1) We use the ImageNet's pre-trained VGG-16 model to initialize JFN, and the rest of the network is randomly initialized with a Gaussian distribution with a mean of zero and a standard deviation of 0.01. In our paper, we set the initial learning rate to 10^{-3}, then it will decrease 10^{-1} every 20k iterations and execute 60k iterations. The model uses a random gradient descent method for training with using 128 small batches. For the EPRN network, when the IOU between the anchor boxes and the ground truth is greater than 0.7, it is considered to be a positive sample; when IOU between the anchor box and the ground truth is less than 0.3, it is used as a negative sample. The soft-NMS with a threshold of 0.6 is used to eliminate a big number of excrescent boxes. Finally, only 800 candidate boxes are left by this method. For FTN networks, when the IOU between the anchor boxes and the ground truth is greater than 0.45, it is considered to be a positive sample, while candidate boxes with $0.1 \le IOU \le 0.3$ are determined to be negative samples. In addition, the FTN network also

applies hard mining techniques, and the top 70% of the samples are selected by us and then deliver to the back propagation.

(2) We will compare the loss of all $d/3\times d/3$ windows and finally choose the window with the highest loss, the window is used to generate a single $d\times d$ mask. Then these spatial masks for n positive region proposals and generate n pairs of training examples $\{(X^{-1},\tilde{M^{-1}}),....,(X^{-n},\tilde{M^{-n}})\}$ are generated by our adversarial network. The idea is that the mask network should learn to generate the masks which can give the detector network high losses. For each sliding window, we drop out the values in all channels whose spatial locations are covered by the window and generate a new feature vector from the region proposal. Then the feature vector is used to compute the loss through classification and box regression layers.

The binary cross entropy loss is used to train the mask network, and it can be formulated as:

$$L_{MaskN}=-\frac{1}{n}\sum_{p}^{n}\sum_{i,j}^{d}[\tilde{M}_{ij}^{p}A_{ij}(X^{P})+(1-\tilde{M}_{ij}^{p})(1-A_{ij}(X^{P}))] \tag{4}$$

where the outputs of the network in location (i, j) given input feature map X^{P} is represented by $A_{ij}(X^{P})$. We conduct 20K iterations through the loss training network. The initial learning rate is set to 10^{-3} and then decreased to 10^{-4} after 20k iterations.

(3) The network outputs a continuous heat-map, instead of a binary mask. We select the most influential 1/2 to generate the mask.

(4) Joint learning: combining the models obtained in steps (1) and (2), the initial learning rate is 10^{-3}, 10^{-1} is reduced every 10k, and iterative training is performed 20k times to obtain the final vehicle detection model. During each training iteration, the relationship between the various parts of the network is optimized. In order to train the vehicle detector, during the forward propagation, MaskN located after ROI Pooling 2 is used to generate a mask on the characteristics and then sampled to generate a binary mask which used to remove the features behind the ROI Pooling 2. Then we calculate the loss by forwarding the modified features and train the detector in a end-to-end way. Parameters before the pooling layer in the Mask network are initialized by using the EPRN network. We initialized the two fully connected layers by using the pre-trained ImageNet network. The Mask network uses its own fully connected layer for training. In this way, the network produces "harder" and diverse examples for training vehicle detector.

2.6 Multi-stage Loss

After the ERPN and FTN networks, the output boxes and scores of the vehicle are (x, y, w, h, s) and (x', y', w', h', s') respectively, $(\hat{x}, \hat{y}, \hat{w}, \hat{h})$ is the location of the positive anchor box. $g=(g_x, g_y.g_{w,}g_h)$ is used to parameterize the location of the ground truth box. $t=(t_x,t_y,t_w,t_h)$ is used to parameterize the the ground-truth box associated with the positive anchor box. $l=(l_x,l_y.l_{w,}l_h)$ is a vector representing 4 parameterized coordinates, it is used to parameterize the box generated by the ERPN network associated with the positive anchor box. $t'=(t_{x'},t_{y'},t_{w'},t_{h'})$ is used to parameterize the the ground-truth box associated with the proposal box generated by the ERPN network. And $l'=(l_x',l_y',l_w',l_h')$ is used to parameterize the FTN network output boxes associated with the proposal box generated by the ERPN network.

$$l_x = (x - \hat{x}) / \hat{w}, l_w = \log(w / \hat{w})$$
$$l_y = (y - \hat{y}) / \hat{h}, l_h = \log(h / \hat{h}) \quad (5)$$

$$t_x = (g_x - \hat{x}) / \hat{w}, t_w = \log(g_w / \hat{w})$$
$$t_y = (g_y - \hat{y}) / \hat{h}, t_h = \log(g_h / \hat{h}) \quad (6)$$

$$l_x' = (x' - x) / \hat{w}, l_w' = \log(w' / w)$$
$$l_y' = (y' - y) / \hat{h}, l_h' = \log(h' / h) \quad (7)$$

$$t_x' = (g_x - x) / \hat{w}, t_w' = \log(g_w / w)$$
$$t_y' = (g_y - y) / \hat{h}, t_h' = \log(g_h / h) \quad (8)$$

Therefore, we use the multi-stage loss L to jointly train for the classification, bounding-box regression(only for foreground classes) of the *EPRN* and the *FTN* and the loss of the Mask network:

$$L = \alpha L_{ERPN} + \alpha L_{FTN} + 2\alpha L_{MaskN}$$
$$L_{ERPN}(l, s) = L_{cls}(s) + \lambda L_{loc}(l, t) \quad (9)$$
$$L_{FTN}(l', s') = L_{cls}(s') + \lambda L_{loc}(l', t')$$

where L_{ERPN}, L_{MaskN} and L_{FTN} are the loss for ERPN, MaskN, and FTN respectively. α is used to balance the two networks of EPRN and FTN. λ is used to balance L_{cls} and L_{reg}. L_{MaskN} is shown in equation 4. In our algorithm, we emphasize on the loss of the MaskN. As is shown from table 3, when α =0.5, λ =1, the accuracy is the highest in our experiments. The classification regression loss is set to $L_{cls}(s) = -\log^s$, for the bbox regression, we use the smooth L1 method in [9], defined as follows:

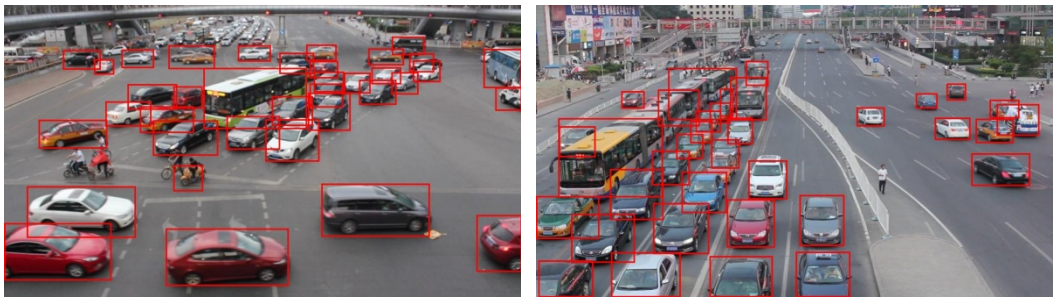
$$L_{\text{loc}}(t, s) = \sum_{i \in \{x, y, w, h\}} \sigma(t_i - s) \quad (10)$$

$$\sigma(\text{x}) = \begin{cases} 0.5x^2, if\ |x|<1 \\ |x|-0.5, others \end{cases} \quad (11)$$

3. Experiments and Discussions

3.1 DETRAC dataset

UA-DETRAC [23] is a highly complex multi-object detection and tracking benchmark. This data set is shot on 24 different locations in Beijing and Tianjin, China, using the Cannon EOS 550D camera. Video is recorded at 25 frames per second (FPS) with a resolution of 960×540 pixels. Among them, 140K frames are marked. 84K images are used for training, 56K images are used for testing, and 28K images are used for verification. DETRAC dataset is challenging, including data in various lighting conditions, such as sunny, cloudy, rainy and night time. According to statistics, each frame contains an average of 8.6 vehicles that are occluded frequently. This experiment only trains one category: car. all motor vehicles are classified as car. **Fig. 2** shows the detection results of our algorithm in DETRAC test set.



(a) (b)

Fig. 2. The detection results of our algorithm in DETRAC dataset

3.2 Comparing with State-of-The-Art

Table 1 shows the controlled experiment that use the different networks in our proposed framework. It also shows the comparison of not using soft-NMS and Mask network respectively, Faster-RCNN, EB [24], and our algorithm. Experimental results show that our algorithm without using soft-NMS exceeds the EB by 1.05% in mAP (mean Average Precision) and exceeds the Faster RCNN by 10.56% in mAP. Meanwhile, our algorithm without using Mask network exceeds the EB by 0.49% and exceeds the Faster RCNN by 10% in mAP. But at the same time it increases the calculation burden. When our algorithm don't use JTN network, the mAP will decrease 9.15%. So fusing the conv1, conv3, and conv5 convolutional layers to obtain the fusion feature map is of vital importance in our algorithm.

We evaluate the running time of our method on DETRAC test dataset, as shown in **Table 2**. With shared conv features, the speed up version only takes 25 ms to generate proposals in EPRN. Because of the deconvolution, the JFN takes 75ms which costs time most. Besides, MaskN takes 10ms, and FTN takes 15ms,The total time is 120 ms.

Table 3 shows the detection results of our algorithm on DETRAC test set by using different values of α and λ in equation (9). The hyper-parameter α and λ are very important for our method. The best performances are highlighted in bold.when α =0.5, λ =1, the accuracy is the highest in our experiments.

Fig. 3 is the comparison of the vehicle detection results of our algorithm and the latest EB. As shown in **Fig. 3**, because of the MaskN, our method can successfully detect most of the vehicles that are occluded.

Fig. 4 is the comparison of vehicle detection results of Faster-RCNN, SSD, YOLO, EB and our algorithm. The detection score threshold is 0.4. Our algorithm can detect vehicles more accurately under the same conditions including the same image resolution and detection threshold. Because adjacent layers are strongly correlated, which indicates that the combination of wider coarse-to-fine CNN features is more important.

Table 1. Comparison of running time and detection accuracy of different algorithm combinations

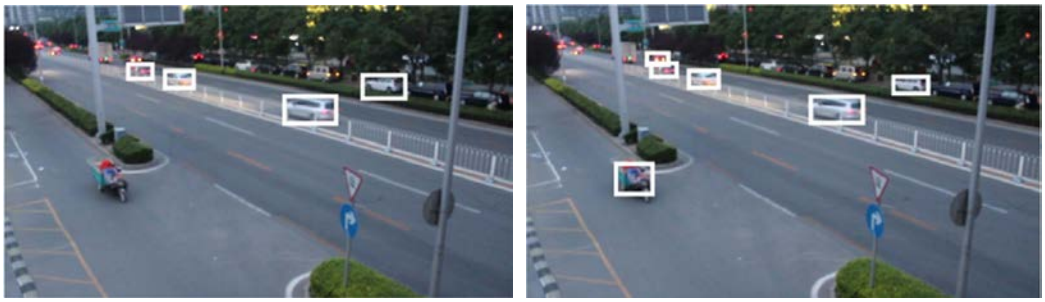
Method	Times/s	mAP
Faster-RCNN	0.087	58.45
EB	0.110	67.96
Our algorithm without JFN	0.090	60.45
Our algorithm without soft-NMS	0.100	69.01
Our algorithm without MaskN	0.110	68.45
Our algorithm	0.120	69.60

Table 2. The running time of the every step in our algorithm in the training

Steps	Times/s
JFN	0.075
EPRN	0.025
MaskN	0.010
FTN	0.015
Total	0.120

Table 3. Detection results (mAP) of our algorithm on DETRAC test set using different values of α and λ in equation (9)

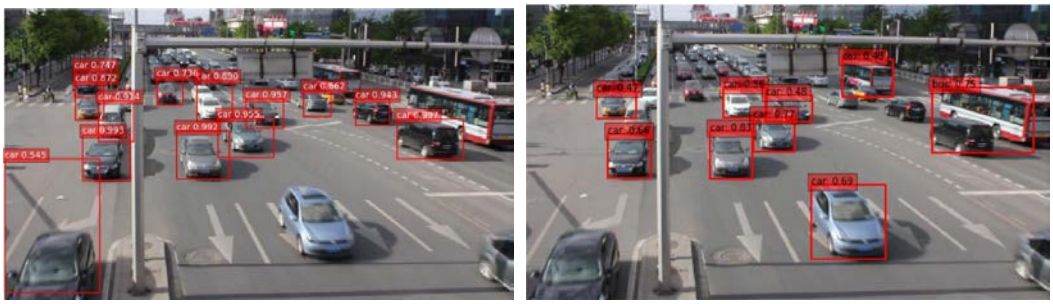
α \ λ	0.5	1	2
0.5	68.05%	**69.60%**	67.45%
0.6	67.26%	69.02%	68.26%



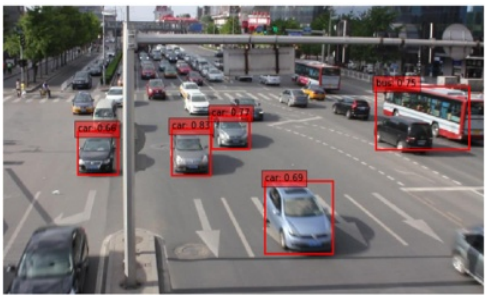
EB algorithm Our algorithm

EB algorithm Our algorithm

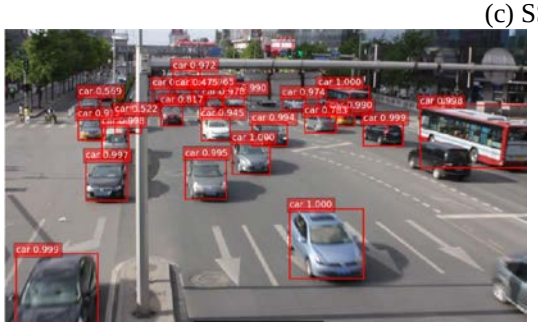
Fig. 3. The comparison of our algorithm and the EB



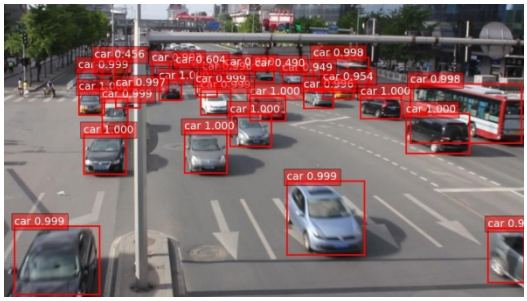
(a) Faster-RCNN (b) SSD



(c) SSD



(d) EB



(e) our algorithm

Fig. 4. The comparison of the detection results of Faster-RCNN, SSD, YOLO, EB and our algorithm

Table 4. The mAP of different excellent vehicle detection algorithms on the DETRAC test dataset (* is tested by ourselves, # is cited from the respective papers)

Method	Overall	Easy	Medium	Difficult	Sunny	Cloudy	Rainy	Night	Speed/s	Environment
DPM#	25.70	34.42	30.29	17.62	24.78	30.91	25.55	31.77	6	CPU@2.4GHz
ACF#	46.35	54.27	51.52	38.07	58.30	35.29	37.09	66.58	1.5	CPU@2.4GHz
RCNN#	48.95	48.95	54.06	39.47	59.73	39.32	39.06	67.52	10	GPU@K40
YOLO*	57.72	83.28	62.25	42.44	57.97	64.53	47.84	69.75	/	GPU@TitanX
Faster-RCNN*	58.45	82.75	63.05	44.25	66.29	69.85	45.16	62.34	**0.090**	GPU@TitanX
EB*	67.96	89.65	73.12	53.64	72.42	73.93	53.40	83.73	0.110	GPU@TitanX
Our algorithm*	**69.60**	**89.87**	**75.09**	**55.69**	**75.27**	**74.62**	**54.85**	**84.79**	0.120	GPU@TitanX

Table 4 shows the comparison of our algorithm with the most advanced vehicle detection methods. The best performances are highlighted in bold. Detection performance is showed including the overall mAP and mAPs in different situations. We compare it with DPM, RCNN, ACF [25], Faster-RCNN, YOLO and EB. Our algorithm is implemented on Caffe [26]. The results of the experiments show that our algorithm is superior to the currently most advanced methods. While our network with a 600×1000 sized input can achieve 11.15% mAP using Nvidia Titan X GPU, it is higher than Faster-RCNN, 11.88% higher than YOLO, and approximately 1.64% higher than EB. Our algorithm has the best performance in all the subcategories. In terms of speed, compared with RCNN, the deep learning network proposed by us is faster, and it is slightly slower than Faster-RCNN and EB.

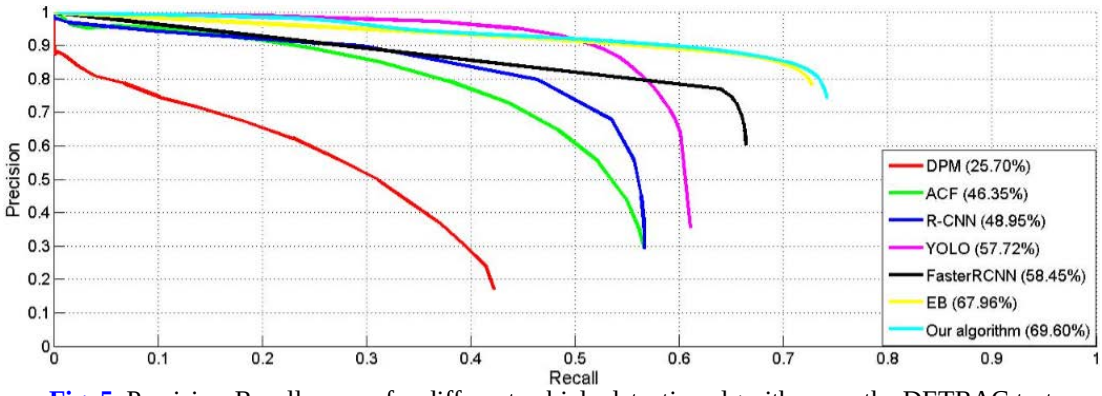


Fig. 5. Precision-Recall curves for different vehicle detection algorithms on the DETRAC test

Fig. 5 shows the precision-recall curves for different vehicle detection methods on the DETRAC test. Our results have reached the leading level in terms of recall rate and precision rate. In terms of recall rate, the recall rate of our algorithm is 2% higher than that of EB. There are more positive samples which are predicted correctly. In terms of precision, our algorithm is basically the same as EB. The appropriate anchor boxes are set according to the size of the object in EPRN, which make our results reach the leading level in terms of recall rate and precision rate.

Fig. 6. Selected examples of successful detection results

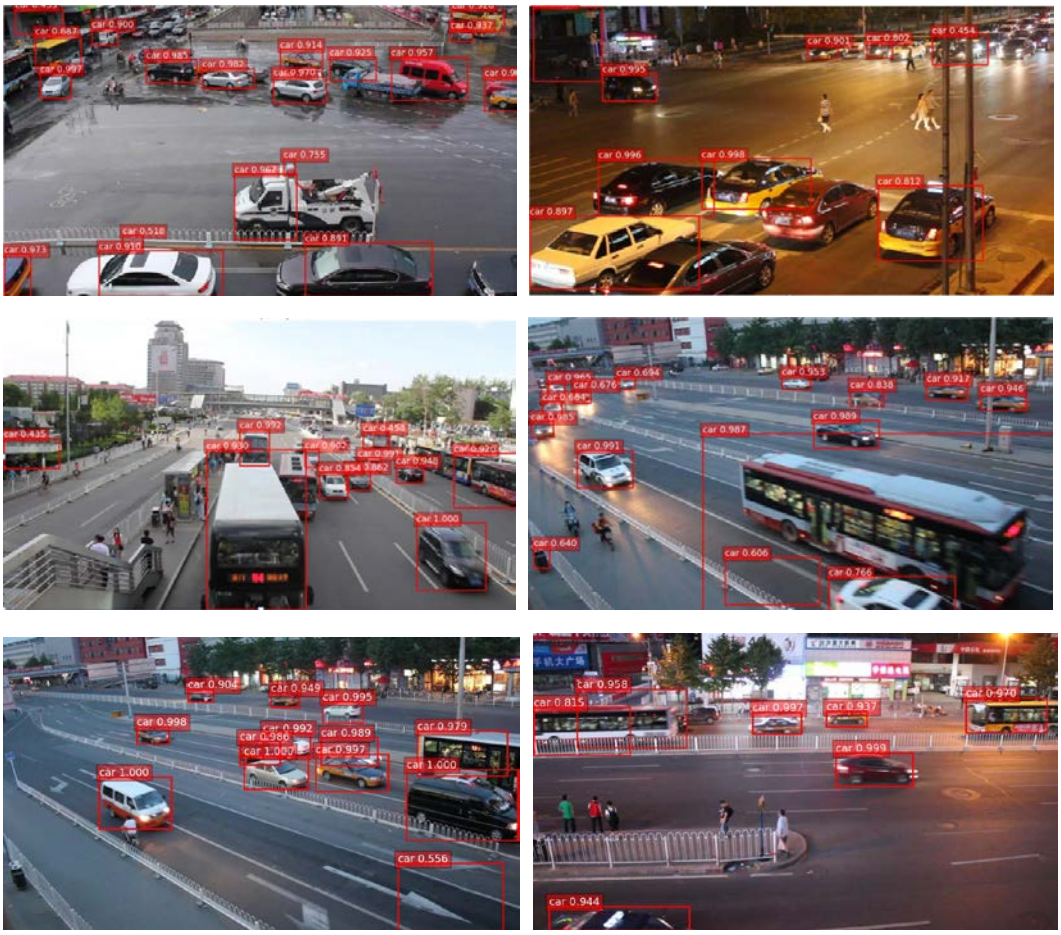


Fig. 7. Selected examples of unsuccessful detection results

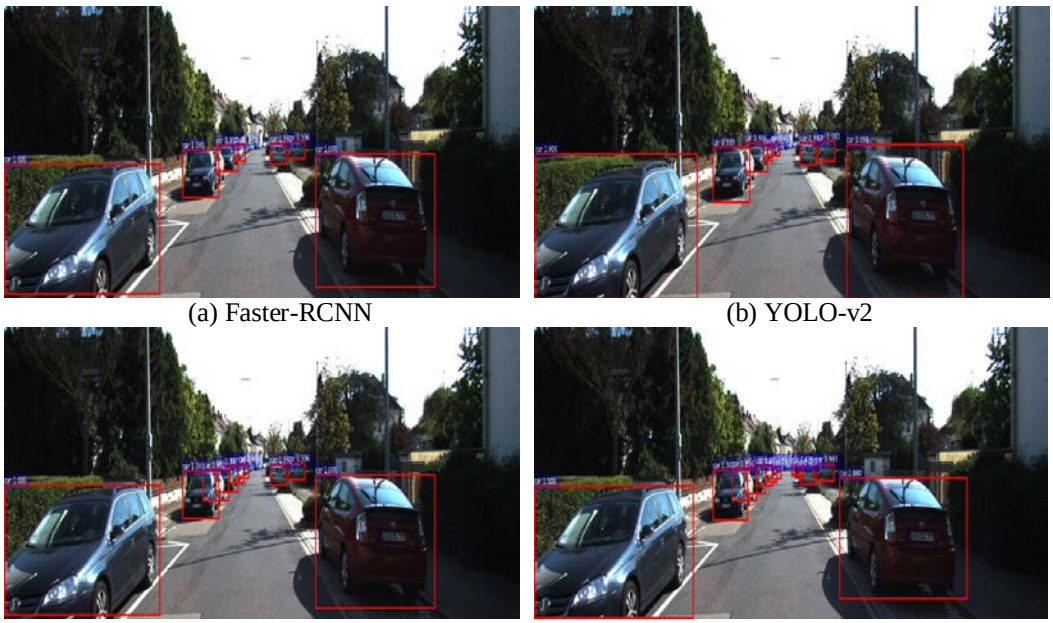
Fig. 6 and **Fig. 7** show the qualitative evaluation of our algorithm on the test dataset, successful and partially unsuccessful results are shown. In different scenarios, especially when the road is seriously blocked or the vehicles are far away from the camera, our algorithm can achieve very good performance. However, our algorithm also has partial failure detection. Due to the introduction of generation adversarial network, the algorithm erroneously judges some black background areas as vehicles.

3.3 Additional Experiments

Table 5. The mAP of different excellent vehicle detection methods on the KITTI test dataset (* is tested by ourselves, # is cited from the respective papers)

Method	Easy	Moderate	Hard	Speed/s	Environment
MS-CNN#	90.46	88.83	74.76	0.4	GPU @ 2.5 Ghz
Ours*	90.15	83.25	75.16	0.12	GPU@TitanX
YOLO-v3*	82.15	80.23	75.29	0.04	GPU@TitanX
RefineNet#	90.10	79.21	65.71	0.2	GPU @ 2.5 Ghz
RetinaNet#	89.93	78.85	68.73	0.2	4cores@2.5Ghz
Faster-RCNN*	86.79	78.24	69.26	0.1	GPU@TitanX
DSSD#	83.89	67.17	59.09	0.06	GPU@TitanX
YOLO-v2*	74.46	64.53	53.14	0.03	GPU@TitanX

Table 5 shows the detection results of different state-of-the-art approaches on KITTI test dataset which consists of 7481 training images and 7518 test images, comprising a total of 80256 labeled objects. We compare our approache with MS-CNN [27], YOLO-v3 [28], RefineNet [29], RetinaNet [30], Faster-RCNN [10], DSSD [31] and YOLO-v2 [32]. While our network with a 600×1000 sized input can achieve top2 using Nvidia Titan X GPU. Our algorithm achieves completely state-of-the-art object detection accuracy. The detection threshod is also 0.4. In our model, more contextual and small-object vehicle information can be obtained by JFN. In EPRN, we use a specific anchor box which allows the model to speed up the convergence and reduce the time of detection. Therefore, our model is also considerable in detection time compared to other methods. The MaskN and FTN are introduced in our algorithm, it can make our results reach the leading level in terms of mAP.



(a) Faster-RCNN (b) YOLO-v2

(c) YOLO-v3 (d) our algorithm

Fig. 8. The comparison of the detection results of Faster-RCNN*, YOLO-v2*, YOLO-v3* and our algorithm

Fig. 8 shows the comparison of the detection results of Faster-RCNN, YOLO-v2, YOLO-v3 and our algorithm. It can be observed that the proposed algorithm in this paper can achieve the best performance. As shown in **Fig. 8**, because the MaskN is applied and the convolutional layers are fused, our method can successfully detect most of the vehicles which are small and occluded.

4. Conclusion

In recent years, there are mainly three directions for improving vehicle detection. The first direction focuses on changing the network structure. The main idea is to use more in-depth networks to extract vehicle characteristics. The second direction is to take more contextual information into account to improve vehicle detection. The third direction is to make better use

of the vehicle dataset itself. We consider the latter two directions and fuse the conv1, conv3, and conv5 convolutional layers to obtain the fusion feature map. It takes more consideration of the context information of the image and processes the vehicle image itself through the Mask network to generate vehicles with occlusion. Good performance is achieved for both class recognition and localization on the DETRAC and KITTI data set. Meanwhile, the runing speed of our algorithm is 8-10 FPS on a moderate commercial GPU Titan X.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants 61871445, 61302156; Key R & D Foundation Project of Jiangsu province under Grant BE20 16001-4; Ministry of Education - China Mobile Research Foundation Project under Grant M CM20150504; the University Natural Science Research Project of Jiangsu province under Gr ant 13KJB510021.

References

[1] Y. Lu, A. Chowdhery, and S. Kandula, "Optasia: A Relational Platform for Efficient Large-Scale Video Analytics," in *Proc. of ACM Symposium on Cloud Computing ACM*, pp.55-70, 2016. Article (CrossRef Link)

[2] K. Park, D. Lee, and Y. Park, "Video-based detection of street-parking violation," in *Proc. of International Conference on Image Processing, Computer Vision, & Pattern Recognition, IPCV 2007*, June 25-28, 2007, Las Vegas Nevada, USA DBLP, pp.152-156, 2007.

[3] C. Stauffer and W. E. L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," *Computer Vision and Pattern Recognition*, vol. 2, pp.252-258, 1999. Article (CrossRef Link)

[4] L. Li, L. Shao, X. Zhen, et al., "Learning Discriminative Key Poses for Action Recognition," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp.1860-1870, 2013. Article (CrossRef Link)

[5] H. Azizpour and I. Laptev, "Object Detection Using Strongly-Supervised Deformable Part Models," *European Conference on Computer Vision*, pp.836-849, 2012. Article (CrossRef Link)

[6] A. Karpathy, et al., "Large-Scale Video Classification with Convolutional Neural Networks," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition IEEE Computer Society*, pp.1725-1732, 2014. Article (CrossRef Link)

[7] K. M. He, et al., "Mask R-CNN," in *Proc. of IEEE International Conference on Computer Vision*, IEEE, pp.2980-2988, 2017. Article (CrossRef Link)

[8] R. Girshick, et al., "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. of the IEEE conference on computer vision and pattern recognition*, pp.580-587, 2014. Article (CrossRef Link)

[9] R. Girshick, "Fast R-CNN," in *Proc. of IEEE International Conference on Computer Vision*, IEEE, pp.1440-1448, 2015. Article (CrossRef Link)

[10] S. Q. Ren,et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 6, pp.1137-1149, 2015. Article (CrossRef Link)

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *Computer Vision and Pattern Recognition IEEE*, pp.3431-3440, 2015. Article (CrossRef Link)

[12] J. Redmon, et al., "You Only Look Once: Unified, Real-Time Object Detection," in *Proc. of the IEEE conference on computer vision and pattern recognition*, pp.779-788, 2016. Article (CrossRef Link)

[13] W. Liu, et al., "SSD: Single Shot MultiBox Detector," in *Proc. of European conference on computer vision*, pp.21-37, 2016. Article (CrossRef Link)

[14] F. Riaz, et al., "A collision avoidance scheme for autonomous vehicles inspired by human social norms," *Computers & Electrical Engineering*, 2018. Article (CrossRef Link)
[15] C. P. Moate , et al., "Vehicle Detection in Infrared Imagery Using Neural Networks with Synthetic Training Data," *Image Analysis and Recognition*, 2018. Article (CrossRef Link)
[16] K. F. Hussain, M. Afifi, and G. Moussa, "A Comprehensive Study of the Effect of Spatial Resolution and Color of Digital Images on Vehicle Classification," *IEEE Transactions on Intelligent Transportation Systems*, pp.1-10, 2018. Article (CrossRef Link)
[17] Li, Jianan, et al., "Perceptual Generative Adversarial Networks for Small Object Detection," *Computer Vision and Pattern Recognition IEEE*, pp.1951-1959, 2017. Article (CrossRef Link)
[18] K. Simonyan, and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *in ICLR*, 2015.
[19] J. Deng, et al., "ImageNet: A large-scale hierarchical image database," in *Proc. of Computer Vision and Pattern Recognition, 2009, CVPR 2009, IEEE Conference on IEEE*, pp.248-255, 2009. Article (CrossRef Link)
[20] T. Kong, et al., "HyperNet: Towards Accurate Region Proposal Generation and Joint Object Detection," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition IEEE Computer Society*, pp.845-853, 2016.Article (CrossRef Link)
[21] S. Ioffe, and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proc. of International Conference on International Conference on Machine Learning*, JMLR.org, pp.448-456, 2015.
[22] Bodla, Navaneeth, et al., "Soft-NMS — Improving Object Detection with One Line of Code," in *Proc. of IEEE International Conference on Computer Vision IEEE Computer Society*, pp.5562-5570, 2017. Article (CrossRef Link)
[23] Wen Long yin, et al., "UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking," *arXiv preprint arXiv:1511.04136*, 2015.
[24] Wang, Li, et al., "Evolving boxes for fast vehicle detection," in *Proc. of IEEE International Conference on Multimedia and Expo, IEEE*, pp.1135-1140, 2017. Article (CrossRef Link)
[25] Dollar, Piotr, et al., "Fast Feature Pyramids for Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp.1532-1545, 2014. Article (CrossRef Link)
[26] Y.Q. Jia, E. Shelhamer, J. Donahue, et al., "Caffe: Convolutional architecture for fast feature embedding," in *Proc. of the 22nd ACM international conference on Multimedia, ACM*, 2014. Article (CrossRef Link)
[27] Cai, Zhaowei, et al., "A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection," *ECCV*, pp.354-370, 2016. Article (CrossRef Link)
[28] Girshick R, et al., "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[29] R. N. Rajaram, et al., "RefineNet: Refining Object Detectors for Autonomous Driving," *IEEE Transactions on Intelligent Vehicles*, vol.1, no. 4, pp.358-368, 2016. Article (CrossRef Link)
[30] T. Y. Lin, et al., "Focal loss for dense object detection," *IEEE transactions on pattern analysis and machine intelligence*, pp.2999-3007, 2018. Article (CrossRef Link)
[31] Fu, Cheng Yang, et al., "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.
[32] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition IEEE*, pp.6517-6525, 2017. Article (CrossRef Link)

Guang Han received the B.S. degree from Shandong University of Technology in 2004, and M.S. and Ph.D. degrees from Nanjing University of Science and Technology, in 2006 and 2010, respectively. Since 2010, he has been with Nanjing University of Posts and Telecommunications, Nanjing, China, where he is currently an associate professor in the Engineering Research Center of Wideband Wireless Communication Technology, Ministry of Education. His current research interests include pattern recognition, video analysis, computer vision and machine learning.

Jin-peng Su received the B.S. degree from Tianjin University of Commerce in 2016. Currently, he is pursuing the M.S. degree in the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, China. His current research interests include computer vision and machine learning.

Cheng-wei Zhang is currently an undergraduate in the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications from 2016. His current research interests include video analysis, computer vision and machine learning.