TFPN: Twin Feature Pyramid Networks for Object Detection

Liang Yi*, Changjian Wang*, Fangzhao Li*, Yuxing Peng,
Yuan Yuan, Zhen Huang
School of Computer
National University of Defense Technology
ChangSha, Hunan, China
{yiliang17a, wangcj, lifangzhao12, pengyuhang,
yuanyuan, huangzhen}@nudt.edu.cn

Qin Lv
Department of Computer Science
University of Colorado Boulder
Boulder, CO USA
qin.lv@colorado.edu

***Abstract*—FPN (Feature Pyramid Networks) is one of the most popular object detection networks, which can improve small object detection by enhancing shallow features. However, there has been limited study on the improvement of large and medium object detection via deeper feature enhancement. One existing approach merges the feature maps of different layers into a new feature map for object detection, but may lead to increased noise and loss of information due to up-sampling and down-sampling. Another approach adds a bottom-up structure after the feature pyramid of FPN, which superimposes the information from shallow layers into the deep feature map but weakens the strength of FPN in detecting small objects. To address these challenges, this paper proposes TFPN (Twin Feature Pyramid Networks), which consists of (1) FPN+, a bottom-up feature enhancement structure that improves large object detection; (2) TPS, a Twin-Pyramid Structure that improves medium object detection; and (3) effective integration of the different structures, which can significantly improve the detection accuracy of large and medium objects while maintaining the advantage of FPN in small object detection. Extensive experiments using the MSCOCO object detection dataset and the BDD100K automatic driving dataset demonstrate that TFPN significantly improves over existing models, achieving up to 2.2 improvement in detection accuracy (e.g., 36.3 for FPN vs. 38.5 for TFPN on COCO Val-17). Using ResNet-50, TFPN can obtain the same accuracy as FPN with ResNet-101, thus requiring fewer parameters.**

***Index Terms*—Object Detection, FPN, TFPN, Twin-Pyramid Structure**

I. Introduction

Object detection is a fundamental task in many real-world applications, and Feature Pyramid Networks (FPN) is one of the most popular deep neural networks for object detection. A number of FPN-based object detectors have been proposed, such as FSSD [9], DSSD [1] and so on. As illustrated in Fig. 1, FPN can extract feature maps of different scales at different layers of deep convolutional networks, thus supporting the detection of multi-scale objects. And FPN adopts a top-down structure to fuse deep-layer semantic information into shallow network layers, which effectively improves the detection of small objects. However, FPN does not pay much attention to the detection of large and medium objects. Intuitively, one can also improve the detection performance of larger objects by fusing the information of shallow layers into deep layers. To this end, two approaches have been proposed in the literature to improve upon FPN. One approach is to analyze and process the feature map of element-wise sum or concatenate after unifying the size of feature maps of different layers, such as Deep Feature Pyramid Reconfiguration [8] (see Fig. 3) and Libra RCNN [15] (see Fig. 4).

Fig. 1. FPN Framework [10]

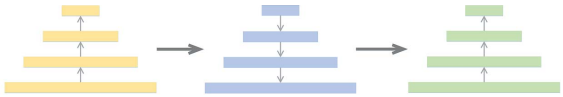


Fig. 2. PANet Framework [13]

However, to resize the scales of different feature maps, small scale feature maps need to be up-sampled and large-size feature maps need to be down-sampled. The former increases the noise while the latter leads to loss of detailed information, thus limiting the potential improvement. The other approach is to add a bottom-up structure after FPN, such as PANet (see Fig. 2). The bottom-up structure performs information fusion by superimposing shallow feature maps into deep feature maps. The results are then used for object detection. Since this approach focuses on enhancing the deep feature maps, it weakens the strength of FPN in small object detection.

To address the above-mentioned limitations, we propose a novel network model for object detection, called TFPN (Twin Feature Pyramid Networks), which effectively fuses feature maps through two feature pyramid networks and a bottom-up feature enhancement module to achieve a richer encoding of multi-scale context (see Fig. 5). Specifically, a small feature pyramid network is added after the original feature pyramid structure in FPN, and is responsible for the information fusion from the shallow layers to the deep layers. The output of both the first and second feature pyramid networks are fed

*Corresponding authors.

2375-0197/19/$31.00 ©2019 IEEE
DOI 10.1109/ICTAI.2019.00251

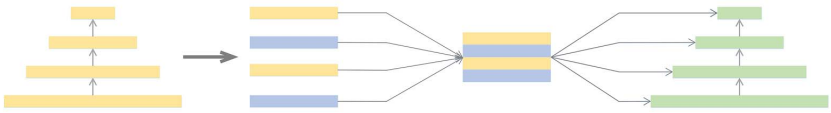


Fig. 3. Deep Feature Pyramid Reconfiguration Networks [8]

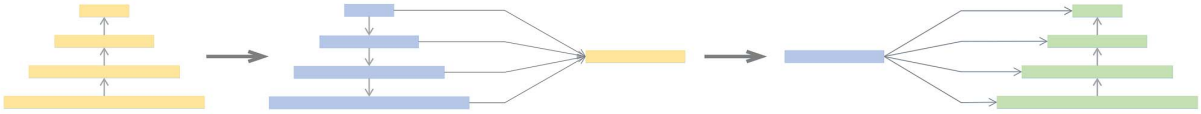


Fig. 4. Libra RCNN Framework [15]

into the bottom-up feature enhancement module, thus enabling effective detection of small, medium, and large objects using the same model.

The contributions of this paper are as follows:

(1) We propose TFPN, a novel Twin Feature Pyramid Networks framework for object detection, which consists of three key innovations:

1) We propose FPN+, a feature enhancement module that effectively fuses shallow-layers information into the deep features to improve the detection of large objects.

2) We propose TPS, a Twin-Pyramid Structure that leverages a second pyramid structure to improve the detection of medium objects.

3) We effectively integrate the two proposed techniques into an overall framework to improve the detection performance of small, medium, and large objects at the same time.

(2) We have implemented the proposed TFPN framework and conducted extensive experiments using the MSCOCO [12] and BDD100K [20] datasets. Our results show that TFPN can significantly improve the detection performance over Faster RCNN, achieving up to 2.2 improvement overall and 3.2 improvement for large objects based on ResNet-50.

II. Related Work

CNN-based Object Detectors. There are two classes of CNN-based object detectors: two-stage detectors and one-stage detectors. RCNN [3] is the earliest two-stage detector. Fast R-CNN [2] put forward ROI pooling to normalize different scale proposals. Faster RCNN [18] firstly designs Region Proposal Networks (RPN) to generate proposals to achieve end-to-end training of object detector. One-stage detectors such as YOLOv1 [17] achieve real-time object detection for the first time. After that, SSD [14] adapts to multi-scale changes of object detection by detecting objects of different sizes at different layers. RetinaNet [11] proposes focal loss to solve the imbalance problem of positive and negative samples in one-stage detectors.

Feature Pyramid for Object Detection. In recent years, many methods have improved object detection performance by integrating features of different layers of CNN. Deep Feature Pyramid Reconfiguration [8] concatenates same-scale feature maps which are resized from different-level feature maps extracted by backbone. Based on the concatenated feature maps, feature maps of different sizes are generated for detection after global attention and local reconfigurations. Pang *et al.* [15] down-samples and up-samples low-level and high-level information is to the same scale, respectively. After element-wise sum, a non-local model [19] that comprehensively balances semantic features is adopted to enhance the original features of FPN, so as to make the features more discriminative.

TABLE I
Key Notations.

Generation stages	Generation feature maps
F Pyramid (FPN): Stage 2, 3, 4, 5	$\{C_2, C_3, C_4, C_5\}$
F Pyramid (FPN): Top-down path	$\{C'_2, C'_3, C'_4, C'_5\}$
S Pyramid: Bottom-up Structure (BS)	$\{P_2, P_3, P_4, P_5\}$
S Pyramid: Top-down Structure (TS)	$\{Q_2, Q_3, Q_4, Q_5\}$
E Module (EM)	$\{F_2, F_3, F_4, F_5\}$

III. Method

In this section, we present in detail the TFPN framework we have developed for object detection, which builds upon FPN's strength for small object detection and further improves the performance for medium and large object detection.

A. Overview and Key Notations

Fig 5 illustrates the overall TFPN framework design. As the name suggests, TFPN contains two feature pyramid structures. We refer to the first feature pyramid networks on the left as the *F Pyramid* and the second feature pyramid networks in the middle as the *S Pyramid*.

The entire network structure includes five convolution stages, which correspond to down-sampling the input image 2, 4, 8, 16, and 32 times, respectively. Here, we use $\{C_2, C_3, C_4, C_5\}$ to represent the generated feature maps at stage 2, 3, 4, 5 in the backbone network, and $\{C'_2, C'_3, C'_4, C'_5\}$ represent the corresponding output feature maps of the *F Pyramid*, respectively. The *S Pyramid* also contains two parts: a bottom-up structure and a top-down structure. The bottom-up structure generates $\{P_2, P_3, P_4, P_5\}$, corresponding to $\{C_2, C_3, C_4, C_5\}$ in the *F Pyramid* respectively. Then $\{Q_2, Q_3, Q_4, Q_5\}$ are produced by the top-down structure corresponding to $\{P_2, P_3, P_4, P_5\}$ in the bottom-up structure.

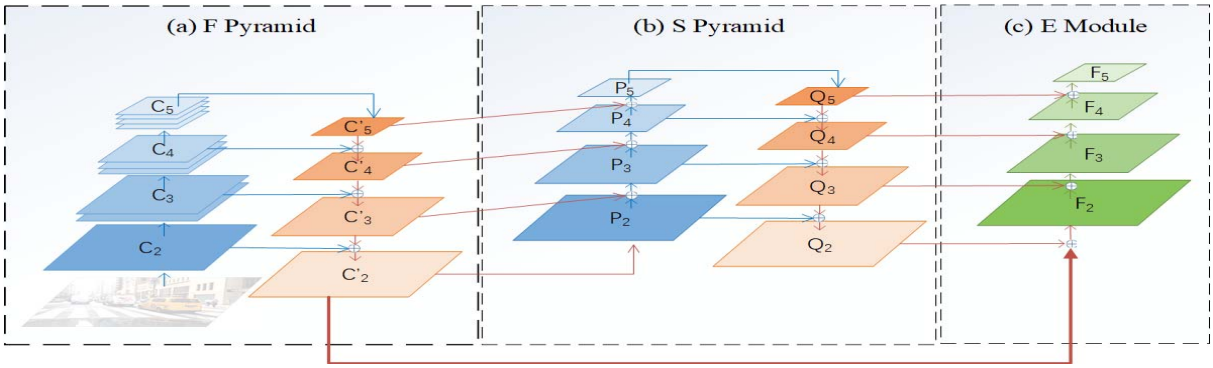


Fig. 5. The Entire Framework of Twin Feature Pyramid Networks (TFPN)

In addition to the Twin-Pyramid Structure, TFPN contains another independent bottom-up structure, referred to as the *E Module*. It takes the outputs of the *F Pyramid* and the *S Pyramid* as inputs and generates $\{F_2, F_3, F_4, F_5\}$ as outputs, which are then fed into a RPN network to generate anchors and proposals for object detection.

B. FPN+: Feature Enhancement Module

We first propose FPN+ to improve large object detection over FPN. As mentioned earlier, FPN improves small object detection by enhancing shallow features. However, it ignores the information fusion from the shallow layers to the deep layers, which can potentially improve large object detection. Specifically, we propose to append a bottom-up path after FPN, referred to as the feature enhancement module (*E Module*), as shown in Fig. 6. This structure only conducts element-wise sum operation on the feature maps $\{C_2', C_3', C_4', C_5'\}$ from shallower layers to deeper layers. This approach is similar to that of PANet [13], but differs in that FPN+ removes the ReLU [4] operation to reduce information loss. As shown in the experiments, FPN+ works effectively for large object detection.

Besides fusing the feature maps $\{C_2', C_3', C_4', C_5'\}$, the *E Module* can be extended to further fuse feature maps $\{P_2, P_3, P_4, P_5\}$ or $\{Q_2, Q_3, Q_4, Q_5\}$ and C_2'. The features generated by the *E Module* are $\{F_2, F_3, F_4, F_5\}$ as listed in Table I. Next, we use $\{Q_2, Q_3, Q_4, Q_5\}$ and C_2' as the input feature maps to illustrate the operations of the *E Module*.

As shown in Fig. 6, the *E Module* has two types of fusion operations. The first type is shown in the lower right, which is responsible for the integration of the lowest feature map C_2' in the *F Pyramid* and the lowest feature map Q_2 in the *S Pyramid* to generate new feature map F_2. The second fusion type is shown in the upper right, which is responsible for the integration of feature map Q_{i+1} in the *F Pyramid* and feature map F_i in the *E Module* to generate new feature map F_{i+1}.

Specifically, we first take the element-wise sum of Q_2 and C_2' to generate a new feature map, which then goes through a 3×3 convolutional layer with stride 1 to generate feature map F_2.

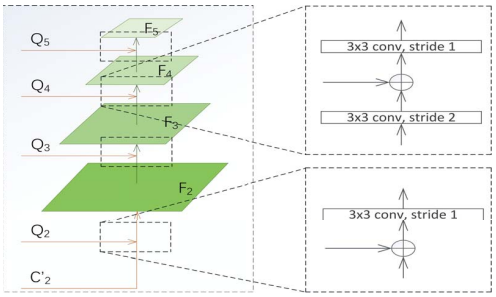


Fig. 6. FPN+: Two building blocks illustrating the feature enhancement module (E Module). The building block on the upper right depicts the fusion of high-level feature maps ($\{Q_3, Q_4, Q_5\}$). The building block on the lower right depicts the fusion of lowest-level feature maps (Q_2 and C_2').

After that, for each $i = 2, 3, 4$, given the higher-resolution feature map F_i and the corresponding feature map Q_{i+1} from the *S Pyramid*, the feature map F_i first goes through a 3×3 convolutional layer with stride 2. We then take element-wise sum of the down-sampled result and the feature map Q_{i+1} by horizontal connection. After that, the fused feature map is processed by another 3×3 convolutional layer to generate a new feature map F_{i+1} for subsequent subnetworks. In the *E Module*, all the feature maps use 256 channels. Note that there are no non-linearities in the *E Module*, which we have empirically determined to have minimum impact on detection performance.

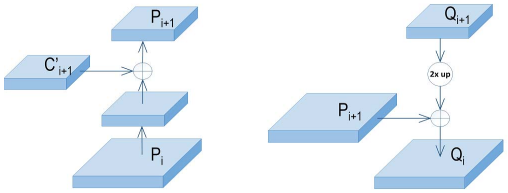
FPN+ is an iterative process from low-level F_2 to high-level F_5. Therefore, it can be described by the following formula:

$$F_i = \begin{cases} \Phi_1(Q_i + C_2'), i = 1 \\ \Phi_1(Q_i + \Phi_2(F_i))), i = 2, 3, 4 \end{cases} \quad (1)$$

Where Φ_1 refers to 3×3 convolution with stride 1, and Φ_2 refers to 3×3 convolution with stride 2.

C. TPS: Twin-Pyramid Structure

We also propose TPS (Twin-Pyramid Structure) to improve medium object detection over FPN. As the name suggests, TPS consists of two pyramid network structures: the *F Pyramid* and the *S Pyramid*.



(a) Bottom-up Structure (b) Top-down Structure

Fig. 7. TPS: Two fusion structures of the S Pyramid.

F Pyramid. For fair comparison, we use FPN as our first feature pyramid networks. It is responsible for image feature extraction and information fusion from deep network layers to shallow network layers. Besides these original functions, the *F Pyramid* also needs to provide feature maps to the *S Pyramid*.

S Pyramid. The *S Pyramid* is used for effective fusion of different depth information under the condition that the feature maps' scale remains unchanged. As illustrated in Fig 7, the *S Pyramid* is divided into two parts: a bottom-up structure (BS) and a top-down structure (TS). The bottom-up structure is responsible for the integration of low-level information into high-level network layers, while the top-down structure is responsible for the integration of high-level information into low-level network layers.

(a) Bottom-up Structure. The bottom-up path starts from P_2, the lowest level, and gradually generates feature maps till P_5. The down-sampling factor from P_2 to P_5 is 2. Here P_2 corresponds to C_2' in a convolution layer.

As shown in Fig. 7. (a), in the bottom-up structure, each fusion operation contains two input maps: a higher-resolution feature map P_i and a corresponding feature map C_{i+1}' from the *F Pyramid*. Each feature map P_i goes through a 3×3 convolutional layer with stride 2. We then compute the element-wise sum of the down-sampled result and the feature map C_{i+1} by horizontal connection. After that, the fused feature map is processed by another 3×3 convolutional layer to generate a new feature map P_{i+1} for subsequent subnetworks. This is an iterative process that ends after approaching P_5. In this structure, all the feature maps use 256 channels. And all convolution layers have a BN [6] and ReLU [4] after them.

(b) Top-down Structure. The top-down path starts from the highest level Q_5 and ends at the lowest level Q_2. The spatial up-sampling factor from Q_2 to Q_5 is 2. Here Q_5 corresponds to P_5 that goes through a convolution layer.

As shown in Fig. 7. (b), each small module has two input maps: a deeper feature map Q_{i+1} and a corresponding feature map P_{i+1} from the bottom-up structure. First of all, Q_{i+1} is resized to the same scale of P_{i+1} by nearest neighbor. Then we compute the element-wise sum of the up-sampling result and the feature map P_{i+1} to produce a unified feature map. The unified feature map will be processed by a 3×3 convolutional layer. As a result, the feature map Q_i will be generated for subsequent subnetworks. This is also an iterative process, terminating at Q_2. In the top-down structure, all the feature maps use 256 channels, but the convolution layers have no BN or ReLU after them.

D. Twin Feature Pyramid Networks

As described above, the original FPN works well for small object detection, while our proposed FPN+ and TPS work well for large and medium object detection, respectively. To achieve overall good performance for object detection of varying sizes, we propose the TFPN (Twin Pyramid Feature Networks) framework, which effectively integrates all three types of networks for general object detection. The complete framework of TFPN is shown in Fig. 5, which consists of two main parts: a Twin-Pyramid Structure and an *E Module*. In TFPN, both the *F Pyramid* and the *S Pyramid* are inputs of the *E Module*. As such, the *E Module* fuses information from both pyramids, making the information fusion process more effective while maintaining the advantages of different structures for object detection.

The first feature pyramid—*F Pyramid*, has the same structure and function as FPN. Firstly, an image is extracted by the left backbone to generate $\{C_2, C_3, C_4, C_5\}$. After that, $\{C_2', C_3', C_4', C_5'\}$ are obtained when the shallow network layers are gradually fused with high-level information via up-sampling. Similar to the *F Pyramid*,*S Pyramid* contains two parts: a bottom-up structure and a top-down structure. The bottom-up structure is responsible for generating new feature maps $\{P_2, P_3, P_4, P_5\}$ through the convolution and gradual fusion of $\{C_2', C_3', C_4', C_5'\}$ from the *F Pyramid*. And the top-down structure can generate $\{Q_2, Q_3, Q_4, Q_5\}$ by fusing $\{P_2, P_3, P_4, P_5\}$ from deep network layers to shallow network layers.

After the Twin-Pyramid Structure, a feature enhancement module—*E Module*, is designed to fuse $\{Q_2, Q_3, Q_4, Q_5\}$ from the *S Pyramid* with C_2' from the *F Pyramid*, which can retain FPN's advantage in small object detection and TPS' advantage in medium object detection, as well as reducing information loss due to the increased pyramid to enhance the detection of large objects. The *E Module* produces final feature maps $\{F_2, F_3, F_4, F_5\}$, which are fed into the RPN for generating anchors and proposals.

IV. Experiments

In this section, we evaluate the effectiveness of TFPN on MSCOCO and BDD100K benchmarks. We also conduct ablation studies to evaluate the effectiveness of our design on MSCOCO.

A. Datasets and Evaluation Metrics

MSCOCO. The MSCOCO dataset consists of natural images from 80 object categories. It consists of 115k images for training (train-2017), 5k images for validation (val-2017) and 20K images for testing (test-2017). Following common practice (FPN and Libra RCNN), we use train-2017 for training and report main results and ablation studies on val-2017 respectively. All reported results follow standard COCO-style Average Precision (AP) metrics that include AP (averaged over IoU thresholds), AP_{50} (AP for IoU threshold 50%), AP_{75} (AP

TABLE II
RESULTS ON COCO [12] VAL-17. "OURS-RE" IS OUR RE-IMPLEMENTATION OF FPN AND "OURS" IS OUR IMPLEMENTED VERSION OF TFPN.

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN (ours-re)	ResNet-50-FPN	36.3	58.4	39.0	21.2	40.3	46.5
Libra RCNN [15]	ResNet-50-FPN	37.7	59.4	40.9	22.4	41.3	49.3
Faster RCNN (ours)	ResNet-50-TFPN	**38.5***	**59.7**	**42.1**	**23.0**	**42.0**	**49.7**

*The accuracy is the same as FPN based on ResNet-101 which is re-implemented in mmdetection [7].

TABLE III
RESULTS ON BDD100K [20] VAL.

Method	Backbone	Mean (AP_{75})
Faster RCNN (ours-re)	ResNet-50-FPN	27.2
Faster RCNN (ours)	ResNet-50-TFPN	**28.5**
Faster RCNN (ours-re)	ResNet-101-FPN	28.9
Faster RCNN (ours)	ResNet-101-TFPN	**29.7**

for IoU threshold 75%). The COCO-style Average Recall (AR) with AR_1, AR_{10}, AR_{100} correspond to the average recall when there are 1, 10 and 100 proposals per image respectively. We also include AP_S and AR_S, AP_M and AR_M, AP_L and AR_L, which correspond to the precision and recall on small, medium and large scales respectively.

BDD100K. BDD100K consists of road scene images for 10 categories in total. There are 70k training images, 10k validation images, and 20k testing images. We only use the training subset for training and report results on the validation subset. Results use standard BDD100K-style AP_{75} metrics.

B. Implementation Details

MSCOCO. Experiments on MSCOCO also are implemented on mmdetection [7] based on PyTorch [16]. Detectors are trained with 4 1080Tis (4 images per GPU with backbone ResNet-50 [5] and 2 images per GPU with backbone ResNet-101 [5]) for 12 epochs with an initial learning rate of 0.02, and decrease it by 0.1 after 8 and 11 epochs respectively if not specifically noted. Other hyper-parameters follow the settings in mmdetection if not specifically noted.

BDD100K. Experiments on BDD100K also are implemented on mmdetection. Detectors are trained with 2 1080Tis (4 images per GPU with backbone ResNet-50 and 2 images per GPU with backbone ResNet-101) for 12 epochs.Other hyper-parameters follow the settings with MSCOCO experiments.

C. Main Results

Results on MSCOCO. Table II compares the object detection performance of TFPN with our re-implemented baseline Faster RCNN and Libra RCNN [15] on COCO val-17. Specifically, our re-implemented baseline Faster RCNN (i.e., FPN) achieves 36.3, 58.4, 39.0, 21.2, 43.0 and 46.5 in AP, AP_{50}, AP_{75}, AP_S, AP_M, and AP_L respectively. Our TFPN approach significantly improves over both FPN and Libra RCNN, with up to 2.2 improvement on AP and 3.2 improvement on AP_L. These results prove that our TFPN framework is more effective in terms of improving FPN than Libra RCNN. Furthermore, using ResNet-50, our TFPN framework achieves the same AP (38.5) as FPN based on ResNet-101(re-implemented in mmdetection). In particular, the most significant improvement (from 46.5 to 49.7, 3.2 improvement) is achieved for large object detection.

Results on BDD100K. Table III shows the results on BDD100K val. We conduct our studies based on both ResNet-50-FPN and ResNet-101-FPN Faster RCNN baselines. The AP_{75} improves from 27.2 to 28.5 on ResNet-50-FPN and from 28.9 to 29.7 on ResNet-101-FPN. These results demonstrate the effectiveness of our TFPN framework, not only on multi-class datasets but also on fewer-class datasets.

D. Ablation Studies

Table IV reports the ablation studies of the three key parts of TFPN: *F Pyramid*, *S Pyramid* and *E Module* on COCO val-17. We divide the *S Pyramid* into the Bottom-up Structure (BS) and the Top-down Structure (TS) to evaluate the effectiveness of the *S Pyramid*.

F Pyramid. The *F Pyramid* is our re-implemented FPN baseline. Its results are different from that of the Faster RCNN re-implemented in mmdetection due to the different experimental environment. The results are shown in the second row in Table IV.

S Pyramid. To demonstrate the effectiveness of the sub-structure of the *S Pyramid*, we extend the *F Pyramid* with TPS (both BS and TS), and the AP, AP_{50} and AP_{75} in the third row in Table IV have respectively improved from 36.3, 58.4, 39.0 to 37.5, 58.8 and 40.6. It is worth noting that detection for medium objects improved by 1.0 while small and large objects improved by 0.5 and 0.6 respectively. We also see significant improvements on the different AR metrics. These results verify the effectiveness of the Twin-Pyramid Structure for large object detection.

E Module. When adding only FPN+ (*E Module*) on top of the *F Pyramid*, the results in the fourth row in Table IV improve AP by more than 0.6. The values for AP and AR on small, medium, and large scales also improved with different degrees. Furthermore, when extending the baseline *F Pyramid* with BS and *E Module*, AP, AP_{50} and AP_{75} in the fifth row in Table IV have improved from 36.9, 58.5, 39.5 to 37.1, 58.6 and 39.9, respectively. Compared with only adding the *E Module*, adding BS achieved 1.7 relative improvement on large objects. At the same time, other metrics have also improved slightly. These results demonstrate the usefulness of the *E Module* for large object detection.

TABLE IV
ABLATION STUDIES OF TFPN ON COCO VAL-17. STARTING WITH OUR RE-IMPLEMENTATION OF FPN (I.E., THE F PYRAMID), WE GRADUALLY ADD THE BOTTOM-UP STRUCTURE (BS), THE TOP-DOWN STRUCTURE (TS) AND THE E MODULE (EM).

Method	FPN	BS	TS	EM	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR_1	AR_{10}	AR_{100}	AR_S	AR_M	AR_L
Baseline	✓				36.3	58.4	39.0	21.2	40.3	46.5	30.5	49.0	51.6	32.6	55.9	64.2
TPS	✓	✓	✓		37.5	58.8	40.6	22.7	41.3	47.1	30.9	49.6	52.3	33.4	56.8	64.3
FPN+	✓			✓	36.9	58.5	39.5	21.7	40.6	47.7	30.9	49.5	52.2	33.4	56.3	65.5
FPN+ (with BS)	✓	✓		✓	37.1	58.6	39.9	21.9	40.5	48.2	30.9	49.3	51.9	32.8	55.8	65.1
TFPN	✓	✓	✓	✓	**38.5**	**59.7**	**42.1**	**23.0**	**42.0**	**49.7**	**31.7**	**50.7**	**53.3**	**34.5**	**57.2**	**66.3**

TFPN. Extending the baseline *F Pyramid* with the *S Pyramid* and *E Module*, we can clearly observe a noticeable AP, AP_{50} and AP_{75} gain from 36.3, 58.4 and 39.0 to 38.5, 59.7, 42.1, as shown in the sixth row in Table IV. There are also significant improvements on the AP and AR metrics for small, medium and large scales. In particular, the largest improvements are from AP_L and AR_L, with 3.2 and 2.1 improvement over the baseline, respectively. These experiments verity the effectiveness of our TFPN design for multi-scale objects.

V. CONCLUSIONS

In this work, we have developed TFPN (Twin Feature Pyramid Networks), a novel framework that optimizes feature pyramid networks for multi-scale object detection. The framework effectively integrates two innovative designs: a twin-pyramid structure (TPS) for medium object detection and a feature enhancement module (FPN+) for large object detection. Extensive results show significant improvement of detection performance for small, medium, and large objects using the same TFPN framework. As such, TFPN provides a practical solution for further optimizing feature pyramid networks. In the future, we plan to apply twin feature pyramid networks to one-stage detectors, such as YOLO, SSD, RetinaNet, and study more effective feature map fusion methods based on FPN.

ACKNOWLEDGMENT

This work is supported in part by the National Key Research and Development Program of China (2016YFB1000100) and the Advanced Research Project of China (31511010203).

REFERENCES

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

[2] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. II

[3] 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. II

[4] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011. III-B, III-C

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. IV-B

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. III-C

[7] Jiaqi Wang Yu Xiong Xiaoxiao Li Shuyang Sun Wansen Feng Ziwei Liu Jianping Shi Wanli Ouyang Chen Change Loy Kai Chen, Jiangmiao Pang and Dahua Lin. mmdetection. https://github.com/open-mmlab/mmdetection. 2018. II, IV-B

[8] Tao Kong, Fuchun Sun, Chuanqi Tan, Huaping Liu, and Wenbing Huang. Deep feature pyramid reconfiguration for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 169–185, 2018. I, 3, II

[9] Zuoxin Li and Fuqiang Zhou. Fssd: feature fusion single shot multibox detector. *arXiv preprint arXiv:1712.00960*, 2017. I

[10] Tsung-Yi Lin, Piotr Dollár, 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*, pages 2117–2125, 2017. 1

[11] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. II

[12] 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. I, II

[13] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018. 2, III-B

[14] 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. II

[15] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 821–830, 2019. I, 4, II, II, IV-C

[16] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration. *PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration*, 6, 2017. IV-B

[17] 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. II

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. II

[19] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. II

[20] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018. I, III