Adaptive Multi-Scale Information Flow for Object Detection

Xiaoyu Chen
xychen9459@gmail.com
Wei Li
weili.cv@gmail.com
Qingbo Wu
qbwu@uestc.edu.cn
Fanman Meng
fmmeng@uestc.edu.cn

School of Information and Communication Engineering
University of Electronic Science and Technology of China

Abstract

Recent CNN-based research reveals that the multi-scale information plays an important role in boosting the performance of object detection. There are several network structures proposed to explore an effective multi-scale feature representation. In these structures, the allocation of information in multi-scale representation has a bias toward very few layers. In this paper, we present a novel module named Adaptive Multi-Scale Information Flow (ASIF) to break the bias and find the proper multi-scale representation for each layer. In ASIF, information from different layers in the feature pyramid is weighted and aggregated. The allocation of information is adaptive from each layer to other layers. To ensure both speed and accuracy at the same time, we follow the SSD detection framework and apply ASIF to it. We evaluate the performance of proposed method on PASCAL VOC and MSCOCO datasets. Experiments show that ASIF is superior to many state-of-the-art methods. Given the image size of 320×320, the mAP could reach 80.2% (45 FPS) on PASCAL VOC 2007 test, and 29.3% on COCO test-dev2015.

1 Introduction

Object detection is an essential issue in the field of computer vision research. In recent years, breakthrough progress has been made in object detectors due to the use of Convolutional Neural Networks (CNN). A main challenge of general object detection comes from the scale variation of the objects across different images. Recently introduced detectors, such as SSD [19], try to solve the problem by using different layers to predict object of different sizes. Large objects and small objects are predicted respectively from deep layers and shallow layers. The propose of this design is to keep consistency between the sizes of objects and filter receptive fields [2].

Based on this straightforward implementation, several improvements of network connections are proposed to explore an efficient multi-scale features representation. For example, FPN [17] and DSSD [5] add top-down connection (Figure 1 (a)) to the bottom-up feed-forward network. Information from the upper layers is propagated down and combined with

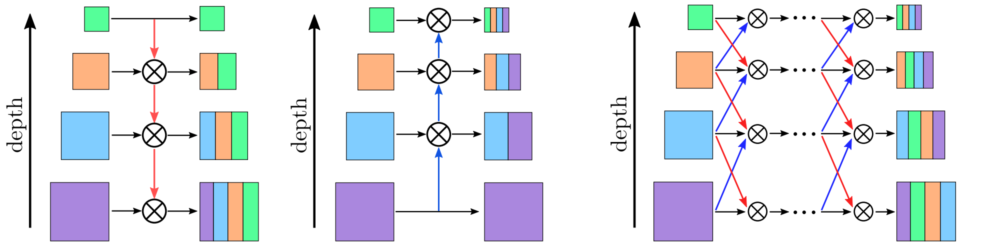


Figure 1: Structures for generating multi-scale feature representation. Symbol $\otimes$ represents feature fusion modules. In each sub-figure, blocks on the left side are input features, and blocks on the right side are output features. Different colors represent information from different features. In top-down (a) and dense connection (b), the information allocation contributes more to top or bottom layers in feature pyramid. However, in the proposed ASIF (c), information are freely transferred between layers, and each layer contains information from all layers. The allocation of information between layers is determined by the training process.

information from lower layers. In addition, DSOD [24] uses dense connection (Figure 1 (b)) to construct feature extraction and detection sub-networks. A methodology called "feature reuse" [11] is adopted. Proceeding from the bottom layer, each layer generates new information and adds it to information from preceding layers.

Though the effectiveness has been demonstrated, we find that the structures of top-down and dense connection are unreasonable. Figure 1 (a) and (b) show that there exists an obvious bias of information allocation between layers in the updated multi-scale feature representation. In the top-down connection, information from all layers is taken to the bottom layer, while the top layer contains only the information of itself. An opposite situation is contained in the dense connection, where the top layer contains much more information than the bottom layer. In fact, the detection operates independently on multiple layers. For each detection layer, it is unknown whether the information from itself is enough to obtain satisfied detection results, or the combination of information from other layers is needed. The structures of top-down and dense connection implicit pre-defined information allocation schemes. These schemes could reduce the flexibility of information selection for different detection layers. Inspired from Ke *et al.*'s work [13] on learnable scale-space representation, we consider that it is a better choice to break the fixed information allocation and make the network learn to generate proper multi-scale representation for each detection layer.

As a result, we propose a novel module called Adaptive Multi-Scale Information Flow (ASIF, shown in Figure 1 (c)) to generate more effective multi-scale feature representation for object detection. There are plenty of bi-directional connections in ASIF. Information on each layer flows to other layers through these connections. Several feature fusion modules are used to weight and aggregate information from different layers. At the output stage, each layer contains information from itself and all the other layers in the feature pyramid. To ensure both speed and accuracy at the same time, we apply our method to the proposal-free detector, such as SSD [19]. Compared with existing methods, the multi-scale representation generated using ASIF is more effective. Experiments on PASCAL VOC and MS COCO datasets show that ASIF can achieve significant improvement over state-of-the-art methods.

2 Related works

Current CNN-based object detectors can be divided into two categories: (1) the proposal-based methods and (2) the proposal-free methods. From R-CNN [7] to its multiple variants [3, 22], the proposal-based methods have made the main contribution to performance improvement in early days. The proposal-based methods first get object proposals and then classify and refine each of them. However, this complex pipeline makes them have no advantage in speed. YOLO [21] is the first detector trying to solve the problem by recasting detection as a straight regression from image to final results, but with the expense of low accuracy. YOLO can be seen as a typical proposal-free method. Recently proposed methods, such as SSD [19] and YOLOv2 [20], can achieve a good balance between speed and accuracy.

Multi-scale detection has become one of the essential technologies of high-performance detectors. Besides multi-scale training and testing [9], the multi-scale hierarchy in CNN is exploited by many detectors. There are several ways to use the multi-scale hierarchy. HyperNet [14] and ION [1] concatenate features from different layers to make prediction. SSD [19] and MS-CNN [2] predict objects at different layers in hierarchy. In addition, recent methods exploit skip layer connections to associate features maps from different layers. FPN [17] and TDM [25] create top-down path with lateral connection to transfer strong semantic information from top to bottom layers. DSOD [24] uses dense connection to fuse and reuse multi-resolution features. RSSD [12] creates rainbow concatenation between different layers so that features in each layer contain information from all the other layers.

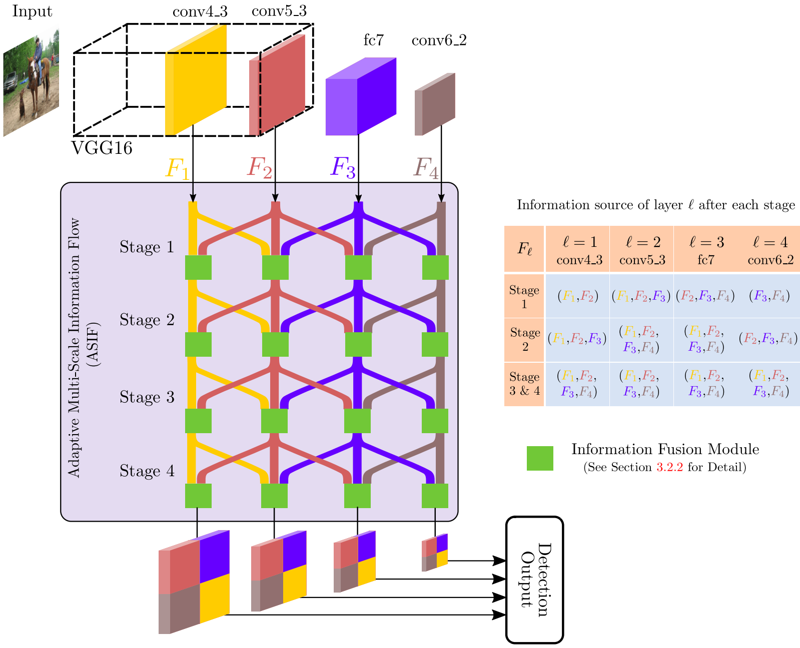
3 Method

In this section, we introduce the proposed Adaptive Multi-Scale Information Flow (ASIF) for object detection. First, we describe our detection framework in Section 3.1. Then, in Section 3.2, we show the implementation of ASIF and how ASIF adaptively allocates information of multi-scale feature pyramid to all detection layers. Finally, we give implementation details in Section 3.3.

3.1 Detection Framework

Figure 2 shows the architecture of proposed detection framework. To ensure accuracy and speed at the same time, we adopt fully convolutional proposal-free detection framework. We select VGG16 [26] as the backbone network [1] to generate feature pyramid. We replace fc6 and fc7 with convolutional layers, and add new layers (conv6_1 and conv6_2) after the VGG16. These modified and added layers decrease the size progressively and form a feature pyramid. Then, the ASIF updates features of each layer in the pyramid by using information from other layers. Finally, several detection sub-networks predict objects using the updated multi-scale feature representation.

[1] We split detection network into two parts: the feature extraction part (such as VGG16), and the detection part. For convenience of description, we call the feature extraction part as 'backbone network'.



F_ℓ	$\ell=1$ conv4_3	$\ell=2$ conv5_3	$\ell=3$ fc7	$\ell=4$ conv6_2
Stage 1	(F_1,F_2)	(F_1,F_2,F_3)	(F_2,F_3,F_4)	(F_3,F_4)
Stage 2	(F_1,F_2,F_3)	(F_1,F_2,F_3,F_4)	(F_1,F_2,F_3,F_4)	(F_2,F_3,F_4)
Stage 3 & 4	(F_1,F_2,F_3,F_4)	(F_1,F_2,F_3,F_4)	(F_1,F_2,F_3,F_4)	(F_1,F_2,F_3,F_4)

Figure 2: The proposed detection framework. The initial multi-scale feature representation comes from four different layers in the backbone network (VGG16). Different cubic colour represents information from different layers. Information from each layer in the multi-scale feature representation is updated by the module called Adaptive Multi-Scale Information Flow(ASIF). The table in figure shows the information sources of each layer after the processing at each stage.

3.2 Adaptive Multi-Scale Information Flow

As mentioned in Section 1, the commonly used top-down and dense connection exist information allocation bias between layers in the feature pyramid. The information allocation is pre-defined to concentrate more on top layers or bottom layers in feature pyramid. This bias could reduce the flexibility of information selection for each detection layer. To generate more effective multi-scale feature representation for object detection, we propose a module called Adaptive Multi-Scale Information Flow (ASIF) to break the allocation bias and adaptively transfer and aggregate information from each layer to all layers.

In the proposed detection framework, each layer in the feature pyramid is responsible for detecting objects within a specified scale range. The advantage of ASIF is that the information needed for each layer is determined by the training process. After the processing of ASIF, each detection layer could get proper information from different layers in the feature pyramid. We consider that this design can achieve better detection results than the existing methods for the detection of objects with different scales.

3.2.1 Multi-stage information aggregation strategy

Figure 2 shows the structure of ASIF. The input of the ASIF is the feature pyramid from the backbone network. The input pyramid has L layers, and the feature in each layer is denoted

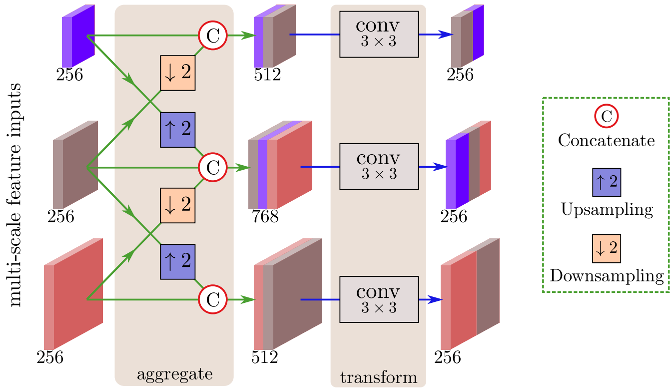


Figure 3: One stage of Adaptive Multi-Scale Information Flow. The upsampling and downsampling are implemented using 2×2 transposed convolution and 2×2 maxpooling respectively. For each layer, information from itself and neighbours is transferred and combined together.

as F_l $(l \in \{1, 2, ..., L\})$. Considering the correlation of information between adjacent layers in the pyramid, we adopt a multi-stage information aggregating strategy to transfer information from each layer to all L layers. At each stage, for layer l, information from F_l is fused with information from F_{l-1} and F_{l+1} through bi-directional connections. Before processing, each layer in the pyramid contains only the information belonging to itself. After the first stage, feature F_l at layer l contains information not only from itself but also from adjacent layers $(l-1)$ or/and $(l+1)$. Starting from the second stage, the range of information aggregation extends gradually from the adjacent layers to all layers in the feature pyramid. The table in Figure 2 shows the information source of each layer after processing by each stage. After the final stage, each layer in the feature pyramid contains weighted and fused information from all layers in the feature pyramid.

3.2.2 Information fusion modules

Figure 3 shows one stage of ASIF. Each layer in a stage contains a information fusion module. These modules are responsible for processing information transferred from different layers and transforming them to a new one. Each module contains two operations, "**aggregate**" and "**transform**". For each layer in the feature pyramid, the "aggregate" process fetches features from itself and adjacent layers, then concatenates them to produce multi-scale intermediate representation. Before concatenation, 2×2 max-pooling and 2×2 transposed convolution are used as down-sampling and up-sampling operation to make features have the same sizes. Next, "transform" process uses 3×3 convolution kernels to weight each information and generate fused output. We set the dimension of input and output features at each stage to 256.

3.3 Implementation Details

Detection Layers Considering the complex network connections in ASIF module (Section 3.2), to ensure the runtime speed, we use **conv4_3**, **conv5_3**, **fc7** and **conv6_2** as detection layers. These four layers are used to construct the input pyramid of ASIF. Because conv4_3 and conv5_3 have different feature scales from other layers, we normalize features using L_2 normalization [18]. In order to fit upsampling operation in ASIF, we resize the input to 320×320, so the feature sizes of detection layers are 40, 20, 10 and 5.

Anchor Boxes We set the scale of anchor boxes on conv4_3, conv5_3, fc7, and conv6_2 to 32, 64, 128 and 256, respectively. The choice of anchor scales is based on the method in [31] showing that controlling the anchor density of each scale to be the same is beneficial to performance. In addition, for each anchor, we set aspect ratios to $a_r \in \{\frac{1}{2}, 1, 2\}$.

Detection sub-network For each detection layer, we use a sub-network with two convolutional layers to produce detection results. Each convolutional layer has a $3 \times 3 \times c_{in} \times c_{out}$ kernel. c_{in} equals to the dimension of input features. The value of c_{out} depends on the usage of kernel. For category prediction $c_{out} = c$ and for bounding box regression $c_{out} = 4c$. c is the number of object categories to predict.

Loss Function We use the multi-task loss $L = \frac{1}{N}\left(L_{loc}(x,l,g) + L_{conf}(x,c)\right)$ for each training sample x. $L_{loc}(x,l,g)$ is the Smooth L1 localization loss [6] between ground truth box g and predicted box l. $L_{conf}(x,c)$ is the classification loss for ground truth category c. N is the number of sampled training anchors for x.

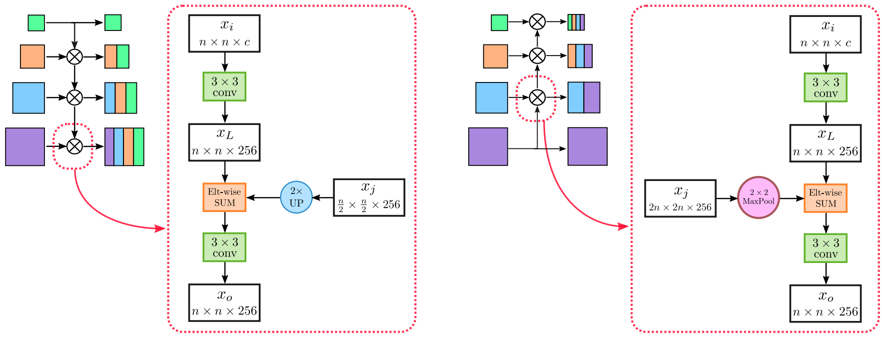
Sampling Training Examples We follow the commonly used Jaccard overlap criterion to sample positive training examples. If an anchor has Jaccard overlap higher than 0.5 with any ground truth box, this anchor is picked up as a positive example for training. In addition, most anchors are negative examples. We use hard negative mining strategy to make the training process faster and more stable. All negatives are sorted in descending order according to their loss L and only the top-n negatives are picked up. The ratio between positives and negatives is $1:3$.

4 Experiments

4.1 Datasets and Training Options

Datasets and Metrics We evaluate the performance on PASCAL VOC [4] and MSCOCO [16] datasets. For PASCAL VOC, we use VOC2007 *trainval* and VOC2012 *trainval* for training, and VOC2007 *test* for testing. For MSCOCO, we use *trainval35k* for training, *minival* for validation and *test-dev2015* for testing. Results on PASCAL VOC are evaluated using mean Average Precision (mAP) across all object categories. For MSCOCO, six different Average Precision (AP) metrics are used: (1) AP with IoU∈[0.5,0.95], (2) AP with IoU=0.5, (3) AP with IoU=0.75, (4) AP for small objects (area$< 32^2$), (5) AP for medium objects ($32^2 <$area$< 96^2$) and (6) AP for large objects (area$> 96^2$).

Optimization We use VGG16 pretrained on ImageNet [23]. All added new layers are initialized with "xavier" method [8]. The network is trained with a mini-batch size 32. For PASCAL VOC, we initialize the learning rate to 4×10^{-3}, then decay it to 4×10^{-4} and 4×10^{-5} at 80k and 100k iterations. For MSCOCO, we use the same learning rate policy as PASCAL VOC, except that the two modifications are set at 280k and 360k iterations. Besides, we set momentum to 0.9 and weight decay to 0.0005.



(a) top-down connection (b) dense connection

Figure 4: The structure of fusion modules in top-down and dense connection. They are used for experiments in Section 4.2.

4.2 Effectiveness of Adaptive Multi-Scale Information Flow

4.2.1 Experiment Settings

To verify the effectiveness of the proposed Adaptive Multi-Scale Information Flow (ASIF), we add four different multi-scale structures to detection framework for comparison: (a) top-down connection, (b) dense connection, (c) top-down+dense connection, and (d) Adaptive Multi-Scale Information Flow (ASIF). Results are obtained using PASCAL VOC dataset. We give details about the first three structures.

- **Top-down connection.** We refer the structure in FPN [ ] to construct the top-down connection. The detail of top-down module is illustrated in Figure 4 (a). The $(n \times n \times c)$ input feature x_i is transformed to $(n \times n \times 256)$ lateral feature x_L. Then, we upsample the updated upper-level $(\frac{n}{2} \times \frac{n}{2} \times 256)$ feature x_j using 2×2 transpose convolution, merge it with the lateral feature x_L by element-wise addition and further transform it with a 3×3 convolutional layer.

- **Dense connection.** We directly use the structure in DenseNet [ ]. Figure 4 (b) shows the operation of the dense connection. At first, each input feature x_i in the pyramid is transformed to lateral one x_L with 256 channels. We use 2×2 max-pooling to downsample the lower-level feature x_j and concatenate it with the lateral feature x_L. Finally, a 3×3 convolutional layer converts the concatenated feature to a new one used for detection.

- **Top-down+dense connection.** We use the combination of top-down and dense connection as an alternative method to solve the information bias mentioned in Section 1, and compare it with our proposed ASIF module. First, we use the dense connection to gather information from all layers in feature pyramid to the top layer. Then, the top-down connection spreads the information to all the other layers in feature pyramid.

4.2.2 Results and Analysis

Overall performance. Table 1 shows the comparison between different structures. Top-down connection and dense connection result in mAP of 78.3% and 77.1% , higher than plain network by 1.8% and 0.6%. These results show that top-down connection is more effective than dense connection. When combining top-down and dense connection, mAP increases to 78.4%, only 0.1% higher than pure top-down connection. Our proposed ASIF gets 79.2%

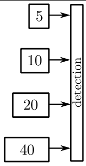
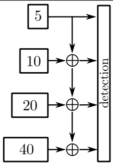
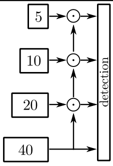
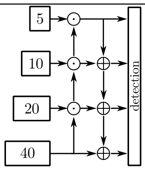
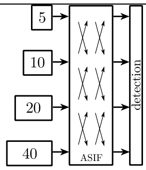
Method	plain	top-down	dense	top-down+dense	ASIF
Topology					
mAP	76.5	78.3	77.1	78.4	**79.2**
mAP_{XS}	13.7	20.6	15.7	18.3	**22.0**
mAP_{S}	49.2	57.3	52.3	55.7	**58.4**
mAP_{M}	73.5	76.6	74.9	75.6	**77.0**
mAP_{L}	80.0	81.8	81.9	82.3	**82.4**
mAP_{XL}	81.0	82.6	82.0	**83.2**	82.4

Table 1: Evaluation on different multi-scale structures. Results are measured using the whole dataset and several subsets with specified object sizes. In topology graphs, $\oplus$ and $\odot$ represent fusion modules in top-down and dense connection respectively. The structure of these modules are illustrated in Figure 4.

Figure 5: Qualitative results of different methods. The first row demonstrates the detection when objects are occluded by others. The second row shows results when an object appears in the ambiguous background (the water ripple looks like a deck).

mAP, which is the best among all the other counterparts. This result also illustrates that ASIF is more effective than the combination of top-down and dense connection when solving the information bias between different layers. Some qualitative results are shown in Figure 5.

Performance for objects with different sizes. In order to evaluate the performance of objects with different sizes, referring to the work of Hoiem *et al.* [10], we split PASCAL VOC into five subsets: XS(extra-small), S(small), M(medium), L(large), XL(extra-large). For top-down connection, detection of small objects has nearly 7%∼8% mAP boost, which reveals that top-level semantics is beneficial to performance on small objects detection [5]. On the other hand, the combination of top-down and dense connection has relatively higher mAP on detection larger objects. Our proposed ASIF achieves the best mAP across almost all scales.

Method	Backbone	Input size	N_{boxes}	Speed(FPS)	mAP
Faster R-CNN[22]	VGG16	~1000×600	300	7	73.2
Faster R-CNN[22]	ResNet-101	~1000×600	300	2.4	76.4
R-FCN[3]	ResNet-101	~1000×600	300	9	80.5
SSD300[19]	VGG16	300×300	8732	46	77.2
SSD321[5]	ResNet-101	321×321	17080	11.2	77.1
DSSD321[5]	ResNet-101	321×321	17080	9.5	78.6
RSSD300[12]	VGG16	300×300	8732	35	78.5
StairNet[27]	VGG16	300×300	19390	30	78.8
DiCSSD[28]	VGG16	300×300	8732	40.8	78.1
DSOD300(plain pred.)[24]	DenseNet	300×300	8732	20.6	77.3
DSOD300(dense pred.)[24]	DenseNet	300×300	8732	17.4	77.7
ASIF-Det320	VGG16	320×320	6375	33	**79.2**
ASIF-Det320+	VGG16	320×320	6375	**48**	**80.2**

Table 2: Results on PASCAL VOC. All networks are trained on VOC2007+VOC2012 trainval and tested on VOC2007 test.

4.3 Comparison with State-of-the-art Methods

4.3.1 Results on PASCAL VOC

Table 2 shows the experiment results. ASIF-Det320 has 79.2% mAP, higher than state-of-the-art proposal-free methods. In addition,ASIF-Det320 can meet the need of real-time detection with a 33FPS runtime speed. DSSD321 and StairNet use the top-down connection to fuse features. DiCSSD uses multi-scale dilated convolution [29] on each detection layer to combine multi-scale information. DSOD300 has 0.4% mAP increment when replacing plain detection structure with dense one. Similar to ASIF-Det320, RSSD300 also generates features by aggregating scale information from other layers. However, ASIF-Det320 has a higher mAP than RSSD300. This reveals that the multi-scale feature representation from ASIF is more effective.

In most cases, only a small amount of bounding boxes cover objects with high confidence. In order to improve the performance of proposed ASIF, we try to refine detection output. We predict objectness score p_{obj} for each anchor after the first stage of ASIF , then at the final stage we generate bounding boxes using anchors satisfying $p_{obj} > \sigma$. This technique is also exploited by several works [15, 30] for simulating two-stage procedure on the one-stage detector. We set σ to 0.005. After the refinement, ASIF-Det320+ increases the mAP to 80.2%, and the runtime speed is increased to 48FPS.

4.3.2 Results on MSCOCO

Table 3 shows the results of different networks. ASIF-Det320 achieves 46.6% AP with 0.5 IoU and 28.1% AP with IoU∈[0.5,0.95], surpassing most of comparable methods. Compared with other SSD-based methods, ASIF-Det320 has higher AP when detecting medium-size objects. However, DSOD300 and DSSD321 have better performance on large objects than ASIF-Det320. One possible reason is that DSOD300 and DSSD321 uses more powerful backbone networks. In addition, we also try to add detection refinement process to ASIF-Det320 when training on COCO dataset. The resulting ASIF-Det320+ achieves 48.8% AP with IoU=0.5 and 31.0% AP with IoU=0.75.

Method	Backbone	AP with different IoU			AP with different size		
		0.5:0.95	0.5	0.75	small	medium	large
Faster R-CNN[22]	VGG16	21.9	32.7	-	-	-	-
R-FCN[3]	ResNet-101	29.9	51.9	-	10.8	32.8	45.0
SSD300[19]	VGG16	25.1	43.1	25.8	6.6	25.9	41.4
SSD321[5]	ResNet-101	28.0	45.4	29.3	6.2	28.3	**49.3**
DSSD321[5]	ResNet-101	28.0	46.1	29.2	7.4	28.1	47.6
RSSD300[12]	VGG16	26.6	45.9	27.3	8.3	28.6	39.7
DiCSSD[28]	VGG16	26.9	46.3	27.7	8.2	27.5	43.4
DSOD300[24]	DenseNet	**29.3**	47.3	30.6	**9.4**	31.5	47.0
ASIF-Det320	VGG16	28.1	46.6	29.3	7.9	32.8	41.2
ASIF-Det320+	VGG16	**29.3**	**48.8**	**31.0**	**9.4**	**32.9**	43.6

Table 3: Results on MSCOCO test-dev2015 set.

5 Conclusion

In this paper, we propose a module called Adaptive Multi-Scale Information Flow (ASIF) to generate more effective multi-scale feature representation for object detection. Compared with other multi-scale structures, ASIF breaks the bias of information between different levels in feature pyramid and adaptively allocates information from all levels to each detection layer. The proposed method achieves state-of-the-art performance on PASCAL VOC and MSCOCO datasets. ASIF is a flexible structure. There are many ways to improve it. For example, according to the detecting difficulty of an object, we can use features from different stages to generate output. This operation could save computing resources and improve the runtime speed.

6 Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant (61601102, 61502084).

References

[1] Sean Bell, C Lawrence Zitnick, Kavita Bala, and Ross Girshick. 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*, pages 2874–2883, 2016.

[2] Zhaowei Cai, Quanfu Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision*, pages 354–370. Springer, 2016.

[3] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[5] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[6] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics (AISTATS)*, pages 249–256, 2010.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer, 2014.

[10] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *European Conference on Computer Vision*, pages 340–353. Springer, 2012.

[11] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[12] Jisoo Jeong, Hyojin Park, and Nojun Kwak. Enhancement of ssd by concatenating feature maps for object detection. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[13] Tsung-Wei Ke, Michael Maire, and Stella X Yu. Multigrid neural architectures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4067–4075, 2017.

[14] Tao Kong, Anbang Yao, Yurong Chen, and Fuchun Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 845–853, 2016.

[15] Tao Kong, Fuchun Sun, Anbang Yao, Huaping Liu, Ming Lu, and Yurong Chen. Ron: Reverse connection with objectness prior networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[17] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[18] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[20] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.

[21] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3):211–252, 2015.

[24] Zhiqiang Shen, Zhuang Liu, Jianguo Li, Yu-Gang Jiang, Yurong Chen, and Xiangyang Xue. Dsod: Learning deeply supervised object detectors from scratch. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[25] Abhinav Shrivastava, Rahul Sukthankar, Jitendra Malik, and Abhinav Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] Sanghyun Woo, Soonmin Hwang, and In So Kweon. Stairnet: Top-down semantic aggregation for accurate one shot detection. *CoRR*, abs/1709.05788, 2017.

[28] Wei Xiang, Dong-Qing Zhang, Vassilis Athitsos, and Heather Yu. Context-aware single-shot detector. *arXiv preprint arXiv:1707.08682*, 2017.

[29] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[30] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. *arXiv preprint arXiv:1711.06897*, 2017.

[31] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, and Stan Z. Li. S3fd: Single shot scale-invariant face detector. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.