Parallel Multi-Branch Convolution Block Net for Fast and Accurate Object Detection

Lei Fu 1*, Wenbin Gu 1*, Lei He 1, Ting Rui 1, Liang Chen 1,2*, Yongbao Ai 1 and Fanjie Meng 1

1 College of Field Engineering, Army Engineering University of PLA, Nanjing 210007, China; fulei10@mails.jlu.edu.cn (L.F.); lgdxhelei@163.com (L.H.); rtinguu@sohu.com (T.R.); chenbb0708@163.com (L.C.); aybjackai@163.com (Y.A.); beilimeng1992@163.com (F.M.)
2 Automobile NCO Academy, Army Military Transportation University, Bengbu 233011, China
* Correspondence: guwenbin1@aliyun.com; Tel.: +86-025-8082-1422

Received: 3 November 2019; Accepted: 17 December 2019; Published: 21 December 2019

Abstract: In order to maintain the high speed advantage of single-stage object detector and improve its detection accuracy, in this paper, we propose a parallel multi-branch convolution block, called PMCB, which can efficiently extract multi-scale object information at a specific layer to form a discriminative feature layer and boost the detection performance with little computational burden. Based on the PMCB module, we build PMCB Net on the basis of the single shot multibox detector (SSD) network by replacing the conventional convolution with PMCB at a specific layer. The performance of the proposed algorithm is compared with that of other state-of-the-art methods on PASCAL VOC2007, MS COCO test datasets. The experimental results show that the proposed algorithm greatly improved detection accuracy performance while only adding a negligible computational burden, which is very important for practical engineering applications.

Keywords: object detection; real-time; parallel multi-branch convolution block; multi-scale information

1. Introduction

Object detection has a wide range of applications in real life, such as unmanned aerial vehicle aerial object detection, pedestrian detection, vehicle detection, traffic sign detection, and optical remote sensing object detection. In recent years, with the progress of convolutional neural network (CNN) architectures [1–9], object detectors have also made great progress. The current object detectors are mainly divided into single-stage object detectors and two-stage object detectors. The two-stage object detectors have made continuous progress in accuracy [10–13]. They divide the detection problem into two steps. The first step is to provide features with stronger representation capabilities through the regional proposal network, and the second step performs classification and regression based on the features to complete object detection. It is widely believed that strong representational capabilities are critical to the performance of object detectors. In order to enhance the representation capabilities, the following methods are mainly adopted: deeper backbone networks and fusion of deep features and shallow features of CNN. For deep networks, the two-stage detector often uses ResNet-101 [3], Hourglass [14] as their backbone. In terms of feature fusion, the most typical representative method is the feature pyramid network (FPN) [15]. By introducing a top-down structure, it enhances the shallow representation and improves the overall performance of the detector. All of these methods are designed to improve the representation ability of the feature layer, making it discriminative, thereby improving the performance of the detector. However, whether a deep network or a feature fusion mechanism, it is inevitable to introduce too much computational load, which makes the detector greatly degraded in terms of detection speed, and it is difficult to meet the applications with high real-time requirements.
In order to satisfy real-time applications, some researchers dive into single-stage detectors, which give up the region proposal step of two-stage object detectors and directly uses the single-step output for classification and regression. The most representative jobs are YOLOv1 [16] and SSD [17]. YOLOv1 performs classification and regression directly on a single feature to obtain test results. SSD obtains detection results by using multiple feature layers for classification and regression. They all showed the advantage of the single-stage object detector in terms of detection speed.

However, due to the abandonment of the regional proposals, that is, the inability to utilize better features with high discriminative power, the detection performance often lags behind the two-stage object detector.

In order to shorten the performance difference between a single-stage object detector and two-stage object detector, people are constantly striving to improve the performance of a single-stage object detector. Subsequently, with a large number of jobs such as [18–23], the performance of a single-stage object detector has been continuously improved, so that they are getting close to the performance of the two-stage object detector. However, some of these performance improvements are due to deeper network structures, and some use more time-consuming FPN structural feature fusion methods, which reduce their detection efficiency, and some even can not meet real-time applications. Based on real-life requirements for detection speed and accuracy, it is particularly important to build a real-time and accurate object detector.

In this paper, we focus on improving single-stage detector’s performance while maintaining high detection efficiency. According to the above discussion, a feasible solution is to use a lightweight network in a single-stage object detector to meet the speed requirement while increasing the feature representation capability of each scale detection layer.

For enhanced feature representation capability, in the field of classification, GoogleNet [24] and its subsequent variants [25–27] use multiple branches of CNN to capture information of different scales. In the segmentation domain, the ASPP Net [28] uses multi convolution with different dilation rates to extract multi-scale feature enhancement representation capability. Inspired by these works, based on an SSD detector, we replaced the original convention convolution layer with a parallel multi-branch convolution block (PMCB) with different receptive fields of view to obtain more discriminative multi-scale object features. In order to reduce the amount of computation, we use atrous convolution and group convolution. The large convolution kernel convolution in each individual branch is decomposed into two smaller convolution kernel convolution, thereby increasing the representation capability, which plays an important role in improving detector performance. Our main contributions are summarized below:

• We propose a PMCB module to enhance the feature representation capability of each detection layer of a single-stage object detector.
• We propose a PMCB Net single-stage object detector, which greatly improves detection accuracy without FPN-like feature fusion, while maintaining real-time speed.
• By conducting experiments on the PASCAL VOC2007 [29] and MS COCO [30] datasets, we have demonstrated that our PMCB Net has certain advantages in speed and accuracy compared to other state-of-the-art single-stage detectors.

2. Related Work

Researchers have put plenty of effort into improving the detection accuracy of single-stage object detector due to its high efficiency compared with two-stage detectors, e.g., R-CNN [31], Fast R-CNN [32], Faster R-CNN [33], and R-FCN (Region-based Fully Convolution Networks) [34]. We classify these works into two categories.

2.1. Detecting at Multiple Feature Layers

The former YOLO detector YOLOv1 can detect objects on the final feature map, which has a faster detection speed but has difficulty in accurately detecting small objects. Naturally, semantic
information of objects can exist in different layers, depending on its size. For a large object, the strong semantic information that is critical for an object detector will emerge in later layers. For a small object, the finely detailed information will disappear in later layers, and less discriminative semantic information in the early layer. Thus, an exploit multiple CNN layer is proposed, SSD makes use of multi-scale feature maps to handle objects of various sizes, and each layer is responsible for certain scale objects. MS-CNN [35] uses a deconvolution layer to upsample feature maps to increase feature map resolution, and object detection is performed at multiple output layers, each focusing on objects within certain scale ranges. Though adopting multiple feature layers, the bottom layers still have wake semantic information, which will hamper the detection ability to small objects.

2.2. Detecting at Multiple Feature Layers That Fused Multi-Scale Features

To further increase the performance of single-stage detector, some recent works prove to not only detect at multiple layers but also combine features from different layers. Representative methods include DSSD (Deconvolutional Single Shot Detector) [18], MDSSD (Multi-scale Deconvolutional Single Shot Detector) [19], RetinaNet [20] and RefineDet [36], and RFBNet [37]. RetinaNet uses a top-down architecture with lateral connections to build high-level semantic feature maps at all scales. The top layer that is spatially coarser, but semantically stronger, is used to enhance the features of bottom pathway via lateral connections and up-sampling. DSSD replaces the backbone of VGG (Visual Geometry Group) to Resnet-101, and adds a deconvolution and skip connection module to integrate information from earlier feature maps and the deconvolution layers. MDSSD enhances feature expression by selectively merging deep layers into shallow features. RefineDet proposes an anchor refinement module to obtain better initialization for subsequent regressor and design a transfer connection to transfer the feature to improve the accuracy. RFBNet takes the relationship between the size and eccentricity of RFs into account, to enhance the feature discriminability and robustness. The following method also adopts the similar fusion method, StaireNet [38] adopts a top-down feature fusion method, and ESSD (Extend Single Shot MultiBox Detector) [39] and CFENet [40] adopt the adjoin layer feature fusion method. We have summarized these feature fusion methods into FPN-like feature fusion methods. Although these fusion methods enhance the representative ability of features, they also consume a lot of computing resources.

For multi-scale feature acquisition, different approaches are used in other areas. In the field of segmentation, DeepLabv3 [28] has achieved comparable performance with other state-of-the-art models on PASCAL VOC 2012 segmentation benchmark. The author attributes it to the modified atrous spatial pyramid polling (ASPP), it adopts multi-branch convolution block with different dilation rates, which is also called dilation convolution. Pyramid scene parsing Net (PSP) [41] performs spatial polling at several scales and also acquires high performances on several semantic segmentation benchmarks. These methods perform feature enhancement at the final convolutional layer, which is different for the object detection to be enhanced in multiple feature layers, and different feature layers have different semantic information. How to better acquire multi-scale information of different layers is therefore worth studying.

3. Proposed Method

In this section, we quickly review the principle of SSD, then introduce our PMCB module and describe the PMCB Net detector architecture, and finally the training schedule.

3.1. SSD

SSD combines predictions from multiple feature maps with different resolutions to handle objects of various sizes. It makes use of a VGG network as a features extractor and adds extra convolution layers to handle multi-scale objects. It abandons the RPN (Region Proposal Network) stage and straightforward makes classification and regression. Predictions at multi-scale improve mean average precision (mAP) and single-stage improves the speed. SSD even outperforms Faster R-CNN in 2016.
For more details, please refer to [17]. However, due to wake semantic information on each layer, the overall performance still needs to be improved.

### 3.2. Parallel Multi-Branch Convolution Block

The Inception Net series [24–27] adopt the inception module. Figure 1a is an example of the inception module in Inception-v4. The inception module is composed of multi-branch convolution. DeepLabv3 makes use of four parallel atrous convolution with different dilation rates at the end of the feature map, which can effectively capture multi-scale information. It adds batch normalization within ASPP. Figure 1b illustrates the structures of the modified ASPP block. PSP Net uses the pyramid pooling module shown in Figure 1c to gather context information. Using four pooling kernels cover the whole, half of, and small portions of the image, then concatenate the four-level feature with the original feature. In order to enhance representation ability, we delicately design parallel multi-branch convolution block (PMCB) to obtain more discriminative features, which are shown in Figure 1d. We fuse the advantage of inception and ASPP, and make lots of modifications. To be specific, first, we modify the dilation rate to be smaller, e.g., 3 and 5, compared with ASPP Net 6, 12, 18, which is more likely to be suitable for detection. Second, we divide $3 \times 3$ convolution to $3 \times 1$ and $1 \times 3$ convolution, and add batch normalization and relu at every convolution layer to increase represent ability. Third, we adopt group convolution at a particular convolution layer to further decrease parameters and prevent overtraining; the details will be explained bellow.

![Figure 1](image1.png)

**Figure 1.** Comparisons of multi-branch feature acquire module; the PMCB module is used to extract multi-scale features at multiple scale feature layers.

There are two different fusion blocks for different scale objects; we respectively design it for specific objects. Figure 2a,b illustrate PMCB-a and PMCB-b block, respectively. They have some common characteristics: they both fuse features under five different pyramid scales. The first branch incorporates global context information by global average polling (GAP). The other branches from left to right adopt gradually increasing dilation rates respectively to acquire different scale information. Features of different scales are concatenated, then pass $1 \times 1$ BC-R, and finally add a shortcut layer.
We use convolution with kernel size 3 as the main convolution. In order to increase representation ability, $3 \times 1$ and $1 \times 3$ convolution replaces the $3 \times 3$ convolution. Every branch block adopts BasicConv (convolution+batchnormalization+relu), which is helpful for training and high performance. We delicately designed the kernel size of each convolution so that it can use group convolution to reduce parameters. Taking conv4 layer as an example, it has 512 out channels, the feature size is $38 \times 38$; when using the architecture of ASPP instead of the L2 norm layer, it would yield about 8.92 M parameters and the FLOPs (Floating Point operations) is 12.5 G, but 0.97 M parameters and 1.36 G FLOPs for PMCB. In addition, for layer conv7 with 1024 channels, the parameters will be 35.66 M and 2.52 M respectively for ASPP and PMCB, respectively. The two blocks also have some differences. First, PMCB-a utilizes convolution with no dilation rate at the second branch of $3 \times 1$ and $1 \times 3$ convolution, however, the PMCB-b utilizes convolution with dilation 2, the first $3 \times 3$ convolution of the fifth branch has dilation rate of 1 and 2, respectively. The reason is that, in the shallow layer, we think convolution with several relatively small receptive field is more appropriate, while, in the deep layer, convolution with a large receptive field is more suitable for larger objects. Second, the stride is one for the first PMCB-b, and two for other PMCB-b; in order to produce a feature pyramid, it will be explained bellow. One more thing needs to be noticed; for the fifth branch, we use $3 \times 3$ convolution for better visualization, in fact; we also use $3 \times 1$ and $1 \times 3$ convolution. In Figure 2, BCG indicates BasicConv plus group convolution, BC indicates BasicConv, and BC-R indicates BasicConv without relu activation. GAP indicates global average pooling.

![The architecture of PMCB-a](image1.png)

![The architecture of PMCB-b](image2.png)

**Figure 2.** PMCB-a is used in the shallow layer (conv4-3), a smaller dilation convolution rate is used to capture smaller objects. PMCB-b is used in another layer; a bigger dilation convolution rate is used to capture bigger objects.

### 3.3. PMCB Net Detection Architecture

Based on SSD, we construct our PMBC Net object detector. Taking PMCB Net300 as an example, we place the L2 norm layer with a PMBC-a block, and replace conv7, conv8, conv9 with the PMBC-b block; the change can be seen in Figure 3. In order to increase the accuracy on the premise of efficient enough, PMCB Net keeps the detection architecture of SSD. It uses a small multi-branch receptive of field convolution for the shallow layer, and bigger dilation convolution for the deep layer. Group convolution is adopted for future reduce parameters. The first PMCB-b module uses a stride equal to one to maintain the original resolution; the others use a stride equal to two to obtain pyramid features, the last two convolutions remain unchanged relative to SSD. For PMCB Net512, we also replace an L2 norm layer with a PMCB-a module, and replace the following four convolution layers of SSD to the PMB-b module, and the last two convolutions remain in reserve.
Figure 3. The architecture of PMCB Net300; we replaced the original L2 norm, conv7, conv8, and conv9 layer for the PMCB module.

4. Experiments Results and Discussion

In order to testify the effectiveness of the PMCB module, we adopt nearly the same training methods as SSD, including hard negative, the default box, and data augmentation, we also use smooth L1 loss for location and softmax loss for classification. We train PMCB Net using PyTorch, on a machine with two Tesla v100 GPUs (NVIDIA, Santa Clara, CA, USA). We evaluate PMCB Net on PASCAL VOC and MS COCO. The metric for evaluating detection performance is mAP.

4.1. PASCAL VOC2007

In this experiment, we train PMCB Net on the union of VOC2007 and VOC2012 train and validation dataset, which includes 16,551 images, and test on the VOC2007 test dataset with 4592 images. The training batch size is set to 32, and total iteration is 120 K, the learning rate warmed up from $1 \times 10^{-6}$ to $4 \times 10^{-3}$ at the first five epochs, and then reduced by 1/10 at 80 K and 100 K iterations. The SGD (Stochastic Gradient Descent) optimizer with a momentum of 0.9 and a decay of 0.0005 is used; the convolution parameters of the newly added layers are all initialized by the Kaiming_normal method [42], the weights of bn layers are all initialized by 1, and the bias of bn layers are all initialized by 0.

We compare PMCB Net with other state-of-the-art detectors in Table 1. Our model with $300 \times 300$ input produces 80.4% mAP. It exceeds the baseline SSD300 by 2.9 points, and even obtains the accuracy of R-FCN [34], which is based on the ResNet-101 backbones and uses a large input size of about 1000 × 600. By using the $512 \times 512$ size, PMCB Net achieves 82.4% mAP; this exceeds the baseline SSD512 by 2.8 points, surpassing the modified version of SSD e.g., DSSD513 [18], MDSSD512 [19], RefineDet512 [36]. Comparing with two-stage methods, PMCB Net512 performs better than ION (Inside-Outside Net) [43], MS-CNN (Multi-scale Convolution Neural Network) [35], and R-FCN [34]. For certain categories, such as bottles and plants belonging to small objects, PMCB Net300 improves the accuracy 10.0 and 5.3 points respectively compared with SSD300, and PMCB Net512 improves the accuracy 5.0 and 6.1 points respectively compared with SSD512. In terms of comprehensive overall performance, it proves that the PMCB module is useful for improving object detection accuracy.
Table 1. Detection results on PASAL VOC2007 test set. All methods are trained on VOC2007 and VOC2012 trainval dataset, and tested on the VOC2007 test set. The performance is measured by mAP, the bold represents the best performance.

| Method       | Network | mAP  | Aero | Bike | Bird | Boat | Bottle | Car  | Cat  | Chair | Cow  | Dog   | Horse | Mbike | Person | Plant | Sheep | Sofa  | Train | tv  |
|--------------|---------|------|------|------|------|------|--------|------|------|-------|------|-------|-------|-------|--------|-------|-------|-------|------|-----|
| Faster [33]  | ResNet-101 | 76.4 | 79.8 | 80.7 | 76.2 | 68.3 | 55.9   | 85.1 | 85.3 | 89.8  | 56.7 | 87.8  | 69.4  | 88.3  | 88.9   | 80.9  | 78.4  | 41.7  | 78.6 | 79.85 | 85.3 | 72.0 |
| ION [43]     | VGG     | 75.6 | 79.2 | 83.1 | 77.6 | 65.6 | 54.9   | 85.4 | 85.1 | 87.0  | 54.4 | 80.6  | 73.8  | 85.3  | 82.2   | 82.2  | 74.4  | 47.1  | 75.8 | 72.7  | 84.2 | 80.4 |
| MS-CNN [35]  | VGG     | 78.2 | 80.3 | 84.1 | 78.5 | 70.8 | 68.5   | 88.0 | 85.9 | 87.8  | 60.3 | 85.2  | 73.7  | 87.2  | 86.5   | 85.0  | 76.4  | 48.5  | 76.3 | 75.52 | 85.0 | 81.0 |
| R-FCN [34]   | ResNet-101 | 80.5 | 79.9 | 87.2 | 81.5 | 72.0 | 69.8   | 86.8 | 88.5 | 89.8  | 67.0 | 88.1  | 74.5  | 89.8  | 90.6   | 79.9  | 81.2  | 53.7  | 81.8 | 81.5  | 85.9 | 79.9 |
| SSD300* [17] | VGG     | 77.5 | 79.5 | 83.9 | 76.0 | 69.6 | 50.5   | 87.0 | 85.7 | 88.1  | 60.3 | 81.5  | 77.0  | 86.1  | 87.5   | 83.9  | 79.4  | 52.3  | 77.9 | 79.5  | 87.6 | 76.8 |
| DSSD321 [18] | ResNet-101 | 78.6 | 81.9 | 84.9 | 80.5 | 68.4 | 53.9   | 85.6 | 86.2 | 88.9  | 61.1 | 83.5  | 78.7  | 86.7  | 88.7   | 86.7  | 79.7  | 51.7  | 78.0 | 80.9  | 87.2 | 79.4 |
| MDSSD300 [19] | VGG     | 78.6 | 86.5 | 87.6 | 78.9 | 70.6 | 55.0   | 86.9 | 87.0 | 88.1  | 58.3 | 84.8  | 73.4  | 84.8  | 89.2   | 88.1  | 78.0  | 52.3  | 78.6 | 74.5  | 86.8 | 80.7 |
| RefineDet320 [36] | VGG     | 80.0 | 83.9 | 85.4 | 81.4 | 73.5 | 60.2   | 86.4 | 88.1 | 89.1  | 62.7 | 83.9  | 77.0  | 85.4  | 87.1   | 86.7  | 82.6  | 55.3  | 82.7 | 78.5  | 88.1 | 79.4 |
| PMCB Net300  | VGG     | 80.4 | 82.8 | 87.5 | 79.6 | 74.3 | 60.5   | 88.0 | 87.3 | 87.8  | 64.0 | 87.3  | 77.9  | 85.4  | 89.0   | 87.5  | 82.0  | 57.6  | 80.2 | 80.4  | 88.0 | 80.0 |
| SSD512* [17] | VGG     | 79.5 | 84.8 | 85.1 | 81.5 | 73.0 | 57.8   | 87.8 | 88.3 | 87.4  | 63.5 | 85.4  | 73.2  | 86.2  | 86.7   | 83.9  | 82.5  | 55.6  | 81.7 | 79.0  | 86.6 | 80.0 |
| DSSD513 [18] | ResNet-101 | 81.5 | 86.6 | 86.2 | 82.6 | 74.9 | 62.5   | 89.0 | 88.7 | 88.8  | 65.2 | 87.0  | 78.7  | 88.2  | 89.0   | 87.5  | 83.7  | 51.1  | 86.3 | 81.6  | 85.7 | 83.7 |
| MDSSD512 [19] | VGG     | 80.3 | 88.8 | 88.7 | 83.2 | 73.7 | 58.3   | 88.2 | 89.3 | 87.4  | 62.4 | 85.1  | 75.1  | 84.7  | 89.7   | 88.3  | 83.2  | 56.7  | 84.0 | 77.4  | 83.9 | 77.6 |
| RefineDet512 [36] | VGG     | 81.8 | 88.7 | 87.0 | 83.2 | 76.5 | 68.0   | 88.5 | 88.7 | 89.2  | 66.5 | 87.9  | 75.0  | 86.8  | 89.2   | 87.8  | 84.7  | 56.2  | 83.2 | 78.7  | 88.1 | 82.3 |
| PMCB Net512  | VGG     | 82.3 | 89.3 | 89.0 | 85.1 | 77.0 | 62.8   | 89.2 | 89.0 | 87.1  | 67.5 | 87.7  | 78.5  | 86.5  | 89.3   | 86.1  | 84.3  | 61.7  | 85.1 | 78.8  | 87.5 | 83.7 |
4.2. Ablation Study

To analyze the effectiveness of different components, we constructed four different experimental parameters and evaluated them on VOC2007. The result is shown in Table 2. In order to make a fair comparison, we all chose the input image size of 300, and all the structures were trained on the union of VOC2007 and VOC2012 train and validation dataset, and tested on the VOC2007 test set. Other training strategies remain unchanged.

Parallel Module: To validate the effectiveness of PMCB module, originally, we design it in the manner of parallel structure; each branch has only one convolution that has the same dilation rate of 3. In this case, it produces 78.5% mAP, delivering a gain of 1.0% in mAP, which indicates that the parallel module is effective for detection.

Moderate Receptive: As previously analyzed, in order to extract features of different scales in a specific layer, we adopt different dilation convolution rates at each branch, 1, 3, 5, and 7 from left to right. As shown in Table 3, 79.2% mAP was obtained in this way. We found that it raised the accuracy 0.7% mAP relative to a sole parallel module with the same dilation rate.

PMCB Module: The existing deep network improves the feature extraction ability by increasing the number of convolutions. In order to minimize the parameters and improve the ability of feature expression, we use the PMCB module. The detailed architecture can be seen from Figure 2. In this design, we also use two methods: one is to remove all RELU in the last convolution layer of each branch, which we call PMCB-RELU; the other is to add RELU in the last convolution layer of each branch, which we call PMCB. In PMCB-RELU, 79.8% of the mAP is obtained, which is 0.6% higher than the parallel convolution module with different receptive fields. When the PMCB module is used, the accuracy reaches 80.4%. We think this is because of the increase of RELU; the features of the model are more restrictive and the generalization ability is stronger.

|                | SSD * | PMCB |
|----------------|-------|-------|
| Parallel Module| ✓ ✓ ✓ ✓|       |
| Moderate Receptive | ✓ ✓ ✓ |       |
| PMCB-RELU       | ✓     |       |
| PMCB            | ✓     |       |
| mAP(%)          | 77.5  | 78.5  |
|                 | 79.2  | 79.8  |
|                 | 80.4  |       |

* indicates that a new data enhancement method is used.

4.3. PASCAL VOC2007 Results Analysis

In order to understand PMCB Net more deeply, we use the method in Diagnosing Error in Object Detectors [44] to analyze PMCB Net300; the experiment is conducted on the PASCAL VOC2007 test dataset. This section makes analysis and research from the following aspects: (1) potential improvements; (2) breaking point analysis; (3) sensitivity and impact of object characteristics.

Potential Improvements: In order to verify the reasons for the potential performance improvement of the algorithm, we selected several classes with larger performance improvement on the SSD baseline for analysis, e.g., bird, boat, and bottle. Figure 4 shows the precision–recall (PR) curves for the three classes. By comparing the PR curve, we can see that our algorithm improves the detection accuracy of small objects more obviously (9.2% for bottle). Medium sized objects also have some improvement (bird for 3.6% and boat for 4.7%). The reasons for the accuracy promotion of small objects are as follows: characteristics of small objects are very weak; especially after a lot of downsampling, the features basically disappear. Because the PMCB module captures the multi-scale features of small object in the shallow layer, it captures the discrimination abilities of the features before its disappearance, and then improves the detection accuracy. The reason for the improvement of the accuracy of the medium-sized object is that the medium-sized object has stronger semantic information than the small object, so the
PMCB module has increased the discrimination of its characteristics on a certain level relative to small objects.

Figure 4. Precision–recall curves of the PASCAL VOC2007 subset classes test dataset.

In order to further verify our viewpoint, we compare the influence of different objects with different sizes on the detection accuracy. Figure 5 shows the comparison between SSD300* and PMCB Net300 for the impact of object sizes. Seven classes were used because they span the major groups of vehicles, animals, and furniture. Comparing the left and right figures, it can be seen that, when the classes are the same, the PMCB Net300 greatly improved the object with a size of XS, and a certain promotion of objects with a size of S, and little promotion of objects with a size of M. PMCB benefits from the performance improvement of these object sizes that improve the overall performance. For each type of L and XL size object, the performance is almost the same. This is precisely because of the effective extraction of shallow features by a PMCB module, which increases the discrimination of features, so that small and medium-sized objects can be successfully detected.

Figure 5. Impact of object size on VOC2007 test using [44]. The plot on the left shows the SSD300* results of BBox Area per class, and the right plot shows the results of PMCB Net300. Key: BBox Area: XS = extra-small; S = small; M = medium; L = large; XL = extra-large; Average Normalized Precision = APN, indicates the precision is normalized across classes; black dashed lines indicate overall APN; ‘+’ with red indicating standard error bars.

Breaking Point Analysis: Although we have greatly improved the accuracy in the three categories above, there is still a big gap with 100% accuracy. We selected some examples of top ranking false positives to further analyze where the algorithm fails. In Figure 6, we present the distribution of top-ranked false positive types. For these three categories, location and background types account for a large proportion. In order to intuitively view the false positive examples, in Figure 7, we visualized the failure case of bird boat and bottle class. It should be noted that, ov equaling 0 means the false positive and true positive don’t belong to the same class; 1-r means the fraction of correct examples that are ranked lower than the given false positive. Taking 1-r = 0.87 as an example, it means that the ranking of this false positive detection is higher than 87% of the correct detection. For the localization type false positive, we think this is because the single-stage detector directly regresses the position of the object, instead of the two-step regression like Faster-RCNN. For the background type false positive,
we think this is because the background has similar features with the object to be detected, and the PMCB module only captures the similar features, which is not enough to make correct detection.

Figure 6. The distribution of top-ranked false positive types. “Loc”: false positive due to bad localization; “Sim”: confusion with similar class; “Oth”: confusion with others; “BG”: confusion with background

Figure 7. Examples of top false positives. The text indicates the type of error (“loc” = localization; “bg” = background; “sim” = confusion with similar object; “oth” = confusion with other VOC objects; “ov” = the amount of overlap with a true object ), and the fraction of correct examples that are ranked lower than the given false positive (“1-r”, for 1-recall).

Sensitivity and Impact of Object Characteristics: In order to analyze the robustness of the algorithm, we analyzed the sensitivity changes of the algorithm to different characteristics using [44]. Figure 8 shows the sensitivity of PMCB Net300 to each characteristic. Here, we use the following characteristics to evaluate the detector, including: occlusion, truncation, size (scale change), aspect ratio, viewpoint (rotation change), and part visibility. Following the method defined in [44], the difference between best and worst indicates sensitivity, the difference between best and overall indicates potential impact of improving robustness. The worst-performing and best-performing subsets for each characteristic are averaged over seven classes. From Figure 8, we can see that PMCB Net300 is very sensitive to occlusion (0.366) and size (0.314), but the impact is small (0.05 and 0.083, respectively) because the most difficult cases are not common. PMCB Net300 is robust to truncation (0.074); for aspect ratio, viewpoint (rotation), and part visibility characteristics, the sensitivity is within 0.19. The size (scale change) and view (rotation change) have greater impact on the detector (0.083 and 0.074, respectively), and the impact of occ, asp, part, and trn gradually become smaller.
4.4. MS COCO

In order to further test the performance of PMCB Net, we also conducted experiments on the MS COCO dataset. We use the trainval35k (118,287 images) for training and testing on standard test-dev (20,288 images) and submit it to the servers for evaluation. All of the training strategies are the same as training on the PASCL VOC except that we decrease the learning rate from \(4 \times 10^{-3}\) to \(4 \times 10^{-4}\) at 80 epochs and divide by 10 at 100 epochs, finally stopping at 120 epochs; each epoch contains 118,287 images.

From Table 3, we can see that PMCB Net300 achieves 30.0% mAP on the test-dev set, which surpasses the baseline SSD300\(^*\) 4.9 points. Our method even slightly surpasses, which uses a larger input size (1000 \(\times\) 600) and employs a ResNet101 as its backbone. PMCB Net300 also surpasses DSSD321, MDSSD300, Yolov2, and RefineDet320.

Table 3. Detection result on COCO test-dev 2015. Almost all of the results are evaluated on the Nvidia Titan X GPU (Santa Clara, CA, USA) except MDSSD, which runs on a 1080 Ti GPU. PMCB Net300\(^*\) and PMCB Net512\(^*\) indicate running on a 1080 Ti GPU, the bold represents the best performance.

| Method            | Data       | Network     | Time   | Avg. Precision, Iou: Avg. Precision, Area: | Avg. Precision, Area: |
|-------------------|------------|-------------|--------|------------------------------------------|-----------------------|
|                   |            |             |        | 0.5:0.95 0.5 0.75 S L M                    | 0.5:0.95 0.5 0.75 S L M |
| Faster [33]       | trainval   | VGG         | 147ms  | 24.2 45.3 23.5 7.7 26.4 37.1             |                      |
| R-FCN [34]        | trainval   | ResNet-101  | 110ms  | 29.9 51.9 - 10.8 85.7 45.0               |                      |
| SSD300\(^*\) [17] | trainval35k| VGG         | 30ms   | 25.1 43.1 25.8 - - -                     |                      |
| DSSD321 [18]      | trainval35k| ResNet-101  | 105ms  | 28.0 46.1 29.2 7.4 28.1 59.3             |                      |
| MDSSD300 [19]     | trainval35k| VGG         | 26ms   | 26.8 46.0 27.7 10.8 - - -                 |                      |
| RefineDet320 [36] | trainval35k| VGG         | -      | 29.4 49.2 31.3 10.0 32.0 44.4             |                      |
| PMCB Net300       | trainval35k| VGG         | 33ms   | 30.0 48.7 31.4 11.5 31.3 45.8             |                      |
| PMCB Net512\(^*\)| trainval35k| VGG         | 22.7ms | 30.0 48.7 31.4 11.5 31.3 45.8             |                      |
| SSD512\(^*\) [17] | trainval35k| VGG         | 53ms   | 28.8 48.5 30.3 - - -                     |                      |
| SSD513 [18]       | trainval35k| VGG         | 182ms  | 33.2 53.3 35.2 13.0 35.4 51.1             |                      |
| MDSSD512 [19]     | trainval35k| VGG         | 58ms   | 30.1 50.5 31.4 13.9 - - -                 |                      |
| YOLOv2 [45]       | trainval35k| darknet     | 25ms   | 21.6 44.0 19.2 5.0 22.4 35.5             |                      |
| YOLOv3 608 [46]   | trainval35k| darknet     | 51ms   | 33.0 57.9 34.4 18.3 35.4 41.9             |                      |
| RefineDet512 [36] | trainval35k| VGG         | -      | 33.0 54.5 35.5 16.3 36.3 44.3             |                      |
| RetinaNet500 [20] | trainval35k| ResNet-101  | 90ms   | 34.4 53.1 36.8 14.7 38.5 49.1             |                      |
| PMCB Net512       | trainval35k| VGG         | 59ms   | 34.1 54.4 36.2 16.3 36.3 48.6             |                      |
| PMCB Net512\(^*\)| trainval35k| VGG         | 41.6ms | 34.1 54.4 36.2 16.3 36.3 48.6             |                      |
For PMCB Net512, it achieves 34.1% mAP on the test-dev set, which surpasses the baseline SSD512* 5.3 points. Our results are even prior to RefineDet512, which simulates a two-stage method for detection and is also prior to DSSD512 that uses ResNet-101 as its backbone. Compared with YOLOv3, we get more accurate accuracy in smaller image sizes (512 vs. 608). PMCB Net512 is slightly inferior to RetinaNet500 (34.1% vs. 34.4%). However, due to the deep backbone and two-stage architecture, its speed is slower than us, which will be discussed below.

4.5. Inference Time

Speed is critical for single-stage object detectors. We intend to improve accuracy while maintaining real-time speed. In order compare with other state-of-the-art detectors, for PASCAL VOC2007, we use 4952 images with batch size 1 to evaluate the inference speed of PMCB Net with a Titan X GPU and 1080 Ti GPU, respectively. The speed can be seen from Table 4. PMCB Net runs at 42 FPS (Frames Per Second) and 20 FPS with input size 300 × 300 and 512 × 512, respectively. Compared with SSD300, PMCB Net300 only sacrifices negligible speed (42 FPS vs 46 FPS) but acquires 2.9 points gains (80.4% vs 77.5%) when using a Titan X GPU. We also tested the performances of PMCB Net on 1080 Ti GPU. PCMB NET300 surpasses MDSSD300, which is designed to improve the detection performance of small objects at speed (72 FPS vs. 38.5 FPS) and accuracy (80.4% vs. 78.6%), and PMCB NET512 is also superior to MDSSD512 in terms of speed (38 FPS vs 17.3 FPS) and accuracy (82.3% vs. 80.3%). Compared with RefineDet320, PMCB NET300 achieves higher accuracy in smaller input size.

Table 4. Comparison of speed and accuracy on PASCAL VOC2007 dataset. All of the methods are trained on the union of VOC2007 and VOC2012 trainval and tested on the VOC2007 test. PMCB Net300* and PMCB Net512* indicate running on a 1080 Ti GPU.

| Method      | Network | GPU   | Input Size | Speed(FPS) | mAP(%) |
|-------------|---------|-------|------------|------------|--------|
| Faster [33] | VGG     | Titan X | ~1000 × 600 | 7          | 73.2   |
| R-FCN [34] | ResNet-101 | Titan X | ~1000 × 600 | 9          | 80.5   |
| SSD300* [17] | VGG | Titan X | 300 × 300 | 46          | 77.5   |
| SSD300* [17] | VGG | 1080 Ti | 300 × 300 | 85          | 77.2   |
| SSD512* [17] | VGG | Titan X | 512 × 512 | 19          | 79.5   |
| DSSD321 [18] | ResNet-101 | Titan X | 321 × 321 | 9.5         | 79.6   |
| DSSD513 [18] | ResNet-101 | Titan X | 513 × 513 | 5.5         | 81.5   |
| MDSSD300 [19] | VGG | 1080 Ti | 300 × 300 | 38.5        | 78.6   |
| MDSSD512 [19] | VGG | 1080 Ti | 512 × 512 | 17.3        | 80.3   |
| RefineDet320 [36] | VGG | Titan X | 320 × 320 | 40.3        | 80.0   |
| RefineDet512 [36] | VGG | Titan X | 512 × 512 | 24.1        | 81.8   |
| PMCB Net300 | VGG | Titan X | 300 × 300 | 42          | 80.4   |
| PMCB Net512 | VGG | Titan X | 512 × 512 | 20          | 82.3   |
| PMCB Net300* | VGG | 1080 Ti | 300 × 300 | 72          | 80.4   |
| PMCB Net512* | VGG | 1080 Ti | 512 × 512 | 38          | 82.3   |

For MS COCO, we also evaluate the detector on a Titan X GPU and a 1080 Ti GPU for fair comparison. MS COCO has 80 categories, and it consumes more time on the Non-Maximum Suppression (NMS) stage. Thus, the whole detection speed is a little slower than that on PACAL VOC. We use 5000 images with batch size 1 to evaluate the inference speed. Table 3 shows that our detector also improves consistently. When using Titan X, PMCB Net runs at 30 FPS and 17 FPS with input size 300 × 300 and 512 × 512, respectively. PMCB Net300 is the most accurate one (30.0% mAP) among the real-time detectors. PMCB Net512 can provide better accuracy and run at a nearly real-time speed. To compare with MDSSD, which is evaluated on 1080 Ti, we also evaluate PMCB Net on a 1080 Ti GPU. The speed can be seen in Table 3. We annotated it as PMCB Net300* and PMCB Net512*, respectively. To make it easier to compare with other state-of-the-art algorithms, we plot the speed accuracy trade-off curve in Figure 9. Our PMCB Net is located in the upper left corner; this shows that,
compared with other state-of-the-art detectors, PMCB Net achieves better trade-off between accuracy and speed.

![Figure 9. Speed accuracy trade-off on MS COCO test-dev.](image)

The accuracy of PMCB Net 512 is a little lower than RetinaNet500 (34.1% vs. 34.4%), but our detector is much closer to real-time than RetinaNet500 (17 FPS vs. 11.1 FPS) on a Titan X GPU. When using a 1080 Ti GPU, PMCB Net512 can reach real-time speed (24 FPS). Two things are worth noting. The first is that we test the SSD300 detector on the same Titan X GPU. The second is that, even if MDSSD uses 1080 Ti, its speed is comparable to the speed at which we use Titan X. Our test results on 1080 Ti surpass the opponent MDSSD in both speed and accuracy.

4.6. Visualization Results

We show the visualization results on the PASCAL VOC2007 test set and MS COCO test dataset in Figures 10 and 11, respectively. For better visual effects, we ignore specific scores and only display the bounding boxes with scores larger than 0.6. All of the results are detected on the size of 300 because we intend on real-time application and an embedded system, which requires high memory and computation. When using 512 × 512 input size, our PMCB Net acquires more accurate results, but consumes more time. The results prove that our detectors are suitable for multi-scale, crowd and small objects. We believe that these improvements are due to our Parallel Multi-branch Convolution Block.

4.7. Discussion

Through the ingenious design of a multi branch feature extraction module, we can not only improve the accuracy of object detection, but also maintain a high detection speed. As we observe an object, if we can combine the features of different feature dimensions of the object, such as the local features of the object, the larger local features, the overall features, and the features with background information, it will improve the accuracy of detection. A multi branch feature extraction module is for simulating this process. However, too many features will reduce the speed of detection, and thus combine with the existing fractional convolution and group convolution to reduce the negative influence as much as possible. As analyzed in Section 4.3, the algorithm causes false positive examples mainly due to localization error and confusion with background; confusion with similar objects without annotation will also produce some false positive examples. The localization type false positive is attributed to a single-stage detector directly regressing the position of the object, instead of the two-step regression like two-stage object detectors. The background type false positive is because of a PMCB module only capturing the similar features, which is not discriminative enough to make correct detection. In terms of overall performance, it still has a large development space for detection speed and accuracy. At present, the updated object detection algorithms, such as the
mechanism of anchor-free [47], detecting objects as paid keypoints [23], and Multi-Level Feature Pyramid Network [48] are promoting the faster and better development of the field of object detection. Whether we can find a better feature fusion method in these new algorithms is worthy of our further discussion.

Figure 10. Qualitative results of PMCB Net300 on the PASCAL VOC 2007 test set (corresponding to 80.4% mAP).
Figure 11. Qualitative results of PMCB Net300 on the MS COCO test data set (corresponding to 30.0% mAP).
5. Conclusions

In this paper, we propose a parallel multi-branch convolution block (PMCB) Net single-stage object detector for fast and accurate object detection. In order to enhance the representation ability of features in different layers, we abandon the ideas of FPN-like methods from the past and adopt a parallel multi-branch convolution block with different dilation rates to extract multi-scale features of objects, and replace the conventional convolution module of SSD in particular layers. Experiments on PASCAL VOC and MS COCO show the effectiveness of PMCB Net for fast and accurate object detection. Our proposed module is light, fast, and easy to integrate with other detection architecture. More importantly, the module can improve accuracy while maintaining real-time inference speed, which is very important for practical applications.

Author Contributions: Conceptualization, L.F. and W.G.; Methodology, L.F.; Software, L.H.; Supervision, T.R.; Validation, L.C.; Resources, Y.A.; Visualization, F.M.; All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Scientific Research Projects of China (No. J20182A020367).

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012.
2. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 2014, arXiv:1409.1556.
3. He, K.; Zhang, X.; Ren, S.; Jian, S. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
4. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv 2016, arXiv:1602.07261.
5. Da, L.; Lin, L.; Xiang, L. Classification of remote sensing images based on densely connected convolutional networks. Comput. Era 2018, 10, 60–63.
6. Ma, N.; Zhang, X.; Zheng, H.T.; Jian, S. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
7. Mahdianpari, M.; Salehi, B.; Rezaee, M.; Mohammadimanesh, F.; Zhang, Y. Very deep convolutional neural networks for complex land cover mapping using multispectral remote sensing imagery. Remote Sens. 2018, 10, 1119. [CrossRef]
8. Liu, W.; Cheng, D.; Yin, P.; Yang, M.; Li, E.; Xie, M.; Zhang, L. Small Manhole Cover Detection in Remote Sensing Imagery with Deep Convolutional Neural Networks. ISPRS Int. J. Geo-Inf. 2019, 8, 49. [CrossRef]
9. Bischke, B.; Helber, P.; Folz, J.; Borth, D.; Dengel, A. Multi-task learning for segmentation of building footprints with deep neural networks. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 1480–1484.
10. Singh, B.; Najibi, M.; Davis, L.S. SNIPER: Efficient multi-scale training. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, Canada, 3–8 December 2018; pp. 9310–9320.
11. Li, Y.; Chen, Y.; Wang, N.; Zhang, Z. Scale-Aware Trident Networks for Object Detection. arXiv 2019, arXiv:1901.01892.
12. Pang, J.; Chen, K.; Shi, J.; Feng, H.; Ouyang, W.; Lin, D. Libra R-CNN: Towards Balanced Learning for Object Detection. In Proceedings of the The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
13. Liu, Y.; Wang, Y.; Wang, S.; Liang, T.; Zhao, Q.; Tang, Z.; Ling, H. CBNet: A Novel Composite Backbone Network Architecture for Object Detection. arXiv 2019, arXiv:1909.03625.
14. Law, H.; Deng, J. CornerNet: Detecting objects as paired keypoints. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 734–750.
15. Lin, T.Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.

16. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.

17. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, Las Vegas, NV, USA, 26 June–1 July 2016.

18. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. arXiv 2017, arXiv:1701.06659.

19. Cui, L. MDSSD: Multi-scale Deconvolutional Single Shot Detector for small objects. arXiv 2018, arXiv:1805.07009.

20. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.

21. Zhu, C.; He, Y.; Savvides, M. Feature Selective Anchor-Free Module for Single-Shot Object Detection. arXiv 2019, arXiv:1903.00621.

22. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. arXiv 2019, arXiv:1904.01355.

23. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. CenterNet: Keypoint Triplets for Object Detection. arXiv 2019, arXiv:1904.08189.

24. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.E.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.

25. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv 2015, arXiv:1502.03167.

26. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2818–2826.

27. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.

28. Chen, L.C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv 2017, arXiv:1706.05587.

29. Everingham, M.; Gool, L.V.; Williams, C.K.I.; Winn, J.M.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. Int. J. Comput. Vis. 2009, 88, 303–338. [CrossRef]

30. Lin, T.Y.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Dollar, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. arXiv 2014, arXiv:1405.0312.

31. Girshick, R.B.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2013; pp. 580–587.

32. Girshick, R.B. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Las Condes, Chile, 11–18 December 2015; pp. 1440–1448.

33. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. 2017, 39, 1137–1149. [CrossRef]

34. Dai, J.; Li, Y.C.; He, K.; Sun, J. R-FCN: Object Detection via Region-based Fully Convolutional Networks. arXiv 2016, arXiv:1605.06409.

35. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection. arXiv 2016, arXiv:1607.07155.

36. Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Single-Shot Refinement Neural Network for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
37. Liu, S.; Huang, D. Receptive field block net for accurate and fast object detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 385–400.
38. Woo, S.; Hwang, S.; Kweon, I.S. StairNet: Top-Down Semantic Aggregation for Accurate One Shot Detection. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1093–1102.
39. Zheng, L.; Fu, C.; Zhao, Y. Extend the shallow part of Single Shot MultiBox Detector via Convolutional Neural Network. arXiv 2018, arXiv:1801.05918.
40. Zhao, Q.; Sheng, T.; Wang, Y.; Ni, F.; Cai, L. CFENet: An Accurate and Efficient Single-Shot Object Detector for Autonomous Driving. arXiv 2018, arXiv:1806.09790.
41. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6230–6239.
42. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Las Condes, Chile, 11–18 December 2015; pp. 1026–1034.
43. Bell, S.; Zitnick, C.L.; Bala, K.; Girshick, R.B. Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2874–2883.
44. Hoiem, D.; Chodpathumwan, Y.; Dai, Q. Diagnosing Error in Object Detectors. In Proceedings of the 12th European Conference on Computer Vision—Volume Part III, Florence, Italy, 7–13 October 2012.
45. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
46. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. arXiv 2018, arXiv:1804.02767.
47. Zhu, C.; He, Y.; Savvides, M. Feature Selective Anchor-Free Module for Single-Shot Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Long Beach, CA, USA, 16–20 June 2019; pp. 840–849.
48. Zhao, Q.; Sheng, T.; Wang, Y.; Tang, Z.; Chen, Y.; Cai, L.; Ling, H. M2Det: A Single-Shot Object Detector based on Multi-Level Feature Pyramid Network. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI, Honolulu, HI, USA, 27 January–1 February 2019.