Two-layer Residual Feature Fusion for Object Detection

Jaeseok Choi[1], Kyoungmin Lee[2], Jisoo Jeong[1] and Nojun Kwak[1]
[1]*Seoul National University, Seoul, Korea*
[2]*SK Holdings, Seoul, Korea*

Keywords: Object Detection, Computer Vision, Machine Learning, Neural Network, Deep Learning.

Abstract: Recently, a lot of single stage detectors using multi-scale features have been actively proposed. They are much faster than two stage detectors that use region proposal networks (RPN) without much degradation in the detection performances. However, the feature maps in the lower layers close to the input which are responsible for detecting small objects in a single stage detector have a problem of insufficient representation power because they are too shallow. There is also a structural contradiction that the feature maps not only have to deliver low-level information to next layers but also have to contain high-level abstraction for prediction. In this paper, we propose a method to enrich the representation power of feature maps using a new feature fusion method which makes use of the information from the consecutive layer. It also adopts a unified prediction module which has an enhanced generalization performance. The proposed method enables more precise prediction, which achieved higher or compatible score than other competitors such as SSD and DSSD on PASCAL VOC and MS COCO. In addition, it maintains the advantage of fast computation of a single stage detector, which requires much less computation than other detectors with similar performance.

1 INTRODUCTION

The development of deep neural networks (DNN) in recent years has achieved remarkable results not only in object detection but also in many other areas. In the early researches of object detection using DNN, much attention has been paid to representation learning that can replace hand-crafted features without much consideration on the speed of detectors. Recently, real-time detectors with low computational complexities have been actively researched.

Researches on two-stage detectors, mostly based on Faster R-CNN (Ren et al., 2015), applied the *region proposal network* (RPN) and RoI pooling to the feature maps extracted by a state-of-the-art classifier, such as ResNet-101 (He et al., 2016). On the other hand, the single-stage methods such as YOLO (Redmon et al., 2016) and SSD (Liu et al., 2016) removed RoI pooling layer and predict bounding boxes and corresponding class confidences directly while enabling faster detection and end-to-end learning.

Especially SSD makes use of multi-scale feature maps generated from a backbone network such as VGG-16 (Simonyan and Zisserman, 2014) to detect objects in various sizes. Since each of the prediction modules composed of 3 $\times$ 3 convolution filters detects bounding boxes on each layer separately, they cannot reflect appropriate contextual information from different scales. It causes the problem named as "Box-in-Box" (Jeong et al., 2017) as shown in Figure 1. In the figure, we can see that SSD often detects

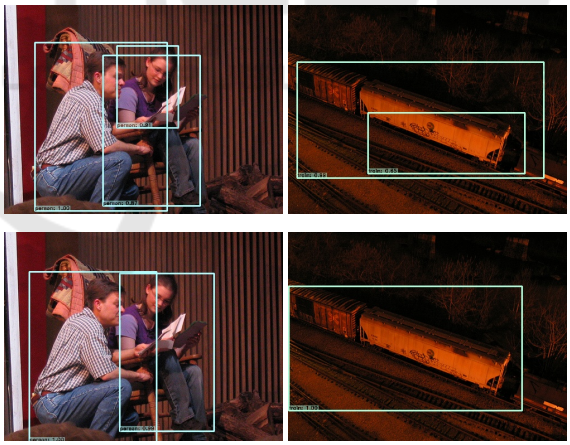

Figure 1: **Box-in-Box problem**. Top: SSD300. Bottom: RUN300 (proposed). SSD detects objects with overlapping boxes which are redundant. RUN solves this problem by utilizing the proposed two-layer feature fusion method and unified prediction module.

*J. Choi and K. Lee equally contributed to this work

Choi, J., Lee, K., Jeong, J. and Kwak, N.
Two-layer Residual Feature Fusion for Object Detection.
DOI: 10.5220/0007306803520359
In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2019)*, pages 352-359
ISBN: 978-989-758-351-3

a single object with two overlapping boxes, of which the smaller box contains partial image such as the upper body of a person or the front of a train.

To solve the problem, (Fu et al., 2017; Lin et al., 2017) used ResNet and *feature pyramid network* (FPN) (Lin et al., 2016) structure to inject larger contextual information through deep convolutional backbone by the use of deconvolution. However, these structures have the disadvantage of increased computational complexity, thus reduces detection speed, which is a key advantage of a single-stage detector.

In this paper, we propose a method to solve the essential problems of multi-scale single stage detectors. More specifically, we introduce a single-stage object detector using a feature fusion method which connects just two consecutive layers. The proposed two-layer feature fusion network has a structure where the Resblock (He et al., 2016) and the deconvolution layer are added on the multi-scale feature maps. It makes detected boxes be determined with larger context and be more reliable. In addition, we also adopt a unified prediction module (Jeong et al., 2017; Lin et al., 2017) by integrating multiple prediction modules, that had been applied separately to each layer, into one to boost the information level of feature maps in the earlier layers. The proposed network is named as RUN which is an abbreviation for two-layer Residual feature fusion with Unified prediction Network. It is not only very compact and fast compared to other ResNet-based two-stage detectors , but it also achieves superior or competitive performance compared to other competitors.

2 RELATED WORKS

Overfeat (Sermanet et al., 2013), SPPNet (He et al., 2014), R-CNN (Girshick et al., 2014), Fast R-CNN (Girshick, 2015), Faster R-CNN (Ren et al., 2015) and R-FCN (Li et al., 2016) which are classified as *region-based convolutional neural networks* (R-CNN) showed a tremendous improvement in performance compared to the previous object detection techniques. These region-based approaches have achieved huge advances over the last few years and are still the state-of-the-art approaches among many object detection techniques. Specifically, these approaches usually use a two-stage method of generating a number of bounding boxes and then assigning a classification score to the bounding boxes. Thus, although classification may be relatively accurate, these are too slow to be used for real-time applications.

Redmon (Redmon et al., 2016) proposed a method named as YOLO to predict bounding boxes and associated class probabilities in a single step by framing object detection as a regression problem. It divides input images to grid maps and regresses bounding boxes for multiple objects on each grid. This was the beginning of single stage detection and subsequently inspired structures such as SSD (Liu et al., 2016). However, since YOLO uses only the highest-level[1] feature maps for object detection, there is a lack of lower-level information, which results in somewhat inaccurate detections, especially for small objects.

In order to solve this problem, SSD (Liu et al., 2016) utilized not only the highest-level features but also lower-level features which have enough resolution to detect small objects. As mentioned in Inside-Outside Net (ION) (Bell et al., 2016) and HyperNet (Kong et al., 2016), each feature maps at different layers have different abstraction levels for an input image. Therefore, it is clear that using multi-scale feature maps can improve detection performance for objects of various scales. In SSD, many default boxes are created in the feature maps and bounding box regression and classification are performed for each box area using $3\times$ 3 convolutions. This method enables multi-scale object detection without using RoI pooling. In addition, it can effectively improve the detection accuracy of small objects which is a disadvantage of YOLO (Redmon et al., 2016).

However, as mentioned in MS-CNN (Cai et al., 2016), SSD has the problem that back-propagation allows the gradient to cause unnecessary deformations in the feature maps since the feature maps of the backbone network are used directly in bounding box regression and classification. Then, it can lead to some instability during learning. In addition, since each classifier only uses single scale feature maps, it cannot reflect larger or smaller contextual information other than the one for the corresponding scale.

Recently, various methods have attempted to enhance the contextual information of each layer while taking advantage of SSD (Liu et al., 2016). DSSD (Fu et al., 2017) could obtain higher accuracy by changing the base network to ResNet-101 (He et al., 2016) and combining the FPN (Lin et al., 2016) using deconvolution layers in combination with the existing multiple layers to reflect the large scale context. However, with the use of deep structure of ResNet-101 and deconvolution layers, the processing speed degrades much (under 16.4 images per second), which prohibits the method to be used for real-time detection problems.

Rainbow SSD (R-SSD) (Jeong et al., 2017) proposed a method to concatenate feature maps not

[1]In this paper, the term *level* is used interchangeably with *layer*. Highest level indicates the the farthest layer from the input layer.

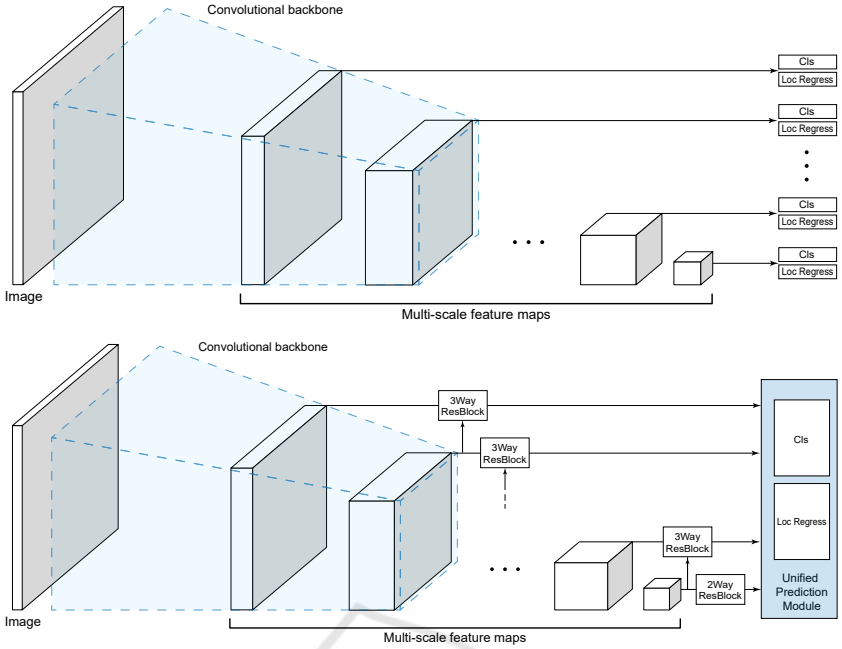

Figure 2: **Architectures of SSD and the proposed RUN**. Top: SSD. Bottom: the proposed RUN. Compared with SSD, the proposed structure has residual blocks and unified prediction module. The arrow from the bottom to the top indicates the deconvolution branch.

only in the adjacent layers but also in all the layers for bounding box regression and classification using pooling and deconvolution. It achieves higher performance than SSD by enhancing representation power of feature maps. Also, by making the dimension of each layer the same, it was made possible to use a unified prediction module instead of different prediction modules for different layers. Woo (Woo et al., 2017) proposed StairNet which utilizes both FPN (Lin et al., 2016) structure of base VGG-16 network and unified prediction of R-SSD.

3 THE PROPOSED ARCHITECTURE

3.1 Two-layer Feature Fusion

Recent CNN models designed for object detection makes use of a backbone network which is originally devised to solve image classification problems. Although the detection network can be trained end-to-end, the backbone network is normally initialized with the weights for the image classification problems. The relation between the features and predictions in the networks used for image classification can be expressed mathematically as follows:

$$\mathbf{x}_n = \mathcal{F}_n(\mathbf{x}_{n-1}) = (\mathcal{F}_n \circ \mathcal{F}_{n-1} \circ \cdots \circ \mathcal{F}_1)(\mathbf{I}) \quad (1)$$

$$\textbf{Scores} = \mathcal{P}(\mathbf{x}_n), \quad (2)$$

where $\mathbf{I}$ is an input image, $\mathbf{x}_n$ is the n^{th}-level feature map, $\mathcal{P}$ is a prediction function, and $\mathcal{F}_n$ is a combination of non-linear transformations such as convolution, pooling, ReLU, etc. Here, the top feature map, $\mathbf{x}_n$, learns information on high-level abstraction. On the other hand, $\mathbf{x}_k$ $(k < n)$ has more local and low-level information as k becomes smaller.

SSD (Liu et al., 2016) applies several feature maps with different scales directly as an input to separate prediction modules to calculate object positions and classification scores, which can be denoted by the following equation:

$$\textbf{Detection} = \left\{\mathcal{P}_1(\mathbf{x}_{s_1}), \mathcal{P}_2(\mathbf{x}_{s_2}), \ldots, \mathcal{P}_k(\mathbf{x}_{s_k})\right\}, \quad (3)$$

where s_1 to s_k are feature indices for source feature maps for multi-scale prediction, $\mathcal{P}_k$ is a function that outputs multiple objects with different positions and scores. Combining (1) and (3), it can be expressed as

$$\textbf{Detection} = \{\mathcal{P}_1(\mathbf{x}_{s_1}), \mathcal{P}_2(\mathcal{F}_{s_1}^{s_2}(\mathbf{x}_{s_1})) \ldots, \mathcal{P}_k(\mathcal{F}_{s_1}^{s_k}(\mathbf{x}_{s_1}))\}, \quad (4)$$

where $\mathcal{F}_a^b(\mathbf{x}_a) \triangleq (\mathcal{F}_b \circ \cdots \circ \mathcal{F}_{a+1})(\mathbf{x}_a)$. Here, the earlier feature map $\mathbf{x}_{s_1}$ needs to learn high-level abstraction to improve the performance of $\mathcal{P}_1(\mathbf{x}_{s_1})$. At the same time, it also needs to learn local features for efficient information transfer to the next feature maps. This not only makes learning difficult, but also causes the overall performance to decrease. It is also a problem in the above-mentioned hourglass structure.

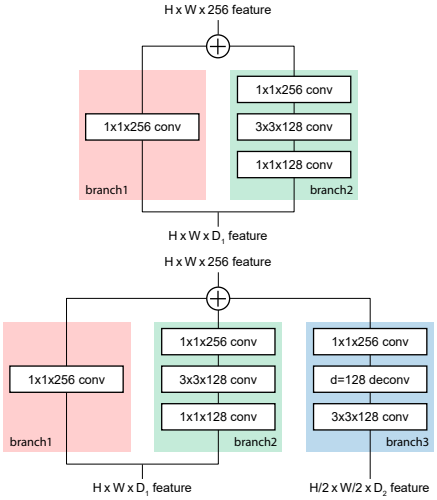

Figure 3: **Feature fusion module**. Top: 2-way Resblock. Bottom: 3-way Resblock with deconvolution branch (branch3).

To resolve this problem, SSD (Liu et al., 2016) added L2 normalization layer between the conv4_3 layer and the prediction module, which results in a reduced magnitude of the gradients from the prediction module. Cai (Cai et al., 2016) tried to solve this problem by adding a convolution layer only to the conv4_3 layer. Since the above problem is not solely on the conv4_3 layer, the aforementioned approaches do not essentially solve the problem. To meet this contradictory requirement of maintaining low-level information while having the flexibility to learn high-level abstraction, it is desired to separate and decouple the backbone network and the prediction module in the training phase.

In order to solve the same problem, we propose a new architecture that decouples backbone network from the prediction module as shown in Figure 2. Instead of directly connecting the feature maps in the backbone network to the prediction module, we inserted a multi-way Resblock[2] for each level of feature maps, which acts like a bumper that secures the information in the backbone network from the prediction modules. The detailed architecture of the proposed multi-way Resblocks are shown in Figure 3.

Convolution layers and nonlinear activation units are used for all branches of the proposed Resblock. This prevents the gradients of the prediction module from flowing directly into the feature maps of the backbone network. Also, it clearly distinguishes the features to be used for prediction from the features to be delivered to the next layer. In other words, the proposed Resblock takes the role of learning high-level abstraction for object detection, while the backbone network containing low-level features is designed to be intact from the high-level detection information. This design helps to improve the feature structure of the SSD (Liu et al., 2016) by forcing it not to learn high-level abstraction and to keep low-level image features.

Also, the depths of the earlier layers (eg. conv4_3) used for small-sized object detection in SSD (Liu et al., 2016) are very shallow. Therefore, in SSD, small objects can not be detected well because the representation power is insufficient to be used in the prediction as it is. To supplement this problem, we used a 3×3 convolution layer in branch2 of the Resblock as shown in Figure 3 to reflect the peripheral contextual information.

Branch3 in the right side of Figure 3 contains a deconvolution layer whose input is the feature maps of the consecutive layer. This two-layer feature fusion is similar to a structure proposed in (Fu et al., 2017) and (Ren et al., 2017), and it is a proper method to propagate large contextual information to a small scale feature map so that even when detecting a small object, information about its surroundings is also utilized. This can reduce the cases of detecting a part of an actual object. Thus, it can be a remedy for the box-in-box problem described earlier. The effect of this is intuitively shown in the right side of Figure 1. Finally, the proposed two-layer feature fusion method in Figure 2 can be expressed as

$$\textbf{Detection} = \left\{\mathcal{P}_1(\hat{\mathbf{x}}_{s_1,s_2}), \mathcal{P}_2(\hat{\mathbf{x}}_{s_2,s_3}), \ldots, \mathcal{P}_k(\hat{\mathbf{x}}_{s_k})\right\}, \tag{5}$$

where $\hat{\mathbf{x}}_{a,b} = \mathcal{B}_1(\mathbf{x}_a) + \mathcal{B}_2(\mathbf{x}_a) + \mathcal{B}_3(\mathbf{x}_b)$ and $\hat{\mathbf{x}}_a = \mathcal{B}_1(\mathbf{x}_a) + \mathcal{B}_2(\mathbf{x}_a)$. Here, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ indicate branch1, branch2 and branch3, respectively.

3.2 Unified Prediction Module

Detecting objects of various sizes has been recognized as an important problem in object detection. Traditionally, (Viola and Jones, 2004; Dollár et al., 2014; Dalal and Triggs, 2005) used a single classifier to predict multi-scale feature maps extracted from the image pyramid. There is another approach of using multiple classifiers on a single input image. The latter has the advantage of reducing the amount of computation for calculating feature maps. However, it requires an individual classifier for each object scale.

Since the neural network has been prevalent, the two-stage detectors applied RoI Pooling to the CNN output to extract feature maps of the same size from

[2]Here, we term the basic components of our feature fusion architecture as *Resblocks* which are slightly different from those in ResNets (He et al., 2016) in that instead of identity mapping, 1×1 convolution is used as shown in the red boxes in Figure 3.

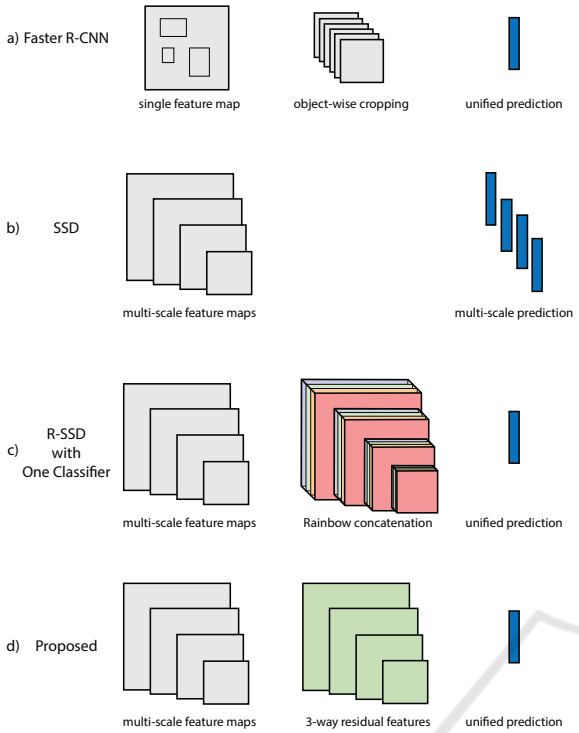

Figure 4: **Comparison of various object detection schemes**: a) R-CNN and its variants need object-wise cropping and the prediction is done by a common unified classifier. b) SSD does not need any cropping but requires a separate classifier for each scale of feature maps. c) R-SSD concatenates feature maps in different layers so that objects in each scale can be predicted with one unified classifier with the same amount of information. d) In the proposed method, Resblock takes the role of feature map concatenation and one unified classifier is used for prediction.

objects of different sizes. These feature maps were used as the input of a single classifier. Meanwhile, other methods using multi-scale features, such as SSD (Liu et al., 2016), adopted multiple classifiers since feature maps in each scale differed not only in length but also in the underlying contextual information. In order to effectively learn the prediction layers of various scales, it is necessary to input objects of various scales. SSD could dramatically increase the detection performance through augmentation which transforms the size of input images.

R-SSD (Jeong et al., 2017) proposed the Rainbow concatenation which combines feature maps in different scales using pooling and deconvolution. This allows to set the depth of the input features for each prediction module to be the same. Thus, R-SSD could use a single classifier that shares the weight of multi-scale prediction modules. Similarly, the proposed two-layer feature fusion through 3-way Resblocks enforces all the feature maps to have the same depth of 256 as shown in Figure 3. Thus, structurally, it is possible to unify convolution layers of different prediction modules like R-SSD. The idea of the unified prediction module is similar to (Jeong et al., 2017), but our method is different from R-SSD in information contained in the input feature maps.

This approach makes differently-scaled feature maps have similar level of information. SSD (Liu et al., 2016) used multiple features of various scales. This results in an improved performance of detecting small objects compared with YOLO (Redmon et al., 2016; Redmon and Farhadi, 2016) which used only the last layer of the back-bone network. However, since its earliest feature map is obtained from much shallower layers than the later feature maps, it still has a limitation of insufficient information for prediction. Because unified prediction applies equally to feature maps of all scales, it forces the output of the 3-way Resblock between the feature map of the backbone and the prediction module to be learned at a similar information level. It means that unified prediction in combination with the residual feature block makes the feature maps in the earliest Resblocks rich in context.

In the R-SSD (Jeong et al., 2017) structure, detection performance for small objects was increased using the unified prediction module, but the overall performance decreased. However, as shown in Table 1, the proposed RUN structure shows a meaningful increase in overall performance due to the unified prediction. This shows that our two-layer feature fusion method using 3-way Resblock is more suitable for unified prediction than the feature fusion method used in R-SSD. The work in (Lin et al., 2017) also mentions unified prediction, but it does not include any details and experiments on it. A brief summary of different object detection schemes is shown in Figure 4.

4 EXPERIMENT

We experimented the proposed method on PASCAL VOC 2007 (Everingham et al., 2010), PASCAL VOC 2012 and MS COCO datasets (Lin et al., 2014). Our implementation is based on the publicly available SSD (Liu et al., 2016)[3]. All the experimental results of SSD are the latest scores with data augmentation mentioned in (Fu et al., 2017). For all the experiments, the reduced VGG-16 model (Simonyan and Zisserman, 2014) pre-trained on the ILSVRC CLS-LOC dataset (Russakovsky et al., 2014) is used as the backbone network. For fair comparison, most of the settings are set to be the same as those of SSD except

[3]https://github.com/weiliu89/caffe/tree/ssd

Table 1: **PASCAL 2007** `test` detection results.

Method	mAP
SSD 300	77.5
SSD 300 + 2WAY	78.3
SSD 300 + 2WAY + Unified Pred	78.6
SSD 300 + 3WAY	78.8
SSD 300 + 3WAY + Unified Pred	**79.2**

the number of proposals. It is different from SSD, because we used 6 default boxes in all the prediction layers for unified prediction while SSD used 4 for the conv4_3 and the top layer, and 6 for the rest.

4.1 PASCAL VOC2007

We trained our model on VOC2007 `trainval` and VOC2012 `trainval`. We set the batch size as 32. For the training of the 2-way model, we used a learning rate of 10^{-3} initially, then it is decreased by a factor of 10 at 80k and 100k iterations, respectively. The training was terminated at 120k iterations. For the 3-way model, we froze all the weights of the pre-trained 2-way model except the prediction module, then fine-tuned the network using the learning rate of 10^{-3} for 40k iterations, 10^{-4} for the next 20k iterations, and 10^{-5} for the final 10k iterations. The end-to-end training was also applied on the 3-way model, but the results were worse than the above training strategy.

Table 1 shows our result on PASCAL VOC 2007 `test` set. Here, *Unified Pred* indicates the application of the unified prediction module and the prediction modules for the ones without this indication were trained separately as in the original SSD. As mentioned above, each 3-way model was fine-tuned on the corresponding 2-way model. In this experiment, we observed that the proposed model with only 2-way Resblock without the deconvolution path achieved 1.1% higher mAP than that of SSD. The 3-way model which further utilizes deconvolution layers was up to 0.6% higher than the 2-way model. The unified prediction module made better advance in the 3-way model than the 2-way model, which scored 79.2% and 78.4% respectively.

4.2 MS COCO

For fair comparison with SSD (Liu et al., 2016), most of the hyper-parameters required for training were set to the same as SSD. For training 2-way models, we used a learning rate of 10^{-3} for the first 240k iterations, 10^{-4} for the next 120k iterations and 10^{-5} for the last 40k. For training 3-way models, we used a learning rate of 10^{-3} for the first 120k iterations, 10^{-4} for the next 60k iterations and 10^{-5} for the last 20k, which are exactly half of those for the 2-way models. Other parameters such as scales and aspect ratios of the prior box were identical to those of SSD.

Table 2 shows the performance of various methods on MS COCO `test-dev`. Despite the proposed methods use a relatively shallow network, VGG-16 (Simonyan and Zisserman, 2014), they achieved enough performance to compare with other methods which use a very deep network. The fourth column indicates that RUN3WAY300 achieved 2.9% better mAP compared to SSD300 (Liu et al., 2016). It was the same performance with SSD321 and DSSD321 (Fu et al., 2017), which adopted ResNet-101 (He et al., 2016) as their back-bone network. Also, RUN3WAY512 achieved 3.6% better mAP than SSD512. In particular, RUN3WAY512 achieved the highest average precision and recall for small objects among compared methods except RetinaNet. It means that the proposed Resblock is a quite effective module to enhance low-level feature maps.

5 DISCUSSION

5.1 Speed vs. Accuracy

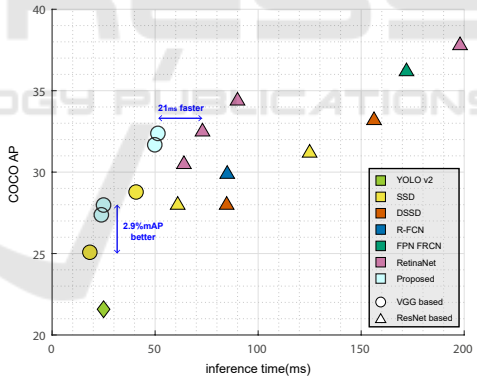

Figure 5: **Speed vs. Accuracy of recent methods using public numbers on COCO**. Our results (sky blue circles) are measured on Titan X. (Best viewed in color).

The single stage detectors, which are represented by YOLO (Redmon et al., 2016) and SSD (Liu et al., 2016), proposed end-to-end neural networks that removed the RoI Pooling of two-stage detectors. They have achieved a lot of speed improvements, but they could not avoid the loss of accuracy. Conversely, recent single stage detectors have been studied to improve performance, while suffering the loss of speed. Unlike other approaches, the proposed RUN is designed to maximize performance at high speeds on the VGG-16 (Simonyan and Zisserman, 2014) back-

Table 2: **MS COCO `test-dev` detection results.**

Method	data	network	Avg. Precision, IoU:			Avg. Precision, Area:			Avg. Recall, #Dets:			Avg. Recall, Area:		
			0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Faster (Ren et al., 2015)	trainval	VGG	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION (Bell et al., 2016)	train	VGG	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
R-FCN (Li et al., 2016)	trainval	ResNet-101	29.9	51.9	-	10.8	32.8	45.0	-	-	-	-	-	-
RetinaNet (Lin et al., 2017)	trainval	ResNet-101	**39.1**	**59.1**	**42.3**	**21.8**	**42.7**	**50.2**	-	-	-	-	-	-
SSD300 (Liu et al., 2016)	trainval35k	VGG	25.1	43.1	25.8	6.6	25.9	41.4	23.7	35.1	37.2	11.2	40.4	58.4
SSD321 (Fu et al., 2017)	trainval35k	ResNet-101	28.0	45.4	29.3	6.2	28.3	49.3	25.9	37.8	39.9	11.5	43.3	64.9
DSSD321 (Fu et al., 2017)	trainval35k	ResNet-101	28.0	46.1	29.2	7.4	28.1	47.6	25.5	37.1	39.4	12.7	42.0	62.6
RUN2WAY300	trainval35k	VGG	27.4	46.1	28.4	8.9	27.9	43.8	25.0	37.3	39.5	14.6	42.6	59.8
RUN3WAY300	trainval35k	VGG	28.0	47.5	28.9	9.9	28.6	43.9	25.3	38.0	40.5	16.2	43.8	60.2
SSD512 (Liu et al., 2016)	trainval35k	VGG	28.8	48.5	30.3	10.9	31.8	43.5	26.1	39.5	42.0	16.5	46.6	60.8
SSD513 (Fu et al., 2017)	trainval35k	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8	28.3	42.1	44.4	17.6	**49.2**	65.8
DSSD513 (Fu et al., 2017)	trainval35k	ResNet-101	33.2	53.3	35.2	13.0	35.4	51.1	**28.9**	**43.5**	**46.2**	21.8	49.1	**66.4**
RUN2WAY512	trainval35k	VGG	31.7	52.1	33.6	13.2	33.9	46.5	27.7	42.2	44.7	22.0	47.9	62.7
RUN3WAY512	trainval35k	VGG	32.4	53.5	34.2	14.7	34.0	46.7	28.0	43.0	45.8	**24.4**	48.1	63.4

Table 3: **Speed & Accuracy on PASCAL VOC2007 `test`.** * is measured by ourselves.

Method	network	mAP	FPS	GPU
Faster R-CNN (Ren et al., 2015)	VGG16	73.2	7	Titan X
Faster R-CNN (He et al., 2016)	ResNet-101	76.4	2.4	K40
R-FCN (Li et al., 2016)	ResNet-101	80.5	9	Titan X
SSD300 (Liu et al., 2016)	VGG16	77.5	54.5*	Titan X
SSD321 (Fu et al., 2017)	ResNet-101	77.1	16.4	Titan X
DSSD321 (Fu et al., 2017)	ResNet-101	78.6	11.8	Titan X
R-SSD300 (Jeong et al., 2017)	VGG16	78.5	37.1*	Titan X
StairNet (Woo et al., 2017)	VGG16	78.8	30	Titan X Pascal
RUN2WAY300	VGG16	78.6	41.8	Titan X
			58.4	Titan X Pascal
RUN3WAY300	VGG16	79.2	40.0	Titan X
			56.3	Titan X Pascal
SSD512 (Liu et al., 2016)	VGG16	79.5	24.5*	Titan X
SSD513 (Fu et al., 2017)	ResNet-101	80.6	8.0	Titan X
DSSD513 (Fu et al., 2017)	ResNet-101	81.5	6.4	Titan X
R-SSD512 (Jeong et al., 2017)	VGG16	80.8	15.8*	Titan X
RUN2WAY512	VGG16	80.6	20.1	Titan X
			31.8	Titan X Pascal
RUN3WAY512	VGG16	80.9	19.5	Titan X
			29.8	Titan X Pascal

bone, which has significantly fewer layers and parameters than ResNet (He et al., 2016). The experimented results demonstrate the performance improvement of RUN.

Table 3 shows that our method outperforms other competitors with less loss of speed. Our experiments were tested using Titan X GPU, cuDNN v5.1 and Intel I7-6700@3.4GHz. For exact comparison, we measured FPS of some methods on the same environment and marked * in the table.

In Figure 5, we show the trade-off relation between the detection accuracy and inference time by plotting the results of RUN and other methods on COCO `test-dev`. The RUN3WAY300 model (25.0ms, 28.0% mAP) is 36% slower but 2.9% better in mAP than the SSD300 (Liu et al., 2016) model (18.3ms, 25.1% mAP). It is about 60% faster than ResNet-101 based SSD321 (Fu et al., 2017) model (61ms, 28.0% mAP) that has a similar performance. Likewise, the RUN3WAY512 (51.4ms, 32.4% mAP) is 26% slower but 3.6% better in mAP than the SSD512 model (40.8ms, 28.8% mAP). It is about 44% faster than RetinaNet-50-500 (Lin et al., 2017) (73ms, 32.5% mAP) that has a similar performance.

In addition, we measured FPS of our methods on Titan X Pascal with the other environment kept the same. Table 3 shows that even the most complex version of our method, RUN3WAY512, can works in real time (29.8 FPS) on Tital X Pascal.

6 CONCLUSION

The proposed RUN architecture for object detection was originated from the awareness of the contradictory requirements for multi-scale features that they should contain low-level information on an image as well as high-level information on objectness. The proposed two-layer feature fusion method using the 3-way Resblock alleviated the gradient exploitation problem and enriched contextual information, an important element for a good prediction performance. We also showed that the generalization performance of multi-scale prediction in our architecture can be improved by adopting the unified prediction module which integrates the separate prediction modules into a common prediction module. This approach, which can be seen to be somewhat simple, resulted in outstanding performance on the PASCAL VOC test. The results on COCO dataset also show how fast and efficient our algorithms are. We expect the proposed method be not restricted to SSD-based methods but also applicable to other structures utilizing multi-scale features.

ACKNOWLEDGEMENTS

The research was supported by the Green Car development project through the Korean MTIE (10063267)

and ICT R&D program of MSIP/IITP (2017-0-00306).

REFERENCES

Bell, S., Zitnick, C. L., Bala, K., and Girshick, R. (2016). Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2874–2883. IEEE.

Cai, Z., Fan, Q., Feris, R., and Vasconcelos, N. (2016). A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

Dollár, P., Appel, R., Belongie, S., and Perona, P. (2014). Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Fu, C.-Y., Liu, W., Ranga, A., Tyagi, A., and Berg, A. C. (2017). Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). 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.

He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Jeong, J., Park, H., and Kwak, N. (2017). Enhancement of ssd by concatenating feature maps for object detection. *arXiv preprint arXiv:1705.09587*.

Kong, T., Yao, A., Chen, Y., and Sun, F. (2016). Hypernet: Towards accurate region proposal generation and joint object detection. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 845–853. IEEE.

Li, Y., He, K., Sun, J., et al. (2016). R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, pages 379–387.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2016). Feature pyramid networks for object detection. *arXiv preprint arXiv:1612.03144*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788.

Redmon, J. and Farhadi, A. (2016). Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*.

Ren, J., Chen, X., Liu, J., Sun, W., Pang, J., Yan, Q., Tai, Y.-W., and Xu, L. (2017). Accurate single stage detector using recurrent rolling convolution. In *CVPR*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2014). Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

Woo, S., Hwang, S., and Kweon, I. S. (2017). Stairnet: Top-down semantic aggregation for accurate one shot detection. *arXiv preprint arXiv:1709.05788*.