Parallel Feature Pyramid Network for Object Detection

Seung-Wook Kim[0000−0002−6004−4086], Hyong-Keun Kook, Jee-Young Sun, Mun-Cheon Kang, and Sung-Jea Ko

School of Electrical Engineering, Korea University, Seoul, South Korea
{swkim,hkkook,jysun,mckang}@dali.korea.ac.kr, sjko@korea.ac.kr

Abstract. Recently developed object detectors employ a convolutional neural network (CNN) by gradually increasing the number of feature layers with a pyramidal shape instead of using a featurized image pyramid. However, the different abstraction levels of CNN feature layers often limit the detection performance, especially on small objects. To overcome this limitation, we propose a CNN-based object detection architecture, referred to as a parallel feature pyramid (FP) network (PFPNet), where the FP is constructed by widening the network width instead of increasing the network depth. First, we adopt spatial pyramid pooling and some additional feature transformations to generate a pool of feature maps with different sizes. In PFPNet, the additional feature transformation is performed in parallel, which yields the feature maps with similar levels of semantic abstraction across the scales. We then resize the elements of the feature pool to a uniform size and aggregate their contextual information to generate each level of the final FP. The experimental results confirmed that PFPNet increases the performance of the latest version of the single-shot multi-box detector (SSD) by mAP of 6.4% AP and especially, 7.8% AP_{small} on the MS-COCO dataset.

Keywords: Real-Time Object Detection, Feature Pyramid.

1 Introduction

Multi-scale object detection is a difficult and fundamental challenge in computer vision. Recently, object detection has achieved a considerable progress thanks to a decade of advances in convolutional neural networks (CNNs).

The early CNN-based object detectors utilize a deep CNN (DCNN) model as part of an object detection system. OverFeat [38] applies a CNN-based classifier to an image pyramid in a sliding window manner [5, 7]. The regions with CNN features (R-CNN) method [10] adopts a region-based approach (also known as a two-stage scheme), where the image regions of object candidates are provided for a CNN-based classifier. Recent region-based detectors such as Fast R-CNN [9] and Faster R-CNN [35] utilize a single-scale feature map, which is transformed by a DCNN model, as shown in Fig. 1(a) (top). In [35], using this single-scale feature, a complete object detection system is formed as an end-to-end CNN model and exhibits state-of-the-art performance.

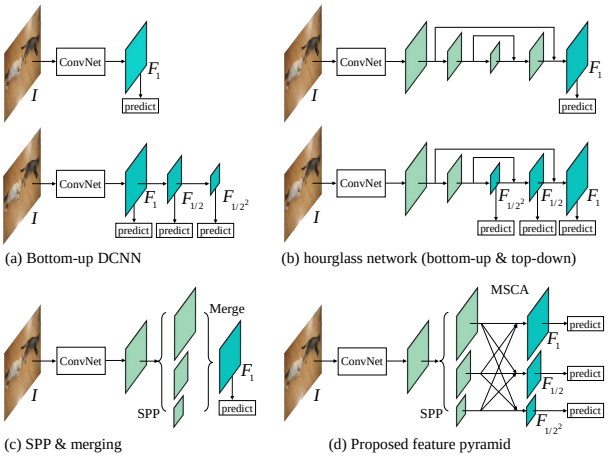

Fig. 1. Variant DCNN models that use a single-scale feature layer for visual recognition and their extensions to feature pyramids: bottom-up DCNN models (a), hourglass networks (b), and SPP-based networks (c); our network model (d) can be viewed as an extended version of (c) for multi-scale object detection.

Inspired by the pyramidal shape of the DCNN feature layers, some researchers have attempted to exploit multiple feature layers to improve the detection performance [2,27]. As shown in Fig. 1(a) (bottom), the single-shot multi-box detector (SSD) [27] utilizes a feature pyramid (FP), each level of which comprises DCNN layers responsible for detecting objects within a certain size range. In addition, SSD, a single-stage detector, is a region-free detector that does not require a region proposal process. By using the FP and a single-stage scheme, SSD exhibits detection performance that is comparable to region-based detectors and high computational efficiency competitive to YOLO [31], which uses a single-scale feature map. However, the object detectors with this FP would poorly perform on the lower feature layers because of their lack of object-level information.

It has been shown that the lower- and upper-level features are complementary to each other and their combinations are beneficial for segmentation [30], keypoint estimation [29], and object detection [22]. In the hourglass model (see Fig. 1(b)), to generate a single high-level feature map, the object knowledge contained in the upper layer is forwarded to the lower layers by appending a top-down module. The low-resolution of the upper layers can ensure invariance to pixel variations, which is helpful for identifying object instances, but it can cause difficulties in pixel-level tasks such as pixel labeling and box regression [13, 28]. Thus, the lateral connections are added between the bottom-up and top-down layers to transfer the information on object details to the top-down layers. A recent trend in single-stage methods is to bring such benefits of the hourglass network into FP-based object detectors [8,20,23,25,43]. The deconvolutional single shot detector (DSSD) [8] forms an hourglass structure that can exploit both the object-level information insensitive to pose and appearance from the upper

layers and the richer spatial information from the lower layers. By appending a top-down module to SSD, DSSD can raise the performance of the region-free object detectors to the level of region-based ones. However, the top-down modules incur additional computational costs, so the speed of region-free methods is no longer their advantage. In recent, the RetinaNet [25] and RefineDet [43] simplify the feature layers of the top-down and lateral paths, and achieve state-of-the-art performance while operating in real time.

In this study, we propose a parallel FP network (PFPNet) to construct an FP by widening the network width instead of increasing its depth. As shown in Fig. 1(d), we first employ the spatial pyramid pooling (SPP) [14] to generate a wide FP pool with the feature maps of different sizes. Next, we apply additional feature abstraction to the feature maps of the FP pool in parallel, which makes all of them have similar levels of semantic abstraction. The multi-scale context aggregation (MSCA) modules then resize these feature maps to a uniform size and aggregate their contextual information to produce each level of the final FP.

To the best of our knowledge, PFPNet is the first attempt to apply the SPP module to a region-free multi-scale object detector using a CNN-based FP. Previous studies have demonstrated that multi-scale representation using the SPP module can greatly improve the performance in terms of object detection [14] and segmentation [44]. These studies utilize the SPP module to produce a single high-level feature vector with a fixed size or a feature map with a fine resolution for making predictions as shown in Fig. 1(c). By contrast, we utilize the SPP module to produce an FP where the predictions are independently made on each level. In summary, this study makes three main contributions:

- We employ the SPP module to generate pyramid-shaped feature maps via widening the network width instead of increasing its depth.
- By using the MSCA module similar to an inception module [41], our model effectively combines the context information at vastly different scales. Since the feature maps of our FP have a similar abstraction level, the difference in performance among the levels of the FP can be effectively reduced.
- We obtained remarkable performance on the public datasets. Using an input size of 512×512, PFPNet achieved the mean average precision (mAP) of 82.3% on the Pascal VOC 2007, 80.3% on the PASCAL VOC 2012, and 35.2% on the MS-COCO, thereby outperforming the state-of-the-art object detectors. Using an input size of 320×320 and 512×512, PFPNet operates at 33 and 24 frames per second (FPS), respectively, on a single Titan X GPU. For small-scale objects, PFPNet increases the performance of the latest version of the SSD [26] by the AP of 7.8% on the MS-COCO dataset.

2 Parallel Feature-Pyramid Network

First, we explain our motivation for designing the proposed architecture for object detection. In [34], Ren *et al.* stated that useful feature maps for robust object detection should have the following properties: 1) the feature maps need to

contain the fine details that represent the structure of objects well; 2) the feature maps must be extracted by using a sufficiently deep transformation function, *i.e.*, the high-level object knowledge should be encoded in the feature maps; 3) the feature maps should have meaningful context information to support predictions of the exact location and the class labels of "hard-to-detect" objects such as occluded, small, blurred, and saturated objects.

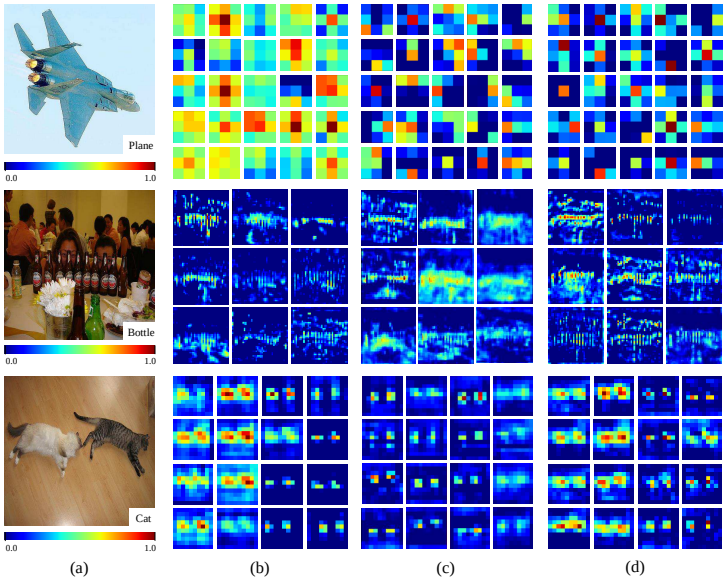

Fig. 2. Examples of input images (a) and their corresponding feature maps using SSD [27] (b), FPN [23] (c), and PFPNet (d). All the models use VGGNet-16 as their backbone network. The objects are detected at the visualized scales of the feature maps (in the second row, the feature maps represent the scale of the beer bottles).

Let us recall FPs discussed in Section 1. The feature maps in the FP based on the bottom-up DCNN might not satisfy the first property for the upper-level feature maps obtained by the deep transformation functions; on the other hand, the lower-level feature maps obtained by the shallow transformation functions may not meet the second property, which impairs the detection performance on small objects. Furthermore, each feature map is only responsible for the output at its corresponding scale, so contextual information at different scales cannot be effectively incorporated. A simple way to overcome these limitations is to utilize the transformation functions with a proper depth to retain both the fine spatial information and the high-level semantic object information. As shown in Fig. 1(d), if the FP is arranged in a row, we can apply such transformation functions with the same depth to generate every level of the FP. We then aggregate different types of contextual information using the proposed MSCA modules to

produce the final feature maps which satisfy the third property required for good feature maps mentioned above.

Fig. 2 shows the images of objects with various sizes and their feature maps at the scales corresponding to the objects. As discussed earlier, for SSD, the visualized channels of the upper-level feature map for a plane are well activated, but the activation values are not sparse, which could diminish the box regression performance. For small bottles, fine details can be observed, but the activations are not consistent despite the similar shape of the bottles. Some studies [20, 42] have shown that masking the activation values concentrated in the object regions can improve the performance of visual recognition tasks. Thus, sparse channel values on the object region can provide more accurate object information for large objects. For FPN [23], which is an hourglass model, and PFPNet, the visualized channels for a plane are well activated and sparser than those for SSD. For small bottles, the channel values of FPN are more activated than those of SSD, but the details somewhat disappear owing to the blurred information from the top-down path. On the other hand, the visualized channels in PFPNet retain not only the fine details of the objects, but also the consistent high activation values overlapping with the exact object location. For the medium-sized objects (the cats in the bottom row), all the feature channels of SSD, FPN, and PFPNet have feature values that are well activated and concentrated in the object region. This observation shows that the proper depth of the transformation function can enhance the quality of feature representation, and the experimental results in Section 3 demonstrate the effectiveness of the proposed structure. In the following, we provide details of the proposed PFPNet.

Base Network. PFPNet is based on VGGNet-16 [40]. In PFPNet, the final fully-connected (fc) layers in VGGNet-16 are replaced with convolutional (Conv) layers by sub-sampling their parameters, and this modified VGGNet-16 were pre-trained on the ILSVRC dataset [37].

Bottleneck Layer. For a feature transformation, we use bottleneck layers [16]. In the bottleneck layer, to improve the computational efficiency, a 1×1 convolution is applied prior to a 3×3 convolution to reduce the number of channels. The batch normalization [17] without scale/shift and the rectified linear unit (ReLU) [12] are used for input normalization and activation. In the bottleneck layer, the 1×1 convolution produces the feature maps with $C/2$ channels, where C is the number of the output channels of the bottleneck layer.

FP Pool. The pooling [21] layer, which is widely used in visual classification tasks [15,40,41], not only reduces the spatial size of the feature maps to a specific size, but it can also aggregate the contextual prior in a sub-region. In [14, 44], the SPP layer with various sizes of pooling sub-regions is utilized to construct an FP for object detection and segmentation .

Inspired by these previous studies, we use the SPP layer to construct an FP pool that is enriched with both the spatial information and multi-scale semantic

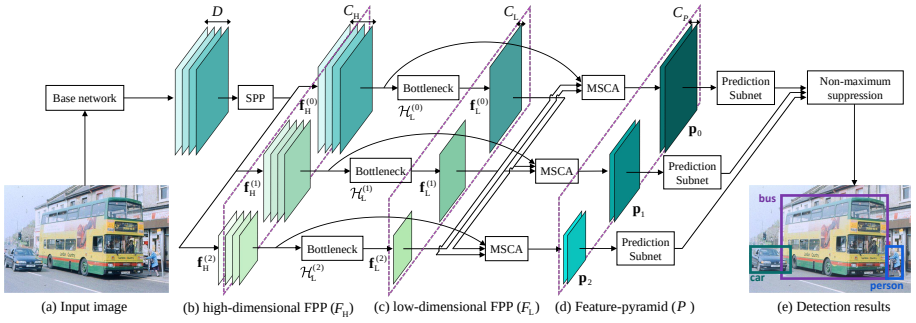

Fig. 3. Overview of PFPNet with $N = 3$. For an input image (a), the base network is employed to obtain the input for PFPNet. The high-dimensional FP pool (F_{H}) (b) is formed by using the SPP module, and the low-dimensional FP pool (F_{L}) (c) is obtained by applying the further transformation to the elements of (b). From these feature pools, the MSCA modules generate the final FP (P) (d) for multi-scale detection. Finally, the FP is fed into the Conv prediction Subnets to obtain the detection results (e).

object-information. Fig. 3 illustrates the architecture of PFPNet for multi-scale object detection. Let the base network produces the $W \times H$ output feature map having D output channels. By using the SPP module, we first form the high-dimensional FP pool, $F_{\mathrm{H}} = \{\mathbf{f}_{\mathrm{H}}^{(0)}, \mathbf{f}_{\mathrm{H}}^{(1)}, \cdots, \mathbf{f}_{\mathrm{H}}^{(N-1)}\}$, where $\mathbf{f}_{\mathrm{H}}^{(n)}$, the feature map with a spatial size of $\frac{W}{2^n} \times \frac{H}{2^n}$, denotes the nth level of F_{H}, and N denotes the number of pyramid levels. Thus, we obtain downsampled feature maps with the channel number of $C_{\mathrm{H}} = D$ by successively decreasing the spatial size by half. We apply the bottleneck layers, denoted as $\mathcal{H}_{\mathrm{L}}^{(n)}(\cdot)$, to each level in parallel to further extract the appropriate contextual feature for each scale and to reduce the channel number of contextual representation. We let $F_{\mathrm{L}} = \{\mathbf{f}_{\mathrm{L}}^{(0)}, \mathbf{f}_{\mathrm{L}}^{(1)}, \cdots, \mathbf{f}_{\mathrm{L}}^{(N-1)}\}$ represent the low-dimensional FP pool, which is the output of the transformation of F_{H} with a reduced channel number, $C_{\mathrm{L}} = D/(N-1)$.

MSCA. Incorporating context information at different scales can facilitate several visual classification tasks [1,13,18,28]. Combining the feature maps by summation is a common approach for collecting contextual information from multiple features [15]. However, Huang *et al.* [16] recently insisted that the summation could weaken the information flow in a network. They introduced an alternative approach, which involves concatenating the feature maps directly to retain the maximum information flow between feature layers. Several architectures for object detection adopted this approach. For instance, in [1], higher-level DCNN layers are fused into a single feature map via concatenation. In [18], multiple feature-layers based on the DCNN or the hourglass network are concatenated as well to exploit the contextual information at different scales.

PFPNet also use concatenation to collect the contextual information in the FP pool. Fig. 4 shows an example of how the MSCA module produces a level of the final FP, $P = \{\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{N-1}\}$. Consider that we generate the level n

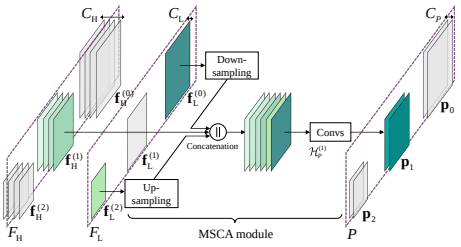

Fig. 4. Multi-scale context aggregation (MSCA) module.

of the FP, $\mathbf{p}_n$, with a size of $\frac{W}{2^n} \times \frac{H}{2^n}$. We assume that the level n of the FP pool contains primary information about the objects in $\mathbf{p}_n$, and the other levels supplement the information for the objects as context priors at different scales. Therefore, we bring $\mathbf{f}_{\mathrm{H}}^{(n)}$ from the high-dimensional FP pool, F_{H}, as the primary information, while we gather the larger- and smaller-sized feature maps from the low-dimensional FP pool, F_{L}, as the supplementary information. To match the sizes of the feature maps from F_{L}, we directly upsample the feature maps with smaller sizes ($> n$) via bilinear interpolation and downsample those with larger sizes ($< n$) via non-overlapping average pooling. Finally, these feature maps are combined via concatenation as described in [44]. The single feature map from F_{H} accounts for half the contents of the concatenated feature map and the $N-1$ feature maps in F_{L} with $D/(N-1)$ channels comprise the other half, *i.e.*, the concatenated feature map has $2D$ channels. We utilize another transformation by using bottleneck module or a series of 3×3 convolutions, denoted as $\mathcal{H}_P^{(n)}(\cdot)$, to refine and aggregate the collected information in the concatenated feature maps, and finally obtain $\mathbf{p}_n$ with C_P channels. Since the MSCA modules reuse the feature maps in F_{L}, we can effectively exploit the multi-scale context information with the improved use of computational resources.

In the MSCA module, the feature map from F_{H} is combined with other feature maps of F_{L} by using the skip connection. This can ease difficult optimization process due to a wide and complex structure of the MSCA module having a number of parameters. We conducted an experiment in which the skip connection was omitted and the concatenated feature map was built using only the feature maps of F_{L}. In this case, to form F_{L}, we let the number of output channels of $\mathcal{H}_{\mathrm{L}}(\cdot)$ be $2D/N$ so that the concatenated feature maps have $2D$ channels. In the experiment, this setting not only increased the number of parameters for $\mathcal{H}_{\mathrm{L}}(\cdot)$, but also decreased the performance of the proposed network slightly, as we expected.

Details of PFPNet. We use 3×3 Conv layers to predict the locations of objects and their class labels. For box regression sub-network (Subnet), a 3×3 Conv layer with $4A$ filters is applied to each level of the FP to calculate the relative offset between the anchor and the predicted bounding box, where A is the number of anchors per location of the feature map. For classification, another

3×3 Conv layer with $(K+1)A$ filters followed by softmax is applied to predict the probability of an object being present at each spatial position for each of the A anchors and K object classes. Since we focus on the contribution of the proposed FP to object detection, we use simple Subnets using a single-layer 3×3 convolution to ensure fair comparisons with SSD [27] and RefineDet [43].

The anchors allow us to allocate the output space to the multiple levels of the FP. For fair comparisons with SSD and RefineDet, we employ two anchor types: the pre-defined anchor boxes identical to those used in [26], denoted by a suffix "-S", and the anchor boxes predicted by the anchor refinement module (ARM) presented in [43], denoted by a suffix "-R". For PFPNet-S, we adopt most of the settings presented in [26, 27] such as anchor design, a matching scheme, and input sizes. We use the input sizes of 300×300 and 512×512, which are denoted as PFPNet-S300 and PFPNet-S512, respectively. For PFPNet-S, a bottleneck module is used for a transformation function $\mathcal{H}_P^{(n)}(\cdot)$, and all of the new Conv layers were initialized using the Gaussian filler with the standard deviation of 0.01. For PFPNet-R, we employ the ARM proposed in [43]. The ARM is a prediction Subnet, which outputs the coordinate information and objectness of the refined anchors. If the objectness of a refined anchor is larger than a threshold θ (θ is empirically set to 0.01), the refined anchor is then used as an input anchor for the final prediction Subnets; otherwise, it will be discarded. As described in [43], we employ PFPNet with the input sizes of 320×320 and 512×512, which are denoted as PFPNet-R320 and PFPNet-R512, respectively. The levels of the final pyramid P has $C_P = 256$ channels, and two 3×3 Conv layers followed by ReLU are applied as a transformation function $\mathcal{H}_P^{(n)}(\cdot)$ for PFPNet-R. All of the new Conv layers were initialized using the Xavier method [11].

We employ the multi-task loss defined in [27] for PFPNet-S and that in [43] for PFPNet-R to optimize the model parameters. Following SSD, the smooth L_1 loss is used to calculate the loss function for bounding box regression. Our experiments were conducted on a single NVIDIA Titan X GPU. For PFPNet-S300 and PFPNet-R320, we use the stochastic gradient descent (SGD) with a mini-batch size of 32 images, and for PFPNet with the size of 512, we employ the SGD with a mini-batch size of 28 images owing to the memory limit. We use a momentum of 0.9 and weight decay of 5×10^{-4}.

3 Experiments

3.1 Datasets

We utilize three datasets, namely Pascal VOC 2007, VOC 2012 [6], and MS COCO (COCO) [24] datasets. Both VOC 2007 and VOC 2012 datasets contain 20 object classes. VOC 2007 is divided into two subsets, *trainval* (5,011 images) and *test* (4,952 images) sets, which are fully annotated with the ground truth bounding boxes of objects. VOC 2012 comprises the *trainval* set (11,540 images), which is annotated, and the *test* set (10,991 images), which has no disclosed label. COCO has 80 object categories for object detection. In our experiments using

COCO, we use the *trainval35k* subset [1] for training, which is a union of 80k images from *train* and a random subset of 35k images from the 40k *val* images. We present the results obtained using the *test-dev* subset of the COCO dataset.

3.2 Experimental Setup

We compare performance of PFPNet with state-of-the-art region-based detectors, Faster R-CNN [35,36] and its variations [15,23,39], HyperNet [19], ION [1], R-FCN [3], Deformable R-FCN [4], and CoupleNet [45], as well as some region-free detectors, the YOLO [31], YOLOv2 [32], and SSD [27]. Note that, for SSD, we use the versions of the latest implementations [26]. For comparisons with the multi-scale region-free detector using the hourglass model, we employed RON [20], R-SSD [18], DSSD [8], RetinaNet [25], and RefineDet [43]. The suffix "+" represents the results obtained with a multi-scale testing.

3.3 PASCAL VOC 2007

Training and Evaluation Metric. In this experiment, the union of VOC 2007 *trainval* and VOC 2012 *trainval* sets denoted as VOC07+12 is used to train all of the networks. For VOC 2007 *test* set, the detection performance is evaluated using the mean AP (mAP) where a predicted bounding box is correct if its intersection over union (IoU) with the ground truth bounding box is higher than 0.5. We train our network for 110k iterations with an initial learning rate of 10^{-3}, which is divided by 10 at 80k iterations and again at 100k iterations.

Table 1. Impact of hyperparameters.

	# pyramid levels (N)					Reduced channel number (C_{L})		
	2	3	**4**	5	6	64	128	**256**
mAP (%)	79.08	80.08	**80.72**	80.68	80.15	80.16	80.34	**80.72**

Ablation Study for the impact of the hyperparameters We conduct experiments to clarify the impact of the hyperparameters, N and C_{L}. As shown in Table 1, for the pyramid levels (with $C_{\mathrm{L}} = 256$), PFPNet shows the best result of 80.72% where $N = 4$. For the reduced channel number of the low-dimensional feature pool F_{L} (with $N = 4$), $C_{\mathrm{L}} = 256$ shows the best mAP value.

Ablation Study for Wide FP and MSCA Module. To verify the effectiveness of the wide FP and MSCA module, we conduct an experiment with or without using the MSCA modules. The prediction Subnets are applied directly to the feature maps of F_{L}. As listed in Table 2, the wide FP obtained by using the SPP layer (*i.e.*, the mAP of 77.9%) increases the performance by 1.7% mAP as compared to the baseline. With the MSCA module, we find that mAP is further increased from 77.9% to 78.5%.

Table 2. Performance comparisons of different FPs.

Method	PFPNet-R320			RefineDet320		FPN320		SSD (baseline)
ARM	✓			✓		✓		
MSCA module	✓	✓						
mAP (%)	80.7	78.5	77.9	80.0	77.3	79.6	77.6	76.2

Performance Comparisons with Other FPs. To demonstrate the effectiveness of the proposed FP, we test four different FPs. For the FP based on the bottom-up model, we use SSD [27] as a baseline. For the FP based on the hourglass model, we adopt two different models, the single-stage detector using the feature pyramid network (FPN) [23] and RefineDet [43]. VGGNet-16 [40] is used as a base network, where the input size was 320×320. For a fair comparison, we use the same parameter settings. Every tested model has four pyramid levels ($N = 4$), and the number of anchors was $A = 3$ where the aspect ratios $\{\frac{1}{2}, 1, 2\}$ are employed. Single-layer 3×3 convolutions are utilized as prediction Subnets. For the proposed FP and the FPs based on hourglass models, we additionally evaluate the performance w/ or w/o ARM [43]. As shown in Table 2, PFPNet-R, RefineDet, and FPN can effectively increase the mAPs as compared to the baseline. Without ARM, FPN shows better performance than RefineDet (77.6% *vs.* 77.3%). As indicated in [8, 32], well designed anchors boost the performance of object detectors. Since ARM adaptively refines the anchor boxes, it has increased the performance of all the models by more than 2% points. Specifically, the mAPs of PFPNet-R, RefineDet, and FPN are increased by 2.2%, 2.7%, and 2.0%, respectively. As a result, by using the ARM, the proposed FP exhibits the best performance (80.7% mAP) among the compared models.

Results. Table 3 shows the performance of PFPNet and other conventional detectors. Recent region-free methods based on the hourglass model such as R-SSD512 [18], DSSD513 [8], and RefineDet512 [43] exhibit the performance competitive to the region-based detectors on this set. For the input size of 300×300, PFPNet-S300 shows the result similar to RefineDet320, which is the first method achieving the mAP of above 80% with such a small-resolution input. PFPNet-R320, *i.e.*, PFPNet using the ARM which was also used in RefineDet, obtains the mAP of 80.7%. Note that, for the larger input size of 512×512, PFPNet-S512 achieves the same result as RefineDet512. As shown in Table 3, PFPNet-R512 exhibits the best mAP among the region-free methods, and both PFPNet-R320 and PFPNet-R512 perform better than most of the region-based methods except CoupleNet [45], which adopts the residual network (ResNet)-101 as its base network and uses an larger input size (1000×600) as compared with PFPNet-R512. Since the input size dramatically affects the detection performance, we also tested PFPNet-R320+ and PFPNet-R512+, which utilizes the multi-scale testing, to reduce the impact of input sizes. PFPNet-R320+ and PFPNet-R512+ achieves the mAPs of 83.5% and 84.1%, respectively. As compared with the real-time object detectors such as SSD, YOLOv2, R-SSD, and RefineDet, PFPNet

Table 3. Detection results on PASCAL VOC 2007 and VOC 2012 test sets.

Method	Backbone	Input resolution	# Boxes	FPS	mAP (%)	
					VOC 2007	VOC 2012
Faster R-CNN [35]	VGGNet-16	∼1000 × 600	300	5	73.2	70.4
HyperNet [19]	VGGNet-16	∼1000 × 600	100	0.9	76.3	71.4
Faster R-CNN [36]	ResNet-101	∼1000 × 600	300	2.4	76.4	73.8
ION* [1]	VGGNet-16	∼1000 × 600	4000	1.3	79.2	76.4
R-FCN [3]	ResNet-101	∼1000 × 600	300	9	80.5	77.6
CoupleNet [45]	ResNet-101	∼1000 × 600	300	8	82.7	80.4
YOLO [31]	GoogleNet [41]	448 × 448	98	45	63.4	57.9
RON384 [20]	VGGNet-16	384 × 384	30600	15	75.4	73.0
SSD300 [26]	VGGNet-16	300 × 300	8732	46	77.2	75.8
R-SSD300 [18]	VGGNet-16	300 × 300	8732	35	78.5	76.4
YOLOv2 [32]	DarkNet-19	544 × 544	845	40	78.6	73.4
DSSD321 [8]	ResNet-101	321 × 321	17080	10	78.6	76.3
SSD512 [26]	VGGNet-16	512 × 512	24564	19	79.8	78.5
R-SSD512 [18]	VGGNet-16	512 × 512	24564	17	80.8	-
DSSD513 [8]	ResNet-101	513 × 513	43688	6	81.5	80.0
RefineDet320 [43]	VGGNet-16	320 × 320	6375	40	80.0	78.1
RefineDet512 [43]	VGGNet-16	512 × 512	16320	24	81.8	80.1
RefineDet320+ [43]	VGGNet-16	-	-	-	83.1	82.7
RefineDet512+ [43]	VGGNet-16	-	-	-	83.8	83.5
PFPNet-S300	VGGNet-16	300 × 300	8732	39	79.9	76.8[1]
PFPNet-R320	VGGNet-16	320 × 320	6375	33	80.7	77.7[2]
PFPNet-S512	VGGNet-16	512 × 512	24564	26	81.8	79.7[3]
PFPNet-R512	VGGNet-16	512 × 512	16320	24	82.3	80.3[4]
PFPNet-R320+	VGGNet-16	-	-	-	83.5	83.0[5]
PFPNet-R512+	VGGNet-16	-	-	-	**84.1**	**83.7**[6]

* ION adopted iterative bbox regression and voting, and regularizing with segmentation labels.
[1] http://host.robots.ox.ac.uk:8080/anonymous/HUJBN7.html
[2] http://host.robots.ox.ac.uk:8080/anonymous/GATL5Q.html
[3] http://host.robots.ox.ac.uk:8080/anonymous/SNRWPN.html
[4] http://host.robots.ox.ac.uk:8080/anonymous/GKGYPV.html
[5] http://host.robots.ox.ac.uk:8080/anonymous/B5AKH8.html
[6] http://host.robots.ox.ac.uk:8080/anonymous/M7K1BM.html

not only has the real-time speed, but also exhibits the best mAPs. The time complexity and detection performance of PFPNet-S, even without using ARM, are similar to those of RefineDet. PFPNet-R320 and PFPNet-R512 operate at 33 FPS and 24 FPS, respectively.

3.4 PASCAL VOC 2012

In this experiment, we used a subset, called VOC07++12, consisting of VOC 2007 *trainval* and *test* sets, and VOC 2012 *trainval* set, for a training, as described in [27, 35]. Using the VOC07++12 set, we trained PFPNet for 240k iterations in total. Starting with an initial learning rate of 10^{-3}, the learning rate is decreased by a factor of 10 at 160k and again at 200k iterations.

Table 3 shows the detection performance of PFPNet and other conventional detectors based on the comp4 (outside data) track from the public leaderboard on PASCAL VOC 2012. For the input size of 300 × 300, PFPNet-S300 obtains the mAP of 76.8%, which is better than most of the region-based methods using the input size of 1000 × 600 and region-free ones using the similar input size to PFPNet-S300. PFPNet-R320 shows the mAP of 77.7%, which is better than

the performance of most region-free detectors with similar input sizes except RefineDet320 [43]. For the input size of 512×512, PFPNet-S512 and PFPNet-R512 exhibit the mAPs of 79.7% and 80.3%, respectively. PFPNet-R512 outperforms other compared models except CoupleNet [45]. To reduce the impact of input sizes for a fair comparison, the multi-scale testing is applied, and as can be seen in Table 3, PFPNet-R320+ and PFPNet-R512+ yield the state-of-the-art mAPs of 83.0% and 83.7%, respectively.

3.5 MS COCO

To validate PFPNet on a more challenging dataset, we conducted experiments using the COCO dataset. The performance evaluation metric for the COCO dataset is slightly different from that for the VOC dataset. The AP over different IoU thresholds from 0.5 to 0.95 is denoted as $AP_{50:95}$, to present the overall performance of the detection models. The APs with IoU thresholds of 0.5 and 0.75 are denoted as AP_{50} and AP_{75}, respectively. In addition, the MS COCO evaluation server provides the AP for diverse scales. The object scales are determined by measuring the number of pixels in the object's segmentation mask, S, as follows: small objects (AP_S): $S < 32^2$; medium objects (AP_M): $32^2 < S < 96^2$; large objects (AP_L): $S > 96^2$. Using the COCO *trainval35k* sets, we trained our model for 400k iterations in total, starting with an initial learning rate of 10^{-3}, and then decreasing it by a factor of 10 at 280k and again at 360k iterations.

Table 4. Detection results on MS COCO test-dev set.

Method	Backbone	Train set	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN [35]	VGGNet-16	trainval	21.9	42.7	-	-	-	-
ION* [1]	VGGNet-16	train	33.1	55.7	34.6	14.5	35.2	47.2
R-FCN [3]	ResNet-101	trainval	29.9	51.9	-	10.8	32.8	45.0
CoupleNet [45]	ResNet-101	trainval	34.4	54.8	37.2	13.4	38.1	50.8
Faster R-CNN+++ [15]	ResNet-101-C4	trainval	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w/ FPN [23]	ResNet-101-FPN	trainval35k	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN w/ TDM [39]	Inception-ResNet-v2-TDM	trainval	37.3	57.8	39.8	17.1	40.3	52.1
Deformable R-FCN [4]	Aligned-Inception-ResNet	trainval	37.5	58.0	40.8	19.4	40.1	52.5
YOLOv2 [32]	DarkNet-19	trainval35k	21.6	44.0	19.2	5.0	22.4	35.5
SSD300 [26]	VGGNet-16	trainval35k	25.1	43.1	25.8	6.6	25.9	41.4
RON384++ [20]	VGGNet-16	trainval	27.4	49.5	27.1	-	-	-
DSSD321 [8]	ResNet-101	trainval35k	28.0	46.1	29.2	7.4	28.1	47.6
SSD512 [26]	VGGNet-16	trainval35k	28.8	48.5	30.3	10.9	31.8	43.5
RefineDet320 [43]	VGGNet-16	trainval35k	29.4	49.2	31.3	10.0	32.0	44.4
RetinaNet400 [25]	ResNet-50	trainval35k	30.5	47.8	32.7	11.2	33.8	46.1
RetinaNet400 [25]	ResNet-101	trainval35k	31.9	49.5	34.1	11.6	35.8	48.5
RefineDet320 [43]	ResNet-101	trainval35k	32.0	51.4	34.2	10.5	34.7	50.4
RetinaNet500 [25]	ResNet-50	trainval35k	32.5	50.9	34.8	13.9	35.8	46.7
RefineDet512 [43]	VGGNet-16	trainval35k	33.0	54.5	35.5	16.3	36.3	44.3
DSSD513 [8]	ResNet-101	trainval35k	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet500 [25]	ResNet-101	trainval35k	34.4	53.1	36.8	14.7	38.5	49.1
RefineDet320+ [43]	VGGNet-16	trainval35k	35.2	56.1	37.7	19.5	37.2	47.0
RefineDet512 [43]	ResNet-101	trainval35k	36.4	57.5	39.5	16.6	39.9	51.4
RefineDet512+ [43]	VGGNet-16	trainval35k	37.6	58.7	40.8	22.7	40.3	48.3
RefineDet320+ [43]	ResNet-101	trainval35k	38.6	59.9	41.7	21.1	41.7	52.3
RetinaNet800** [25]	ResNet-101-FPN	trainval35k	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet800** [25]	ResNeXt-101-FPN	trainval35k	40.8	61.1	44.1	24.1	44.2	51.2
RefineDet512+ [43]	ResNet-101	trainval35k	**41.8**	**62.9**	**45.7**	**25.6**	**45.1**	**54.1**
PFPNet-S300	VGGNet-16	trainval35k	29.6	49.6	31.1	10.6	32.0	44.9
PFPNet-R320	VGGNet-16	trainval35k	31.8	52.9	33.6	12.0	35.5	46.1
PFPNet-S512	VGGNet-16	trainval35k	33.4	54.8	35.8	16.3	36.7	46.7
PFPNet-R512	VGGNet-16	trainval35k	35.2	57.6	37.9	18.7	38.6	45.9
PFPNet-R320+	VGGNet-16	trainval35k	37.8	60.0	40.7	22.2	40.4	49.1
PFPNet-R512+	VGGNet-16	trainval35k	39.4	61.5	42.6	25.3	42.3	48.8

* ION adopted iterative bbox regression and voting, and regularizing with segmentation labels.
** RetinaNet800 was trained with scale jitter and for 1.5× longer than RetinaNet500.

As shown in Table 4, PFPNet-S300 and PFPNet-R320 produce the APs of 29.6% and 31.8%, which outperform the VGGNet-16-based models using an input size of around 300. PFPNet-R320 even produces the better results than R-FCN using ResNet-101 [15], and the similar results to RetinaNet400 using ResNet-101 with the larger input size of 400×400. Without using the ARM, PFPNet-S512 shows the comparable performance to the hourglass models using the similar input size such as RetinaNet500 [23], DSSD513 [8], which are based on ResNet-101, and RefineDet512 [43]. With the ARM, PFPNet-R512 achieves the AP of 35.2%. By using VGGNet-16, ARM, and the same input sizes, PFPNet-R320 and PFPNet-R512 increase the overall APs of RefineDet by 2.4% and 2.2%, respectively. As can be seen in Table 4, the performance of PFPNet-R512 is much better than the state-of-the-art region-free detectors with an input size of around 512, and superior to most of the region-based detectors except Faster R-CNN w/ FPN [23], Faster R-CNN w/ TDM [39], and Deformable R-FCN [39], which use complex ResNet-based backbones with a large input size of 1000×600. To reduce the impact of input sizes for a fair comparison, we also employed the multi-scale testing on this set. PFPNet-R320+ obtains the even higher AP than all of the compared region-based object detectors, and PFPNet-R512+ attains the best AP of 39.4% among the compared object detection models based on VGGNet. As compared with the detectors based on recent backbone networks, PFPNet shows comparable detection performance, especially on small objects

Note that PFPNet-S512 and PFPNet-R512 show the particularly good AP values at a small scale (16.3% and 18.7%) among the compared models. Detecting the small-scaled objects is one of the most challenging problem for both the region-based and region-free methods. The experimental results demonstrate that the feature representation power can be improved by using the proposed FP architecture, especially at a small scale, as discussed in Section 2. As provided in [24], more than 70% of COCO dataset is composed of objects, the sizes of which are smaller than 10% of the input size (cf. the VOC dataset has less than 50% of objects with such sizes). In addition, the images of COCO dataset possess more valuable contextual information than those of VOC dataset, which can be estimated by investigating the average number of object categories per image (3.4 vs. 1.4). Since the proposed MSCA modules aggregate the multi-scale context, the COCO dataset has more information available than VOC dataset. As a result, PFPNet has great advantages in detecting small-scale objects and utilizing the multi-scale context information, which yields a significant improvement on COCO dataset.

Speed versus accuracy trade-off: Fig. 5 shows the speed versus AP of single-stage detectors on COCO test-dev dataset. As compared with RetinaNet and RefineDet models yielding similar APs to PFPNet, PFPNet operates more than twice as fast. For the similar speed, PFPNet outperforms SSD [26], RefineDet [43], and recently-released YOLOv3 [33].

In addition, we obtain the performance on VOC 2012 test set using the fine-tuned models pretrained on COCO. The models were first trained on COCO

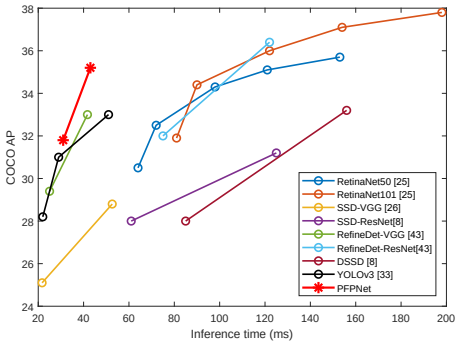

Fig. 5. Speed (ms) versus accuracy (AP) of single-stage detectors on COCO test-dev.

trainval35k set, and then fine-tuned on VOC07++12 set. The mAP results of PFPNet-R320 and PFPNet-R512 are 84.5% and 85.7%, respectively. With the same backbone and train set, PFPNet-R320 and PFPNet-R512 increase the mAPs of RefineDet [43] by 1.8% and 0.7%, respectively. By using the multi-scale test scheme, PFPNet-R (87.8% mAP) was ranked 7th place on the leaderboard among the overall architectures at the time of submission.

4 Conclusions

In this paper, we proposed an effective FP for object detection. In contrast to the conventional FPs, we designed the FP having a wide structure. This allows transformation functions to have a proper and uniform depth. Thus, all elements of the FP could retain both the fine-structure of objects and the high-level object knowledge. By using the proposed MSCA module, we efficiently reused the elements in the FP pool to collect the various contextual information at the different scale and produce the final FP. A single-shot object detection method developed using the wide FP, called PFPNet, achieved 82.3% on the Pascal VOC 2007, 80.3% on the PASCAL VOC 2012, and 35.2% on the MS-COCO, in terms of the mAP. By employing the multi-scale testing, PFPNet exhibits the state-of-the-art performance. In particular, PFPNet has the advantage in detecting small-scale objects with the AP_S of 18.7% on the MS-COCO dataset.

Although we concentrated on object detection, we believe that the feature representation using the proposed wide FP will be beneficial for a variety of other computer vision tasks.

Acknowledgements. This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (2014-0-00077, Development of global multi-target tracking and event prediction techniques based on real-time large-scale video analysis).

References

1. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2016)
2. Cai, Z., Fan, Q., Feris, R.S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: Proc. European Conf. Comput. Vis. (2016)
3. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Proc. Neural Inf. Process. Syst. (2016)
4. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proc. IEEE Int. Conf. Comput. Vis. (2017)
5. Dollár, P., Appel, R., Belongie, S., Perona, P.: Fast feature pyramids for object detection. IEEE Trans. Pattern Anal. Mach. Intell. **36**(8), 1532–1545 (2014)
6. Everingham, M., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC) Challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010)
8. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: Dssd: Deconvolutional single shot detector. arXiv preprint arXiv:1701.06659 (2017)
9. Girshick, R.: Fast r-cnn. In: Proc. IEEE Int. Conf. Comput. Vis. (2015)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2014)
11. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS (2010)
12. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
13. Hariharan, B., Arbelaez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2015)
14. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1904–1916 (2015)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2016)
16. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
17. Ioffe, S., Szegedy, C.: Batch normalization : Accelerating deep network training by reducing internal covariate shift. In: Proc. Int. Conf. Mach. Learn. (2015)
18. Jeong, J., Park, H., Kwak, N.: Enhancement of ssd by concatenating feature maps for object detection. In: Proc. British Mach. Vis. Conf. (2017)
19. Kong, T., Yao, A., Chen, Y., Sun, F.: Hypernet: Towards accurate region proposal generation and joint object detection. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2016)
20. Kong, T., Sun, F., Yao, A., Liu, H., Lu, M., Chen, Y.: Ron: Reverse connection with objectness prior networks for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)

21. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
22. Li, H., Liu, Y., Ouyang, W., Wang, X.: Zoom out-and-in network with map attention decision for region proposal and object detection. arXiv preprint arXiv:1709.04347 (2017)
23. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
24. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Proc. European Conf. Comput. Vis. (2014)
25. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proc. IEEE Int. Conf. Comput. Vis. (2017)
26. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. arXiv preprint arXiv:1512.02325 (2015)
27. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proc. European Conf. Comput. Vis. (2016)
28. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2015)
29. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Proc. European Conf. Comput. Vis. (2016)
30. Pinheiro, P.O., Lin, T.Y., Collobert, R., Dollár, P.: Learning to refine object segments. In: Proc. European Conf. Comput. Vis. (2016)
31. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2016)
32. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
33. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
34. Ren, J., Chen, X., Liu, J., Sun, W., Pang, J., Yan, Q., Tai, Y.W., Xu, L.: Accurate single stage detector using recurrent rolling convolution. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
35. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Proc. Neural Inf. Process. Syst. (2015)
36. 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. **39**(6), 1137–1149 (2017)
37. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Li, F.F.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (Mar 2015)
38. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. In: Proc. Int. Conf. Learn. Representations
39. Shrivastava, A., Sukthankar, R., Malik, J., Gupta, A.: Beyond skip connections: Top-down modulation for object detection. arXiv preprint arXiv:1612.06851 (2016)
40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014)
41. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2015)

42. Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., Tang, X.: Residual attention network for image classification. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
43. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. arXiv preprint arXiv:1711.06897 (2017)
44. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proc. IEEE Comput. Vis. Pattern Recognit. (2017)
45. Zhu, Y., Zhao, C., Wang, J., Zhao, X., Wu, Y., Lu, H.: Couplenet: Coupling global structure with local parts for object detection. In: Proc. IEEE Int. Conf. Comput. Vis. (2017)