Deep Feature Pyramid Reconfiguration for Object Detection

Tao Kong[1], Fuchun Sun[1], Wenbing Huang[2], and Huaping Liu[1]

[1] Department of Computer Science and Technology, Tsinghua University,
Beijing National Research Center for Information Science and Technology (BNRist).
{kt14@mails, fcsun@mail, hpliu@mail}.tsinghua.edu.cn
[2] Tencent AI Lab.
hwenbing@126.com

Abstract. State-of-the-art object detectors usually learn multi-scale representations to get better results by employing feature pyramids. However, the current designs for feature pyramids are still inefficient to integrate the semantic information over different scales. In this paper, we begin by investigating current feature pyramids solutions, and then reformulate the feature pyramid construction as the feature reconfiguration process. Finally, we propose a novel reconfiguration architecture to combine low-level representations with high-level semantic features in a highly-nonlinear yet efficient way. In particular, our architecture which consists of global attention and local reconfigurations, is able to gather task-oriented features across different spatial locations and scales, globally and locally. Both the global attention and local reconfiguration are lightweight, in-place, and end-to-end trainable. Using this method in the basic SSD system, our models achieve consistent and significant boosts compared with the original model and its other variations, without losing real-time processing speed.

Keywords: Object Detection, Feature Pyramids, Global-Local Reconfiguration

1 Introduction

Detecting objects at vastly different scales from images is a fundamental challenge in computer vision [1]. One traditional way to solve this issue is to build feature pyramids upon image pyramids directly. Despite the inefficiency, this kind of approaches have been applied for object detection and many other tasks along with hand-engineered features [7, 12].

We focus on detecting objects with deep ConvNets in this paper. Aside from being capable of representing higher-level semantics, ConvNets are also robust to variance in scale, thus making it possible to detect multi-scale objects from features computed on a single scale input [39, 17]. However, recent works suggest that taking pyramidal representations into account can further boost the detection performance [30, 20, 16]. This is due to its principle advantage of producing

multi-scale feature representations in which all levels are semantically strong, including the high-resolution features.

There are several typical works exploring the feature pyramid representations for object detection. The Single Shot Detector (SSD) [34] is one of the first attempts on using such technique in ConvNets. Given one input image, SSD combines the predictions from multiple feature layers with different resolutions to naturally handle objects of various sizes. However, SSD fails to capture deep semantics for shallow-layer feature maps, since the bottom-up pathway in SSD can learn strong features only for deep layers but not for the shallow ones. This causes the key bottleneck of SSD for detecting small instances.

To overcome the disadvantage of SSD and make the networks more robust to object scales, recent works (e.g., FPN [30], DSSD [15], RON [26] and TDM [44]) propose to combine low-resolution and semantically-strong features with high-resolution and semantically-weak features via lateral connections in a top-down pathway. In contrast to the bottom-up fashion in SSD, the lateral connections pass the semantic information down to the shallow layers one by one, thus enhancing the detection ability of shallow-layer features. Such technology is successfully used in object detection [15, 31], segmentation [19], pose estimation [47, 5], etc.

Ideally, the pyramid features in ConvNets should: (1) reuse multi-scale features from different layers of a single network, and (2) improve features with strong semantics at all scales. The FPN works [30] satisfy these conditions by lateral connections. Nevertheless, the FPN, as demonstrated by our analysis in § 3, is actually equivalent to a linear combination of the feature hierarchy. Yet, the linear combination of features is too simple to capture highly-nonlinear patterns for more complicate and practical cases. Several works are trying to develop more suitable connection manners [25, 46, 48], or to add more operations before combination [28].

The basic motivation of this paper is to enable the networks learn information of interest for each pyramid level in a more flexible way, given a ConvNet's feature hierarchy. To achieve this goal, we explicitly reformulate the feature pyramid construction process as feature reconfiguration functions in a highly-nonlinear yet efficient way. To be specific, our pyramid construction employs a global attention to emphasize global information of the full image followed by a local reconfiguration to model local patch within the receptive field. The resulting pyramid representation is capable of spreading strong semantics to all scales. Compared to previous studies including SSD and FPN-like models, our pyramid construction is more advantageous in two aspects: 1) the global-local reconfigurations are non-linear transformations, thus depicting more expressive power; 2) the pyramidal precessing for all scales are performed simultaneously and are hence more efficient than the layer-by-layer transformation (e.g. in lateral connections).

In our experiments, we compare different feature pyramid strategies within SSD architecture, and demonstrate the proposed method works more compet-

itive in terms of accuracy and efficiency. The main contributions of this paper are summarized as follows:

- We propose the global attention and local reconfiguration for building feature pyramids to enhance multi-scale representations with semantically strong information;
- We compare and analysis popular feature pyramid methodologies within the standard SSD framework, and demonstrate that the proposed reconfiguration works more effective;
- The proposed method achieves the state-of-the-art results on standard object detection benchmarks (i.e., PASCAL VOC 2007, PASCAL VOC 2012 and MS COCO) without losing real-time processing speed.

2 Related work

Hand-engineered feature pyramids: Prior to the widely development of deep convolutional networks, hand-craft features such as HOG [45] and SIFT [35] are popular for feature extraction. To make them scale-invariant, these features are computed over image pyramids [9, 13]. Several attempts have been performed on image pyramids for the sake of efficient computation [4, 7, 8]. The sliding window methods over multi-scale feature pyramids are usually applied in object detection [10, 14].

Deep object detectors: Benefited by the success of deep ConvNets, modern object detectors like R-CNN [18] and Overfeat [41] lead dramatic improvement for object detection. Particularly, OverFeat adopts a similar strategy to early face detectors by applying a ConvNet as the sliding window detector on image pyramids; R-CNN employs a region proposal-based strategy and classifies each scale-normalized proposal with a ConvNet. The SPP-Net [20] and Fast R-CNN [17] speed up the R-CNN approach with RoI-Pooling that allows the classification layers to reuse the CNN feature maps. Since then, Faster R-CNN [39] and R-FCN [6] replace the region proposal step with lightweight networks to deliver a complete end-to-end system. More recently, Redmon et al. [37, 38] propose a method named YOLO to predict bounding boxes and associate class probabilities in a single step.

Deep feature pyramids: To make the detection more reliable, researchers usually adopt multi-scale representations by inputting images with multiple resolutions during training and testing [20, 21, 3]. Clearly, the image pyramid methods are very time-consuming as them require to compute the features on each of image scale independently and thus the ConvNet features can not be reused. Recently, a number of approaches improve the detection performance by combining predictions from different layers in a single ConvNet. For instance, the HyperNet [27] and ION [3] combine features from multiple layers before making detection. To detect objects of various sizes, the SSD [34] spreads out default boxes of different scales to multiple layers of different resolutions within a single ConvNets. So far, the SSD is a desired choice for object detection satisfying the speed-vs-accuracy trade-off [24]. More recently, the lateral connection

(or reverse connection) is becoming popular and used in object detection [15, 30, 26]. The main purpose of lateral connection is to enrich the semantic information of shallow layers via the top-down pathway. In contrast to such layer-by-layer connection, this paper develops a flexible framework to integrate the semantic knowledge of multiple layers in a global-local scheme.

3 Method

In this section, we firstly revisit the SSD detector, then consider the recent improvements of lateral connection. Finally, we present our feature pyramid reconfiguration methodology.

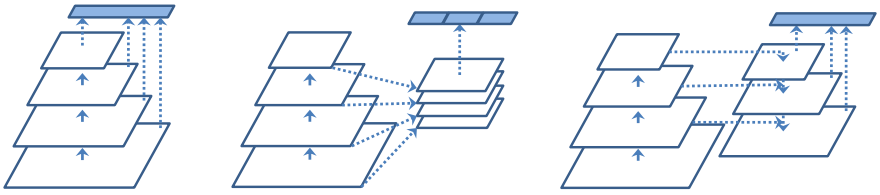

Fig. 1. Different feature pyramid construction frameworks. left: SSD uses pyramidal feature hierarchy computed by a ConvNet as if it is a featurized image pyramid; middle: Some object segmentation works produce final detection feature maps by directly combining features from multiple layers; right: FPN-like frameworks enforce shallow layers by top-down pathway and lateral connections.

ConvNet Feature Hierarchy: The object detection models based on ConvNets usually adopt a backbone network (such as VGG-16, ResNets). Consider a single image x_0 that is passed through a convolutional network. The network comprises L layers, each of which is implemented by a non-linear transformation $\mathcal{F}_l(\cdot)$, where l indexes the layer. $\mathcal{F}_l(\cdot)$ is a combination transforms such as convolution, pooling, ReLU, etc. We denote the output of the l^{th} layer as x_l. The total backbone network outputs are expressed as $X_{net} = \{x_1, x_2, ..., x_L\}$.

Without feature hierarchy, object detectors such as Faster R-CNN [39] use one deep and semantic layer such as x_L to perform object detection. In SSD [34], the prediction feature map sets can be expressed as

$$X_{pred} = \{x_P, x_{P+1}, \ldots, x_L\}, \tag{1}$$

where $P \gg 1$[3]. Here, the deep feature maps x_L learn high-semantic abstraction. When $P < l < L$, x_l becomes shallower thus has more low-level features. SSD uses deeper layers to detect large instances, while uses the shallow and high-resolution layers to detect small ones[4]. The high-resolution maps with limited-

[3] For VGG-16 based model, $P = 23$ since we begin to predict from *conv4_3* layer.

[4] Here the 'small' means that the proportion of objects in the image is small, not the actual instance size.

semantic information harm their representational capacity for object recognition. It misses the opportunity to reuse deeper and semantic information when detecting small instances, which we show is the key bottleneck to boost the performance.

Lateral Connection: To enrich the semantic information of shallow layers, one way is to add features from the deeper layers[5]. Taking the FPN manner [30] as an example, we get

$$\begin{aligned} x_L^{'} &= x_L, \\ x_{L-1}^{'} &= \alpha_{L-1} \cdot x_{L-1} + \beta_{L-1} \cdot x_L, \\ x_{L-2}^{'} &= \alpha_{L-2} \cdot x_{L-2} + \beta_{L-2} \cdot x_{L-1}^{'}, \\ &= \alpha_{L-2} \cdot x_{L-2} + \beta_{L-2}\alpha_{L-1} \cdot x_{L-1} + \beta_{L-2}\beta_{L-1} \cdot x_L, \end{aligned} \tag{2}$$

where α, β are weights. Without loss of generality,

$$x_l^{'} = \sum_{l=P}^{L} w_l \cdot x_l, \tag{3}$$

where w_l is the generated final weights for l^{th} layer output after similar polynomial expansions. Finally, the features used for detection are expressed as:

$$X_{pred}^{'} = \{x_P^{'}, x_{P+1}^{'}, \ldots, x_L^{'}\}. \tag{4}$$

From Eq.3 we see that the final features $x_l^{'}$ is equivalent to the linear combination of $x_l, x_{l+1}, \ldots, x_L$. The linear combination with deeper feature hierarchy is one way to improve information of a specific shallow layer. And the linear model can achieve a good extent of abstraction when the samples of the latent concepts are linearly separable. However, the feature hierarchy for detection often lives on a non-linear manifold, therefore the representations that capture these concepts are generally highly non-linear function of the input [29, 33, 23]. It's representation power, as we show next, is not enough for the complex task of object detection.

3.1 Deep Feature Reconfiguration

Given the deep feature hierarchy $X = [x_P, x_{P+1}, \ldots, x_L]$ of a ConvNet, the key problem of object detection framework is to generate suitable features for each level of detector. In this paper, the feature generating process at l^{th} level is viewed as a non-linear transformation of the given feature hierarchy (Fig. 2):

$$x_l^{'} = \mathcal{H}_l(X) \tag{5}$$

[5] When the resolutions of the two layers are not the same, usually upsample and linear projection are carried out before combination.

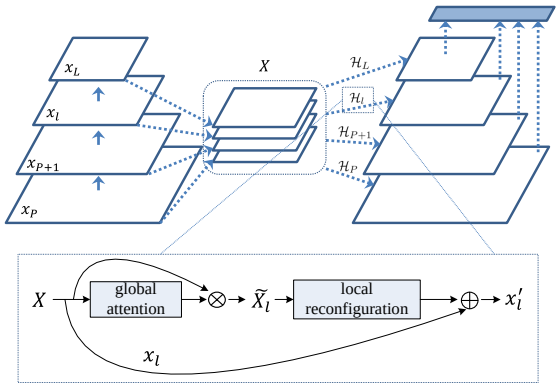

Fig. 2. Top: Overview of the proposed feature pyramid building networks. We firstly combine multiple feature maps, then generate features at a specific level, finally detect objects at multiple scales. Down: A building block illustrating the global attention and local reconfiguration.

where X is the feature hierarchy considered for multi-scale detection. For ease of implementation, we concatenate the multiple inputs of $\mathcal{H}_l(\cdot)$ in Eq.5 into a single tensor before following transformations[6].

Given no priors about the distributions of the latent concepts of the feature hierarchy, it is desirable to use a universal function approximator for feature extraction of each scale. The function should also keep the spatial consistency, since the detector will activate at the corresponding locations. *The final features for each level are non-linear transformations for the feature hierarchy, in which learnable parameters are shared between different spatial locations.*

In this paper, we formulate the feature transformation process $\mathcal{H}_l(\cdot)$ as global attention and local reconfiguration problems. Both global attention and local reconfiguration are implemented by a light-weight network so they could be embedded into the ConvNets and learned end-to-end. The global and local operations are also complementary to each other, since they deal with the feature hierarchy from different scales.

Global Attention for Feature Hierarchy Given the feature hierarchy, the aim of the global part is to emphasise informative features and suppress less useful ones globally for a specific scale. In this paper, we apply the Squeeze-and-Excitation block [23] as the basic module. One Squeeze-and-Excitation block consists of two steps, *squeeze* and *excitation*. For the l^{th} level layer, the *squeeze* stage is formulated as a global pooling operation on each channel of X which

[6] For a target scale which has $W \times H$ spatial resolution, adaptive sampling is carried out before concatenation.

has $W \times H \times C$ dimensions:

$$z_l^c = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} x_l^c(i,j) \tag{6}$$

where $x_l^c(i,j)$ specifies one element at c^{th} channel, i^{th} column and j^{th} row. If there are C channels in feature X, Eq.8 will generate C output elements, denoted as $\mathbf{z}_l$.

The excitation stage is two fully-connected layers followed by sigmoid activation with input $\mathbf{z}_l$:

$$\mathbf{s}_l = \sigma(W_l^1 \delta(W_2^l \mathbf{z}_l)) \tag{7}$$

where δ refers to the ReLU function, σ is the sigmoid activation, $W_l^1 \in R^{\frac{c}{r}}$ and $W_2^2 \in R^c$. r is set to 16 to make dimensionality-reduction. The final output of the block is obtained by rescaling the input X with the activations:

$$\tilde{\mathbf{x}}_l^c = s_l^c \otimes \mathbf{x}^c \tag{8}$$

then $\tilde{X}_l = [\tilde{x}_l^P, \tilde{x}_l^{P+1}, \ldots, \tilde{x}_l^L]$, $\otimes$ denotes channel-wise multiplication. More details can be referred to the SENets [23] paper.

The original SE block is developed for explicitly modelling interdependencies between channels, and shows great success in object recognition [2]. In contrast, we apply it to emphasise channel-level hierarchy features and suppress less useful ones. By dynamically adopting conditions on the input hierarchy, SE Block helps to boost feature discriminability and select more useful information globally.

Local Reconfiguration The local reconfiguration network maps the feature hierarchy patch to an output feature patch, and is shared among all local receptive fields. The output feature maps are obtained by sliding the operation over the input. In this work, we design a residual learn block as the instantiation of the micro network, which is a universal function approximator and trainable by back-propagation (Fig.3).

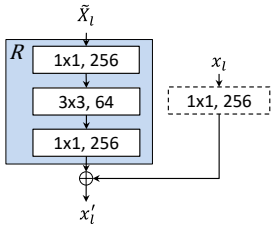

Fig. 3. A building block illustrating the local reconfiguration for level l.

Formally, one local reconfiguration is defined as:

$$x_l^{'} = R(\tilde{X}_l) + W_l x_l \tag{9}$$

where W_l is a linear projection to match the dimensions[7]. $R(\cdot)$ represents the residual mapping that improves the semantics to be learned.

Discussion A direct way to generate feature pyramids is just use the term $R(\cdot)$ in Eq.9. However, as demonstrated in [21], it is easier to optimize the residual mapping than to optimize the desired underlying mapping. Our experiments in Section 4.1 also prove this hypothesize.

We note there are some differences between our residual learn module and that proposed in ResNets [21]. Our hypothesize is that the semantic information is distributed among feature hierarchy and the residual learn block could select additional information by optimization. While the purpose of the residual learn in [21] is to gain accuracy by increasing network depth. Another difference is that the input of the residual learning is the feature hierarchy, while in [21], the input is one level of convolutional output.

The form of the residual function $R(\cdot)$ is also flexible. In this paper, we involve a function that has three layers (Fig.3), while more layers are possible. The element-wise addition is performed on two feature maps, channel by channel. Because all levels of the pyramid use shared operations for detection, we fix the feature dimension (numbers of channels, denoted as d) in all the feature maps. We set $d = 256$ in this paper and thus all layers used for prediction have 256-channel outputs.

4 Experiments

We conduct experiments on three widely used benchmarks: PASCAL VOC 2007, PASCAL VOC 2012 [11] and MS COCO datasets [32]. All network backbones are pretrained on the ImageNet1k classification set [40] and fine-tuned on the detection dataset. We use the pre-trained VGG-16 and ResNets models that are publicly available[8]. Our experiments are based on re-implementation of SSD [34], Faster R-CNN [39] and Feature Pyramid Networks [30] using PyTorch [36]. For the SSD framework, all layers in $\boldsymbol{X}$ are resized to the spatial size of layer conv8_2 in VGG and conv6_x in ResNet-101 to keep consistency with DSSD. For the Faster R-CNN pipeline, the resized spatial size is as same as the conv4_3 layer in both VGG and ResNet-101 backbones.

4.1 PASCAL VOC 2007

Implementation details. All models are trained on the VOC 2007 and VOC 2012 trainval sets, and tested on the VOC 2007 test set. For *one-stage* SSD, we set the learn rate to 10^{-3} for the first 160 epochs, and decay it to 10^{-4} and 10^{-5}

[7] When dimensions are the same, there is no need to use it, denoted as dotted line in Fig.3

[8] https://github.com/pytorch/vision

for another 40 and 40 epochs. We use the default batch size 32 in training, and use VGG-16 as the backbone networks for all the ablation study experiments on the PASCAL VOC dataset. For *two-stage* Faster R-CNN experiments, we follow the training strategies introduced in [39]. We also report the results of ResNets used in these models.

Baselines For fair comparisons with original SSD and its its feature pyramid variations, we conduct two baselines: Original SSD and SSD with feature lateral connections. In Table 1, the original SSD scores 77.5%, which is the same as that reported in [34]. Adding lateral connections in SSD improves results to 78.5% (SSD+lateral). When using the global and local reconfiguration strategy proposed above, the result is improved to 79.6%, which is 1.6% better than SSD with lateral connection. In the next, we discuss the ablation study in more details.

Table 1. Effectiveness of various designs with SSD300.

method	backbone	FPS	mAP(%)
SSD (Caffe) [34]	VGG-16	46	77.5
SSD (ours-re)	VGG-16	44	77.5
SSD+lateral	VGG-16	37	78.5
SSD+Local only	VGG-16	40	79.0
SSD+Local only(no res)	VGG-16	40	78.6
SSD+Global-Local	VGG-16	39.5	**79.6**

How important is global attention? In Table 1, the fourth row shows the results of our model without the global attention. With this modification, we remove the global attention part and directly add local transformation into the feature hierarchy. Without global attention, the result drops to 79.0% mAP (-0.6%). The global attention makes the network to focus more on features with suitable semantics and helps detecting instance with variation.

Comparison with the lateral connections Adding global and local reconfiguration to SSD improves the result to 79.6%, which is 2.1% better than SSD and 1.1% better than SSD with lateral connection. This is because there are large semantic gaps between different levels on the bottom-up pyramid. And the global and local reconfigurations help the detectors to select more suitable feature maps. This issue cannot be simply remedied by just lateral connections. We note that only adding local reconfiguration, the result is better than lateral connection (+0.5%).

Only use the term $R(\cdot)$ One way to generate the final feature pyramids is just use the term $R(\cdot)$. in Eq.9. Compared with residual learn block, the result drops 0.4%. The residual learn block can avoid the gradients of the objective function to directly flow into the backbone network, thus gives more opportunity to better model the feature hierarchy.

Use all feature hierarchy or just deeper layers? In Eq.3, the lateral connection only considers feature maps that are deeper (and same) than corresponding levels. To better compare our method with lateral connection, we conduct a experiment that only consider the deep layers too. Other settings are the same with the previous baselines. We find that just using deeper features drops accuracy by a small margin(-0.2%). We think the difference is that when using the total feature hierarchy, the deeper layers also have more opportunities to re-organize its features, and has more potential for boosting results, similar conclusions are also drawn from the most recent work of PANet [33].

Accuracy vs. Speed We present the inference speed of different models in the third column of Table 1. The speed is evaluated with batch size 1 on a machine with NVIDIA Titan X, CUDA 8.0 and cuDNN v5. Our model has a 2.7% accuracy gain with 39.5 *fps*. Compared with the lateral connection based SSD, our model shows higher accuracy and faster speed. In lateral connection based model, the pyramid layers are generated serially, thus last constructed layer considered for detection becomes the speed bottleneck ($x_P^{'}$ in Eq. 4). In our design, all final pyramid maps are generated simultaneously, and is more efficient.

Under Faster R-CNN pipeline To validate the generation of the proposed feature reconfiguration method, we conduct experiment under *two-stage* Faster R-CNN pipeline. In Table 2, Faster R-CNN with ResNet-101 get mAP of 78.9%. Feature Pyramid Networks with lateral connection improve the result to 79.8% (+0.9%). When replacing the lateral connection with global-local transformation, we get score of 80.6% (+1.8%). This result indicate that our global-and-local reconfiguration is also effective in *two-stage* object detection frameworks and could improve its performance.

Comparison with other state-of-the-arts Table 3 shows our results on VOC2007 test set based on SSD [34]. Our model with 300×300 achieves 79.6% mAP, which is much better than baseline method SSD300 (77.5%) and on par with SSD512. Enlarging the input image to 512×512 improves the result to 81.1%. Notably our model is much better than other methods which try to include context information such as MRCNN [10] and ION [3]. When replace the backbone network from VGG-16 to ResNet-101, our model with 512×512 scores 82.4% without bells and whistles, which is much better than the one-stage DSSD [15] and two-stage R-FCN [6].

Table 2. Effectiveness of various designs within Faster R-CNN.

method	backbone	mAP(%)
Faster [39]	VGG-16	73.2
Faster [6]	ResNet-101	76.4
Faster(ours-re)	ResNet-50	77.6
Faster(ours-re)	ResNet-101	78.9
Faster+FPNs	ResNet-50	78.8
Faster+FPNs	ResNet-101	79.8
Faster+Global-Local	ResNet-50	79.4
Faster+Global-Local	ResNet-101	**80.6**

Table 3. PASCAL VOC 2007 test detection results. All models are trained with 07+12 (07 trainval + 12 trainval). The entries with the best APs for each object category are bold-faced.

Method	backbone	mAP(%)	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster[39]	VGG-16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
ION[3]	VGG-16	76.5	79.2	79.2	77.4	69.8	55.7	85.2	84.2	89.8	57.5	78.5	73.8	87.8	85.9	81.3	75.3	49.7	76.9	74.6	85.2	82.1
MRCNN[16]	VGGNet	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	75.5	85.0	81.0
Faster[39]	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	79.8	85.3	72.0
R-FCN[6]	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[34]	VGG-16	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
SSD512[34]	VGG-16	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
StairNet[46]	VGG-16	78.8	81.3	85.4	77.8	72.1	59.2	86.4	86.8	87.5	62.7	85.7	76.0	84.1	88.4	86.1	78.8	54.8	77.4	79.0	88.3	79.2
RON320[26]	VGG-16	76.6	79.4	84.3	75.5	69.5	56.9	83.7	84.0	87.4	57.9	81.3	74.1	84.1	85.3	83.5	77.8	49.2	76.7	77.3	86.7	77.2
DSSD321[15]	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
DSSD513[15]	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**
Ours300	VGG-16	79.6	84.5	85.5	77.2	72.1	53.9	87.6	87.9	89.4	63.8	86.1	76.1	87.3	88.8	86.7	80.0	54.6	80.5	81.2	88.9	80.2
Ours512	VGG-16	81.1	90.0	87.0	79.9	75.1	60.3	88.8	**89.6**	89.6	65.8	**88.4**	79.4	87.5	90.1	85.6	81.9	54.8	79.0	80.8	87.2	79.9
Ours300	ResNet-101	80.2	89.3	84.9	79.9	**75.6**	55.4	88.2	88.6	88.6	63.3	87.9	78.8	87.3	87.7	85.5	80.5	55.4	81.1	79.6	87.8	78.5
Ours512	ResNet-101	**82.4**	**92.0**	**88.2**	81.1	71.2	65.7	88.2	87.9	**92.2**	65.8	86.5	**79.4**	**90.3**	90.4	89.3	**88.6**	**59.4**	**88.4**	75.3	**89.2**	78.5

To understand the performance of our method in more detail, we use the detection analysis tool from [22]. Figure 4 shows that our model can detect various object categories with high quality. The recall is higher than 90%, and is much higher with the 'weak' (0.1 jaccard overlap) criteria.

4.2 PASCAL VOC 2012

For VOC2012 task, we follow the setting of VOC2007 and with a few differences described here. We use 07++12 consisting of VOC2007 trainval, VOC2007 test, and VOC2012 trainval for training and VOC2012 test for testing. We see the same performance trend as we observed on VOC 2007 test. The results, as shown in Table 4, demonstrate the effectiveness of our models. Compared with SSD [34] and other variants, the proposed network is significantly better(+2.7% with 300×300).

Compared with DSSD with ResNet-101 backbone, our model gets similar results with VGG-16 backbone. The most recently proposed RUN [28] improves the results of SSD with skip-connection and unified prediction. The method add several residual blocks to improve the non-linear ability before prediction. Compared with RUN, our model is more direct and with better detection performance. Our final result using ResNet-101 scores 81.1%, which is much better than the state-of-the-art methods.

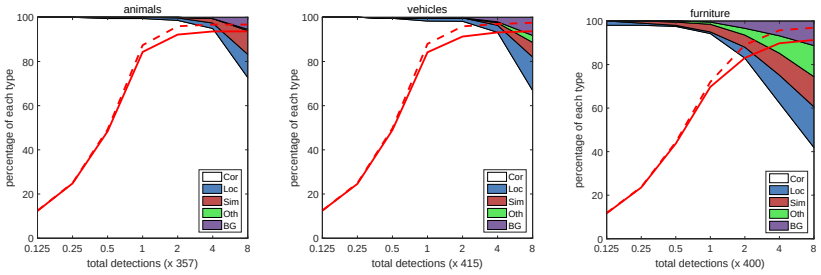

Fig. 4. Visualization of performance for our model with VGG-16 and 300×300 input resolution on animals, vehicles, and furniture from VOC2007 test. The Figures show the cumulative fraction of detections that are correct (Cor) or false positive due to poor localization (Loc), confusion with similar categories (Sim), with others (Oth), or with background (BG). The solid red line reflects the change of recall with the 'strong' criteria (0.5 jaccard overlap) as the number of detections increases. The dashed red line uses the 'weak' criteria (0.1 jaccard overlap).

Table 4. PASCAL VOC 2012 test detection results. All models are trained with 07++12 (07 trainval+test + 12 trainval). The entries with the best APs for each object category are bold-faced

Method	network	mAP(%)	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster[39]	ResNet-101	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
R-FCN[6]	ResNet-101	77.6	86.9	83.4	**81.5**	63.8	62.4	81.6	81.1	93.1	58.0	83.8	60.8	92.7	86.0	84.6	84.4	59.0	80.8	68.6	86.1	72.9
ION[3]	VGG-16	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	85.2	73.5
SSD300[34]	VGG-16	75.8	88.1	82.9	74.4	61.9	47.6	82.7	78.8	91.5	58.1	80.0	64.1	89.4	85.7	85.5	82.6	50.2	79.8	73.6	86.6	72.1
SSD512[34]	VGG-16	78.5	90.0	85.3	77.7	64.3	58.5	85.1	84.3	92.6	61.3	83.4	65.1	89.9	88.5	88.2	85.5	54.4	82.4	70.7	87.1	75.6
DSSD321[15]	ResNet-101	76.3	87.3	83.3	75.4	64.6	46.8	82.7	76.5	92.9	59.5	78.3	64.3	91.5	86.6	86.6	82.1	53.3	79.6	75.7	85.2	73.9
DSSD513[15]	ResNet-101	80.0	92.1	86.6	80.3	68.7	58.2	84.3	85.0	**94.6**	63.3	85.9	65.6	**93.0**	88.5	87.8	86.4	57.4	85.2	73.4	87.8	76.8
YOLOv2[38]	Darknet-19	75.4	86.6	85.0	76.8	61.1	55.5	81.2	78.2	91.8	56.8	79.6	61.7	89.7	86.0	85.0	84.2	51.2	79.4	62.9	84.9	71.0
DSOD[42]	DenseNet	76.3	89.4	85.3	72.9	62.7	49.5	83.6	80.6	92.1	60.8	77.9	65.6	88.9	85.5	86.8	84.6	51.1	77.7	72.3	86.0	72.2
RUN300[28]	VGG-16	77.1	88.2	84.4	76.2	63.8	53.1	82.9	79.5	90.9	60.7	82.5	64.1	89.6	86.5	86.6	83.3	51.5	83.0	74.0	87.6	74.4
RUN512[28]	VGG-16	79.8	**90.0**	87.3	80.2	67.4	62.4	84.9	**85.6**	92.9	61.8	84.9	66.2	90.9	89.1	88.0	86.5	55.4	**85.0**	72.6	87.7	76.8
StairNet[46]	VGG-16	76.4	87.7	83.1	74.6	64.2	51.3	83.6	78.0	92.0	58.9	81.8	**66.2**	89.6	86.0	84.9	82.6	50.9	80.5	71.8	86.2	73.5
Ours300	VGG-16	77.5	89.5	85.0	77.7	64.3	54.6	81.6	80.0	91.6	60.0	82.5	64.7	89.9	85.4	86.1	84.1	53.2	81.0	74.2	87.9	75.9
Ours512	VGG-16	80.0	89.6	**87.4**	80.9	68.3	61.0	83.5	83.9	92.4	63.8	85.9	63.9	89.9	89.2	88.9	86.2	56.3	84.4	75.5	89.7	78.5
Ours300	ResNet-101	78.7	89.4	85.7	80.2	65.1	58.6	84.3	81.8	91.9	63.6	84.2	65.6	89.6	85.9	86.0	85.0	54.4	81.9	**75.9**	87.8	77.5
Ours512	ResNet-101	**81.1**	87.4	85.7	81.4	**71.1**	**64.3**	**85.1**	84.8	92.2	**66.3**	**87.6**	66.1	90.3	**90.1**	**89.6**	**87.2**	**60.0**	84.4	75.7	**89.7**	**80.1**

4.3 MS COCO

To further validate the proposed framework on a larger and more challenging dataset, we conduct experiments on MS COCO [32] and report results from test-dev evaluation server. The evaluation metric of MS COCO dataset is different from PASCAL VOC. The average mAP over different IoU thresholds, from 0.5 to 0.95 (written as 0.5:0.95) is the overall performance of methods. We use the 80k training images and 40k validation images [32] to train our model, and validate the performance on the test-dev dataset which contains 20k images. For ResNet-101 based models, we set batch-size as 32 and 20 for 320×320 and 512×512 model separately, due to the memory issue.

With the standard COCO evaluation metric, SSD300 scores 25.1% AP, and our model improves it to 28.4% AP (+3.3%), which is also on par with DSSD with ResNet-101 backbone(28.0%). When change the backbone to ResNet-101, our model gets 31.3% AP, which is much better than the DSSD321(+3.3%). The

method	train Data	input size	network	Average Precision 0.5	0.75	0.5:0.95
two-stage						
OHEM++[43]	trainval	$\sim 1000 \times 600$	VGG-16	45.9	26.1	25.5
Faster[39]	trainval	$\sim 1000 \times 600$	VGG-16	42.7	-	21.9
R-FCN[6]	trainval	$\sim 1000 \times 600$	ResNet-101	51.9	-	29.9
CoupleNet[49]	trainval35k	$\sim 1000 \times 600$	ResNet-101	**54.8**	37.2	34.4
one-stage						
SSD300[34]	trainval35k	300×300	VGG-16	43.1	25.8	25.1
SSD512[34]	trainval35k	512×512	VGG-16	48.5	30.3	28.8
SSD513[15]	trainval35k	513×513	ResNet-101	50.4	33.1	31.2
DSSD321[15]	trainval35k	321×321	ResNet-101	46.1	29.2	28.0
DSSD513[15]	trainval35k	513×513	ResNet-101	53.3	35.2	33.2
RON320[26]	trainval	320×320	VGG-16	47.5	25.9	26.2
YOLOv2[38]	trainval35k	544×544	DarkNet-19	44.0	19.2	21.6
RetinaNet[31]	trainval35k	500×500	ResNet-101	53.1	36.8	34.4
Ours300	trainval	300×300	VGG-16	48.2	29.1	28.4
Ours512	trainval	512×512	VGG-16	50.9	32.2	31.5
Ours300	trainval	300×300	ResNet-101	50.5	32.0	31.3
Ours512	trainval	512×512	ResNet-101	54.3	**37.3**	**34.6**

Table 5. MS COCO test-dev2015 detection results.

accuracy of our model can be improved to 34.6% by using larger input size of 512×512, which is also better than the most recently proposed RetinaNet [31] that adds lateral connection and focal loss for better object detection.

Table 6 reports the multi-scale object detection results of our method under SSD framework using ResNet-101 backbone. It is observed that our method achieves better detection accuracies than SSD and DSSD for the objects of all scales.

Methods	AP_s	AP_m	AP_l	AP
SSD513	10.2	34.5	49.8	31.2
DSSD513	13.0	35.4	51.1	33.2
Ours512	14.7	38.1	51.9	34.6

Table 6. MS COCO test-dev2015 detection results on small (AP_s), medium (AP_m) and large (AP_l) objects.

5 Conclusions

A key issue for building feature pyramid representations under a ConvNet is to reconfigure and reuse the feature hierarchy. This paper deal with this problem

with global-and-local transformations. This representation allows us to explicitly model the feature reconfiguration process for the specific scales of objects. We conduct extensive experiments to compare our method to other feature pyramid variations. Our study suggests that despite the strong representations of deep ConvNet, there is still room and potential to building better pyramids to further address multiscale problems.

Acknowledgement This work was jointly supported by the National Science Fundation of China(NSFC) and the German Research Foundation(DFG) joint project NSFC 61621136008/DFG TRR-169 and the National Natural Science Foundation of China(Grant No: 61327809,61210013).

Fig. 5. Qualitative detection examples on VOC 2007 test set with SSD300 (77.5% mAP) and Ours-300 (79.6% mAP) models. For each pair, the left is the result of SSD and right is the result of ours. We show detections with scores higher than 0.6. Each color corresponds to an object category in that image.

References

1. Adelson, E.H., Anderson, C.H., Bergen, J.R., Burt, P.J., Ogden, J.M.: Pyramid methods in image processing. RCA engineer **29**(6), 33–41 (1984)

2. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence **39**(12), 2481–2495 (2017)
3. Bell, S., Lawrence Zitnick, C., Bala, K., Girshick, R.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2874–2883 (2016)
4. Benenson, R., Mathias, M., Timofte, R., Van Gool, L.: Pedestrian detection at 100 frames per second. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. pp. 2903–2910. IEEE (2012)
5. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. arXiv preprint arXiv:1711.07319 (2017)
6. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Advances in neural information processing systems. pp. 379–387 (2016)
7. Dollár, P., Appel, R., Belongie, S., Perona, P.: Fast feature pyramids for object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(8), 1532–1545 (2014)
8. Dollar, P., Appel, R., Kienzle, W.: Crosstalk cascades for frame-rate pedestrian detection. In: Computer Vision–ECCV 2012, pp. 645–659. Springer (2012)
9. Dollar, P., Belongie, S.J., Perona, P.: The fastest pedestrian detector in the west. In: BMVC. vol. 2, p. 7 (2010)
10. Dollar, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. IEEE transactions on pattern analysis and machine intelligence **34**(4), 743–761 (2012)
11. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010)
12. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. pp. 1–8. IEEE (2008)
13. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9), 1627–1645 (2010)
14. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9), 1627–1645 (2010)
15. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: Dssd: Deconvolutional single shot detector. arXiv preprint arXiv:1701.06659 (2017)
16. Gidaris, S., Komodakis, N.: Object detection via a multi-region and semantic segmentation-aware cnn model. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1134–1142 (2015)
17. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
18. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014)
19. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. arXiv preprint arXiv:1703.06870 (2017)

20. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence **37**(9), 1904–1916 (2015)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
22. Hoiem, D., Chodpathumwan, Y., Dai, Q.: Diagnosing error in object detectors. In: European conference on computer vision. pp. 340–353. Springer (2012)
23. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. arXiv preprint arXiv:1709.01507 (2017)
24. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al.: Speed/accuracy trade-offs for modern convolutional object detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7310–7311 (2017)
25. Jeong, J., Park, H., Kwak, N.: Enhancement of ssd by concatenating feature maps for object detection. arXiv preprint arXiv:1705.09587 (2017)
26. 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. vol. 1, p. 2 (2017)
27. Kong, T., Yao, A., Chen, Y., Sun, F.: Hypernet: Towards accurate region proposal generation and joint object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 845–853 (2016)
28. Lee, K., Choi, J., Jeong, J., Kwak, N.: Residual features and unified prediction network for single stage detection. arXiv preprint arXiv:1707.05031 (2017)
29. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
30. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. arXiv preprint arXiv:1612.03144 (2016)
31. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. arXiv preprint arXiv:1708.02002 (2017)
32. 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: European conference on computer vision. pp. 740–755. Springer (2014)
33. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. arXiv preprint arXiv:1803.01534 (2018)
34. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
35. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)
36. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
37. 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. pp. 779–788 (2016)
38. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. arXiv preprint arXiv:1612.08242 (2016)
39. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)

40. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)
41. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229 (2013)
42. Shen, Z., Liu, Z., Li, J., Jiang, Y.G., Chen, Y., Xue, X.: Dsod: Learning deeply supervised object detectors from scratch. In: The IEEE International Conference on Computer Vision (ICCV). vol. 3, p. 7 (2017)
43. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 761–769 (2016)
44. Shrivastava, A., Sukthankar, R., Malik, J., Gupta, A.: Beyond skip connections: Top-down modulation for object detection. arXiv preprint arXiv:1612.06851 (2016)
45. Wang, X., Han, T.X., Yan, S.: An hog-lbp human detector with partial occlusion handling. In: CVPR (2009)
46. Woo, S., Hwang, S., Kweon, I.S.: Stairnet: Top-down semantic aggregation for accurate one shot detection. arXiv preprint arXiv:1709.05788 (2017)
47. Yang, W., Li, S., Ouyang, W., Li, H., Wang, X.: Learning feature pyramids for human pose estimation. In: The IEEE International Conference on Computer Vision (ICCV). vol. 2 (2017)
48. 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)
49. Zhu, Y., Zhao, C., Wang, J., Zhao, X., Wu, Y., Lu, H.: Couplenet: Coupling global structure with local parts for object detection. In: Proc. of Intl Conf. on Computer Vision (ICCV) (2017)