Reverse Densely Connected Feature Pyramid Network for Object Detection

Yongjian Xin[1,3], Shuhui Wang[1⋆], Liang Li[1], Weigang Zhang[2,3], and Qingming Huang[3]

[1] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
[2] School of Computer Science and Technology, Harbin Institute of Technology, Weihai 264209, China
[3] University of Chinese Academy of Sciences, Beijing 100049, China
{yongjian.xin,liang.li}@vipl.ict.ac.cn, wangshuhui@ict.ac.cn
wgzhang@hit.edu.cn, qmhuang@ucas.ac.cn

Abstract. The wide and extreme diversity of object size is an everlasting challenging issue in object detection research. To address this problem, we propose Reverse Densely Connected Feature Pyramid Network (Rev-Dense FPN), a novel multi-scale feature transformation and fusion method for object detection. Through reverse dense connection, we directly fuse all the feature maps of higher levels than the current one. This avoids useful contextual information on the higher level to vanish when passed down to lower levels, which is a key disadvantage of widely used feature fusion paradigms such as recursive top-down connection. Therefore, a more powerful hierarchical representation structure can be obtained by effectively aggregating multi-level contexts. We apply Rev-Dense FPN on SSD framework, which reaches 81.1% mAP (mean average precision) on the PASCAL VOC 2007 dataset and 31.2 AP on the MS COCO dataset. The results show that Rev-Dense FPN is more effective in dealing with diversified object sizes.

Keywords: Object Detection · Convolutional Neural Networks · Feature Pyramid.

1 Introduction

Object detection is one of the key techniques for significant applications in vision research, such as autonomous driving, intelligent video surveillance, and so on. Due to the prevalence of deep convolutional neural networks (CNNs), recent years have witnessed a remarkable progress in object detection literature. Following either the two-stage paradigm stemming from the R-CNN [7] and Faster R-CNN [25] series, or the one-stage framework stemming from YOLO [23] and SSD [21], endeavors have been devoted to improving detection performance. Among these efforts, a critical class of approaches [21, 18, 27, 15] are proposed

⋆ indicates corresponding author.

to deal with the multi-scale problem caused by a wide and extreme diversity of object sizes, which is an ever-lasting challenging issue in object detection.

Generally, existing works are naturally conducted on the basis of multi-scale detection frameworks, *i.e.*, multi-scale single stage detectors like SSD or multi-scale variants of Faster R-CNN. Then feature fusion techniques are developed to enhance low-level features for small objects, such as FPN [18], TDM [27], FSSD [17], etc. The main idea of these approaches is to recursively fuse deeper-layer features onto shallower layers, making lower-level features more discriminative and boosted by contextual information. However, for a complicated image with multiple objects of different scales, it is hard to determine which upper-layer features above the current layer are helpful. The recursive top-down connection which is widely used in existing model solutions such as FPN or TDM may cause the useful information from higher level to vanish along the layer-by-layer propagation path, which makes it hard for lower-level features to benefit from the contexts delivered by upper-layer features. To address this problem, in this paper, we propose Reverse Densely Connected Feature Pyramid Network (Rev-Dense FPN), a novel multi-scale feature fusion method which obtains feature pyramid by reverse dense connections.

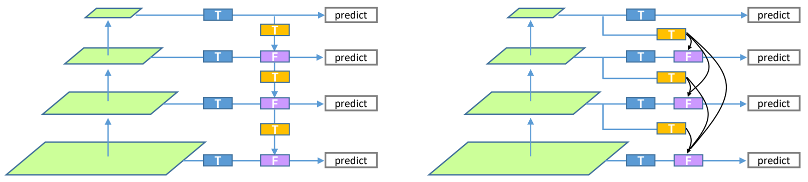

Fig. 1. The recursive top-down connection used in [18, 27](left) vs. the proposed reverse dense connection (right). T denotes a learnable layer and F denotes fusion operation. Different colors of modules indicate different functions. In the left part, blue Ts are lateral layers while yellow ones are top-down layers. The fusion operation F is element-wise addition for FPN and concatenation for TDM. In the right part, blue Ts are layers for adjusting the receptive fields while the yellow ones are layers for channel reduction. F is concatenation.

As shown in the right part of Fig. 1, for each level of layer, a reverse dense connection directly aggregates features from all the upper-level layers. Through this type of connection, it avoids the aforementioned problem of recursive connection that higher-level contextual information may vanish as it is passed recursively to the lower levels. Since we concatenate several higher-level feature maps with increasing receptive fields, the representation obtained for the current level is highly hierarchical and contains rich multi-level conditional information, thus it tends to be more capable of dealing with the diversified object sizes. Moreover, by applying direct linking, the reverse dense connection also makes information flow forward and backward more smoothly through the network, so that the optimization of deep model parameters can be more sufficient. In our work, we choose a set of feature maps from the backbone network and apply reverse dense connection for each of them. This will generate a new set of feature maps which are taken as our final feature pyramid for the consequent operations. The corresponding feature pyramid transformation is denoted as Rev-Dense FPN since reverse dense connection is the major operation for construction of such pyramidal representation.

We apply Rev-Dense FPN to the SSD detection framework and show that SSD with Rev-Dense FPN can obtain a significant performance gain compared with the original SSD baseline as well as other feature fusion methods. Reaching higher accuracy, our method only suffers a small speed drop. We conduct object detection experiments on the PASCAL VOC 2007 dataset [5] and the MS COCO dataset [20]. Results demonstrate the effectiveness and efficiency of the proposed Rev-Dense FPN. Using only the moderate-sized VGG16 [28] network as backbone, our approach obtains 81.1% mAP on the PASCAL VOC 2007 dataset and 31.2 AP on the MS COCO dataset, outperforming a number of multi-scale feature fusion methods as well as some state-of-the-art detectors.

Our main contributions can be summarized as follows:

– We propose Rev-Dense FPN, a multi-scale feature fusion method based on reverse dense connection, which directly aggregates features from all the feature levels higher than the current one, forming a hierarchical feature pyramid representation that is boosted by aggregating multi-level contexts more comprehensively.
– We apply Rev-Dense FPN to the SSD network and show that SSD with Rev-Dense FPN can obtain a significant performance gain compared with other feature fusion methods. Besides, our method can also be applied to other state-of-the-art object detection frameworks.
– Experimental results demonstrate the effectiveness and efficiency of the proposed Rev-Dense FPN on the PASCAL VOC 2007 dataset and the MS COCO dataset.

2 Related Work

2.1 Two General Frameworks for Object Detectors

One-Stage Detectors. One-stage detector, also called single-stage detector, can be termed as a more powerful class-aware Region Proposal Network (RPN) [25]. Typical works for one-stage detectors include SSD [21], DSSD [6], YOLO v2 [24], RON [14], RetinaNet [19] and so on. Taking a resized image as input, the detector runs in a fully-convolutional manner. Through one forward pass, the detector outputs category probabilities and bounding box offsets for all pre-defined anchors, which are then interpreted and post-processed by removing duplicate boxes. One-stage detectors are usually endowed with relatively faster speed, but with lower potential of accuracy.

Two-Stage Detectors. Another paradigm for CNN-based object detectors is introduced by the R-CNN [7] and Faster R-CNN [25] series, which are called two-stage detectors. Typical works are SPPNet [10], ION [1], R-FCN [3], Mask R-CNN [8], Light-Head R-CNN [16], etc. These detectors first generate a sparse set of candidate boxes (*i.e.*, object proposals or region proposals) via a fully convolutional class-agnostic weak detector, *i.e.*, the Region Proposal Network. Then the generated proposals are processed by a head subnet that classifies these proposals and offsets their locations. Because the feature for each Region-of-Interest (RoI) is better aligned and more fine-grained than the single-stage framework, two-stage approaches usually get high accuracy. The weakness of two-stage methods is their relatively lower detection efficiency.

2.2 Multi-Scale Feature Fusion

Detecting objects of a wide range of scales has long been a difficult problem. Before the advent of multi-scale detection frameworks, there have been algorithms that fuse features from different layers to build "hyper-feature map". These methods typically choose one single feature map with the most appropriate spatial scale, pick some features from other layers (often higher-level layers) and then fuse them together to construct the final feature map on which the detection task happens. Such operations have been applied on HyperNet [15], MS-CNN [2] and PVANET [12].

Since a single feature map is hard to cover objects of various scales, there come up multi-scale detectors like SSD and multi-scale Faster R-CNN. Under multi-scale detection frameworks, the feature used for detecting objects is naturally a feature pyramid. Objects with different scales are respectively distributed onto the corresponding levels of the feature pyramid. To enhance pyramidal representation for multi-scale detection, there are several works based on feature pyramid transformation.

Feature Pyramid Network (FPN) [18] is the first work that performs feature pyramid transformation to get better features, especially for the early stage feature maps to which small objects are assigned. The main idea of FPN is to recursively fuse the more discriminative and semantically consistent features from higher levels onto the current one. Through the aggregation, objects assigned to lower levels are better detected. However, FPN uses element-wise addition as its fusion manner, which can make features to be highly miscellaneous. Instead of element-wise addition, TDM [27] uses concatenation to fuse higher level features. But to avoid information loss, TDM keeps a large channel number when applying concatenation, making it computationally inefficient. TDM also keeps recursive linking as used in FPN. This may cause higher-level information to diminish when it is propagated top-down, especially when there are relatively more feature levels. Besides FPN and TDM, FSSD [17] involves another fusion manner which takes some levels in early stages with relatively larger spatial scales and fuses them together, then down-samples the fused feature map to get higher pyramid levels. FSSD is concise and lightweight, but it suffers the drawback of early-stage fusion, which will limit the capacity of the representation.

Additionally, there are also works based on reorganizing features, like STDN [29]. STDN expands a group of smaller feature maps to a larger plane. Since it uses DenseNet [13] as its backbone network which naturally combines features from previous layers, explicit feature aggregation across different layers is not used.

3 Rev-Dense FPN

3.1 Reverse Dense Connection

For convenience of illustration, we first formulate the mechanism of multi-scale detectors (with feature pyramid transformation) as follows:

Given n feature maps of decreasing spatial scales $\mathbf{C} = \{C_i\}_{i=1}^n$ chosen from the backbone network, a group of transformations $\mathbf{\Phi} = \{\phi_i\}_{i=1}^n$ is then conducted on $\mathbf{C}$ to yield a feature pyramid $\mathbf{P} = \{P_i\}_{i=1}^n$. For each ϕ_i, it takes a subset of $\mathbf{C} \cup \{P_k\}_{k=i+1}^n$ as input and outputs P_i. Then a series of detection operations $\{det_i(\cdot)\}_{i=1}^n$ are applied on $\mathbf{P}$ correspondingly to get the detection results $\boldsymbol{\mathcal{D}} = \bigcup_{i=1}^n \mathcal{D}_i$, where $\mathcal{D}_i = det_i(P_i)$. More specific mathematical expression is shown as follows (Algorithm 1):

Algorithm 1 Multi-scale detectors with feature pyramid transformation.

```
Input: Chosen backbone stages: $\mathbf{C} = \{C_i\}_{i=1}^n$.
Output: Transformed feature pyramid: $\mathbf{P} = \{P_i\}_{i=1}^n$; Detection results: $\boldsymbol{\mathcal{D}} = \bigcup_{i=1}^n \mathcal{D}_i$.
1: Initialize $\mathbf{P} = \varnothing$, $\boldsymbol{\mathcal{D}} = \varnothing$.
2: for $i = n : -1 : 1$
3:     select $\mathcal{S}_i \subseteq \mathbf{C} \cup \mathbf{P}$
4:     $P_i = \phi_i(\mathcal{S}_i)$
5:     update $\mathbf{P} = \mathbf{P} \cup \{P_i\}$
6: for $i = 1 : n$
7:     $\mathcal{D}_i = det_i(P_i)$
8:     update $\boldsymbol{\mathcal{D}} = \boldsymbol{\mathcal{D}} \cup \mathcal{D}_i$
```

Through the formulation it is obvious that the transformation procedure $\mathbf{\Phi}$ is critical for the quality of the feature pyramid $\mathbf{P}$ used for detection. For works like FPN [18] and TDM [27], ϕ_i is defined as a recursive aggregating operation, which takes the last-step generated P_{i+1} and the current backbone stage C_i as inputs and outputs the current-level fused feature map P_i. The context information of higher-level layer C_i is first aggregated with C_{i-1} and then passed to lower layers $C_j(i-j \geq 2)$. Such recursive aggregating operation may cause the context information C_i to diminish in $C_j(i-j \geq 2)$, which is swiped away and of no contribution for detection in $C_j(i-j \geq 2)$.

In our work, we introduce reverse dense connection instead of recursive connection and use concatenation as our feature fusion strategy. Specifically, to obtain P_i, we take all $\{C_k\}_{k \geq i}$ as the input of ϕ_i. For each C_k, some operations are applied to either reduce the channel number and match its spatial scale with

C_i or adjust its receptive field. Then the transformed feature maps are concatenated together to form P_i. Since all the feature map levels higher than the current one are taken as the input of ϕ_i and their information is linked to the current level, this type of connection is naturally called reverse dense connection.

There are several advantages of reverse dense connection.

First, our reverse dense connection passes contextual information of various higher levels directly to the current stage, which avoids information to diminish. It is commonly recognized that detecting objects of various scales in a comprehensive image needs contextual information, especially for small objects whose features often appear at shallow layers and are less discriminative. However, due to the high complexity of data, it is hard to distinguish which level of contextual information is helpful for classifying objects on the current one. Therefore, it is better to pass all contextual information of various higher levels directly to the current stage.

Second, concatenating several higher levels that have increasing receptive fields yields hierarchical representation. Boosted by multi-level contexts, such hierarchical representations tend to be more discriminative.

Third, direct linking in reverse dense connection reduces the number of repeating times for stacked learnable modules, leading to more efficient information flow and smoother optimization.

In the next section, we will describe in detail both the formulation of $\{\phi_i\}_{i=1}^n$ and our Rev-Dense FPN based on reverse dense connection.

3.2 Feature Pyramid Transformation for Rev-Dense FPN

In this section we describe in detail Rev-Dense FPN. Suppose we have chosen n backbone stages $\mathbf{C} = \{C_i\}_{i=1}^n$. After the Rev-Dense FPN transformation, it outputs n fused feature pyramid levels $\mathbf{P} = \{P_i\}_{i=1}^n$.

As shown in Fig. 2, for backbone stage C_i, we apply a 3×3 convolution with k_1 output channels. For each backbone stage in $\{C_{i+1}, C_{i+2}, \ldots, C_n\}$, a 1×1 convolution with k_2 output channels is applied, followed by a bilinear up-sample operation to match the spatial scale of C_i. These convolution operations are used for the sake of projecting the representations of $\{C_i, C_{i+1}, \ldots, C_n\}$ into a common space for fusion. Finally, we fuse all the resulted feature maps by concatenation and get P_i.

It should be noted that the 1×1 convolution is performed only once for one certain stage C_i for computation efficiency. And then we apply several bilinear up-sample operations with different spatial outputs to match different scales of $C_1, C_2, \ldots, C_{i-1}$ for later fusion.

k_1, k_2 are tunable parameters. Typically, k_1 is set larger while k_2 is much smaller, similarly to the growth rate in DenseNet [13]. For all the convolution layers used in Rev-Dense FPN, we do not apply non-linearity to their outputs.

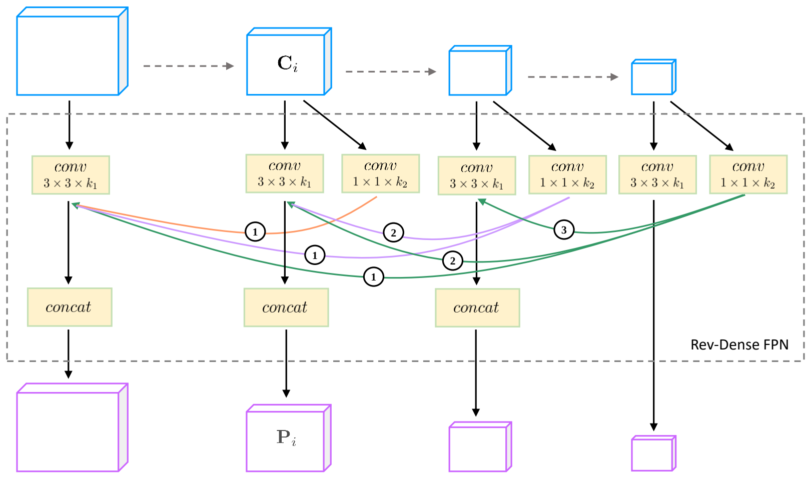

Fig. 2. Structure of Rev-Dense FPN. Taking a set of backbone stages with decreasing spatial scales, it outputs a feature pyramid of the same sizes. The gray dashed arrow lines indicate the propagating direction of the backbone network. Each of the colored arrow lines corresponds to a bi-linear up-sample operation, which is marked by a small circle. The number in the small circle suggests the level number for the targeting spatial size when up-sample is performed.

According to the description above, the feature pyramid transformation $\mathbf{\Phi} = \{\phi_i\}_{i=1}^{n}$ for Rev-Dense FPN can be mathematically formulated as follows:

$$P_i = \phi_i(C_i, C_{i+1}, \ldots, C_n) = \mathcal{F}^i_{3\times3}(C_i) \oplus \left(\bigoplus_{k=i+1}^{n} \text{Up}\left(\mathcal{F}^k_{1\times1}(C_k), \text{size}(C_i)\right) \right) \quad (1)$$

Where $\mathcal{F}$ denotes convolution, $\oplus$ denotes concatenation and Up is bilinear up-sample operation.

3.3 Network Architecture

We adopt SSD as our baseline algorithm to validate the effectiveness of Rev-Dense FPN. The backbone network is VGGNet which is exactly the same as original SSD.

For input size 300×300, the SSD baseline extracts 6 feature maps to detect objects of different scales. As shown in Fig. 3, the feature maps extracted are $conv4\_3$, $conv7$, $conv8\_2$, $conv9\_2$, $conv10\_2$, and $conv11\_2$. The corresponding spatial scales are 38, 19, 10, 5, 3, 1. To apply Rev-Dense FPN, we first omit the last level $conv11\_2$ for its extreme small spatial size. Then for $conv7$ that has 1024 channels, we add a 1×1 convolution layer followed by relu activation to reduce the channel number to 512. Such channel reduction is for the sake of computation conservation and the reduced feature map is denoted as $conv7'$. Therefore, the chosen backbone stages are $\mathbf{C} = \{conv4\_3, conv7', conv8\_2, conv9\_2, conv10\_2\}$. Then Rev-Dense FPN transformation $\mathbf{\Phi}$ is applied to $\mathbf{C}$ (with $n = 5$), yielding $\mathbf{P} = \{P_i\}_{i=1}^{5}$. To keep the number of feature levels consistent with baseline, we add $conv11\_2$ in and take $\mathbf{P} \cup \{conv11\_2\}$ as our final feature pyramid. At last we add classification and localization layers on the final feature pyramid to obtain detection outputs.

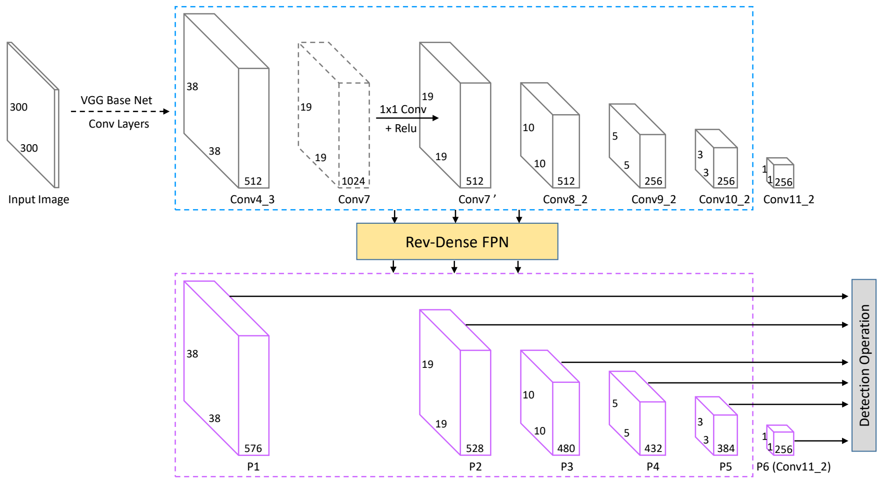

Fig. 3. Network architecture for applying Rev-Dense FPN on SSD300. The blue dashed rectangle encloses the input for Rev-Dense FPN transformation, while the purple dashed rectangle denotes the output. The cube with gray dashed border denotes that it is not contained in the input maps. The channel numbers of the output feature pyramid are in accordance with configuration $k_1 = 384$, $k_2 = 48$.

For input size 512×512 where there are 7 levels extracted by the baseline SSD, we also adopt similar strategy, *i.e.*, omit the last level when apply Rev-Dense operation, reduce the dimension for 1024-channel map and apply classification and localization layers on the final feature pyramid.

4 Experiments

We conduct experiments on the PASCAL VOC 2007 dataset [5] and the MS COCO dataset [20], which have 20 and 80 categories respectively. In the following sections, we first describe the implementation details of our approach (Section 4.1). Then in Section 4.2 and Section 4.3, we will describe training and testing settings on the two datasets and show the experimental results. To demonstrate the flexibility of our method, we also extend Rev-Dense FPN to two-stage method (multi-scale Faster R-CNN with FPN), which is shown in Section 4.4.

4.1 Implementation Details

We implement SSD with Rev-Dense FPN using the deep learning framework PyTorch [22]. The training objective is a combination of classification loss and localization loss. And the definitions of both the two components are consistent with the original SSD. Since there are many up-sample operations with large scaling factors in Rev-Dense FPN, we set the weight of localization loss to 1.5 for better spatial alignment of the final feature pyramid. Base VGGNet pre-trained on ImageNet [4] is used for initialization. For all the layers without ImageNet pre-trained parameters, we use the uniform version of the MSRA method [9] to initialize their weights randomly. At the beginning of training procedure, we apply the linear warm-up strategy. Specifically, we set the learning rate to 1/3 of the base learning rate at the first step, linearly increase it to the base learning rate through the following 300 steps and then keep on. The other details are kept the same as the original SSD, such as sampling strategy, hard example mining rules, data augmentation, etc.

4.2 PASCAL VOC 2007

Settings of Training and Testing. For the PASCAL VOC 2007 dataset, we use the *voc07trainval* and *voc12trainval* for training and test the models on the *voc07test* set. We use a batch size of 32 and the SGD optimizer. The momentum is set to 0.9 and the weight decay is 0.0001. We train our models for 250 epochs in total (173k iterations). The learning rate is 10^{-3} for the first 100k iterations (using the warm-up strategy described in Section 4.1), 10^{-4} for the next 40k iterations and 10^{-5} for the rest iterations. At the testing stage, NMS (Non-Maximum Suppression) with 0.45 IoU threshold is performed as post-processing operation to remove duplicate boxes. Finally, we evaluate the mean average precision (mAP) on the *voc07test* set.

We train our models with 4 Nvidia GTX 1080Ti GPUs and test on a single GPU. We implement two Rev-Dense FPNs including Rev-Dense FPN (SSD300) and Rev-Dense FPN (SSD512) with input size 300×300 and 512×512 respectively. We set $k_1 = 384$ and $k_2 = 48$ for Rev-Dense FPN (SSD300) and $k_1 = 256$ and $k_2 = 48$ for Rev-Dense FPN (SSD512).

Study on the Effectiveness of Rev-Dense FPN. To demonstrate our argument that reverse dense connection is superior to recursive top-down connection for multi-scale feature fusion or feature pyramid transformation, we conduct the following experiments with SSD300 as baseline:
Transplant FPN to SSD We transplant FPN to the SSD detector. We implement the connection structure defined by FPN and set output channel number of the lateral layers to 256, which is most commonly used by the community. The backbone stages chosen for the FPN transformation are all the 6 feature maps extracted by the baseline. We do not omit the last feature map. The reason is that in methods based on recursive top-down connection, up-sample operations always happen between two consecutive feature levels. Therefore, there is no

extreme scaling factor for up-sampling in FPN. The resulted model reaches 78.3% mAP (see Table 1), which is by 0.9% lower than Rev-Dense FPN.

Transplant TDM to SSD Similar to transplanting FPN on SSD, we add modules as described in [27] to SSD. The output channel numbers for lateral layers, top-down layers and out layers are set to 128, 128 and 256, respectively. This is slightly different with [27] in which the three numbers are inconsistent at each stage. We choose the largest ones and keep them consistent, since we think this will enhance the representation power of the model. All the 6 feature maps extracted by SSD are taken as the input of TDM. As shown in Table 1, SSD with TDM reaches an mAP of 78.2%, which is by 1.0 point lower than Rev-Dense FPN.

Remove Reverse Dense Connections To verify that it is the reverse dense connections from higher-level layers that contribute to our method rather than the projection operations which happen on the current levels, we remove reverse dense connections from higher-level layers for ablation study. With the removal of reverse dense connections, the mAP point drops to 77.9% (see Table 1), which is by 1.3 points lower than the model with reverse dense connections from higher levels.

Table 1. Experimental results of recursive top-down connection vs. reverse dense connection for multi-scale feature fusion (feature pyramid transformation). The baseline for this group of experiments is SSD300. All the models are trained on *voc07trainval* and *voc12trainval* and tested on *voc07test*.

Feature pyramid transformation	Connection style	mAP (%)
None (Baseline)	-	77.5
FPN [18]	Recursive top-down	78.3
TDM [27]	Recursive top-down	78.2
Rev-Dense FPN (SSD300) w/o reverse connections	-	77.9
Rev-Dense FPN (SSD300)	Reverse dense	**79.2**

The experiments above validate that reverse dense connection is better than recursive top-down connection. As shown in Table 1, using reverse dense connection, Rev-Dense FPN (SSD300) reaches 79.2% mAP. But for the methods based on recursive top-down connection FPN and TDM, the mAPs are around 1 point lower than our Rev-Dense FPN (SSD300), which supports our point of view. The removal of reverse dense connections as ablation (77.9% mAP vs. 79.2% mAP) also verifies its contribution.

Ablation Study on Which Backbone Stages to Choose. To further study the contributions of different feature levels, we conduct another ablation experiment in which various combinations of input backbone stages are involved. With the basic setting $\{C_1, C_2, C_3, C_4, C_5, C_6\} = \{conv4\_3$, $conv7'$, $conv8\_2$, $conv9\_2$, $conv10\_2$, $conv11\_2\}$, we consider the following combinations of $\{C_i\}_{i=1}^{6}$: $(C_2, C_3,$

C_4, C_5, C_6), (C_1, C_3, C_5), (C_1, C_2, C_3, C_4) and (C_1, C_2, C_3, C_4, C_5), using SSD300 as the baseline. The selected stages in a combination are taken as the input of Rev-Dense FPN transformation while the backbone stages excluded are kept the same in the output feature pyramid. Results are shown in Table 2.

Table 2. Ablation study on which backbone stages to choose. The baseline is SSD300. All the models are trained on *voc07trainval* and *voc12trainval* and tested on *voc07test*.

Backbone stages	Number of levels	mAP (%)
None (Baseline)	-	77.5
$(C_2, C_3, C_4, C_5, C_6)$	5	78.4
(C_1, C_3, C_5)	3	78.7
(C_1, C_2, C_3, C_4)	4	78.8
$(C_1, C_2, C_3, C_4, C_5)$	5	**79.2**

As the results suggest, leaving the lowest level out entails the poorest result (78.4% mAP). This indicates that small objects can benefit largely from reverse dense connections. Without being involved in the fusion transformation, low-level features for small objects become less of representation power. Taking the interleaved 3 stages as the input of Rev-Dense FPN yields similar performance as using (C_1, C_2, C_3, C_4) (78.7% mAP vs. 78.8% mAP). But adding C_2 and C_4 back reaches the highest performance (79.2% mAP), which proves their importance. Through the comparison between the last two items, the contribution of level C_5 is also confirmed.

Results on PASCAL VOC 2007. We show the experimental results on PASCAL VOC 2007 for Rev-Dense FPN (SSD300) and Rev-Dense FPN (SSD512). As shown in Table 3, using only the VGGNet as backbone, our Rev-Dense FPN (SSD300) reaches 79.2% mAP, surpassing many other SSD-like detectors, including DSSD321, DSOD300, STDN300 and FSSD300. Reaching high accuracy, Rev-Dense FPN (SSD300) runs at a high speed (70.9 fps), which is only of a small speed drop compared to SSD300 and is faster than FSSD300 as well. For input size 512×512, our Rev-Dense FPN (SSD512) reaches 81.1% mAP, which is also higher than STDN513 and FSSD512. Though DSSD513 is slightly higher than our method, it is much slower than out Rev-Dense FPN (SSD512). Note that there is huge gap of model capacity between the two backbone networks (VGGNet [28] for Rev-Dense FPN vs. ResNet-101 [11] for DSSD). ResNet-101 is much more powerful than VGGNet. With such huge gap, our Rev-Dense FPN (SSD512) is by a small margin lower than DSSD513, which validates the effectiveness of our reverse dense connection.

Table 3. Experimental results on the PASCAL VOC 2007 dataset. The baselines (SSD300 and SSD512) reported here are our own implementation, with SSD300 slightly higher than the original paper (77.2% mAP) and SSD512 the same. For Rev-Dense FPN (SSD300), we set $k_1 = 384$ and $k_2 = 48$. For Rev-Dense FPN (SSD512), we set $k_1 = 256$ and $k_2 = 48$. All the entries listed use *voc07trainval* and *voc12trainval* as training data and use *voc07test* as test data.

Method	Backbone	Input size	GPU	Speed (fps)	mAP (%)
Faster R-CNN [25]	ResNet-101	600×1000	K40	2.4	76.4
R-FCN [3]	ResNet-50	600×1000	-	-	77.0
SSD300 [21]	VGGNet	300×300	1080Ti	**83.3**	77.5
SSD512 [21]	VGGNet	512×512	1080Ti	**39.2**	79.8
YOLOv2 [24]	DarkNet-19	544×544	Titan X	40	78.6
DSOD300 [26]	DS/64-192-48-1	300×300	Titan X	17.4	77.7
DSSD321 [6]	ResNet-101	300×300	Titan X	9.5	78.6
DSSD513 [6]	ResNet-101	513×513	Titan X	5.5	**81.5**
STDN300 [29]	DenseNet-169	300×300	Titan Xp	41.5	78.1
STDN513 [29]	DenseNet-169	513×513	Titan Xp	28.6	80.9
FSSD300 [17]	VGGNet	300×300	1080Ti	65.8	78.8
FSSD512 [17]	VGGNet	512×512	1080Ti	35.7	80.9
Rev-Dense FPN (SSD300)	VGGNet	300×300	1080Ti	**70.9**	**79.2**
Rev-Dense FPN (SSD512)	VGGNet	512×512	1080Ti	**38.0**	**81.1**

4.3 MS COCO

Settings of Training and Testing. For the MS COCO dataset, we train our model on the *COCO2017 train* set and test on the *COCO test-dev* set. The batch size is 32 and the SGD optimizer is used. We set the momentum and the weight decay to 0.9 and 0.0001. The model is trained for 110 epochs in total (around 403k iterations). The learning rate is 10^{-3} for the first 280k iterations (using the warm-up strategy described in Section 4.1), 10^{-4} for the next 90k iterations and 10^{-5} for the rest. At testing stage, NMS with 0.5 IoU threshold is applied.

We implement Rev-Dense FPN (SSD512) on the COCO dataset and set $k_1 = 256$, $k_2 = 48$.

Results on COCO. We evaluate our model on the *COCO test-dev* set, using the official evaluation server. Results are shown in Table 4. On the *COCO test-dev* set, Rev-Dense FPN (SSD512) reaches 31.2 AP, which is by a large margin higher than the SSD512 baseline. It also reaches comparable result as the SSD513 which uses the much powerful ResNet-101 for backbone. Though the AP of Rev-Dense FPN (SSD512) is slightly lower than STDN513 and FSSD512, it is worth noticing that for the performance on small objects (AP_S), our result is the highest (15.5) among the listed methods. Since small objects are mostly distributed onto lower-level layers which benefit most from reverse dense connections, the result further suggests the effectiveness of reverse dense connection

Table 4. Experimental Results on the *COCO test-dev* set. All the entries are trained on the *COCO2017 train* set or the *trainval35k* set which is identical. Column 3∼5 show the AP points corresponding to different IoU thresholds (AP: AP points averaged over IoU thresholds 0.50:0.05:0.95. AP_{50}, AP_{75}: AP point when IoU threshold is 0.5 or 0.75). The last 3 columns show the AP points corresponding to different object scales. Subscripts S, M, L represent small, medium, large respectively.

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv2 [24]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD512 [21]	VGGNet	28.8	48.5	30.3	10.9	31.8	43.5
SSD513 [6]	ResNet-101	31.2	50.4	33.3	10.2	34.5	**49.8**
STDN513 [29]	DenseNet-169	**31.8**	51.0	**33.6**	14.4	**36.1**	43.4
FSSD512 [17]	VGGNet	**31.8**	52.8	33.5	14.2	35.1	45.0
Rev-Dense FPN (SSD512)	VGGNet	31.2	**52.9**	32.4	**15.5**	32.9	43.9

and the representation power of its resulted features boosted by hierarchical multi-level contexts. Some visualized samples are shown in Fig. 4.

4.4 Rev-Dense FPN on Two-Stage Method

To demonstrate the flexibility of Rev-Dense FPN, we apply our approach to a state-of-the-art two-stage detector, Faster R-CNN with Feature Pyramid Network (FPN). In the original configuration of [18], the chosen backbone stages are $\mathbf{C} = \{res2\_2, res3\_3, res4\_5, res5\_2\}$ in ResNet50. Then FPN is applied on these maps, yielding P_1 to P_4. Finally, a 1×1 2-stride max-pooling is performed on P_4 to get P_5. For the experimental settings of ours, we keep the same $\mathbf{C}$ as the input of Rev-Dense FPN. k_1 is set as an increasing sequence of {240, 320, 400, 480} and k_2 is set to 80. As FPN does, we use 3×3 convs to post-process the fused maps and output 256 channels. The MS COCO dataset is adopted. We train both models for 180k iterations, using batch size of 8. The learning rate is 0.01 and it is then divided by 10 at 120k and 160k iterations.

Table 5. Results of FPN vs. Rev-Dense FPN for two-stage method. Both models are trained on the *COCO2017 train* (*trainval35k*) set and tested on the *COCO minival* set.

Method	Backbone	Input size	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN w/ FPN [18]	ResNet50	$\sim 600 \times 1000$	35.4	56.8	37.9	18.2	38.5	48.0
Faster R-CNN w/ Rev-Dense FPN	ResNet50	$\sim 600 \times 1000$	**35.9**	**57.4**	**38.6**	**19.0**	**38.6**	**48.8**

As the results shown in Table 5, with Rev-Dense FPN, the detection performance is better than FPN in all the metrics, suggesting that Rev-Dense FPN is flexible to two-stage method and can make effective improvement as well.

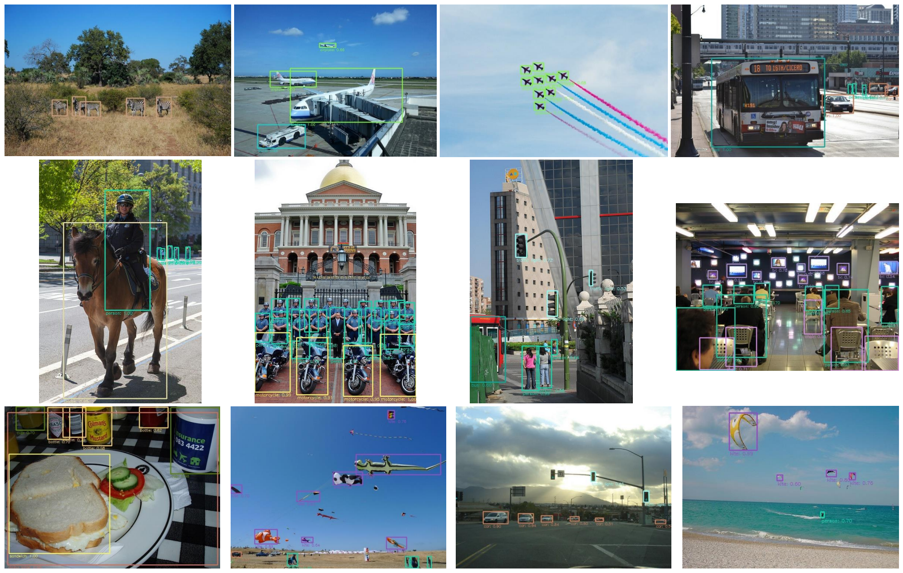

Fig. 4. Some visualized results of Rev-Dense FPN (SSD512) on the *COCO test-dev* set. Score threshold is set to 0.5 for displaying.

5 Conclusions

We develop a new multi-scale feature fusion mechanism for object detection. Our approach, denoted as the Rev-Dense FPN, performs reverse dense connection that directly fuses features from all the higher levels to the current one. Taking a set of feature maps from the base network, for each of them, we first project the current stage, then compress dimension for all the higher levels and concatenate the obtained maps onto the projected current stage after scale matching. By applying Rev-Dense FPN to SSD, we show that Rev-Dense FPN composed by reverse dense connections can achieve higher performance than approaches based on recursive top-down fusion manner. Experimental results demonstrate that reverse dense connection is more effective and Rev-Dense FPN can learn a more powerful representation which benefits from multi-level contexts and contributes to detecting multi-scale objects, especially small objects.

6 Acknowledgements

This work was supported in part by National Natural Science Foundation of China: 61672497, 61332016, 61771457, 61732007, 61620106009, 61650202 and U1636214, in part by National Basic Research Program of China (973 Program): 2015CB351802, in part by Key Research Program of Frontier Sciences of CAS: QYZDJ-SSW-SYS013, and in part by Shandong Provincial Natural Science Foundation, China: ZR2017MF001.

References

1. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.B.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 2874–2883 (2016)
2. Cai, Z., Fan, Q., Feris, R.S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV. pp. 354–370 (2016)
3. 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 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain. pp. 379–387 (2016)
4. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA. pp. 248–255 (2009). https://doi.org/10.1109/CVPRW.2009.5206848, https://doi.org/10.1109/CVPRW.2009.5206848
5. Everingham, M., Gool, L.J.V., Williams, C.K.I., Winn, J.M., Zisserman, A.: The pascal visual object classes (VOC) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010). https://doi.org/10.1007/s11263-009-0275-4, https://doi.org/10.1007/s11263-009-0275-4
6. Fu, C., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: DSSD : Deconvolutional single shot detector. CoRR **abs/1701.06659** (2017)
7. Girshick, R.B., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014. pp. 580–587 (2014)
8. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 2980–2988 (2017)
9. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015. pp. 1026–1034 (2015). https://doi.org/10.1109/ICCV.2015.123, https://doi.org/10.1109/ICCV.2015.123
10. 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)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90, https://doi.org/10.1109/CVPR.2016.90
12. Hong, S., Roh, B., Kim, K., Cheon, Y., Park, M.: Pvanet: Lightweight deep neural networks for real-time object detection. CoRR **abs/1611.08588** (2016)
13. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 2261–2269 (2017)

14. Kong, T., Sun, F., Yao, A., Liu, H., Lu, M., Chen, Y.: RON: reverse connection with objectness prior networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 5244–5252 (2017)
15. Kong, T., Yao, A., Chen, Y., Sun, F.: Hypernet: Towards accurate region proposal generation and joint object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 845–853 (2016)
16. Li, Z., Peng, C., Yu, G., Zhang, X., Deng, Y., Sun, J.: Light-head R-CNN: in defense of two-stage object detector. CoRR **abs/1711.07264** (2017)
17. Li, Z., Zhou, F.: FSSD: feature fusion single shot multibox detector. CoRR **abs/1712.00960** (2017)
18. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 936–944 (2017)
19. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 2999–3007 (2017)
20. Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V. pp. 740–755 (2014). https://doi.org/10.1007/978-3-319-10602-1_48, https://doi.org/10.1007/978-3-319-10602-1_48
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: ECCV (2016)
22. 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)
23. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 779–788 (2016)
24. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 6517–6525 (2017)
25. Ren, S., He, K., Girshick, R.B., 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)
26. Shen, Z., Liu, Z., Li, J., Jiang, Y., Chen, Y., Xue, X.: DSOD: learning deeply supervised object detectors from scratch. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 1937–1945 (2017)
27. Shrivastava, A., Sukthankar, R., Malik, J., Gupta, A.: Beyond skip connections: Top-down modulation for object detection. CoRR **abs/1612.06851** (2016)
28. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014), http://arxiv.org/abs/1409.1556
29. Zhou, P., Ni, B., Geng, C., Hu, J., Xu, Y.: Scale-transferrable object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)