IET Image Processing

Special issue

Call for Papers

Be Seen. Be Cited.
Submit your work to a new IET special issue

"Advancements in Fine Art Pattern Extraction and Recognition"

Guest Editors: Fabio Bellavia, Gennaro Vessio, Giovanna Castellano and Sinem Aslan

Read more

The Institution of Engineering and Technology

Received: 11 October 2020 | Revised: 18 May 2021 | Accepted: 10 June 2021

IET Image Processing

DOI: 10.1049/ipr2.12297

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Accurate and efficient vehicle detection framework based on SSD algorithm

Min Zhao | **Yuan Zhong** | **Dihua Sun** | **Yuhao Chen**

School of Automation, Chongqing University, Chongqing, P. R. China

Correspondence
Min Zhao, School of Automation, Chongqing University, Chongqing 400044, P. R. China.
Email: zhaomin@cqu.edu.cn

[Correction added on 23-July-2021, after first online publication: New co-author Yuhao Chen has been added in this version]

Abstract

Vehicle detection plays an important role in intelligent transportation systems and security. Using the original Single Shot MultiBox Detector (SSD) directly for vehicle detection, lacks accuracy and stability. Moreover, most of the state-of-the-art methods need cost a lot of time to inference. Vehicle detection is often used in complex traffic environments. Therefore, faster detection speed and higher detection accuracy are required. This study is aimed at developing a trade-off between accuracy and speed vehicle detection framework based on the SSD algorithm. To improve the multi-scale detection performance of SSD, semantic information, detailed features and receptive fields are combined to propose the feature pyramid enhancement strategy (FPES). On the other hand, the cascade detection mechanism is proposed to strengthen the positioning capability of SSD and an adaptive threshold acquisition method for object detection module (ODM) stage to improve model accuracy. Finally, a more efficient convolutional network is deployed through network slimming. Experimental results demonstrate that the proposed framework achieves state-of-the-art performance on UA-DETRAC and Udacity benchmarks. Interestingly, the inference time is the lowest for the proposed method than the state-of-the-art methods, promising its application for fast and effective vehicle detection.

1 | INTRODUCTION

Vehicle detection is widely used in computer vision applications such as intelligent transportation and security. With the development of deep learning, the accuracy of 2D vehicle detection is constantly improving, but the cost is that the detection speed is slower. In complex traffic scenes, not only vehicles of different scales and types exist, but also vehicles are easily occluded from other targets, thus fast and accurate vehicle detection is one of the most challenging tasks.

Recently, deep network has achieved remarkable results on image classification tasks, such as convolutional neural networks (CNN) have been widely used in vehicle detection instead of machine learning methods [1–3]. Specifically, CNN-based vehicle detectors can be categorized into multi-stage detectors and single-stage detectors. Multi-stage vehicle detectors usually consist of two steps: The first step is to generate some proposals that may contain vehicle targets, commonly used methods include the region proposal network (RPN) [4], EdgeBoxes [5], and so on. The second step focuses on the classification and regression of the proposals. Multi-stage detectors achieve the highest detection accuracy on multiple popular benchmarks. However, even if compression techniques such as network clipping and quantization are used later, they still have high requirements for the hardware environment, which greatly limits the deployment of vehicle object detection algorithms.

Single-stage anchor point detectors, such as by SSD [6] and YOLO [7], abandon the region proposal generation process, which significantly shorten the algorithmic reasoning time. Even on Titan X graphics cards, real-time detection speeds of more than 20 fps (frames per second) can be obtained. YOLO v3 and SSD are often regarded as the best detectors for the trade-off between accuracy and speed.

The anchor-free detectors [8–10] currently do not lag behind those with anchors in accuracy. However, due to its demand for high-resolution and high-semantic feature maps, the feature pyramid networks (FPN) or hourglass structure has become one of the standard 'accessories', making the inference speed far from real-time detectors such as SSD and YOLO v3. Due to the dual high requirements for accuracy and speed of

vehicle object detection, SSD and YOLO v3 are obviously the most suitable baseline algorithms. However, considering that vehicle object detection is often deployed on the edge of low computing power, YOLO v3 uses a heavier backbone network and high-resolution input, compared to VGG-SSD. Although it has obtained a corresponding improvement in accuracy, video memory has also grown significantly. Therefore, we choose the SSD as the baseline algorithm.

While the SSD algorithm obtains efficient inference speed, it also loses the robustness and precise positioning performance for multi-scale object detection of two-stage detectors such as FPN. To construct a real-time yet powerful vehicle detector, while exploiting the proposal-free scheme, we propose the feature pyramid enhancement strategy (FPES) and cascade detection mechanism (CDM). Specifically, FPES enhances the original SSD feature pyramid from the three perspectives of semantic information, detailed features and receptive fields to improve the multi-scale vehicle detection performance. Based on the cascade detection proposed by RefineDet [11], CDM improves from the two perspectives of feature alignment and sample distribution. Then, by analysing the IOU threshold and sample distribution of the cascaded network, an adaptive threshold acquisition method is proposed. Finally, the speed of the model is improved by network slimming without losing the accuracy of the model.

In summary, the main contributions of this research are described as follows:

1. Comprehensively considering semantic information, detailed features and receptive fields, we propose the FPES method to enhance the quality of feature pyramids, which is beneficial for detecting multi-scale vehicles.
2. Inspired by RefineDet [11] and cascade R-CNN [12], we propose CDM, aiming to strengthen the regression ability of the detector to bridge the gap with the multi-stage detectors. And introducing deformable convolution [13] ensures the alignment between the features of the object detection module (ODM) stage and anchors.
3. Different objects may have different distributions, we propose an adaptive threshold acquisition method for the ODM stage to improve model accuracy.

This paper is organized as follows. We first review the related work of vehicle detection, and then introduce the method of this research in detail. Finally, we present the experiment results on UA-DETRAC [14] and Udacity [15] and conclude this paper.

2 | RELATED WORK

Vehicle detection is a sub-topic of target detection, so it is necessary to study the vehicle object detection algorithm. Vehicle object detection includes 2D vehicle detection and 3D detection. 3D detection has made some progress [16], but the precision and speed are lower than that of 2D detection. However, there are still challenges for accurate and speed 2D vehicle detection.

Since 2015, anchor-based detection algorithms have emerged in an endless stream, which can be roughly divided into two categories: One is multi-stage detectors such as faster R-CNN and cascade R-CNN; the other is single-stage, such as the SSD and the RetinaNet detector. Anchor is a set of rectangular boxes with different aspect ratios and areas for each pixel predefined in a picture. This is helpful for dealing with different scales and aspect ratios of objects. In addition, anchor can also reduce the number of ambiguity samples. The introduction of anchor has greatly promoted the development of the target detection field, but it inevitably brought defects. Hyperparameters are difficult to adjust and generalization performance is poor. A limited anchor is difficult to cover in the shape of all instances, such as small objects and particularly elongated objects. Few effective positive samples and difficult-to-negative samples result in an imbalance of samples and difficulty in training. A number of anchors have significantly increased the memory/video memory requirements, and the amount of calculation has also become very large, increasing the training time.

A vehicle detector based on deep learning always selects an object detection algorithm as a baseline, and then makes some modifications to the suitable data domain to improve performance. The basic idea of the two-stage vehicle detection algorithm is that at first, using sliding windows or RPN generates region proposals that may contain vehicles, then, region proposals are reclassified by support vector machine (SVM) or fully connected layers. One typical example is networks based on faster R-CNN. Therefore, there are two directions to enhance the performance of these networks: improve RPN or the detector head network. Considering setting the same number of anchors for each scale in RPN, GP-FRCNN [17] establishes scene geometric relationships to sparse anchor distribution, which effectively improves the detection performance of faster R-CNN on vehicle targets of extremely large and small scale. Experiments on the UA-DETRAC vehicle detection data set show that compared with faster R-CNN, GP-FRCNN improves the accuracy by 19 points, but the inference speed also decreases from 11 to 4 fps. Through the analysis of RPN, researchers found that the classification of anchors by RPN does not use the feature of the internal region of anchors. To obtain more precise proposals, they designed evolving boxes (EB) [18], which performs roI pooling at the RPN stage to obtain anchor's region feature. On the UA-DETRAC vehicle detection dataset, the accuracy of EB is nine points higher than that of faster R-CNN, and the inference speed is only 1 fps. SINet [19] has made two improvements relative to faster R-CNN: One is to propose CARoI pooling, which performs RoI pooling and fusion in multiple different feature layers to obtain semantics and detailed feature, to solve the problem when the vehicle scale is smaller than the RoI pooling output size. The second is to use different weights heads to predict for different scale of the proposals. There is still other work to improve the performance of vehicle detection from the perspective of multi-task learning.

In addition to improving the existing detection algorithm, vehicle target detection has another solution – multi-task learning to share knowledge and improve detection performance. To solve each task, the network will share features, thus enriching the expression of features and improving the performance of each task. Detection and annotation for vehicles (DAVE) [20] completely decouples the two steps of the multi-stage detector.

Fast vehicle proposal network (FVPN) is a shallow network to quickly generate proposals containing vehicles. Attributes learning network (ALN) uses a deep backbone network to perform feature extraction and multi-attribute learning on proposals, such as type, colour and so on. These algorithms are modified based on multi-stage detectors and have achieved extremely high detection accuracy on multiple public benchmarks (KITTI [21], UA-DETRAC), but they are still far away from the real-time detection. For example, the inference speed of EB on the TITAN X card is only 4 fps.

All the above-mentioned vehicle objection detection algorithms can be regarded as two-stage detectors, whose core is focused on how to improve the detection accuracy, while ignoring the detection speed. In fact, vehicle target detection is often used in intelligent driving and security situation. The accuracy of the algorithm is important, but the inference speed and the number of parameters are also important.

Recently, vehicle detectors based on single-stage target detection algorithms have been working more and more in the past two years. CMNet [22] uses YOLO V3 [7] as the baseline algorithm and employs a carefully designed Connetct-and-Merge Block, which can effectively improve the flow of information in CNN. Based on this, a new backbone, connect-and-merge residual network (CMRN), was proposed and instead of Darknet to detect vehicles. LRF-Net [23] designs a lightweight feature pyramid without pre-trained weights and merges it with the original SSD feature pyramid to supplement detailed features, and then introduces an additional top-down path to enhance semantic information at all levels of the feature pyramid. In the UAVDT [24] dataset, the accuracy of LRF-Net exceeds RetinaNet by about four points, which effectively improves the detection performance of small-scale vehicles. However, the network adds more convolution layers and upsampling layers than the SSD, which reduces the inference speed of the detector. Compared with LRF-Net, our work not only supplements the semantic information and detailed features of the feature pyramid, but also improves the quality of features from the perspective of the receptive field, and further improves the performance of detecting small-scale vehicles. Inspired by RefineDet, we also introduced a cascade detection strategy for a single-stage detector, which can significantly improve the performance of the algorithm.

3 | OUR APPROACH

In this section, we give a detailed introduction to our method. Section 3.1 describes the FPES, and then introduces the CDM in Section 3.2. Finally, Section 3.3 describes the overall structure of our method.

3.1 | Feature pyramid enhancement strategy

It has been explained in [25] that the useful features for multi-scale object detection have the following three properties: (1) contains fine details to present the structure of the object; (2) obtained by a deep enough transformation function, so that the high-level knowledge of the target is included in the features. Taking the vehicle as an example, the low-level knowledge may feature such as colour and shape, and the high-level features may be more discriminative features such as the shadow of the car's bottom and the relative position of the window. These features often need to be obtained by a deep network; (3) encode meaningful semantic information to support the prediction of the exact location and category labels of targets that are difficult to detect due to insufficient appearance information of the object (such as being obscured and blurred shapes).

The first prediction layer in the SSD feature pyramid comes from the shallow Conv4-3. Most of the improvement work [26, 27] on SSD are adding the context information and high-level knowledge extracted from the high-level features to Conv4-3 to help detect small scale targets, like FPN [28]. However, these works ignore two issues: (1) FPN only supplements the semantic information of shallow features but ignores the detailed features of high-level features. (2) Compared with high-level features, conv4-3 is shallow, but it still experienced ten-time convolution and three-time pooling, thus which could lose some important low-level details. The visualization results of conv4-3 feature maps are shown in Figure 2. Even though some vehicle targets are detected in the feature map, but the activation area is very small and the information contained is extremely limited.

According to the above discussion, to construct a feature pyramid suitable for multi-scale vehicle detection, we design an independent shallow network branch parallel to the backbone network, downsample the input image and input it into the branch, to obtain the detailed features that can represent the structure of the vehicle. Specifically, to obtain the low-level features of the object details, the training image is pooled downsampled to the resolution of conv4-3 layer, then downsampled image is processed by several convolution modules; to obtain the high-level knowledge of the object, the fc7 layer, the last layer of the backbone network, is sampled up and fused with conv4-3 layer in the channel dimension. Considering the redundancy and partial lack of context information contained in fc7 layer, therefore, in order to extract meaningful context information in fc7 layer, 128 convolution kernels (1024 is the channel number of fc7 layer feature graph) are added to fc7 layer for online feature screening before fusion. By fusing the original features of conv4-3, the features extracted from the downsampling image and the meaningful features selected from fc7 layer, the three 'good' features mentioned above can be satisfied at the same time.

Furthermore, inspired by RFBNet [29], we employ the atrous spatial pyramid pooling (ASPP) [30] module to effectively expand the receptive field to enhance the detection performance of small-scale vehicles. Based on the above analysis, we propose the FPES.

As shown in Figure 3, first, downsampling the image and extracting features through three convolution modules, in which the convolution module is composed of convolution layer, batch normalization (BN) layer and ReLu layer. It will obtain the features of the encoded fine details. On the other hand, sufficient semantic information features (fc7) and high-resolution features (Conv4-3) are generated by the backbone network, then these features are aggregated through a concatenation operation

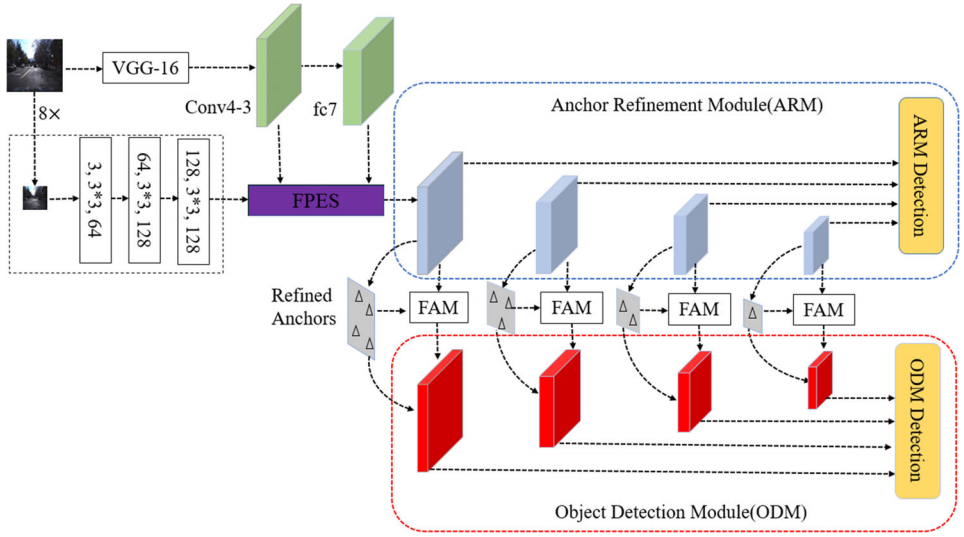

FIGURE 1 An overview of our proposed framework. We exploit FPES to enhance the original features, and then employ upsampling and downsampling to build a feature pyramid at the ARM stage, and finally use the FAM module to generate the feature pyramid at the ODM stage

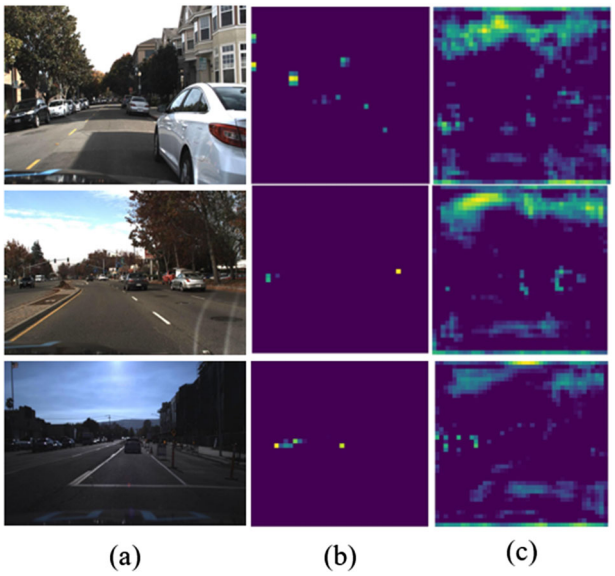

FIGURE 2 Feature map visualization. From left to right: Original image, SSD features from Conv4-3, and refined features from the feature pyramid enhancement strategy. The original features of the Conv4-3 activation area and the receptive field are small, and the vehicle target of the corresponding scale can not be well detected. The feature by refined is significantly improved

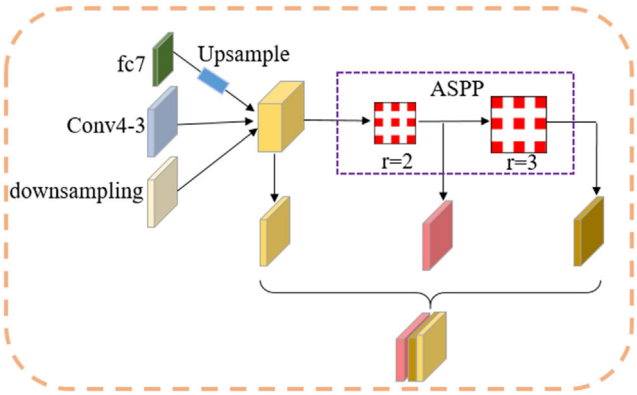

FIGURE 3 Schematic diagram of FPES structure. Downsampling indicates the features of shallow branch extraction parallel to the backbone network. Aggregate fc7, Conv4-3 and downsampling features, and exploit ASPP module to obtain multi-level receptive field features to comprehensively improve feature quality

and the ASPP module with three branches. Finally, we obtain the feature maps that encode enrichment of semantic information, fine details and sufficient receptive fields, and then build a feature pyramid by upsampling and downsampling operations based on this feature.

3.2 | Cascade detection mechanism

The performance of a single-stage detector is weaker than that of a multi-stage detector, mainly due to the following reasons: (1) multi-stage detector has more detection times than a single-stage detector; (2) due to the filtering of a large number of negative samples in the pre-stage, the multi-stage detector greatly eases the imbalance between positive and negative samples in the training process; 3) multi-stage detector has independent features for each stage of classification and regression; 4) RoI pooling or RoI align operation makes the multi-stage detector exploit a region feature instead of point feature, which is also the most fundamental difference from the single-stage detector. According to the above analysis, inspired by RefineDet, we introduced CDM to the SSD to improve vehicle detection accuracy.

Similar to RefineDet, we call the first detection process is anchor refinement module (ARM) and the second detection process is ODM. If the cascade detection is applied directly to the SSD, it will cause a serious misalignment of the ODM

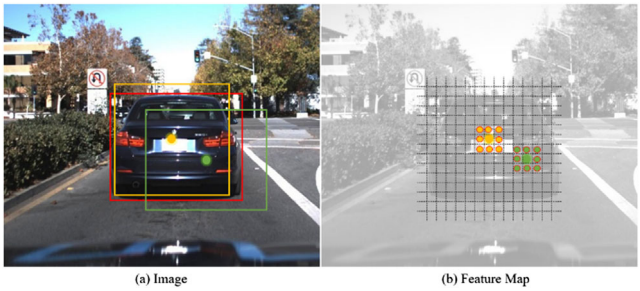

FIGURE 4 Schematic diagram of the misalignment between feature and anchor into the ODM stage. Green box denotes the original anchor box, yellow box represents refined proposal by ARM stage, and red box is ground truth. The regression of the ARM stage caused the position shift of the original anchor

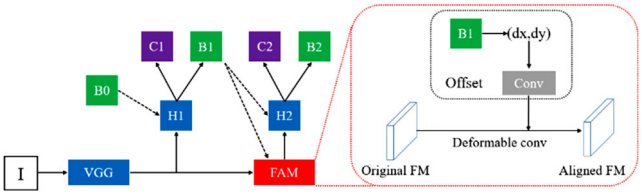

FIGURE 5 Schematic diagram of FAM. The original anchor is refined to the proposal after the ARM regression, and at the same time uses the FAM module to modify the original features to complete the second detection of the proposal

stage features and anchor. As shown in Figure 4, the original anchor (green box) is regression yellow proposal after the ARM stage, the position and size change significantly, and the centre point shifts accordingly. At the ODM stage, the yellow proposal is used as its anchor, and it is classified and returned again. For a single-stage detector, the feature that represents the anchor is the point feature whose centre point corresponds to the location in the feature map. If the original feature map is used directly or a few convolution modules are simply added for the second prediction, then the features used for the yellow proposal in ODM are still obtained from the green points in the feature map, thus leading to the misalignment between anchor and feature.

Inspired by [31], we proposed the feature alignment module (FAM) by introducing deformable convolution to mitigate the misalignment. To better encode the changes in the position of the anchor into the feature map of the ODM stage, we convolve the output of the ARM regression branch to obtain the offset of the deformable convolution [13], thereby more efficiently and trying to ensure the alignment between anchors and features of ODM stage.

As shown in Figure 5, 'H' the network header, 'B' the bounding box and 'C' classification. B0, B1 denote original anchor and refined anchor by ARM, respectively. B1 outputs four-dimensional variables (dx, dy, dh, dw), where (dx, dy) corresponds to the offset of the spatial position, (dh, dw) represents the offset on the scale. The deformable convolution offset is generated by convolution of (dx, dy). The whole process of FAM can be expressed by the formula:

$$\begin{aligned} \Delta p^i &= f(dx^i, dy^i) \\ FM^{i+1} &= Deformable\,(FM^i, \Delta p^i) \end{aligned} \tag{1}$$

where FM denotes feature map, f denotes convolution operation. Specifically, 2D convolution includes three steps: first, sample in the input feature map; second, use convolution kernel weight matrix for weighted summation; finally, restore the matrix back to the image. For any point p_0 on the conventional convolution output feature map, it can be calculated by the following formula:

$$y(p_0) = \sum_{p_n \in \mathfrak{R}} w(p_n) \cdot x(p_0 + p_n) \tag{2}$$

where w denotes the weight matrix and $x$denotes the input feature map. For deformable convolution, $\mathfrak{R}$ will be modified by the offset $\{\Delta p_n | n = 1, \ldots, N\}$, here $N = |\mathfrak{R}|$. Then the calculation formula for any point on the deformable convolution output feature map becomes the following formula:

$$y(p_0) = \sum_{p_n \in \mathfrak{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n) \tag{3}$$

On account of the regression of the ARM stage, the ODM stage obtained more positive sample anchors and high-quality anchors (such as anchors with a target IoU greater than 0.8). Cascade R-CNN pointed out that in the classification task, specifying different IoU thresholds to divide positive and negative samples will cause the regression branch to behave very differently. Specifically, when a low IoU threshold is used, samples with low IoU perform better; conversely, samples with high IoU are more suitable for high IoU thresholds. This essentially reveals that the distribution of training samples must be adjusted as the anchor distribution changes. The anchor in the ODM stage has high quality and many positive samples, so a higher IoU threshold is set to divide positive and negative samples.

In summary, compared with Refinedet, our CDM uses the spatial prediction value of ARM to generate the deviation of the convolution sampling points, thereby using deformable convolution to alleviate the misalignment between feature and anchor in the ODM stage. Further, to employ high-quality anchors, we set a higher IoU threshold to determine positive and negative samples, inspired by the cascade R-CNN.

3.3 | Overall architecture

As shown in Figure 1, the FPES strategy is used to enhance the SSD feature pyramid, which is then used as the binary classification (foreground and background) and first regression of the ARM stage, then the refined anchor of the ARM stage as the initial anchor of the ODM stage. FAM is a FAM proposed in CDM. To ensure the inference speed of the model, we set the number of channels on all FAM output feature maps to 256.

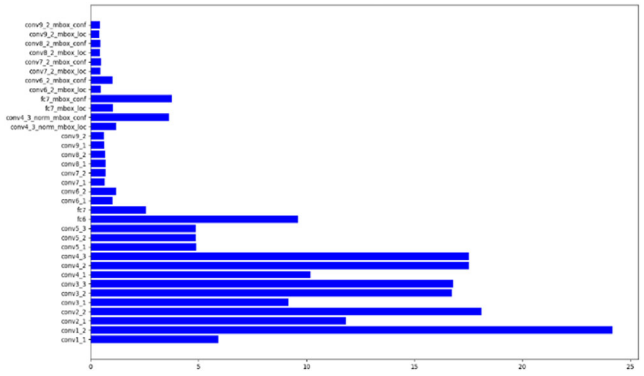

FIGURE 6 Inference time of each layer of SSD on Jeston TX1

The network loss function consists of two parts: the loss in the ARM stage and the loss in the ODM stage. The classification loss function uses the SoftMax cross-entropy loss function, and the regression loss function uses the smooth-l1 loss function. In short, the loss function of an ARM can be calculated as:

$$Loss_{ARM} = \frac{1}{N_{ARM}}\left(\sum_i L_{b-cls}(p_i, [l_i^* \geq 1]) + \lambda \sum_i [l_i^* \geq 1] L_{reg}(x_i, g_i^*)\right) \quad (4)$$

where N_{ARM}denotes the positive sample number of the ARM stage, p_i, x_i, l_i^*andg_i^*represent confidence, regression value, category label and the location attribute of the matched target of the i-th anchor, respectively. $[l_i^* \geq 1]$represents a symbolic function, $L_{b-cls}$and L_{reg}represents binary loss function and regression loss function, respectively. Similarly, the loss function of ODM can be calculated as:

$$Loss_{ODM} = \frac{1}{N_{ODM}}\left(\sum_i L_{cls}(c_i, l_i^*) + \lambda \sum_i [l_i^* \geq 1] L_{reg}(t_i, g_i^*)\right) \quad (5)$$

The total loss function is the loss sum of the ARM and ODM stage.

3.4 | Network slimming

As shown in Figure 6, we can find SSD inference time is mainly concentrated in the front layer of the network. The reason is that the feature map of the front layer has a large resolution. Therefore, this paper starts with a shallow feature map to prune the model layer by layer.

VGG-16 uses pooling layers with a stride step of two to downsample after Conv1-2, Conv2-2, onv3-3. Changing the stride step of these convolution layers to two and remove these pooling layers, so the resolution of the output feature map is reduced by four times. In addition, the running time of the pooling layer is reduced, which further improves the inference speed. All modified convolutional layers are initialized with the weights corresponding to the original model. To reduce the loss of model accuracy, this paper uses the feature map after the original model pooling to constrain the output of the modified convolutional layer. The loss function of the fine-tuning process is shown in the equation:

$$Loss = Loss_{ARM} + Loss_{ODM} + \alpha \cdot \sum_{i=1}^{3} \left\| Y_0^i - Y_1^i \right\|_2^2 \quad (6)$$

Taking Conv1-2 as an example, increasing its step size to 2, $Y_1^1$is the output of Conv1-2 after network slimming, $Y_0^1$is the output of the original model pool1 layer. Add the L2 norm of the difference between Y_1^1 and Y_0^1 to the loss function to constrainY_1^1. Make it as close to Y_0^1 as possible. Probably close. $\alpha$is the balance factor.

4 | EXPERIMENT

4.1 | Data set

In this section, we present a series of experiments on the UA-DETRAC vehicle benchmark with 140K captured frames and 1.2 M labelled vehicles, which contain 83,791 images for training and 56,340 images for testing. The UA-DETRAC dataset is extremely challenging owing to its large variation. Udacity is another computer vision algorithm evaluation data set in the field of autonomous driving, created by Udacity and the founder of Google's driverless car. The Udacity data set contains two subsets: Udacity (CrowdAI) and Udacity (Autti). The first subset is mainly about driving in Mountain View, California and neighbouring cities under daylight conditions. A total of 9423 pictures with a resolution of 1920 × 1200 are provided by CrowdAI via machine learning and by humans. The second subset has nearly 5500 more images than the first subset, and all are annotated by humans.

Evaluation is done using the Matlab code provided by the authors of datasets. A strict training and validation protocol were followed, and the testing data was never seen by the network before the final evaluation. The metric used for evaluation is the average precision (AP), the AP, with a minimum IOU of 0.7 between inferred and ground-truth bounding boxes. The minimum IOU is the minimum overlap of a bounding box with the ground truth to be considered a true detection. The IOU is computed as the intersection of the boxes divided by the union of the boxes.

To evaluate the effectiveness of our proposed method, we concentrated on the following questions: (1) The influence of different components on vehicle detection performance in our proposed method; (2) the performance comparison of our algorithm with other advanced vehicle detection algorithms. For a fair comparison, we split the overall train set with 56K images for training and 28K images for validating. Therefore, we report the results of the first set of experiments on the validation set and compare the results of the test set with other

TABLE 1 Anchor size setting of Udacity dataset

Feature resolution	Anchor size
40 ×40	(19,18),(30,26)
20 ×20	(47,35),(75,52)
10 ×10	(121,91)
5 ×5	(253,210)

TABLE 2 Anchor size setting of UA-DETRAC dataset

Feature resolution	Anchor size
128 ×128	(29,24), (41,32)
64 ×64	(51,45),(68,38),(68,59),(95,49)
32 ×32	(94,87),(130,62)
16 ×16	(160,104)(235,104)
8 ×8	

state-of-the-art methods. All experiments are implemented on Ubuntu 16.04 with four GPUs (NVIDIA GeForce RTX 2080TI).

4.2 | Training details

We train all models on Pytorch 1.1 framework. We set the batch size to 10, the initial learning rate to 10^{-6} and linear growth to 4×10^{-3}, next apply cosine attenuation strategy for learning rate, ending up in the 100 epochs. In the meantime, we employ a weight decay of 5×10^{-4} for regularization and momentum of 0.9.

We use *k*-means clustering algorithm to cluster the anchors into six categories for the Udacity dataset. The cluster centre box is assigned to four levels of feature pyramid as anchor size, and the results are shown in Table 1. The feature resolution corresponds to the input size 320 ×320 in Udacity dataset. (19, 18) means that 19 is the width of anchor and 18 is the length of anchor. The input size is 512 ×512 in UA-DETRA dataset.

The selection of the IoU threshold for the ARM and the ODM is the key to getting high-quality anchors. According to cascade R-CNN, in the classification task, specifying different IoU thresholds to divide the positive and negative samples will cause different performance of the regression branch. This essentially reveals that the distribution of training samples must be adjusted as the anchor distribution changes. Because the ODM stage has more high-quality anchors, we need to set a higher IoU threshold. What's more, different objects may have different distributions, for example, the maximum IoU of some targets and anchor is only 0.6, while some targets may have multiple anchors whose IoU exceeds 0.9. So, for each target, different IoU thresholds need to be set to adapt to the distribution of samples.

Based on what we discussed, we propose an adaptive threshold acquisition method. The method is as follows:

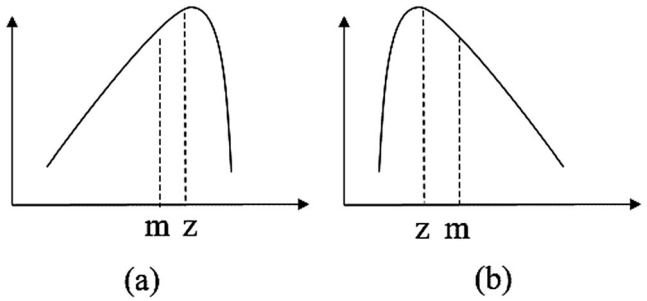

FIGURE 7 Mean and median positions in left and right-skewed distributions

1. For any targets, calculate the IoU between it and all anchors, and put the anchor with IoU greater than 0.5 in **Ω**;
2. Calculate m, the mean value of IoU of all anchors in **Ω**and targets;
3. Calculate p, the skewness of IoU between all anchors in **Ω**and the target;
4. If $p < 0$, $I = \max(m, 0.7)$; if $p > 0$,$I = \min(m, 0.7)$

$$p = E\left[\left(\frac{IoU_i - m}{\delta}\right)^3\right] \tag{7}$$

where $IoU_i$is *i*th IoU in**Ω**, I is IoU threshold, the IoU threshold used in ARM is 0.5, which is consistent with standard SSD. For the ODM stage, first, we use 0.5 as the IoU threshold to get all samples that have a higher coincidence with the target. Then calculate $m$and$p$. 0.7 in the fourth step is the reference threshold, because the prediction box with IoU over 0.7 is regarded as the correct result in UA-DETRAC. If $p < 0$, IoU is a left-skewed distribution with a minimum value. As shown in Figure 7a, $m$and$z$the mean and median respectively. At this time, the average value will be pulled towards the minimum value. If the mean is higher than 0.7, it indicates that there are more anchors with IoU higher than 0.7. At this time, 0.6 is used as the IoU threshold to be harmful to the regression of higher IoU samples, so take the larger value of the mean and 0.7 As the IoU threshold. And if $p > 0$, it means that the IoU is a right-skewed distribution and there is a maximum value, as shown in Figure 7b. At this time,m will be pulled toward the maximum value. If $m <$ 0.7, it indicates that the IoU values of most anchors are less than 0.7. Then if 0.7 is used as the IoU threshold, the number of positive samples will be too small, and overfitting is prone to occur. Therefore, the minimum value between $m$and 0.7 is used as the IoU threshold.

4.3 | Ablation study

For a fair comparison, all ablation experiments apply the same training strategy.

Table 3 demonstrates the ablation study of switching off different components of our proposed FPES. The SSD baseline algorithm achieved 78.45% AP, semantic in Table 3 indicates

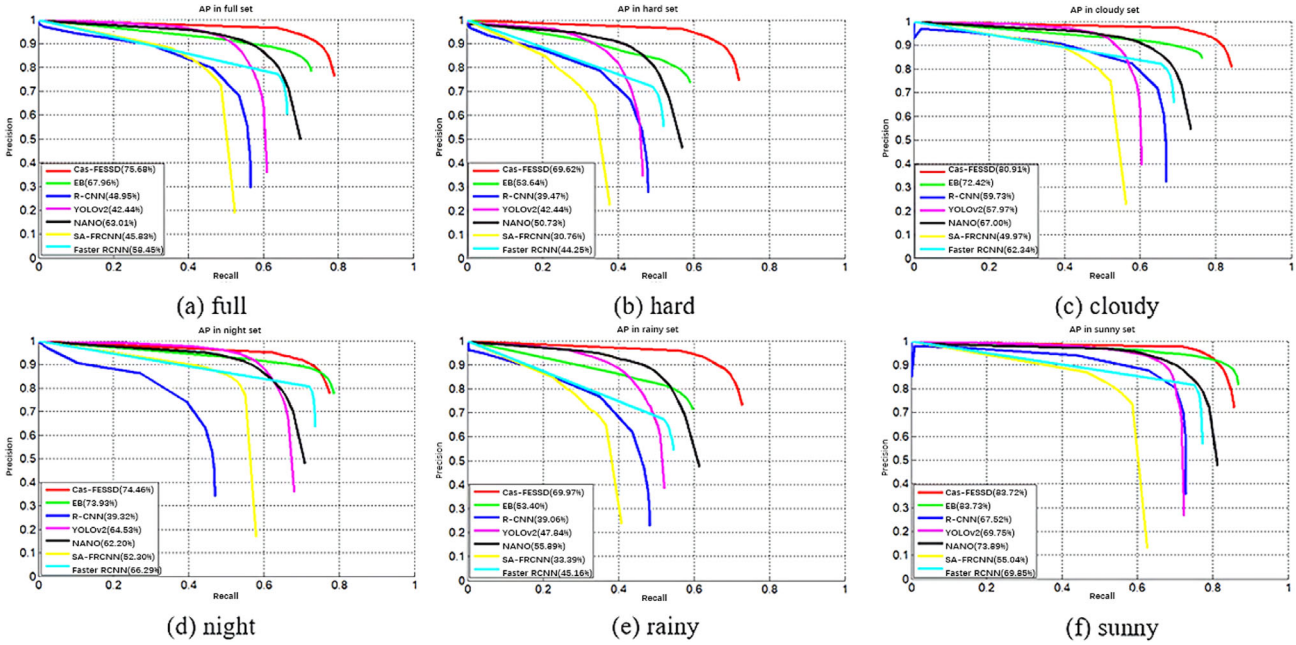

FIGURE 8 Comparison of precision-recall (P-R) curve between our proposed method and other vehicle detection algorithm on the test set of full, hard, cloudy, night, rainy and sunny respective

TABLE 3 Ablation studies on switching off different components of our proposed FPES. We illustrate average precisions (AP) on the UA-DETRAC validation set

Algorithmic setting	AP (%)
SSD	78.45
Semantic	79.02
Semantic + Detail	80.23
Semantic + Detail + ASPP	80.84

TABLE 4 Control experiments on switching off different components of our proposed CDM. Cascade indicates that only the original SSD's prediction was classified and regressed again, FAM and sample mean feature alignment and increased IoU

Algorithmic setting	AP (%)
SSD+FPES	80.84
Cascade	82.56
Cascade + FAM	83.51
Cascade + FAM + sample	84.33

that we upsample the features of fc7 and fuse with conv4-3 to compensate for the semantic information of the shallow layer, which can slightly improve detection performance (e.g., 0.57%). The detail of Table 3 represents a short path that retains fine details in parallel with the backbone network, which can significantly improve detection accuracy, and proves the importance of sufficient detailed features for vehicle detection and the pyramid of SSD seriously loses the detailed information. In addition, we also introduced the ASPP module to expand the receptive field, which further improves the detection accuracy of our method.

Table 4 indicates the importance of different elements of CDM. The baseline result is SSD + FPES. We first directly introduced the cascade detection, just like RefineDet, which greatly boost the detection performance to 82.56%. Then we employ deformable convolution to mitigate the misalignment between features and anchor in the ODM stage, as shown in the third row of Table 4, an one-percent detection performance improvement indicates the effectiveness of deformable convolution. Finally, we exploited the increasing IoU threshold to determine positive and negative samples, as illustrated in the last row of Table 4 which improved accuracy 0.82% AP.

4.4 | Comparison with state-of-the-art frameworks

Figure 8 and Table 5 demonstrate the accuracy and spent comparison of our method with the state-of-the-art vehicle detection frameworks. Precision–recall curves and APs are reported. DPM [32] is a representative algorithm for vehicle detection using the machine learning method; however, its accuracy is only 25.70%, which is far lower than the CNN-based vehicle detection algorithm. Selective search for object recognition (SSOR) [33] uses the extremely lightweight ResNet-10 as the backbone network and becomes the fastest detection algorithm on UA-DETRAC benchmark, which achieves 34 fps inference speed on 1080 GPU card, but its accuracy is only 59.07%. Compared with multi-stage vehicle detectors such as

TABLE 5 Comparison of accuracy (overall) and speed (fps) on the UA-DETRAC test set. The method lateral CNN in blue fonts is an anonymous submission algorithm

Method	Overall	fps	GPU	Easy	Hard	Cloudy	Night	Rainy	Sunny
DPM [32]	25.70	0.17	–	34.42	17.62	24.78	30.91	25.55	31.77
R-CNN [36]	48.95	0.10	–	59.31	39.47	59.73	39.32	39.06	67.52
Faster-CNN [4]	58.45	11	Titan X	82.75	44.25	66.29	69.85	45.16	62.34
YoLov2 [37]	57.72	–	–	83.28	42.44	57.97	64.53	47.84	69.75
SSDR [33]	59.07	34	1080	77.84	45.98	62.79	60.88	48.55	74.32
EB [18]	67.96	10	Titan X	89.65	53.64	72.42	73.93	53.40	83.73
R-FCN [34]	69.87	6	Titan X	93.32	54.31	74.38	75.09	56.21	84.08
CSP [38]	77.67	4	K40	93.65	64.54	86.81	80.63	61.39	89.66
GP FR-CNN [17]	77.96	4	K40	92.74	67.22	83.23	77.75	70.17	86.56
HAT [35]	78.64	3.6	Titan X	93.44	68.04	86.27	78.00	67.97	88.78
FG-BR [39]	79.96	10	M40	93.49	70.78	87.36	78.42	70.50	89.89
Lateral CNN	67.25	21.5	Titan X	89.56	51.61	69.11	74.36	55.77	78.66
Cas-FESSD	75.68	34	2080Ti	83.46	68.32	80.91	74.38	69.14	82.54

TABLE 6 Comparison of pruning scheme in udacity

Method	GPU	AP (%)	fps
SSD	2080Ti	74.78	**116**
Cas-FESSD	2080Ti	**80.86**	79
Directly prune	2080Ti	80.12	91
Loss (α= 0.2)	2080Ti	80.52	91
Loss (α= 1.0)	2080Ti	80.49	91
Loss (α= 0.5)	2080Ti	**80.63**	**91**

EB, R-FCN and GP faster R-CNN, the inference of our method is three times faster than theirs without losing accuracy. As an anchor-free detector, center and scale prediction (CSP) [34] has an overall performance of 77.67%, but it is limited to a heavy backbone network with a speed of only 4 fps. Lateral CNN is currently the fastest anonymous submission detection algorithm on the leader board, but whether it is accuracy or speed, it lags far behind our proposed method. FG-BR [35] is one of the best vehicle target detection algorithms, its accuracy exceeds our approach by more than four points, but the speed lags by more than three times. Specifically, FG-BR is significantly better than our method in the easy part of the test set and performs similarly in the hard part. Therefore, on the UA-DETRAC dataset, our algorithm is currently one of the best vehicle detection algorithms for accuracy and speed trade-offs. Figure 9 demonstrates qualitative evaluations of our method on the test set. We have successfully detected most vehicles with different appearances, especially in the case of heavy occlusion or when the vehicle is far from the camera so that the scale of the vehicle is too small. However, our algorithms still face some challenges. Our algorithm only has an 83.46% AP for easy-to-detect datasets. From Figure 9, we can find that our algorithm is not very good for large object detection, for example, if a bus or truck is close to the camera, it is easy to increase the false detection rate. What's more, the easy part with more large-scale targets does not perform well due to the excessive background noise introduced by high-level features. In addition, the 'grey area' introduced by random erasure reduces the two scene areas of extremely high brightness and extremely low brightness during training, so the performance of Cas-FESSD is not ideal in night and sunny parts.

In conclusion, the detection results of our method are of high quality and can be used for other downstream tasks, such as vehicle re-identification and colour recognition.

4.5 | Layer pruning strategy of backbone network

Compared with the SSD baseline algorithm, Cas-FESSD significantly improved the detection accuracy. The AP increased by 6.08%, but the inference speed was also reduced from 11 6 to 79 fps. If the first three pooling layers in VGG-16 are directly cropped and retrained, the final accuracy drops by 0.74% and the speed increases by 12 fps. Using the output of the pooling layer in Cas-FESSD proposed in Section 3.4 to supervise the corresponding convolutional layer and setting differences to balance the multi-task loss function, the accuracy after retraining finally only drops by 0.23% (α= 0.5). Compared with the standard SSD algorithm, the accuracy is still improved by about 6% while maintaining the inference speed of 91 fps.

As can be seen from Table 7, our network after pruning is called Cas-FESSD (prune), and the time in the table represents the average time of network inference.

Cas-FESSD not only maintains the fast detection performance of SSD, but also significantly improves the accuracy of vehicle target detection, which is mainly due to the FPES and the cascade detection algorithm. The input sizes of YOLO v3 and RetinaNet affect their speed.

FIGURE 9 Detection results on the UA-DETRAC vehicle test set

TABLE 7 Comparison of performance of this paper and another detection algorithm on the Udacity dataset

Method	Backbone	Input Size	GPU	AP (%)	Time(ms)
SSD	VGG-16	300 × 300	Titan X P	74.78	10
YOLO v3	Darknet53	416 × 416	Titan X P	78.93	26
RetinaNet	ResNet-101	832 × 500	Titan X P	82.04	84
Cas-FESSD	VGG-16	320 × 320	Titan X P	80.86	16
Cas-FESSD (prune)	VGG-16	320 × 320	Titan X P	80.63	14

In general, our network is a trade-off between accuracy and speed vehicle detection framework based on the SSD algorithm.

5 | CONCLUSION

In this paper, we propose a novel and effective strategy named feature pyramid enhancement strategy (FPES) based on semantic information, detailed features and receptive field, which significantly enhance the quality of the feature pyramid. In addition, inspired by RefineDet, we introduced cascade detection into vehicle detectors to improve positioning capabilities. In order to maximize the advantages of cascade detection, we proposed the CDM from the perspective of alignment between feature and anchor and input sample distribution. Extensive experiments on the UA-DETRAC benchmark indicate our framework is able to produce remarkably excellent detection performance in respect of both accuracy and speed.

ORCID

Yuan Zhong https://orcid.org/0000-0003-3548-823X
Dihua Sun https://orcid.org/0000-0001-6559-1495

REFERENCES

1. He, K., et al.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016
2. Krizhevsky, A., et al.: ImageNet classification with deep convolutional neural networks. In: Proceeding IEEE Conference on Neural Information Processing Systems (NIPS), Lake Tahoe, Nevada, USA, 1097–1105, 3-8 December 2012
3. Szegedy, C., et al.: Rethinking the inception architecture for computer vision. In: Proceeding IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016
4. Ren, S., et al.: Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39(6), 1137–1149 (2017)
5. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: European Conference on Computer (ECCV), Zurich, Switzerland, 6–12 Sep. 2014

6. Liu, W., et al.: SSD: single shot multiBox detector. In:European Conference on Computer (ECCV), Amsterdam, The Netherlands, 11–14 Oct. 2014
7. Joseph, R., Ali, F.: YOLOv3: an incremental improvement. available at https://arxiv.org/abs/1804.02767, accessed 8 Apr 2018
8. Law, H., Deng, J.: CornerNet: detecting objects as paired keypoints. In: European Conference on Computer (ECCV) Munich, Germany, 8–14 Sep. 2018
9. Duan, K., et al.: CenterNet: keypoint triplets for object detection. In: IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 27 Oct.–2 Nov. 2019
10. Tian, Z., et al: FCOS: fully convolutional one-stage object detection. In: IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 27 Oct.–2 Nov. 2019
11. Zhang, S., et al.: Single-shot refinement neural network for object detection. In: Proceeding IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018
12. Cai, Z., Vasconcelos, N.: Cascade R-CNN: delving into high quality object detection. In: Proceeding IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018
13. Dai, J., et al.: Deformable convolutional networks. In: IEEE International Conference on Computer Vision, Venice, Italy, 22–29 Oct. 2017
14. Lyu, S., et al.: UA-DETRAC 2017: report of AVSS2017 & IWT4S challenge on advanced traffic monitoring. In Proceeding 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), Lecce,Italy, 29 Aug.–1 Sept. 2017
15. Udacity self driving car, available at https://github.com/udacity-/self-driving-car, accessed 2016
16. Dai, D., et al.: Image guidance based 3D vehicle detection in traffic scene. Neurocomputing 428, 1–11 (2021)
17. Amin, S., Galasso, F.: Geometric proposals for faster R-CNN. In: IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 29 Aug.–1 Sept. 2017
18. Wang, L., et al.: Evolving boxes for fast vehicle detection. In: IEEE International Conference on Multimedia and Expo (ICME), Hong Kong, China, 10–14 July 2017
19. Hu, X., et al.: SINet: a scale-insensitive convolutional neural network for fast vehicle detection. IEEE Trans. Intell. Transp. Syst. 20(3), 1010–1019 (2019)/bib>
20. Zhou, Y., et al.: Fast automatic vehicle annotation for urban traffic surveillance. IEEE Trans. Intell. Transp. Syst. 19(6), 1973–1984 (2018)
21. Geiger, A., et al.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012
22. Zhang, F., et al.: CMNet: a connect-and-merge convolutional neural network for fast vehicle detection in urban traffic surveillance. IEEE Access 7, 72660–72671 (2019)
23. Wang, T., et al.: Learning rich features at high-speed for single-shot object detection. In:IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 27 Oct.–2 Nov. 2019
24. Yu, H., et al.: The unmanned aerial vehicle benchmark: object detection, tracking and baseline. Int. J. Comput. Vision 128(5), 1–19 (2020)
25. Ren, J., et al.: Accurate single stage detector using recurrent rolling convolution. In: IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017
26. Fu, C., et al.: DSSD: deconvolutional single shot detector. available at https://arxiv.org/abs/1701.06659, accessed 23 Jan 2017
27. Zhao, Q., et al.: M2Det: a single-shot object detector based on multi-level feature pyramid network. In: AAAI Conference on Artificial Intelligence. Hawaii, USA, 2—9 Feb. 2021
28. Lin, T., et al.: Feature pyramid networks for object detection. In: IEEE Conference on IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017
29. Liu, S., et al.: Receptive field block net for accurate and fast object detection. In: European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 Sep. 2018
30. Zhang, H., et al.: Cascade RetinaNet: maintaining consistency for single-stage object detection. available at https://arxiv.org/abs/1907.06881, accessed 16 Jul 2019
31. Dollar, P., et al.: Fast feature pyramids for object detection. IEEE Trans. Pattern Anal. Mach. Intell. 36(8), 1532–1545 (2014)
32. Object detection with discriminatively trained partbased models. IEEE Trans. PAMI 32(9), 1627–1645, (2010)
33. Uijlings, J., et al: Selective search for object recognition. Int. J. Comput. Vision 104(2), 154–17 (2013)
34. Dai, J., et al.: R-FCN: object detection via region-based fully convolutional networks. In: Proceeding IEEE Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain, 5–10 Dec. 2016
35. Wu, S., et al.: Hierarchical attention for part-aware face detection. Int. J. Comput. Vision 127(6), 560–578 (2019)
36. Girshick, R., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Columbus, OH,USA, 23–28 June 2014
37. Redmon, J., Ali, F.: YOLO9000: better, faster, stronger. In: IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017
38. Liu, W., et al.: High-level semantic feature detection: a new perspective for pedestrian detection. In: IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019
39. Fu, Z., et al.: Foreground gating and background refining network for surveillance object detection. IEEE Trans. Image Process. 28(12), 6077–6090 (2019)
40. Yang, Z., et al: RepPoints: point set representation for object detection. In: IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 27 Oct.–2 Nov. 2019
41. Lin, T., et al.: Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. 42(2), 318–327 (2020)

How to cite this article: Zhao, M., et al.: Accurate and efficient vehicle detection framework based on SSD algorithm. IET Image Process. 15, 3094–3104 (2021). https://doi.org/10.1049/ipr2.12297