Journal of Southeast University (English Edition) Vol. 36, No. 3, pp. 252 - 263 Sept. 2020 ISSN 1003—7985

Neighborhood fusion-based hierarchical parallel feature pyramid network for object detection

Mo Lingfei Hu Shuming

(School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China)

Abstract: In order to improve the detection accuracy of small objects, a neighborhood fusion-based hierarchical parallel feature pyramid network (NFPN) is proposed. Unlike the layer-by-layer structure adopted in the feature pyramid network (FPN) and deconvolutional single shot detector (DSSD), where the bottom layer of the feature pyramid network relies on the top layer, NFPN builds the feature pyramid network with no connections between the upper and lower layers. That is, it only fuses shallow features on similar scales. NFPN is highly portable and can be embedded in many models to further boost performance. Extensive experiments on PASCAL VOC 2007, 2012, and COCO datasets demonstrate that the NFPN-based SSD without intricate tricks can exceed the DSSD model in terms of detection accuracy and inference speed, especially for small objects, e.g., 4% to 5% higher mAP (mean average precision) than SSD, and 2% to 3% higher mAP than DSSD. On VOC 2007 test set, the NFPN-based SSD with 300 × 300 input reaches 79.4% mAP at 34.6 frame/s, and the mAP can raise to 82.9% after using the multi-scale testing strategy.

Key words: computer vision; deep convolutional neural network; object detection; hierarchical parallel; feature pyramid network; multi-scale feature fusion

DOI: 10.3969/j.issn.1003 - 7985.2020.03.002

Considering the problem of small object detection, a neighborhood fusion-based hierarchical parallel feature pyramid network is proposed. This network extracts features with rich context information and local details by fusing the shallow features of similar scales, that is, the constructed feature pyramid network has no connection between the upper and lower layers. This network is integrated into the SSD framework to achieve object detection.

Object detection is an important computer vision task that uses image processing algorithms and models to detect instance objects of a specific class in digital images, providing fundamental information for further image understanding. Objects can appear anywhere in the image in a variety of shapes and sizes. Besides, detection is also affected by perspective, illumination conditions, and occlusion, making it a Gordian knot. Early object detectors are closely related to hand-engineered features that are applied to dense image meshes to locate objects in the sliding-window paradigm, such as the Viola-Jones (VJ) face detector[1-2], the histogram of oriented gradient (HOG) pedestrian detector[3], and deformable part models (DPMs)[4]. Recently, with the emergence of convolutional neural networks (CNN) and the rapid development of deep learning, object detection has also made great strides forward. Researchers have proposed many excellent object detectors, which can be roughly classified into the one-stage models and the two-stage models.

The two-stage model divides the detection tasks into two phases and has become the dominant paradigm of object detection. It generates a sparse set of candidate regions (ROI) first (via selective search[5] or region proposal network[6]), and then classifies these ROIs into a particular category and refines their bounding boxes. R-CNN[7] and SPPNet (spatial pyramid pooling convolutional network)[8] are classic works that implement this idea. After years of research, the superior performance of the two-stage detectors on several challenging datasets (e.g., PASCAL VOC[9] and COCO[10]) has been demonstrated by many methods[6, 11-16].

In contrast, the one-stage model can directly predict the category and location of objects simultaneously, thus abandoning the region proposal. Its straightforward structure grants it a one-stage model higher detection efficiency with a slight performance degradation exchange. SSD[17] and YOLO[18] achieve ultra-real-time detection with a tolerable performance, renewing interest in one-stage methods. Many extensions of these two models have been proposed[19-25].

Current state-of-the-art object detectors have achieved excellent performance on several challenging benchmarks. However, there are still many conundrums in the field of object detection: 1) Feature fusion and reusing. Features rich in high-level semantic information is beneficial for object detection. Lin et al.[15, 19-20, 26] obtained more characterized features through multiple feature fusion. 2) The trade-off between performance and speed. For the sake of practical applications, a delicate balance

Received 2020-01-13, **Revised** 2020-08-05.

Biography: Mo Lingfei (1981—), male, doctor, associate professor, lfmo@seu.edu.cn.

Foundation item: The National Natural Science Foundation of China (No. 61603091).

Citation: Mo Lingfei, Hu Shuming. Neighborhood fusion-based hierarchical parallel feature pyramid network for object detection[J]. Journal of Southeast University (English Edition), 2020, 36(3): 252 - 263. DOI: 10.3969/j.issn.1003 - 7985.2020.03.002.

must be struck between performance and detection speed.

This paper rethinks the feature fusion and reusing while taking into account the inference speed. As an efficient feature extraction method, FPN[15] has been adopted by many state-of-the-art detectors[14, 19, 21, 27]. Heuristically, this paper reconstructs the feature pyramid network in a hierarchical parallel manner (neighborhood fusion-based hierarchical parallel feature pyramid network), and then it is integrated into the SSD framework to verify its effectiveness. The main contributions are summarized as follows: 1) Proposing a simple and efficient method for constructing the context-rich feature pyramid network; 2) Integrating the hierarchical parallel feature pyramid network into the SSD framework and showing its performance improvement on standard object detection benchmarks compared with some FPN based models.

1 Feature Pyramid Network

Many experiments have confirmed that it is profitable to use multi-scale features for detection. For example, SSD[17] uses the multiple spatial resolution features of VGG nets[28]. These multi-scale features taken directly from the backbone network can be consolidated into a primary feature pyramid network, in which the top layer has rich semantic information but a lower resolution, while the bottom layer has less semantic information but a higher resolution. Fig. 1 summarizes some typical feature pyramid networks. In this figure, the circle outline represents the feature map, its larger size represents higher resolution, and its darker color represents stronger semantic information. FPN[15] performs a top-down layer-by-layer fusion of the primary feature pyramid network with additional lateral connections for building high-level semantic features on all scales. On the basis of FPN, PANet (path aggregation network)[29] adds a bottom-up route to enhance its network structure, which is conducive to shortening the information propagation path while using low-level features to locate objects. NAS-FPN[30] can be regarded as the pioneering work of NAS application in object detection. It automatically learns the structure of the feature pyramid network by designing an appropriate search space, but it requires thousands of GPU hours during searching, and the resulting network is irregular and difficult to interpret or modify.

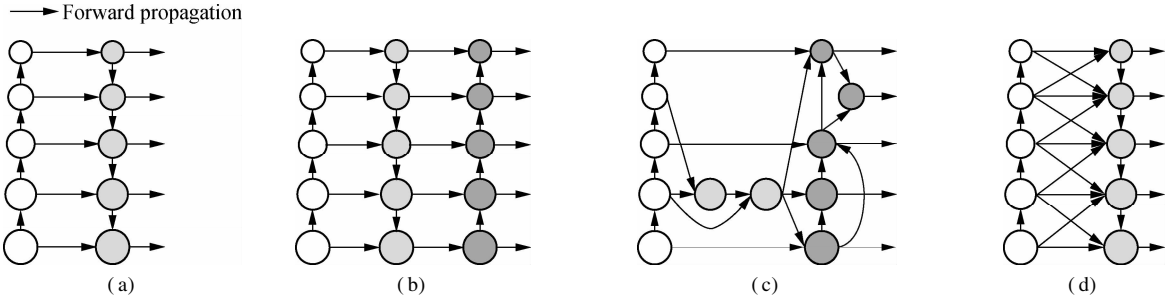


Fig. 1 Feature pyramid network. (a) FPN; (b) PANet; (c) NAS-FPN; (d) NFPN

Fig. 2 shows several similar methods for constructing the feature pyramid network based on the SSD framework. In this figure, the rectangular outline represents the feature map, its width represents the number of channels, and its height represents the resolution (i.e., $512 \times 38 \times 38$). For brevity, some similar connections are omitted. DSSD[19] is inherited from SSD and FPN, building a more representative feature pyramid network by layer-by-layer fusing. In this case, low-resolution features are constantly up-sampled to mix with high-resolution features. RSSD (rainbow single shot detector)[20] constructs a feature pyramid network in a fully connected manner. That is, the feature map of each resolution in the feature pyramid network is obtained by fusing all inputs. In the structure of Fig. 1(b), Fig. 2(b) and Fig. 2 (c), the input features inevitably undergo multiple consecutive resampling for constructing the feature pyramid network, which also causes some additional sampling noises and information loss.

In order to alleviate this potential contradiction, and take into account the training and inference speed, this paper manually limits the resampling times of the input features, ensuring that each feature map will undergo up-sampling and down-sampling at most once. The model structure is shown in Fig. 1(d) and Fig. 2(d), which only fuses shallow features on similar scales, abandoning multiple consecutive resampling used in DSSD and RSSD. Subsequent experimental results demonstrate the effectiveness of this approach. On the VOC 2007 test set, the NFPN-based SSD with 300×300 input size achieves 79.4% mAP at 34.6 frame/s, DSSD[19] with 321×321 input achieves 78.6 mAP at 9.5 frame/s, and RSSD[20] with 300×300 input achieves 78.5% mAP at 35.0 frame/s. After using the multi-scale testing strategy, the NFPN-based SSD can achieve 82.9% mAP.

2 Network Architecture

The neighborhood fusion-based hierarchical parallel feature pyramid network (NFPN) is shown in Fig. 1(d) and Fig. 2(d), introducing a neighborhood fusion-based hierarchical parallel architecture for constructing the context-rich feature pyramid network. The basic unit of NFPN

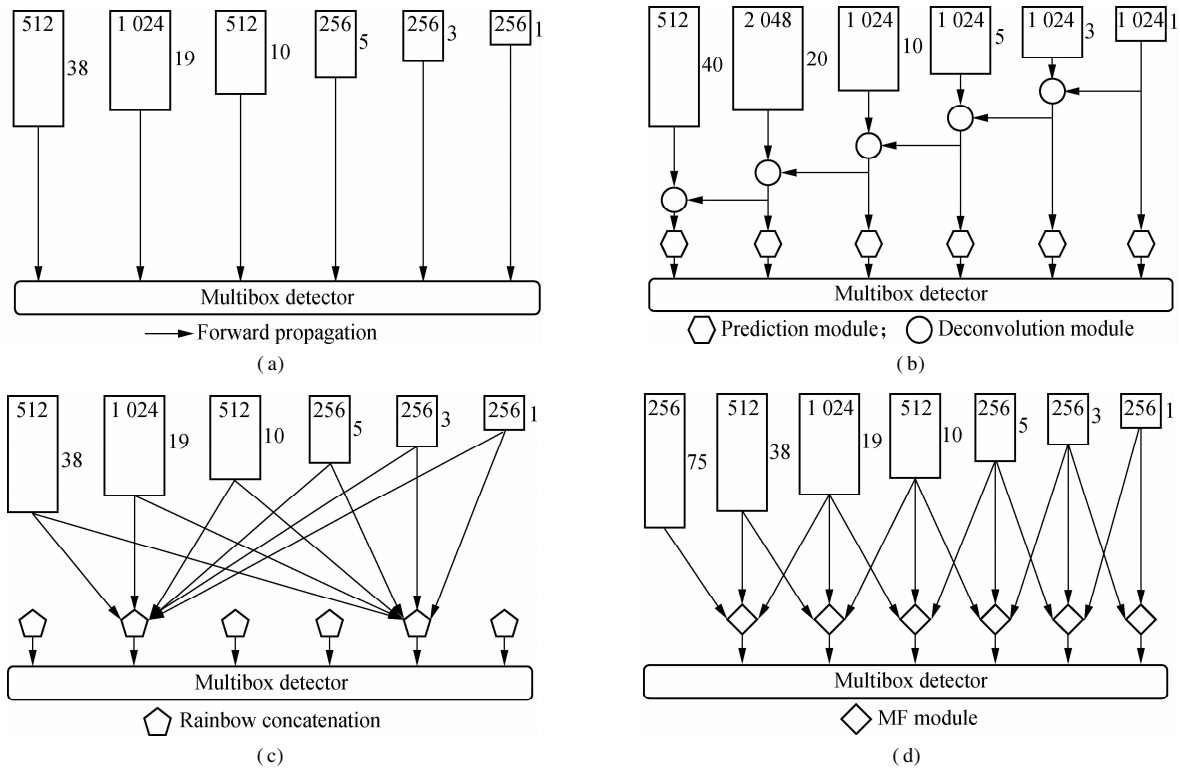


Fig. 2 Methods for constructing the feature pyramid network based on SSD framework. (a) SSD; (b) DSSD; (c) RSSD; (d) NFPN-based SSD

is the multi-scale feature fusion module (MF module), which is used to aggregate multi-scale features. This section will introduce the MF module first, and then discuss how to integrate MF modules to construct NFPN and embed it into the SSD[17] framework.

2.1 MF module

In order to aggregate multi-scale features obtained from the backbone network, this paper introduces a simple multi-scale feature fusion module called the MF module, as shown in Fig. 3. The MF module takes three different scale features as inputs (i. e., $4\times$, $2\times$, $1\times$) and resamples these features to the same resolution ($2\times$) through the down-sampling branch ($4\times \rightarrow 2\times$) and the up-sampling branch ($1\times \rightarrow 2\times$). The subsequent 3×3 convolution layer is designed to refine features and limit noise caused by resampling. Instead of the element-wise operation used in FPN[15] and DSSD[19], the MF module adopts concatenation to combine multiple features, where the status of different features is characterized by the number of channels. The output channel of the basic branch ($2\times$ resolution) is set to be 256 to preserve more of its features, while the output channel of the down-sampling and up-sampling branches is set to be 128 to complement the context information. By convention, each convolutional layer is followed by a batch normalization (BN) layer and a ReLU(rectified linear unit) activation layer.

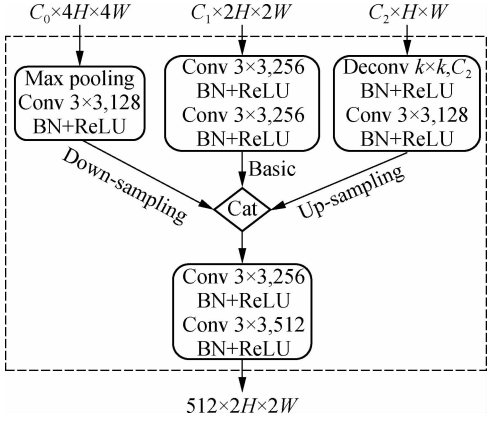


Fig. 3 Multi-scale feature fusion module

2.2 Stacking of MF module

Referring to the structure shown in Fig. 2(d), NFPN can be implemented by stacking MF modules in parallel among layers of the primary feature pyramid network. SSD[17] performs detection tasks using six scale features (denoted as C_4, C_5, …, C_9, respectively) of the VGG-16[28]. Fu et al.[19] proposed the deconvolution module and prediction module to build a context-rich feature pyramid network in a layer-by-layer fusion manner, while Jeong et al.[20] proposed the rainbow concatenation similar to a fully connected network. These structures can be briefly formulated as

$$\left.\begin{aligned} C_i^* &= \text{deconv}(C_i, C_{i+1}^*) \\ P_i &= \text{predict}(C_i^*) \end{aligned}\right\} \tag{1}$$

$$P_i = \text{rainbow}(C_4, C_5, \cdots, C_9) \tag{2}$$

where deconv and predict are the deconvolution module and prediction module in DSSD[19], respectively; rainbow is the rainbow concatenation in RSSD[20]. Finally, $\{P_i\}$ combines into a context-rich pyramid network for multi-scale detection. Considering DSSD and RSSD, the input features of the feature pyramid network inevitably undergo multiple consecutive resampling, which can aggregate more contextual information, but introduce additional resampling noise.

To mitigate this contradiction, this paper manually limits the times of resampling that each input feature needs to perform, ensuring that each input can be up-sampled and down-sampled at most, once. This method is a hierarchical parallel stacking of the MF module and can be formulated as

$$P_i = \mathrm{MF}(C_{i-1}, C_i, C_{i+1}) \tag{3}$$

where MF represents the MF module shown in Fig. 3. Intuitively, NFPN reduces resampling times required for each input feature, and only fuses shallow features with similar resolution. Furthermore, the hierarchical parallel structure does not increase the depth of the computational graph as much as the layer-by-layer structure and is more in line with parallel computing.

This paper is based on the SSD framework to verify the performance of NFPN. In detail, the NFPN-based SSD additionally introduces "conv3_3" (C_3 in Fig. 4) as the down-sampling branch of P_4 and deletes the up-sampling branch of P_9 while increasing the number of channels of its down-sampling branch from 128 to 256. In summary, the NFPN-based SSD contains six parallel stacked MF modules, as shown in Fig. 4.

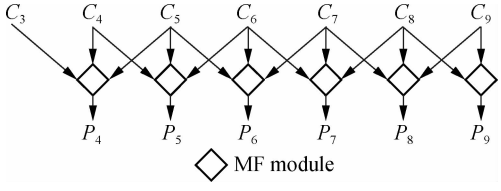


Fig. 4 Stacking of MF module

3 Experiments

Experiments were conducted on three widely used object detection datasets: PASCAL VOC 2007, 2012[9], and MS COCO[10], which have 20, 20, and 80 categories, respectively. The results on the VOC 2012 test set and COCO test-dev set were obtained from the evaluation server. The experimental code is based on Caffe[31].

3.1 Training strategies

For the sake of fairness, almost all the training policies are consistent with SSD[17], including the ground-truth box matching strategy, training objective, anchor set, hard negative mining, and data augmentation. The model loss is a weighted sum between localization loss (Smooth L1) and classification loss (Softmax). NFPN takes seven scale features selected directly from the backbone network as inputs, whose resolutions are 75^2, 38^2, 19^2, 10^2, 5^2, 3^2, and 1^2, respectively. The convolution and deconvolution layers in NFPN do not use bias parameters, and their weights are initialized by a Gaussian function with a mean value of 0 and a standard deviation of 0.01. For the BatchNorm layer, the moving average fraction is set to be 0.999, while the weight and bias are initialized to be 1 and 0, respectively. All the models are trained with the SGD solver on 4 GTX 1080 GPUs, CUDA 9.0, and cuDNN v7 with Intel Xeon E5-2620v4@2.10 GHz. Considering the limitation of GPU memory, this paper uses VGG-16[28] as the backbone network with a batch size of 32, and only trains the model with input size 300×300.

3.2 Testing strategies

Inspired by RefineDet[25], this paper performed both single-scale testing and multi-scale testing. The spatial resolutions of the output features of MF module P_4 to P_9 are 38^2, 19^2, 10^2, 5^2, 3^2, and 1^2, respectively. To perform multi-scale testing, P_8, P_9, and their associated layers are directly removed from the trained model with no other modifications. This will shrink the mAP of the VOC 2007 test set from 79.4% to 75.5% when performing single-scale testing. Multi-scale testing works by imposing different resolution inputs to the model and aggregating all the detection results together, and then uses the NMS with a threshold of 0.45 to obtain the final result on PASCAL VOC dataset while using Soft-NMS on the COCO dataset. The default input resolution is $S_1 \in \{176^2, 240^2, 304^2, 304 \times 176, 304 \times 432, 368^2, 432^2, 496^2, 560^2, 624^2, 688^2\}$. Additionally, the image horizontal flip operation is also used.

3.3 PASCAL VOC 2007

For the PASCAL VOC 2007 dataset, all the models are trained on the union of VOC 2007 and VOC 2012 trainval sets (16 551 images) and tested on the VOC 2007 test set (4 952 images). This paper adopts a fully convolutional VGG-16 net used in ParseNet[32] as the pre-trained model and fine-tunes it using SGD with a momentum of 0.9 and a weight decay of 2×10^{-4}. The initial learning rate is set to be 0.001, then reduced by a factor of 10 at iterations 6×10^4 and 10^5, respectively. The training cycle is 1.2×10^5 iterations.

Tab. 1 shows the results for each category on the VOC 2007 test set. Taking the GPU memory constraint into account, this paper only trains the model with input resolution 300×300 (i.e., NFPN-SSD300). Compared with some models based on the feature pyramid network with a similar input resolution (e.g., RSSD300, DSSD321, and FSSD300), the NFPN-based SSD without intricate tricks shows performance improvements in most catego-

ries, producing the increase of mAP of 0.9%, 0.8%, and 0.6%, respectively. Although its accuracy is inferior to some two-stage models, it guarantees real-time detection. After using the multi-scale testing strategy, the NFPN-based SSD can achieve 82.9% mAP, which is much better than single-scale testing (79.4% mAP).

Tab. 1 Detection results on the VOC 2007 test set %

Method		SSD300[17]	RSSD300[20]	DSSD321[19]	NFPN-SSD300	NFPN-SSD300*
Backbone		VGG-16	VGG-16	ResNet-101	VGG-16	VGG-16
APof each category	Aero	79.4	80.6	81.9	84.8	89.5
	Bike	84.0	85.2	84.9	86.4	88.2
	Bird	75.8	77.9	80.5	79.3	84.3
	Boat	69.7	68.1	68.4	72.5	78.9
	Bottle	50.7	54.5	53.9	56.7	66.5
	Bus	86.8	87.9	85.6	86.7	89.2
	Car	85.9	86.8	86.2	87.4	89.2
	Cat	88.6	87.3	88.9	87.6	88.0
	Chair	60.1	62.4	61.1	62.7	68.5
	Cow	82.2	83.6	83.5	86.2	89.1
	Table	77.3	76.0	78.7	76.9	77.0
	Dog	86.4	86.6	86.7	86.9	88.1
	Horse	87.6	88.4	88.7	88.3	89.2
	Mbike	84.3	86.4	86.7	87.3	88.8
	Person	79.6	80.0	79.7	80.2	84.7
	Plant	52.5	54.9	51.7	53.5	61.8
	Sheep	79.1	79.2	78.0	78.5	85.5
	Sofa	79.5	78.8	80.9	79.3	81.6
	Train	87.9	88.3	87.2	88.6	88.0
	TV	77.3	76.9	79.4	77.8	82.3
mAP		77.7	78.5	78.6	79.4	82.9

Note: * multi-scale testing.

Tab. 2 shows the inference speed and average precision (AP) of some state-of-the-art methods. NFPN-SSD300 takes 28.9 ms to process an image (i.e., 34.6 frame/s and 79.4% mAP) with no batch processing on a GTX 1080 GPU, which exceeds DSSD321 in both inference speed and detection precision but is slightly inferior to RefineDet320. Compared with some methods with large input sizes, NFPN-SSD300 only retains the superiority of inference speed.

In summary, the NFPN-based SSD exhibits a significant improvement in inference speed and detection precision compared to the structure-oriented modified model RSSD[20] and DSSD[19]. Referring to Fig. 5, with the SSD300 model as a baseline, the mAP of NFPN-SSD300 is increased by 2% to 6% for classes with specific backgrounds (e.g., airplane, boat, and cow), while increased by 6% for the bottle class, whose instances are usually small. It is proved that NFPN can extract more fertile contextual information, which is beneficial for the detection of small objects and classes with a unique context. Due to the transportability of this structure, it can also be easily embedded in other detectors to further boost their performance.

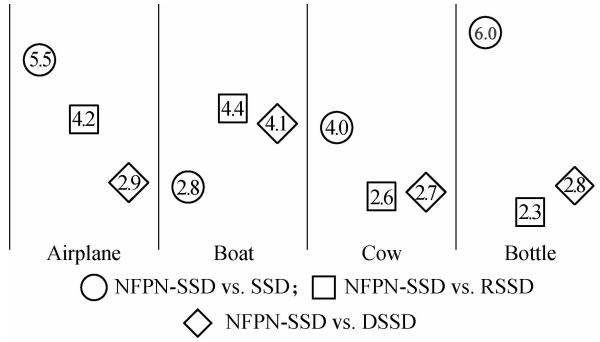


Fig. 5 Comparison of improved average precision (%) of specific classes on VOC 2007 test set

3.4 Ablation study on PASCAL VOC 2007

To demonstrate the effectiveness of the structure shown in Fig. 2(d), this paper designs four variants and evaluates them on the PASCAL VOC 2007 dataset. The training strategies are inherited from Section 3.3. In particular, the batch size is set to be 12, which is the largest batch size that a single GTX 1080 GPU can accommodate. The basic module of NFPN is shown in Fig. 3, which has three input branches, i.e., down-sampling, up-sampling, and the basic branch. By cutting out different branches, there are three variants: 1) down-sampling + basic; 2) basic + up-sampling; 3) down-sampling + basic + up-sampling. Furthermore, the fourth variant is a cascade of the feature pyramid network and can be formulated as

$$\left.\begin{aligned} P_i^* &= \mathrm{MF}(C_{i-1},\ C_i,\ C_{i+1}) \\ P_i &= \mathrm{MF}(P_{i-1}^*,\ P_i^*,\ P_{i+1}^*) \end{aligned}\right\} \tag{4}$$

Tab. 2 Inference speed vs. detection accuracy on PASCAL VOC dataset

Type	Method	Backbone	Input size	mAP/%		Speed/ (frame · s^{-1})	Batch size	GPU
				VOC 2007	VOC 2012			
Two stages	Faster R-CNN[6]	VGG-16	~1 000 × 600	73.2	70.4	7.0	1	Titan X
	Faster R-CNN[33]	ResNet-101	~1 000 × 600	76.4	73.8	2.4	1	K40
	MR-CNN[12]	VGG-16	~1 000 × 600	78.2	73.9	0.03	1	Titan
	R-FCN[13]	ResNet-101	~1 000 × 600	80.5	77.6	9.0	1	Titan X
	CoupleNet[16]	ResNet-101	~1 000 × 600	82.7	80.4	8.2	1	Titan X
One stage	YOLOv1[18]	VGG-16	448 × 448	66.4		21.0	1	Titan X
	YOLOv2[22]	Darknet-19	544 × 544	78.6	73.4	40.0	1	Titan X
	SSD512[17]	VGG-16	512 × 512	79.8	78.5	19.0	1	Titan X
	SSD513[19]	ResNet-101	513 × 513	80.6	79.4	6.8	1	Titan X
	RefineDet512[25]	VGG-16	512 × 512	81.8	80.1	24.1	1	Titan X
	RefineDet512 *[25]	VGG-16	512 × 512	83.8	83.5			
	SSD321[19]	ResNet-101	321 × 321	77.1	75.4	11.2	1	Titan X
	SSD300[17]	VGG-16	300 × 300	77.2	75.8	46.0	1	Titan X
	DSOD300[23]	DS/64-192-48-1	300 × 300	77.7	76.3	17.4	1	Titan X
	RefineDet320[25]	VGG-16	320 × 320	80.0	78.1	40.3	1	Titan X
	RefineDet320 *[25]	VGG-16	320 × 320	83.1	82.7			
One stage with FPN	RSSD512[20]	VGG-16	512 × 512	80.8		16.6	1	Titan X
	FSSD512[34]	VGG-16	512 × 512	80.9				
	DSSD513[19]	ResNet-101	513 × 513	81.5	80.0	5.5	1	Titan X
	RSSD300[20]	VGG-16	300 × 300	78.5	76.4	35.0	1	Titan X
	FSSD300[34]	VGG-16	300 × 300	78.8				
	DSSD321[19]	ResNet-101	321 × 321	78.6	76.3	9.5	1	Titan X
	SSD300	VGG-16	300 × 300	77.7	75.8	41.4	1	1080
	SSD300	VGG-16	300 × 300	77.7	75.8	48.1	8	1080
	NFPN-SSD300	VGG-16	300 × 300	79.4	77.1	34.6	1	1080
	NFPN-SSD300	VGG-16	300 × 300	79.4	77.1	37.7	8	1080
	NFPN-SSD300 *	VGG-16	300 × 300	82.9	82.4			

Note: * multi-scale testing.

where $\{P_i^*\}$ constitutes a junior feature pyramid network and $\{P_i\}$ constitutes a senior feature pyramid network.

Tab. 3 records the evaluation results. It is observed that the variant with three input branches shows better detection performance than those variants with two input branches (mAP 78.6% vs. 78.1% and 78.2%). The cascading of feature pyramid networks can also contribute to the detection performance (i.e., 0.5% higher), but the corresponding memory consumption will also increase. Accordingly, this paper selects the variant model with three input branches in other experiments. Attempts have also been made to integrate the FPN structure into the SSD framework, but it cannot converge to a matching detection performance.

Tab. 3 Ablation study on the VOC 2007 test set

Method	Basic branch	Down-sampling branch	Up-sampling branch	Cascade Eq. (4)	mAP/%
SSD300					76.7
NFPN-SSD300	✓	✓			78.1
NFPN-SSD300	✓		✓		78.2
NFPN-SSD300	✓	✓	✓		78.6
NFPN-SSD300	✓	✓	✓	✓	79.1

Note: ✓ denotes choosing this branch or strategy.

3.5 PASCAL VOC 2012

For PASCAL VOC 2012 dataset, all the models are trained on the union of VOC 2007 and VOC 2012 trainval sets plus VOC 2007 test set (21 503 images) and tested on the VOC 2012 test set (10 991 images). The training set is an augmentation of the training set used in Section 3.3, attaching about 5 000 images. Therefore, the NFPN-based SSD model that iterates 6×10^4 times in Section 3.3 is used as a pre-trained model to shorten the training cycle. Other training strategies are consistent with those discussed in Section 3.3. The evaluation results are recorded in Tab. 2 and Tab. 4. Considering similar input sizes (i.e., 300^2, 321^2), the NFPN-based SSD still shows performance improvement compared with SSD, RSSD, and DSSD. After using the multi-scale testing strategy, the mAP of NFPN-SSD300 can also catch up with RefineDet320.

3.6 MS COCO

MS COCO[10] is a large-scale object detection, segmentation, and captioning dataset. For the object detection task, COCO train, validation, and test sets contain more than 2×10^5 images and 80 object categories. Object detection performance metrics include AP and average recall (AR).

Tab. 4 Detection results on the VOC 2012 test set %

Method		SSD300[17]	RSSD300[20]	DSSD321[19]	NFPN-SSD300	NFPN-SSD300*
Backbone		VGG-16	VGG-16	ResNet-101	VGG-16	VGG-16
AP of each category	Aero	88.0	88.0	87.3	88.1	92.1
	Bike	82.8	83.8	83.3	83.4	87.7
	Bird	74.5	74.8	75.4	76.4	84.2
	Boat	61.7	60.8	64.6	66.3	72.4
	Bottle	47.5	48.9	46.8	50.4	65.9
	Bus	83.1	83.9	82.7	83.2	86.6
	Car	78.9	78.5	76.5	80.2	87.8
	Cat	91.7	91.0	92.9	91.4	93.2
	Chair	58.2	59.5	59.5	60.6	68.1
	Cow	80.1	81.4	78.3	82.2	87.9
	Table	63.9	66.1	64.3	65.1	67.7
	Dog	89.5	89.0	91.5	90.3	92.2
	Horse	85.7	86.3	86.6	87.8	91.0
	Mbike	85.5	86.0	86.6	85.6	89.7
	Person	82.5	83.0	82.1	83.4	89.2
	Plant	50.3	51.3	53.3	50.8	62.5
	Sheep	79.6	80.9	79.6	81.8	87.0
	Sofa	73.6	73.7	75.7	73.5	73.0
	Train	86.7	86.9	85.2	87.6	89.4
	TV	72.2	73.8	73.9	74.6	80.0
mAP		75.8	76.4	76.3	77.1	82.4

Note: * multi-scale testing.

By convention[35], this paper uses a union of 8×10^4 images from the COCO train set and random 3.5×10^4 images from the COCO validation set for training (the trainval35k split), and uses the test-dev evaluation server to evaluate the results. The pre-trained model and optimizer setting are the same as Section 3.3, while the training cycle is set to be 4×10^5 iterations. The initial learning rate is 10^{-3}, then decays to 10^{-4} and 10^{-5} at 2.8×10^5 iterations and 3.6×10^5 iterations, respectively. The experimental results are shown in Tab. 5, and some qualitative test results are shown in Fig. 8 and Fig. 9.

Similarly, to verify the validity of the feature pyramid network constructed in the NFPN-based SSD, the detection results are first compared with DSSD, which integrates FPN and ResNet-101 into the SSD framework with no other tricks. As can be seen from Fig. 6, the NFPN-based SSD has been improved in various detection evaluation metrics (e.g., AP and AR) compared with SSD. Additionally, compared with DSSD, the NFPN-based SSD is only inferior in the detection of large objects. That is, the NFPN-based SSD has more performance improvements for small objects. Fig. 7 visualizes the first 8 channels of the feature map used to detect small objects. For some images that only contain small instances, the feature map in SSD300 used for detection only has few activated neurons, which makes it difficult to locate objects. The combination of NFPN and SSD gives the model a stronger feature extraction capability so that the feature map used for detection can retain more useful information, as shown in the fourth row of Fig. 7. Comparing the feature maps in DSSD and NFPN-SSD, it is not difficult to find that NFPN-SSD can extract more detailed information, which is especially beneficial for the detection of small objects.

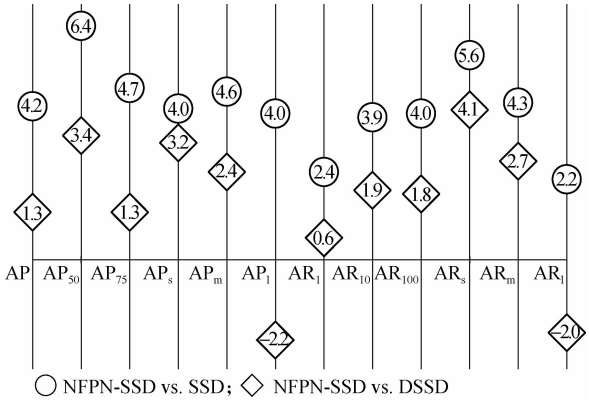


Fig. 6 Comparison of improved AP (%) and AR (%) on MS COCO test-dev set

Considering additional FPN-based detectors (e.g., RetinaNet400, NAS-FPN), the NFPN-based SSD is still inferior in detection accuracy, but it is superior in inference speed (34.6 vs. 15.6 and 17.8 frame/s). Due to the portability of NFPN, it can also complement these methods to further improve their performance.

Tab. 5 Detection results on MS COCO test-dev set

Type	Method	Backbone	Input size	AP/%	AP_{50}/%	AP_{75}/%	AP_s/%	AP_m/%	AP_l/%	Speed/(frame · s^{-1})	GPU
Two stages	Faster R-CNN** [6]	VGG-16	~1 000 × 600	21.9	42.7					7	Titan X
	R-FCN** [13]	ResNet-101	~1 000 × 600	29.9	51.9		10.8	32.8	45.0	9	Titan X
	CoupleNet** [16]	ResNet-101	~1 000 × 600	34.4	54.8	37.2	13.4	38.1	50.8	8.2	Titan X
	Faster R-CNN** [33]	ResNet-101	~1 000 × 600	34.9	55.7					2.4	K40
One stage	YOLOv2 [22]	Darknet-19	544 × 544	21.6	21.6	19.2	5.0	22.4	35.5	40	Titan X
	SSD512 [17]	VGG-16	512 × 512	28.8	48.5	30.3	10.9	31.8	43.5	19	Titan X
	SSD513 [19]	ResNet-101	513 × 513	31.2	50.4	33.3	10.2	34.5	49.8	6.8	Titan X
	YOLOv3 [24]	Darknet-53	608 × 608	33.0	57.9	34.4	18.3	35.4	41.9	19.6	Titan X
	RefineDet512 [25]	VGG-16	512 × 512	33.0	54.5	35.5	16.3	36.3	44.3	24.1	Titan X
	RefineDet512* [25]	VGG-16	512 × 512	37.6	58.7	40.8	22.7	40.3	48.3		Titan X
	SSD300 [17]	VGG-16	300 × 300	25.1	43.1	25.8	6.6	25.9	41.4	46	Titan X
	SSD321 [19]	ResNet-101	321 × 321	28.0	45.4	29.3	6.2	28.3	49.3	11.2	Titan X
	DSOD300** [23]	DS/64-192-48-1	300 × 300	29.3	47.3	30.6	9.4	31.5	47.0	17.4	Titan X
	RefineDet320 [25]	VGG-16	320 × 320	29.4	49.2	31.3	10.0	32.0	44.4	40.3	Titan X
	RefineDet320* [25]	VGG-16	320 × 320	35.2	56.1	37.7	19.5	37.2	47.0		Titan X
Two stages with FPN	Faster R-CNN [15]	ResNet-101-FPN	~1 333 × 800	36.2	59.1	39.0	18.2	39.0	48.2	6	M40
	Faster R-CNN [36]	ResNet-50-FPN	~1 333 × 800	36.6	58.5	39.2	20.7	40.5	47.9	13.5	V100
	Mask R-CNN [36]	ResNet-50-FPN	~1 333 × 800	37.4	58.9	40.4	21.7	41.0	49.1	10.2	V100
	Cascade R-CNN [36]	ResNet-50-FPN	~1 333 × 800	40.4	58.5	43.9	21.5	43.7	53.8	10.9	V100
One stage with FPN	DSSD513 [19]	ResNet-101	513 × 513	33.2	53.3	35.2	13.0	35.4	51.1	5.5	Titan X
	RetinaNet800 [21]	ResNet-50-FPN	800 × 800	35.7	55.0	38.5	18.9	38.9	46.3	6.5	M40
	RetinaNet800 [21]	ResNet-101-FPN	800 × 800	39.1	59.1	42.3	21.8	42.7	50.2	5.1	M40
	NAS-FPN [30]	ResNet-50	640 × 640	39.9						17.8	P100
	NAS-FPN [30]	ResNet-50	1 024 × 1 024	44.2						10.9	P100
	NAS-FPN [30]	ResNet-50	1 280 × 1 280	44.8						7.6	P100
	DSSD321 [19]	ResNet-101	321 × 321	28.0	46.1	29.2	7.4	28.1	47.6	9.5	Titan X
	RetinaNet400 [21]	ResNet-50-FPN	400 × 400	30.5	47.8	32.7	11.2	33.8	46.1	15.6	M40
	RetinaNet400 [21]	ResNet-101-FPN	400 × 400	31.9	49.5	34.1	11.6	35.8	48.5	12.3	M40
	NFPN-SSD300	VGG-16	300 × 300	29.3	49.5	30.5	10.6	30.5	45.4	34.6	1080
	NFPN-SSD300*	VGG-16	300 × 300	35.8	57.6	38.3	20.6	38.5	48.2		1080

Notes: * multi-scale testing; ** using the COCO trainval split as the training set.

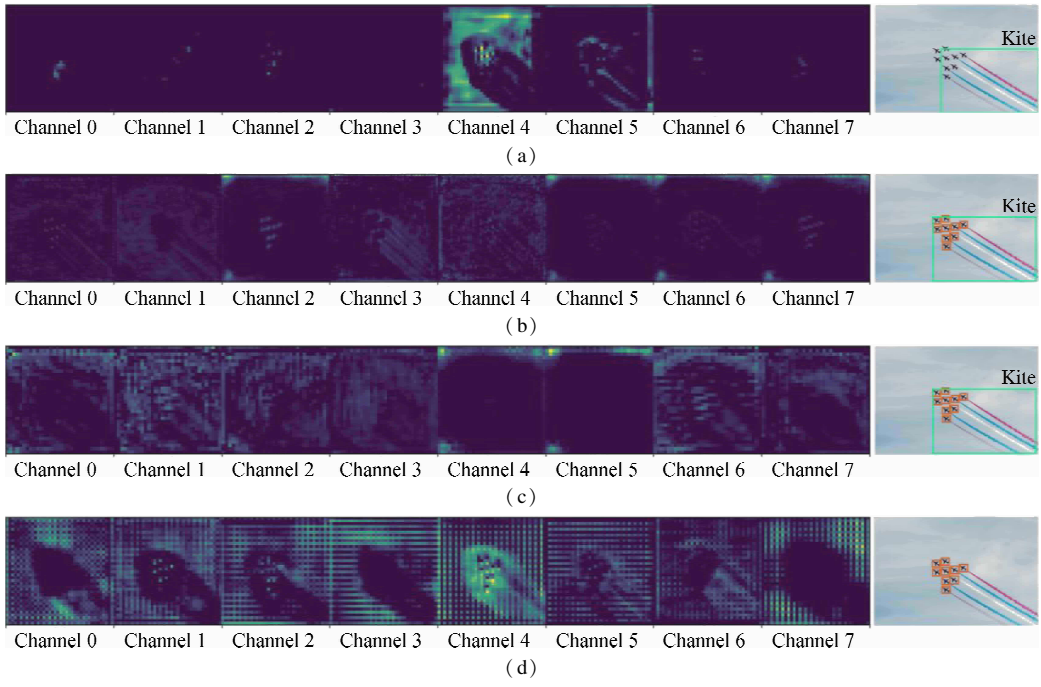
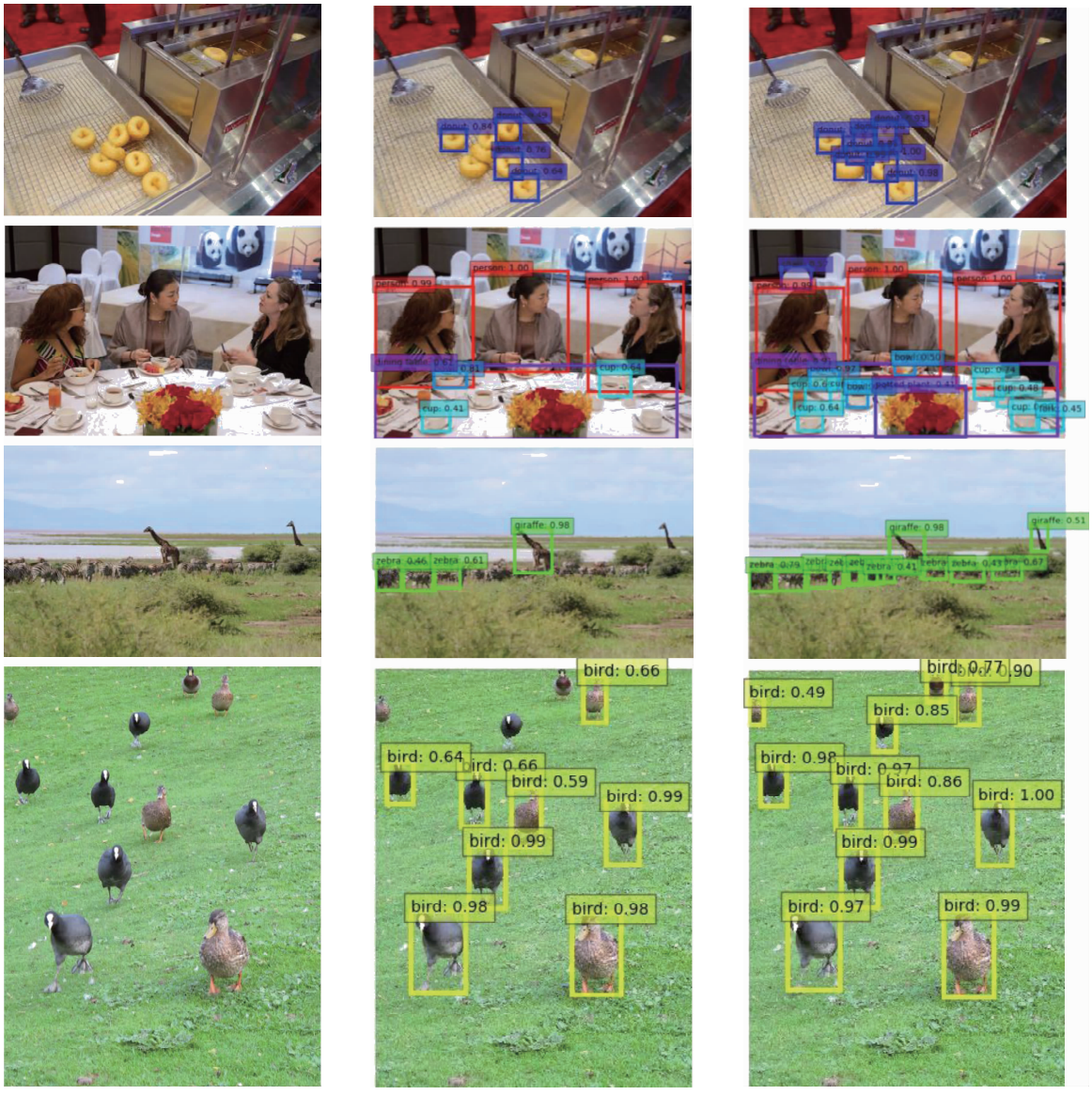


Fig. 7 Visualization of features used to locate small objects. (a) Feature map with a resolution of 38 × 38 in SSD300; (b) Feature map with a resolution of 64 × 64 in DSSD513; (c) Feature map with a resolution of 32 × 32 in DSSD513; (d) Feature map with a resolution of 38 × 38 in NFPN-SSD300



(a) (b) (c)

Fig. 8 Qualitative results for dense objects on COCO test-dev set. (a) Input images; (b) Results of SSD300; (c) Results of NFPN-SSD300

4 Conclusions

1) The hierarchical parallel structure of NFPN eliminates the successive resampling of features and does not increase the depth of the computational graph as much as the layer-by-layer structure. Additionally, the gradient can be passed back to the shallow layer along a shorter path, which is beneficial to the optimization of the model.

2) The hierarchical parallel feature pyramid network is more conducive to the parallel acceleration of GPU.

3) NFPN is highly portable and can be embedded in many methods to further boost their performance. It is demonstrated to be effective by integrating NFPN into the SSD framework, and extensive experimental results show that NFPN is more proficient for detecting small objects and classes with specific backgrounds.

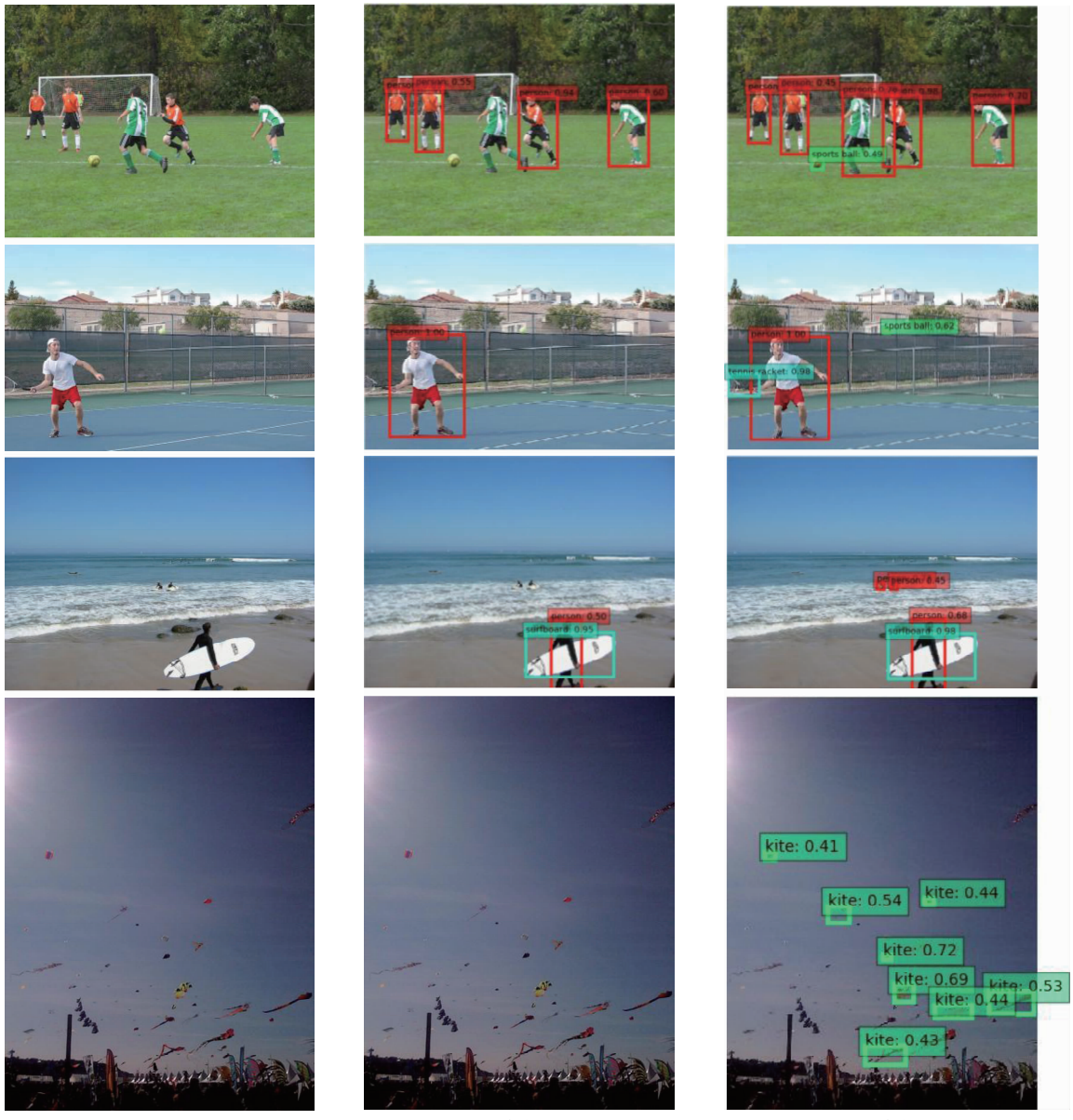
References

[1] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features[C]//*Proceedings of the* 2001 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Kauai, HI, USA, 2001: 511 - 518. DOI:10.1109/CVPR.2001.990517.

[2] Viola P, Jones M J. Robust real-time face detection[J]. *International Journal of Computer Vision*, 2004, **57**(2): 137 - 154. DOI:10.1023/b:visi.0000013087.49260.fb.

[3] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Diego, CA, USA, 2005: 886 - 893. DOI: 10.1109/CVPR.2005.177.

[4] Felzenszwalb P F, Girshick R B, McAllester D. Cascade object detection with deformable part models[C]//2010 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA, 2010 2241 - 2248. DOI:10.1109/CVPR.2010.5539906.



(a) (b) (c)

Fig. 9 Qualitative results for small objects on COCO test-dev set. (a) Input images; (b) Results of SSD300; (c) Results of NFPN-SSD300

[5] Uijlings J R R, Sande K, Gevers T, et al. Selective search for object recognition[J]. *International Journal of Computer Vision*, 2013, **104**(2): 154 - 171. DOI:10.1007/s11263-013-0620-5.

[6] Ren S Q, He K M, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017, **39**(6): 1137 - 1149. DOI: 10.1109/TPAMI.2016.2577031.

[7] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//2014 *IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA, 2014: 580 - 587. DOI:10.1109/CVPR.2014.81.

[8] He K M, Zhang X Y, Ren S Q, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015, **37**(9): 1904 - 1916. DOI:10.1109/TPAMI.2015.2389824.

[9] Everingham M, van Gool L, Williams C K I, et al. The pascal visual object classes (VOC) challenge[J]. *International Journal of Computer Vision*, 2010, **88**(2): 303 - 338. DOI:10.1007/s11263-009-0275-4.

[10] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: common objects in context[M]//*Computer Vision—ECCV* 2014. Cham: Springer International Publishing, 2014: 740 - 755. DOI:10.1007/978-3-319-10602-1_48.

[11] Girshick R. Fast R-CNN[C]//2015 *IEEE International Conference on Computer Vision*. Santiago, Chile, 2015: 1440 - 1448. DOI:10.1109/ICCV.2015.169.

[12] Gidaris S, Komodakis N. Object detection via a multi-region and semantic segmentation-aware CNN model[C]// 2015 *IEEE International Conference on Computer Vision*. Santiago, Chile, 2015: 1134 - 1142. DOI:10.1109/IC-

CV. 2015. 135.

[13] Dai J F, Li Y, He K M, et al. R-FCN: Object detection via region-based fully convolutional networks [J/OL]. arXiv preprint arXiv:1605. 06409, 2016. https://arxiv.org/abs/1605. 06409.

[14] He K M, Gkioxari G, Dollár P, et al. Mask r-cnn[C]// 2017 *IEEE International Conference on Computer Vision*. Venice, Italy, 2017: 2980 - 2988. DOI:10. 1109/ICCV. 2017. 322.

[15] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection [C]//2017 *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, HI, USA., 2017: 936 - 944. DOI: 10. 1109/CVPR. 2017. 106.

[16] Zhu Y S, Zhao C Y, Wang J Q, et al. CoupleNet: Coupling global structure with local parts for object detection [C]//2017 *IEEE International Conference on Computer Vision*. Venice, Italy, 2017: 4146 - 4154. DOI: 10. 1109/ICCV. 2017. 444.

[17] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector[M]//*Computer Vision—ECCV* 2016. Cham: Springer International Publishing, 2016: 21 - 37. DOI:10. 1007/978-3-319-46448-0_2.

[18] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection [C]//2016 *IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas, NV, USA, 2016: 779 - 788. DOI: 10. 1109/CVPR. 2016. 91.

[19] Fu C Y, Liu W, Ranga A, et al. DSSD: Deconvolutionalsingle shot detector [J/OL]. arXiv preprint arXiv: 1701. 06659, 2017. https://arxiv. org/abs/1701. 06659.

[20] Jeong J, Park H, Kwak N. Enhancement of SSD by concatenating feature maps for object detection[C]//*British Machine Vision Conference*. London, UK, 2017. DOI: 10. 5244/C. 31. 76.

[21] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection [C]//2017 *IEEE International Conference on Computer Vision*. Venice, Italy, 2017: 2999 - 3007. DOI:10. 1109/ICCV. 2017. 324.

[22] Redmon J, Farhadi A. YOLO9000:Better, faster, stronger[C]//2017 *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, HI, USA, 2017: 6517 - 6525. DOI:10. 1109/CVPR. 2017. 690.

[23] Shen Z Q, Liu Z, Li J G, et al. DSOD:Learning deeply supervised object detectors from scratch[C]//2017 *IEEE International Conference on Computer Vision*. Venice, Italy, 2017: 1937 - 1945. DOI: 10. 1109/ICCV. 2017. 212.

[24] Redmon J, Farhadi A. YOLOv3: An incremental improvement [J/OL]. arXiv preprint arXiv:1804. 02767, 2018. https://arxiv. org/abs/1804. 02767.

[25] Zhang S F, Wen L Y, Bian X, et al. Single-shot refinement neural network for object detection [C]//2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018: 4203 - 4212. DOI:10. 1109/CVPR. 2018. 00442.

[26] Huang G, Liu Z, van der Maaten L, et al. Densely connected convolutional networks[C]//2017 *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, HI, USA, 2017: 2261 - 2269. DOI: 10. 1109/CVPR. 2017. 243.

[27] Tian Z, Shen C H, Chen H, et al. FCOS: fully convolutional one-stage object detection[C]//2019 *IEEE/CVF International Conference on Computer Vision*. Seoul, Korea, 2019: 9626 - 9635. DOI:10. 1109/ICCV. 2019. 00972.

[28] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition [J/OL]. arXiv preprint arXiv: 1409. 1556, 2014. https://arxiv. org/abs/1409. 1556.

[29] Liu S, Qi L, Qin H F, et al. Path aggregation network for instance segmentation[C]//2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018: 8759 - 8768. DOI: 10. 1109/CVPR. 2018. 00913.

[30] Ghiasi G, Lin T Y, Le Q V. NAS-FPN: Learning scalable feature pyramid architecture for object detection[C]// 2019 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Long Beach, CA, USA, 2019: 7029 - 7038. DOI:10. 1109/CVPR. 2019. 00720.

[31] Jia Y Q, Shelhamer E, Donahue J, et al. Caffe: convolutional architecture for fast feature embedding[C]//*Proceedings of the ACM International Conference on Multimedia*. Orlando, FL, USA, 2014: 675 - 678. DOI:10. 1145/2647868. 2654889.

[32] Liu W, Rabinovich A, Berg A C. ParseNet: Looking-wider to see better [J/OL]. arXiv preprint arXiv:1506. 04579, 2015. https://arxiv. org/abs/1506. 04579.

[33] He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition[C]//2016 *IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas, NV, USA, 2016: 770 - 778. DOI: 10. 1109/CVPR. 2016. 90.

[34] Li Z, Zhou F. FSSD: Featurefusion single shot multibox detector [J/OL]. arXiv preprint arXiv: 1712. 00960, 2017. https://arxiv. org/abs/1712. 00960.

[35] Bell S, Zitnick C L, Bala K, et al. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks [C]//2016 *IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas, NV, USA, 2016: 2874 - 2883. DOI: 10. 1109/CVPR. 2016. 314.

[36] Chen K, Wang J, Pang J, et al. MMDetection: Open MMLabdetection toolbox and benchmark [J/OL]. arXiv preprint arXiv: 1906. 07155, 2019. https://arxiv. org/abs/1906. 07155v1.

用于目标检测的邻域融合与分层并行特征金字塔网络

莫凌飞　　胡书铭

(东南大学仪器科学与工程学院, 南京 210096)

摘要: 为了提升对小目标的检测精度,提出了一种基于邻域融合的分层并行特征金字塔网络(NFPN). 与特征金字塔网络(FPN)和反卷积单次多框检测器(DSSD)中采用的逐层递进融合方式(特征金字塔网络的底层特征依赖于顶层特征)不同,NFPN 仅对具有相似尺度的浅层特征进行融合,所构建的特征金字塔网络上下层之间没有依赖关系. NFPN 具有高度的可移植性,可嵌入到许多检测模型中来进一步提升性能. 在 PASCAL VOC 2007、2012 和 COCO 数据集上进行的大量实验表明,基于 NFPN 的 SSD 模型在检测精度和推理速度方面均优于 DSSD 模型,尤其是对于小目标而言,NFPN-SSD300 精度比 SSD300 高 4% ~5%,比 DSSD321 高 2% ~3%. 在 VOC 2007 测试集上,输入分辨率为 300×300 的基于 NFPN 的 SSD 模型可以实现 79.4% 的检测精度和 34.6 frame/s 的推理速度,在使用多尺度测试方法后,其精度可提升至 82.9%.

关键词: 计算机视觉;深度卷积神经网络;目标检测;分层并行;特征金字塔网络;多尺度特征融合

中图分类号: TP391.4