Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 3808064, 11 pages
https://doi.org/10.1155/2019/3808064

Research Article

Improving Faster R-CNN Framework for Fast Vehicle Detection

Hoanh Nguyen

Faculty of Electrical Engineering Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Vietnam

Correspondence should be addressed to Hoanh Nguyen; nguyenhoanh@iuh.edu.vn

Received 21 August 2019; Revised 16 October 2019; Accepted 5 November 2019; Published 22 November 2019

Academic Editor: Daniel Zaldivar

Vision-based vehicle detection plays an important role in intelligent transportation systems. With the fast development of deep convolutional neural networks (CNNs), vision-based vehicle detection approaches have achieved significant improvements compared to traditional approaches. However, due to large vehicle scale variation, heavy occlusion, or truncation of the vehicle in an image, recent deep CNN-based object detectors still showed a limited performance. This paper proposes an improved framework based on Faster R-CNN for fast vehicle detection. Firstly, MobileNet architecture is adopted to build the base convolution layer in Faster R-CNN. Then, NMS algorithm after the region proposal network in the original Faster R-CNN is replaced by the soft-NMS algorithm to solve the issue of duplicate proposals. Next, context-aware RoI pooling layer is adopted to adjust the proposals to the specified size without sacrificing important contextual information. Finally, the structure of depthwise separable convolution in MobileNet architecture is adopted to build the classifier at the final stage of the Faster R-CNN framework to classify proposals and adjust the bounding box for each of the detected vehicle. Experimental results on the KITTI vehicle dataset and LSVH dataset show that the proposed approach achieved better performance compared to original Faster R-CNN in both detection accuracy and inference time. More specific, the performance of the proposed method is improved comparing with the original Faster R-CNN framework by 4% on the KITTI test set and 24.5% on the LSVH test set.

1. Introduction

Vision-based vehicle detection is an essential prerequisite in many intelligent transportation systems, such as advanced driving assistance systems, autonomous driving, intelligent traffic management systems, and so on. Traditional methods usually use motion and handcrafted features to detect vehicles from images directly. In recent years, deep convolutional neural networks (CNNs) have achieved incredible success on object detection tasks as well as vehicle detection [1]. However, when applying CNNs to vehicle detection, real-time vehicle detection in driving environment is still very challenging. These challenges come from many occluded and truncated vehicles with large vehicle scale variations in traffic images. Thus, the popular CNN-based object detectors such as Faster R-CNN [2] and SSD [3] without modification did not achieve very good performance on vehicle detection. Many recent methods are based on modifying the popular CNN-based object detectors to enhance the performance of detection results. These methods focus on modifying the base network to fit different scales by applying multiscale feature maps of CNN [4] or utilizing input images with multiple resolutions [3]. In most public test datasets, these methods show better detection accuracy compared to traditional CNN-based object detectors. However, these methods still need significant computation cost and thus are still incapable of real-time vehicle detection.

In view of the aforementioned research challenges, this paper proposes an improved framework based on Faster R-CNN for real-time vehicle detection. First, MobileNet architecture [5] is adopted to build the base network instead of VGG architecture in the original Faster R-CNN framework. MobileNet splits the convolution into a 3×3 depthwise convolution and a 1×1 pointwise convolution, effectively reducing both computational cost and number of parameters. Thus, the proposed framework improves both computation cost and inference time. In the region proposal network, nonmaximum suppression algorithm is replaced by soft nonmaximum suppression algorithm [6] in order to solve the issue of heavy vehicle occlusion. Furthermore, context-aware RoI pooling [7] is used instead of RoI pooling to maintain the original structures of the small objects. Finally, a classifier based on MobileNet architecture is built to classify proposals and adjust the bounding box for each of the proposal. The proposed approach is evaluated on the KITTI benchmark dataset and the LSVH dataset. The results show that the proposed approach achieved better performance compared to other traditional deep CNN-based object detectors. More specific, the performance of the proposed method is improved comparing with the Faster R-CNN framework by 4% average with the KITTI test set and 24.5% with the LSVH test set.

This paper is organized as follows: an overview of previous methods on vehicle detection is presented in Section 2. Section 3 describes in detail the proposed method. Section 4 demonstrates experimental results. Finally, the conclusion is made in Section 5.

2. Theoretical Basis

In this section, this paper introduces previous methods on vehicle detection, including traditional methods and recently proposed methods based on deep CNN.

Vision-based vehicle detection system firstly locates vehicle candidate regions. Then, a classifier is constructed to eliminate false vehicle candidate regions. Traditional methods can be divided into two categories: motion-based methods and static appearance feature-based methods. Motion-based methods use the motion to detect the vehicles in the image frame. Background subtraction methods [8] are most widely used. The background removal methods include Kalman filter [9], single Gaussian pixel distribution [10], Gaussian mixture model (GMM) [11], and wavelets [12]. Another method of motion feature is based on the optical flow [13]. Optical flow is widely used in vehicle detection since it is less susceptible to occlusion issues. Static appearance feature-based methods focus on external physical features such as color, texture, edge, and shape. A variety of feature descriptors have been used in this field such as HOG [14], SURF [15], Gabor [16], and Haar-like [17]. These feature descriptors are usually followed by classifiers like SVM [14], artificial neural network [16], and AdaBoost [17]. Traditional methods showed high accuracy in limited conditions. However, with the effect of shadow, occluded vehicle, complex scenarios, and environments, traditional methods showed poor performance.

Recently, deep CNN-based methods have become the leading method for high-quality general object detection, including vehicle detection. Faster region-based convolutional neural network (Faster R-CNN) [2] defined a region proposal network (RPN) for generating region proposals and a network using these proposals to detect objects. RPN shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. This method has achieved state-of-the-art detection performance and becomes a commonly employed paradigm for general object detection. MS-CNN [4] extends the detection over multiple scales of feature layers, which produce good detection performance improvement. SSD framework [18] skips the region proposal stage and directly uses multiple feature maps with different resolutions to perform object localization and classification. YOLOv2 [19] introduces improvements of batch normalization, high-resolution classifier, convolutional with anchor boxes, and dimension clusters compared to original YOLO [20]. Comparing to YOLO, YOLOv2 achieves higher accuracy and higher speed. To better handle the detection problem of vehicles in complex conditions, Chu et al. [21] proposed a vehicle detection scheme based on multitask deep CNN in which learning is trained on four tasks: category classification, bounding box regression, overlap prediction, and subcategory classification. A region of interest voting scheme and multilevel localization are then used to further improve detection accuracy and reliability. Experimental results on the standard test dataset showed better performance than other methods. In [22], the authors proposed the deep model for vehicle detection which consists of feature extraction, deformation processing, occlusion processing, and classifier training using the back propagation algorithm. Li et al. [23] proposed a multivehicle detection method which consists of YOLO under the Darknet framework. To make the full use of the advantages of the depth information of lidar and the obstacle classification ability of vision, Wang et al. [24] proposed a real-time vehicle detection algorithm which fuses vision and lidar point cloud information. The experimental results showed that the proposed algorithm significantly improved the vehicle detection accuracy at different detection difficulty levels compared to the original YOLOv3 algorithm, especially for the vehicles with severe occlusion. In [25], the authors presented a two-stage detector based on Faster R-CNN for high-occluded vehicle detection. The part-aware RPN is proposed to replace the original RPN at the first stage of the Faster R-CNN module, and the part-aware NMS is proposed to refine final results. Kim et al. [26] proposed to integrate additional prediction layers into conventional YOLOv3 using spatial pyramid pooling to complement the detection accuracy of the vehicle for large-scale changes or being occluded by other objects. This architecture showed a state-of-the-art mAP detection ratio against the other vehicle detection approaches with reasonable run-time speed.

3. The Proposed Framework

Figure 1 shows the overall framework of the proposed approach. To differentiate from the original Faster R-CNN framework, the proposed enhancements are highlighted by red boxes in Figure 1. In the first stage, MobileNet architecture [5] is used to build the base convolution layer instead of VGG-16 architecture [27] in the original Faster R-CNN framework. In the region proposal network, soft-NMS algorithm is used to solve the issue of heavy vehicle occlusion. RoI pooling layer is then replaced by the context-aware RoI pooling layer to maintain the original structures of the small vehicles. The classifier based on MobileNet architecture is built at the final stage to classify proposals into the vehicle and background and adjust the bounding box for each of the

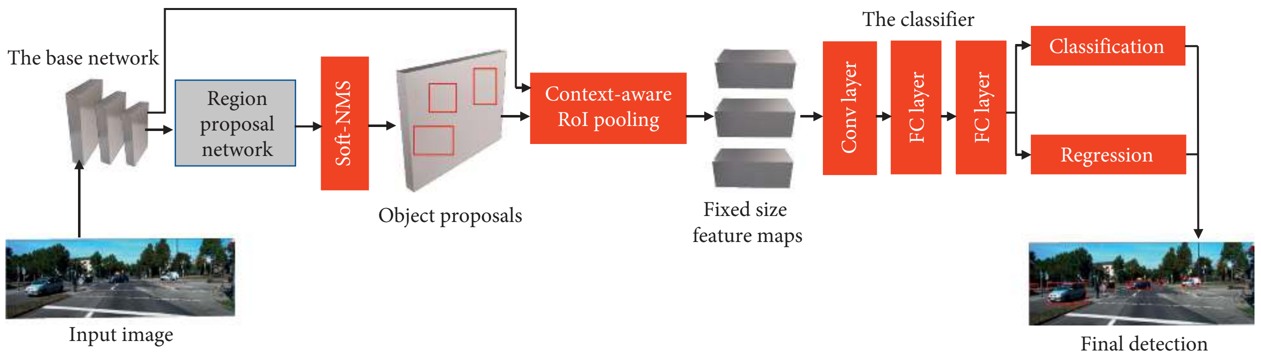


Figure 1: The overall framework of the proposed approach.

detected vehicle. In the following section, the proposed approach is explained in detail.

3.1. The Base Network. The original Faster R-CNN framework used VGG-16 [27] as the base network. In [18], Liu et al. proved that about 80% of the forward time is spent on the base network so that using a faster base network can greatly improve the speed of the whole framework. MobileNet architecture [5] is an efficient network which splits the convolution into a 3×3 depthwise convolution and a 1×1 pointwise convolution, effectively reducing both computational cost and number of parameters. Table 1 shows the comparison of MobileNet and VGG-16 on ImageNet [28]. As shown, MobileNet is nearly as accurate as VGG-16 while being 32 times smaller and 27 times less compute intensive. With the purpose of real-time vehicle detection in traffic scenes, MobileNet architecture is used as the base network in this study. MobileNet introduces two parameters which can be used to tune to fit the resource/accuracy trade-off, including width multiplier and resolution multiplier. The width multiplier allows us to thin the network, while the resolution multiplier changes the input dimensions of the image, thus reducing the internal representation at every layer. In this study, MobileNet is adopted to build the base convolutional layers in Faster R-CNN instead of VGG-16 in the original framework for fast vehicle detection. Since this paper uses only the convolution layers in MobileNet architecture, the size of the input image does not have to be fixed. Supposing the size of the input image is $224 \times 224 \times 3$, the architecture of the base network is defined, as shown in Table 2.

In Table 2, "Conv" represents as a standard convolution; "Conv dw" represents as a depthwise separable convolution; "s1" represents that the convolution stride is 1×1; and "s2" represents that the convolution stride is 2×2.

Depthwise separable convolution is made up of two layers: depthwise convolutions and pointwise convolutions. Depthwise convolutions are used to apply a single filter per each input channel, while pointwise convolution, a simple 1×1 convolution, is used to create a linear combination of the output of the depthwise layer. MobileNet architecture uses both batch norm and ReLU nonlinearities for both layers. The reduction of computational cost is in proportion to the number of output feature map channel and the square

Table 1: Comparison of the MobileNet model with the VGG model.

Model	ImageNet accuracy (%)	Multiply-adds (million)	Parameters (million)
MobileNet	70.6	569	4.2
VGG-16	71.5	15300	138

Table 2: The architecture of the base network.

Type/stride	Filter shape	Input size
Conv/s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw/s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv/s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw/s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv/s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw/s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv/s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw/s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv/s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw/s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv/s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw/s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv/s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5 \times$ conv dw/s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5 \times$ conv/s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$

of kernel size. More details about MobileNet architecture can be found in [5].

3.2. Region Proposal Network (RPN). The RPN first generates a set of anchor boxes from the convolution feature map generated by the base network. An anchor is centered at the sliding window and is associated with a scale and aspect ratio. For the trade-off between recall and processing speed, three anchor box scales of 128, 256, and 512 and three anchor box ratios of 1 : 1, 1 : 2, and 2 : 1 are used for each anchor in this paper as in [2], yielding 9 anchors at each sliding position. For a convolutional feature map of a size 14×14, there are 1,764 anchors in total, as shown in Figure 2.

The RPN then takes all the anchor boxes and outputs two different outputs for each of the anchors. The first one is objectness score, which means the probability that an anchor is an object. The second output is the bounding box regression for adjusting the anchors to better fit the object, as

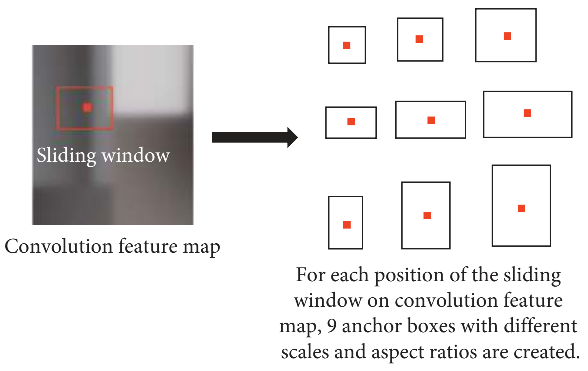

Sliding window
Convolution feature map
For each position of the sliding window on convolution feature map, 9 anchor boxes with different scales and aspect ratios are created.

Figure 2: Anchor boxes generated by RPN.

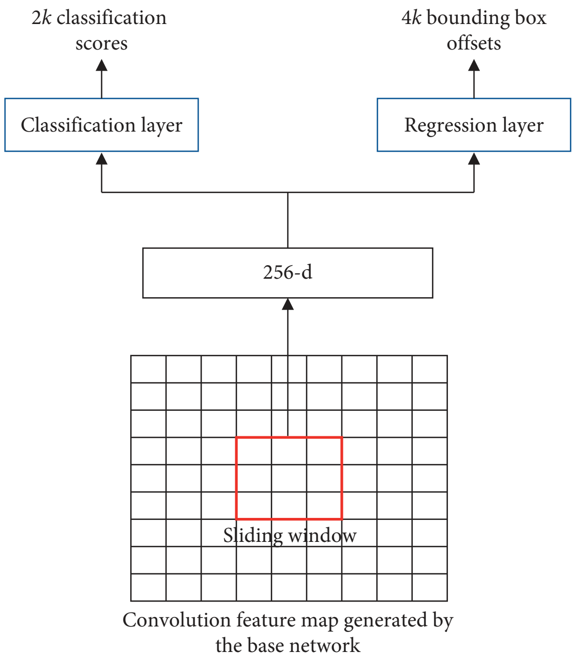


Figure 3: The region proposal network.

shown in Figure 3. Using the final proposal coordinates and their objectness score, a good set of proposals for vehicles is created. Since anchors usually overlap, proposals end up also overlapping over the same object. Soft nonmaximum suppression (NMS) algorithm [6] is used to solve the issue of duplicate proposals. In most state-of-the-art object detections, including Faster R-CNN, NMS algorithm is used to remove duplicate proposals. Traditional NMS removes any other proposal that has an overlap more than a predefined threshold with a winning proposal. Due to heavy vehicle occlusion in traffic scenes, traditional NMS algorithm may remove positive proposals unexpectedly (as shown in Figure 4). To address the NMS issue with occluded vehicles, this paper adopts soft-NMS algorithm. With soft-NMS, the neighbor proposals of a winning proposal are not completely suppressed. Instead they are suppressed based on updated objectiveness scores of the neighbor proposals, which are computed according to the overlap level of the neighbor proposals and the winning proposal. Soft nonmaximum

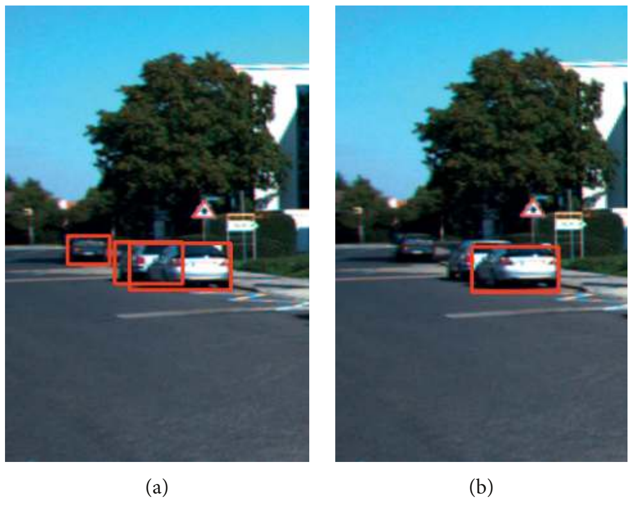


Figure 4: Detection results with (a) Soft-NMS and (b) NMS. Due to heavy vehicle occlusion, NMS removed one car in detection results, while soft-NMS detected two cars separately.

suppression (NMS) algorithm will be discussed more in detail in the following section. Figure 4 shows the example of detection results with NMS (right) and soft-NMS (left). As shown, NMS removed one car due to high overlap between two cars, while soft-NMS kept two cars separately.

3.2.1. Soft Nonmaximum Suppression Algorithm. Let $P_{in} = \{p_1, p_2, p_3, \ldots, p_n\}$ denote an initial proposal set output from the object proposal layers, in which the proposals are sorted by their objectiveness scores. For a proposal p_i, any other proposal that has an overlap more than a predefined threshold T with proposal p_i is called a neighbor proposal of proposal p_i. In this paper, the neighbor proposal threshold T is set to 0.5 by cross-validation. Let S_i denote the objectiveness score of p_i, which is the maximum value in the classification score vector of p_i. For a proposal set, the proposal with the highest objectiveness score is called the winning proposal. Let p_i be a winning proposal and p_j be a neighbor proposal of p_i. The updated objectiveness score of p_j (denoted by S_j^u) is computed by the following formula [6]:

$$S_j^u = S_i\left(1 - O_{p_i,p_i}\right), \tag{1}$$

where O_{p_i,p_i} denotes the intersection of union (IoU) between proposal p_i and proposal p_j and is computed by the following formula:

$$O_{p_i,p_i} = \frac{\text{area}\left(p_i \cap p_j\right)}{\text{area}\left(p_i \cup p_j\right)}. \tag{2}$$

Soft-NMS algorithm is described by the flowchart in Figure 5.

3.3. Context-Aware RoI Pooling. In most two-stage object detection algorithms, such as Fast R-CNN, Faster R-CNN, and so on, the RoI pooling layer [29] is used to adjust the size of proposals to the fixed size. The principle of an RoI

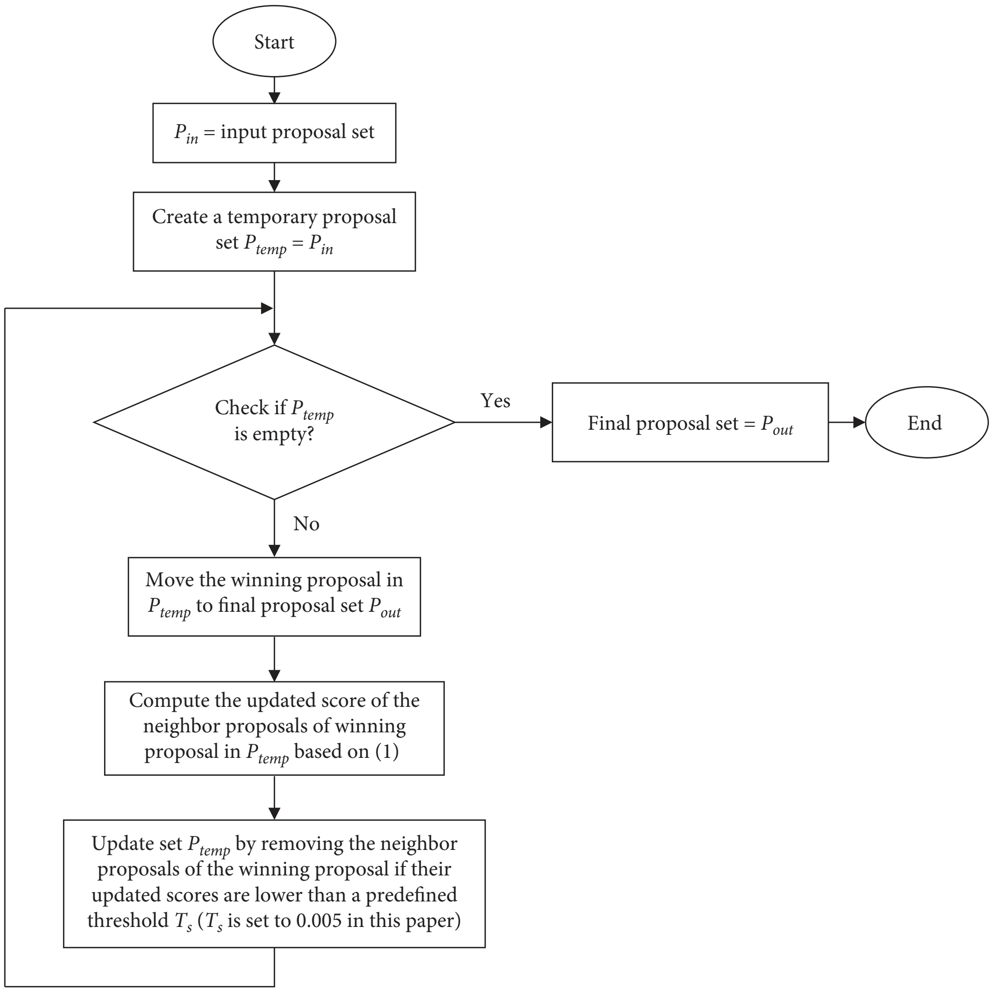

Start
P_{in} = input proposal set
Create a temporary proposal set $P_{temp} = P_{in}$
Check if P_{temp} is empty?
Yes
Final proposal set = P_{out}
End
No
Move the winning proposal in P_{temp} to final proposal set P_{out}
Compute the updated score of the neighbor proposals of winning proposal in P_{temp} based on (1)
Update set P_{temp} by removing the neighbor proposals of the winning proposal if their updated scores are lower than a predefined threshold T_s (T_s is set to 0.005 in this paper)

FIGURE 5: The flowchart of the soft-NMS algorithm.

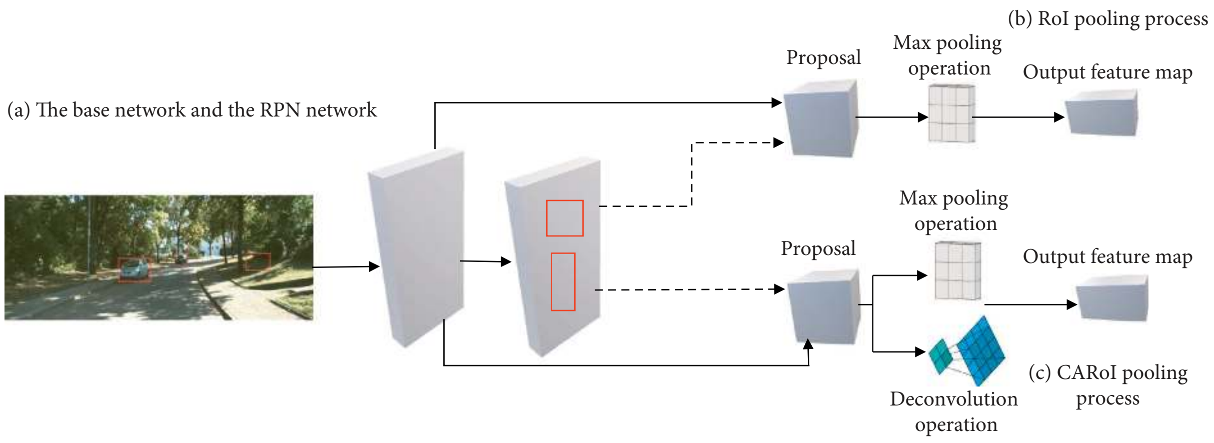


FIGURE 6: Context-aware RoI pooling scheme. (a) Feature maps and proposals generated by the base network and the RPN. (b) Traditional RoI pooling process. (c) CARoI pooling process.

pooling layer is illustrated in Figure 6(b). The RoI pooling layer uses max pooling to convert the features inside any valid region of interest into a small feature map with a fixed spatial extent of $H \times W$. RoI max pooling works by dividing the $h \times w$ RoI proposal into an $H \times W$ grid of subwindows of approximate size $(h/H) \times (w/W)$, and then max pooling the values in each subwindow into the corresponding output grid cell. If a proposal is smaller than $H \times W$, it will be enlarged to $H \times W$ by adding replicated values to fill the new space. RoI pooling avoids repeatedly computing the convolutional layers, so it can significantly speed up both train and test time. However, adding replicated values to small proposals is not appropriate, especially with small vehicles, as it may destroy the original structures of the small vehicles. Moreover, adding replicated values for small proposals will lead to inaccurate representations in the forward propagation and accumulation of errors in the backward propagation during the training process. Thus, the performance of detecting small vehicles will be reduced. To adjust the size of proposals to the fixed size without destroying the original structures of the small vehicles and enhance the performance of the proposed approach on detecting small vehicles, context-aware RoI pooling (CARoI pooling) [7] is used in this paper. The principle of context-aware RoI pooling layer is illustrated in Figure 6(c). In CARoI pooling process, if the size of a proposal is larger than the fixed size of the output feature map, max pooling is used to reduce the size of the proposal to a fixed size as in traditional RoI pooling. If the size of a proposal is smaller than the fixed size of the output feature map, deconvolution operation is applied to enlarge the size of the proposal to a fixed size as the following formula:

$$y_k = F_k \oplus h_k, \tag{3}$$

where y_k represents the output feature map with the fixed size, F_k represents the input proposal, and h_k is the kernel of the deconvolution operation. The size of the kernel is equal to the ratio between the size of the output feature map and the size of the input proposal. Moreover, when the width of a proposal is larger than the fixed width of the output feature map and the height of this proposal is smaller than the fixed height of the output feature map, deconvolution operation as in (3) is adopted to enlarge the height of this proposal, and max pooling is applied to this proposal to reduce the width of this proposal. With the CARoI pooling layer, the size of proposals has been adjusted to the fixed size while discriminative features from the small proposals can be still extracted.

3.4. *The Classifier.*

The classifier is the final stage in the proposed framework. After extracting features for each of proposals via context-aware RoI pooling, these features are used for classification. The classifier has two different goals: classify proposals into the vehicle and background class and adjust the bounding box for each of the detected vehicle according to the predicted class. The proposed classifier is defined in Table 3. This study uses the structure of depthwise separable convolution in MobileNet architecture to build the classifier. The classifier has two fully connected (FC) layers, a box classification layer and a box regression layer. The first FC layer is fed into the softmax layer to compute the confidence probabilities of being vehicle and background. The second FC layer with linear activation functions regresses the bounding box of the detected vehicle. All convolutional layers are followed by a batch normalization layer and a ReLU layer. The loss function and the parameterization of coordinates for bounding box regression are the same as in the original Faster R-CNN framework [2]. The loss function of the RPN and the loss function of the classifier share the same form but are optimized separately.

Table 3: The architecture of the classifier.

Type/stride	Filter shape	Input size
Conv dw/s2	$3\times3\times512$ dw	$14\times14\times512$
Conv/s1	$1\times1\times512\times1024$	$7\times7\times512$
Conv dw/s2	$3\times3\times1024$ dw	$7\times7\times1024$
Conv/s1	$1\times1\times1024\times1024$	$7\times7\times1024$
Avg pool/s1	Pool 7×7	$7\times7\times1024$
FC/s1	1024×2	$1\times1\times1024$
Softmax	Classification	RoI $\times\,2$
FC/s1	1024×4	$1\times1\times1024$
Linear	Regression	RoI $\times\,4$

4. Results and Discussion

In order to compare the effectiveness of the proposed approach with other state-of-the-art approaches on vehicle detection, this paper conducts experiments on widely used public datasets: KITTI dataset [30] and LSVH dataset [7]. The proposed approach is implemented on a Window system machine with Intel Core i7 8700 CPU, NVIDIA GeForce GTX 1080Ti GPU and 16 Gb of RAM. TensorFlow is adopted for implementing deep CNN frameworks, and OpenCV library is used for real-time processing.

4.1. *Dataset.*

KITTI dataset [30] is a widely used dataset for evaluating vehicle detection algorithms. This dataset consists of 7,481 images for training with available ground truth and 7,518 images for testing with no available ground truth. Images in this dataset include various scales of car in different scenes and conditions and were divided into three difficulty level groups: easy, moderate, and hard. If the bounding box size was larger than 40 pixels, a completely unshielded vehicle was considered to be an easy object, if the bounding box size was larger than 25 pixels but smaller than 40 pixels, a partially shielded vehicle was considered as a moderate object, and a vehicle with the bounding box size smaller than 25 pixels and an invisible vehicle that was difficult to see with the naked eye were considered as hard objects. LSVH dataset [7] contains 16 videos captured under different scenes, time, weathers, and resolutions and is divided into two groups: sparse and crowded. A video scene containing more than 15 vehicles per frame on average is considered as a crowded scene. Otherwise, it is considered as a sparse scene. As in [7], this paper uses the eight videos in the sparse group as the training data and the left four videos in the sparse group as the testing data.

4.2. *Evaluation Metrics.*

This paper uses the average precision (AP) and intersection over union (IoU) metrics [31] to evaluate the performance of the proposed method in all three difficulty level groups of the KITTI dataset and LSVH dataset. These criteria have been used to assess various object detection algorithms [30, 31]. The IoU is set to 0.7 in this paper, which means only the overlap between the detected bounding box and the ground truth bounding box greater than or equal to 70% is considered as a correct detection.

4.3. Training. For the base network, this paper uses the MobileNet pretrained model on the ImageNet dataset [32] and further fine-tuned on the KITTI dataset and the LSVH dataset. To accelerate training and reduce overfitting, the weights of each batch normalization layer in the pretrained model are frozen during the training process. The RPN and the classifier are trained by turns. First, the RPN is trained on a mini-batch, and the parameters of the RPN and the base network are updated once. Then, positive proposals and negative proposals generated by the RPN are used to train and update the classifier. The parameters of the classifier are updated once and the parameters of the base convolutional layers are updated once again. The RPN and the MobileNet-based classifier share the base convolutional layers. The loss function and the parameterization of coordinates for bounding box regression in this study are the same as those in original Faster R-CNN. The balancing parameter λ is set to 1 in the loss function. The Adam algorithm [33] is adopted for optimizing the loss functions. The initial learning rates of the RPN and the classifier are set to 0.0001 with the learning rate decay of 0.0005 per mini-batch. The network is trained for 200 epochs.

4.4. Performance Results. In this section, this study first checks the effectiveness of each enhanced module, and then compares the performance of the proposed approach with other state-of-the-art approaches on the KITTI dataset and the LSVH dataset, including original Faster R-CNN [2], SSD [18], YOLO [20], YOLOv2 [19], and MS-CNN [4].

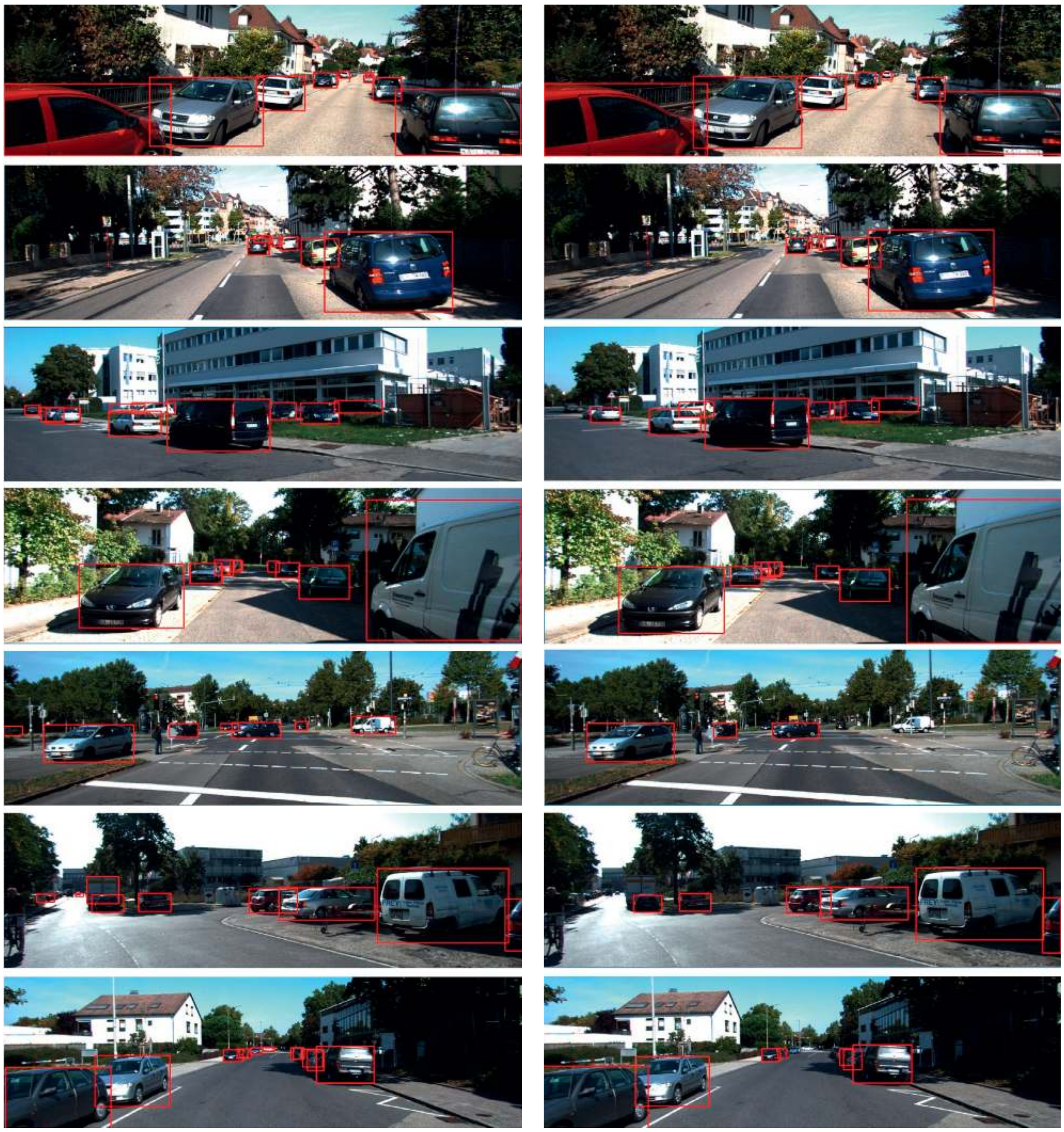
4.4.1. Experimental Results on the KITTI Validation Set. Since the ground truth of the KITTI test set is not publicly available, this paper splits the KITTI training images into a train set and a validation set to conduct experiments as in [3], which results in 3,682 images for training and 3,799 images for validation. To examine the effectiveness of each proposed enhancement, this paper conducts separate experiments on each of the enhanced module and compares the results with the original Faster R-CNN framework. In the first experiment, the NMS algorithm is replaced by the soft-NMS algorithm, while other modules in original Faster R-CNN are kept unchanged. In the second experiment, context-aware RoI pooling is adopted to replace RoI pooling process, and the NMS algorithm is kept unchanged in this experiment. Finally, MobileNet architecture is adopted as the base network instead of VGG-16 architecture, while the NMS algorithm and RoI pooling layer are kept unchanged. Table 4 reports the AP results and the inference time of each enhanced module and the original Faster R-CNN for vehicle detection over the KITTI validation set. It can be observed that soft-NMS improves the performance in all groups with no extra computation time. More specific, the AP with soft-NMS increases by 1.33%, 0.6%, and 0.01% in "easy," "moderate," and "hard" groups, respectively, compared to original Faster R-CNN. These results demonstrate the effectiveness of soft-NMS on solving the issue of duplicate vehicles in driving environments. Comparing with RoI pooling in the original Faster R-CNN, context-aware RoI pooling process dramatically improves the accuracy while no extra time is introduced (as shown in the 3rd row). Particularly, the improvements are significant with the "moderate" and "hard" groups. These results demonstrate that the recovered high-resolution semantic features are very useful for detecting small vehicles. The last row in Table 4 shows the AP results of Faster R-CNN with MobileNet architecture. As shown, MobileNet is nearly as accurate as VGG while dramatically improving the inference time. More specific, Faster R-CNN with MobileNet needs 0.15 second to process an image, while Faster R-CNN with VGG-16 needs up to 2 seconds.

Figure 7 presents some examples of detection results of the proposed method (shown in the left column) and the original Faster R-CNN framework (shown in the right column) on the KITTI validation set. As shown in this figure, with the contribution of context-aware RoI pooling, the proposed approach can detect more small vehicles compared to Faster R-CNN. Furthermore, with the contribution of soft-NMS postprocessing, the proposed method can avoid removing positive vehicle proposals unexpectedly (shown in the first row and the third row in Figure 7).

4.4.2. Experimental Results on the KITTI Test Set. Next, this study trains the proposed network with the KITTI training set and compares the results of the proposed method with recently published methods over the KITTI test set. Table 5 shows the comparison of detection results on all three categories of the KITTI test set. As shown from Table 5, the performance of the proposed method is improved comparing with the Faster R-CNN framework by 2.49%, 5.92%, and 3.6% in "easy," "moderate," and "hard" groups, respectively. Furthermore, comparing with the SSD framework, the proposed algorithm improves by 11.49%, 23.8%, and 18.55% in "easy," "moderate," and "hard" groups, respectively. For the computational efficiency, the proposed method takes 0.15 second for processing an image, while the original Faster R-CNN framework takes up to 2 seconds. The MobileNet architecture dramatically improves processing speed of the proposed approach. Thus, the proposed approach meets the real-time detection standard and can be applied to the road driving environment of actual vehicles. Comparing the average precision and the processing time results in Table 5, it can be concluded that there is no absolute winner with dominant performance over all the comparison aspects. Among the compared leading approaches, MS-CNN [4] ranked the first. However, MS-CNN has the second longest processing time (0.4 second), while the proposed approach needs only 0.15 second. Other one-stage deep learning-based detectors (YOLO, YOLOv2, and SSD) are faster than the proposed approach, but with very low accuracy. Figure 8 shows detection results of the proposed method on the KITTI test set (the left column).

Table 4: The AP results and the inference time of each enhanced module and the original Faster R-CNN.

Method	Average precision			Processing time (s)
	Easy (%)	Moderate (%)	Hard (%)	
Faster R-CNN [2]	87.33	86.67	76.78	2
Soft-NMS	88.66	87.27	76.79	2
Context-aware RoI pooling	88.05	90.84	80.16	2
MobileNet	86.25	86.07	76.18	0.15

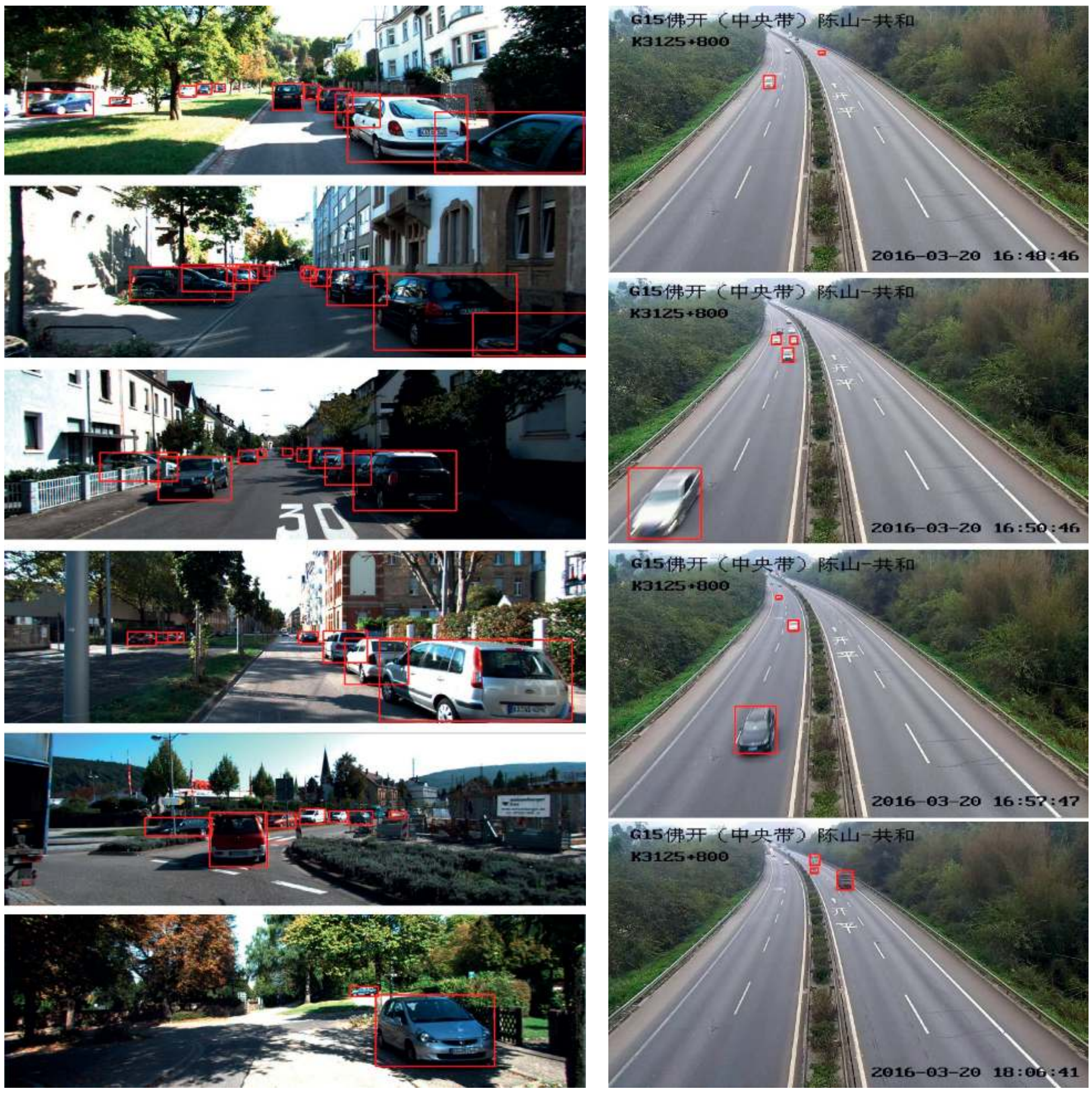


(a) (b)

Figure 7: Detection results of the proposed method (a) and the original Faster R-CNN framework (b) on the KITTI validation set. As shown, the proposed approach can detect more small vehicles and avoid removing positive vehicle proposals unexpectedly (shown in the first row and the third row) compared to the original Faster R-CNN.

Table 5: Detection results of the proposed method and other methods on the KITTI test set.

Method	Average precision			Processing time (s)
	Easy (%)	Moderate (%)	Hard (%)	
Faster R-CNN [2]	86.71	81.84	71.12	2
SSD [18]	77.71	64.06	56.17	0.06
MS-CNN [4]	90.03	89.02	76.11	0.4
YOLO [20]	47.69	35.74	29.65	0.03
YOLOv2 [19]	76.79	61.31	50.25	0.03
Proposed approach	89.20	87.86	74.72	0.15



(a) (b)

Figure 8: Detection results of the proposed method on the KITTI test set (a) and the LSVH test set (b).

Table 6: Comparison of the results of the proposed method and other methods on the LSVH test set.

Method	Average precision (%)	Processing time (s)
Faster R-CNN [2]	40.22	0.48
MS-CNN [4]	72.66	0.35
YOLO [20]	23.78	0.03
YOLOv2 [19]	54.00	0.03
Proposed approach	64.72	0.10

4.4.3. Experimental Results on the LSVH Test Set. This paper also compares the proposed method on another public dataset: LSVH dataset [7]. In this paper, the eight videos in the sparse group are used for training the proposed network, and the left four videos in the sparse group are used for testing the proposed network. To avoid retrieving similar images, this paper extracts one frame in every seven frames of these videos as the training/testing images as in [7]. Table 6 shows the comparison of detection results on the LSVH test set. As shown in Table 6, the performance of the proposed method is improved comparing with the Faster R-CNN framework by 24.5%. Figure 8 shows detection results of the proposed method on the LSVH test set (the right column).

5. Conclusions

Most of state-of-the-art approaches on vehicle detection are focused on detection accuracy. In driving environment, apart from the detection accuracy, the inference speed is also a large concern. Moreover, vehicles are unlikely to be equipped with high-end graphic cards as powerful as used in research environments. Thus, it is necessary to build a faster framework for vehicle detection in driving environments. In this paper, an improved Faster R-CNN framework for fast vehicle detection is proposed. To improve the detection accuracy and the inference time in the challenging driving environment such as large vehicle scale variation, vehicle occlusion, and bad light conditions, MobileNet architecture is first adopted to build the base network of the Faster R-CNN framework. Soft-NMS algorithm is used after the region proposal network to solve the issue of duplicate proposals. Context-aware RoI pooling is then used to adjust the proposals to the specified size without sacrificing important contextual information. Furthermore, the structure of depthwise separable convolution in MobileNet architecture is adopted to build the classifier at the final stage of the Faster R-CNN framework to classify proposals and adjust the bounding box for each of the proposal. The proposed approach is evaluated on the KITTI dataset and the LSVH dataset. Compared with the original Faster R-CNN framework, the proposed approach showed better results in both detection accuracy and processing time. The results demonstrated that the proposed network is simple, fast, and efficient. Moreover, compared with other state-of-the-art methods on vehicle detection, the proposed framework can easily be extended and applied to the detection and recognition of other types of objects encountered in the driving environment, such as license plate, pedestrian, traffic sign, and so on. The good performance of the proposed algorithm on vehicle detection has a high reference value in the field of intelligent driving. In the future, this paper will investigate more enhancements to improve detection results.

Data Availability

The codes used in this paper are available from the author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

References

[1] J. Huang, V. Rathod, C. Sun et al., “Speed/accuracy trade-offs for modern convolutional object detectors,” 2017, https://arxiv.org/abs/1611.10012.

[2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: towards real-time object detection with region proposal networks,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 91–99, Montreal, Canada, December 2015.

[3] X. Chen, K. Kundu, Y. Zhu et al., “3d object proposals for accurate object class detection,” in *Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS 2015)*, Montreal, Canada, December 2015.

[4] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, “A unified multi-scale deep convolutional neural network for fast object detection,” *Computer Vision—ECCV 2016*, pp. 354–370, Springer, Berlin, Germany, 2016.

[5] A. G. Howard, M. Zhu, B. Chen et al., “MobileNets: efficient convolutional neural networks for mobile vision applications,” 2017, https://arxiv.org/pdf/1704.04861.pdf.

[6] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, “Improving object detection with one line of code,” 2017, https://arxiv.org/abs/1704.04503.

[7] X. Hu, X. Xu, Y. Xiao et al., “SINet: a scale-insensitive convolutional neural network for fast vehicle detection,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 3, pp. 1010–1019, 2019.

[8] N. C. Mithun, N. U. Rashid, and S. M. M. Rahman, “Detection and classification of vehicles from video using multiple time-spatial images,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1215–1225, 2012.

[9] S. Messelodi, C. M. Modena, and M. Zanin, “A computer vision system for the detection and classification of vehicles at urban road intersections,” *Pattern Analysis and Applications*, vol. 8, no. 1-2, pp. 17–31, 2005.

[10] B. T. Morris and M. M. Trivedi, “Learning, modeling, and classification of vehicle track patterns from live video,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 425–437, 2008.

[11] J. Zheng, Y. Wang, N. L. Nihan, and M. E. Hallenbeck, “Extracting roadway background image,” *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1944, no. 1, pp. 82–88, 2006.

[12] T. Gao, Z.-G. Liu, W.-C. Gao, and J. Zhang, “A robust technique for background subtraction in traffic video,” *Advances in Neuro-Information Processing*, pp. 736–744, Springer, Berlin, Germany, 2009.

[13] A. Ottlik and H.-H. Nagel, "Initialization of model-based vehicle tracking in video sequences of inner-city intersections," *International Journal of Computer Vision*, vol. 80, no. 2, pp. 211–225, 2008.

[14] Q. Yuan, A. Thangali, V. Ablavsky, and S. Sclaroff, "Learning a family of detectors via multiplicative kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 514–530, 2011.

[15] J.-W. Hsieh, L.-C. Chen, and D.-Y. Chen, "Symmetrical SURF and its applications to vehicle detection and vehicle make and model recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 6–20, 2014.

[16] R. M. Z. Sun and G. Bebis, "Monocular precrash vehicle detection: features and classifiers," *IEEE Trans. Image Process*, vol. 15, no. 7, pp. 2019–2034, 2006.

[17] W. C. Chang and C. W. Cho, "Online boosting for vehicle detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 3, pp. 892–902, 2010.

[18] W. Liu, D. Anguelov, D. Erhan et al., "Single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, pp. 21–37, Springer, Amsterdam, The Netherlands, October 2016.

[19] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," 2017, https://arxiv.org/abs/1612.08242.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," 2016, https://arxiv.org/abs/1506.02640.

[21] W. Chu, Y. Liu, C. Shen, D. Cai, and X.-S. Hua, "Multi-task vehicle detection with region-of-interest voting," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 432–441, 2018.

[22] Y. Cai, Z. Liu, X. Sun, L. Chen, H. Wang, and Y. Zhang, "Vehicle detection based on deep dual-vehicle deformable Part Models," *Journal of Sensors*, vol. 2017, Article ID 5627281, 10 pages, 2017.

[23] X. Li, Y. Liu, Z. Zhao, Y. Zhang, and L. He, "A deep learning approach of vehicle multitarget detection from traffic video," *Journal of Advanced Transportation*, vol. 2018, Article ID 7075814, 11 pages, 2018.

[24] H. Wang, X. Lou, Y. Cai, Y. Li, and L. Chen, "Real-time vehicle detection algorithm based on vision and lidar point cloud fusion," *Journal of Sensors*, vol. 2019, Article ID 8473980, 9 pages, 2019.

[25] W. Zhang, Y. Zheng, Q. Gao, and Z. Mi, "Part-aware region proposal for vehicle detection in high occlusion environment," *IEEE Access*, vol. 7, pp. 100383–100393, 2019.

[26] K.-J. Kim, P.-K. Kim, Y.-S. Chung, and D.-H. Choi, "Multi-scale detector for accurate vehicle detection in traffic surveillance data," *IEEE Access*, vol. 7, pp. 78311–78319, 2019.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105, Lake Tahoe, NV, USA, December 2012.

[29] R. Girshick, "Fast R-CNN," 2015, https://arxiv.org/abs/1504.08083.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, Providence, RI, USA, June 2012.

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2009.

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, Miami, FL, USA, June 2009.

[33] D. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.