Latent Feature Pyramid Network for Object Detection

Jin Xie, Yanwei Pang, Jing Nie, Jiale Cao, Jungong Han

***Abstract*—Object detection methods based on Convolution Neural Networks (CNN) usually utilize feature pyramid networks to detect objects with various scales. The state-of-the-art feature pyramid networks improve detection accuracy by enhancing multi-level feature representations. Fusing multi-level features is the most effective manner to enhance the feature representations. However, the existing feature pyramid networks usually fuse multi-level features by element-wise operations. It leads to the lack of long-range dependencies in the feature fusion. To address the problem, we propose a simple yet efficient feature pyramid network named latent feature pyramid network (LFPN). LFPN can enhance the feature representations by modeling inner-scale and cross-scale long-range dependencies through conducting inner-scale and cross-scale feature fusion in the latent space. Comprehensive experiments are performed on two challenge object detection datasets: MS COCO and Pascal VOC. The experimental results show consistent improvements on various feature pyramid networks, backbones, and object detectors, which demonstrates the effectiveness and generality of our LFPN.**

***Index Terms*—Object detection, feature pyramid network, long-range dependency, latent space.**

I. Introduction

OBJECT detection [1], [2], [3] is a challenging computer vision problem and serves as an important component in numerous real-world systems. With the development of deep neural networks [4], [5], [6], [7], object detection based on deep neural networks has achieved significant improvements. In real-word scenes, detecting objects with various scales is still a challenging problem.

To address the problem, SSD [8] and MS-CNN [9] propose to build pyramidal feature representations and detect different sized objects with different layers of the pyramid (see Fig. 1(a)). Large-sized objects are usually detected in deep-layer features which comprise rich semantic information and small feature resolution, whereas small-sized objects are detected in shallow layer features with large feature resolution. It has been proved that these methods can improve the detection accuracy of different sized objects. However, the feature representations are not strong enough to detect objects with different scales, *e.g.*, the features in shallow layers lack semantic information and features in deep layers lack detailed information. To solve the problem, several methods [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20] were proposed to enhance the feature representations by fusing multi-level features. Among these methods, FPN [10] is one of the most popular feature pyramid methods (see Fig. 1(b)). It is a top-down feature pyramid network where feature representations with rich semantics of deep layers are combined with high-resolution feature maps of shallow layers to enrich their semantic information. These feature pyramid networks which fuse multi-level features by element-wise operations (*e.g.*, addition, concatenation, element-wise multiplication, and *etc.*) have been proven to improve detection accuracy effectively.

The work of [21] demonstrates that capturing long-range dependencies is of crucial importance in convolution neural networks. Obviously, the element-wise operation is incapable of capturing long-range dependencies. Therefore, it becomes necessary to design an efficient module incorporating long-range dependencies into feature pyramid networks to improve the object detection performance. To capture long-range dependencies in convolution neural networks, several works were proposed [21], [22], [23], [24], [25]. Among these methods, non-local network [21] is able to enhance the feature representations by exploring relations between all positions. GloRe [24] was proposed to model long-range dependencies by projecting feature maps into the interaction space and utilizing a graph convolution on the projected graph. While these methods are effective to model long-range dependencies on the single-level feature representations, they are not scalable enough to model the long-range dependencies across multi-level features.

The relation between objects is an important clue to detect them, and objects with different scales are detected in different level features. Obviously, modeling long-range dependencies on the single-level feature is not enough for the feature pyramid, and capturing long-range dependencies across multi-level feature representations can improve object detection performance. To this end, we propose a latent feature pyramid network (LFPN) which can model long-range dependencies not only in the single-level feature representations but also across multi-level feature representations (see Fig. 1(c)).

In summary, the novelty, contribution, and characteristic of

This work was supported in part by National Key Research and Development Program of China (No. 2018AAA0102800), in part by Tianjin Science and Technology Program (No. 19ZXZNGX00050), in part by China Postdoctoral Science Foundation under Grant 2021M700613, and in part by CAAI-Huawei MindSpore Open Fund. (Corresponding author: Yanwei Pang and Jin Xie.)

J. Xie is with the Tianjin Key Laboratory of Brain-Inspired Intelligence Technology, School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China, and also with the School of Big Data and Software Engineering, Chongqing University, Chongqing 401331, China. (e-mail:jinxie@tju.edu.cn)

Y. Pang, J. Nie, and J. Cao are with the Tianjin Key Laboratory of Brain-Inspired Intelligence Technology, School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China. (e-mail:{pyw, jingnie, connor}@tju.edu.cn)

J. Han is with the School of Computer Science, Aberystwyth University, UK. (E-mail: jungonghan77@gmail.com)

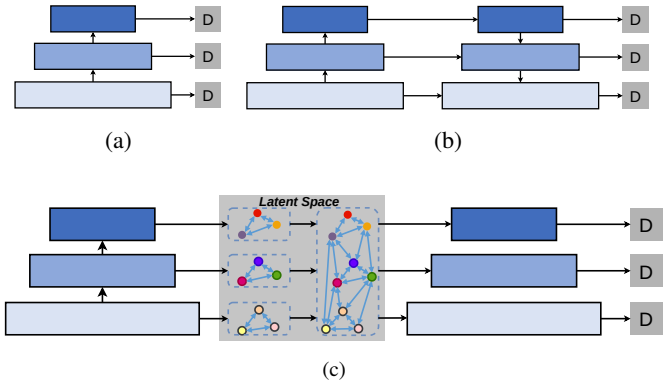


Fig. 1. Comparison of our approach with different feature pyramid networks. (a) Feature pyramid network employed in [8], [9]. (b) Top-down feature pyramid network [10] which combines deep-layer features and shallow-layer features. (c) The proposed latent feature pyramid network (LFPN) conducts feature fusion in the latent space. It can be observed that our LFPN is different from other feature pyramid networks.

the proposed LFPN are as follows.

(1) We propose a simple yet effective feature pyramid network termed as latent feature pyramid network (LFPN) consisting of projection, inner-scale fusion, cross-scale fusion, and reverse-projection.

(2) The projection and reverse-projection are employed to project the feature from grid space to latent space and reverse-project the features from the latent space to the original grid space, respectively. Feature fusion is conducted in the obtained latent space to model the long-range dependencies. The projection and reverse-projection processes are realized by employing a project matrix and a reverse-projection matrix. Both of the two matrices can be updated by end-to-end learning. Thus, the latent space can be optimized by data-driven training.

(3) In the latent space, two efficient feature fusion modules including inner-scale fusion and cross-scale fusion are proposed to enrich contextual information by capturing the long-range dependencies on single-level and cross-level feature representations.

(4) The proposed LFPN has good generality, which can be seamlessly integrated into various object detectors, different backbones, and different feature pyramid structures. In addition, our proposed LFPN improves detection accuracy with negligible computational burdens.

II. Related Work

In recent years, object detection approaches [26], [27], [28], [29], [30], [8], [31] based on the convolution neural networks (CNN) have shown continuous performance improvements. CNN-based detectors can be roughly divided into two categories: anchor-based object detectors [30], [8], [31], [32], [33] and anchor-free object detectors [34], [35], [36], [37]. Anchor-based object detectors can be divided into single-stage methods and two-stage methods. Single-stage detectors [28], [38], [39] usually directly regress the offsets of the anchors and predict object categories. Two-stage detectors [40], [41], [42], [43] first generate proposals and then classify and regress these proposals. Faster RCNN is one of the most popular object detectors. It employs a region proposal network (RPN) to generate region proposals in the first stage. In the second stage, a RoI layer is used to extract fixed sized features for all proposals, and these features go through a network to generate classification scores and regress bounding-box coordinates. Different from the anchor-based detectors, in recent years, anchor-free detectors [34], [35], [36], [37] were proposed to directly predict categories, locations, and scales of objects. For both anchor-based detectors and anchor-free detectors, detecting objects with different scales is still one of the most challenging problems. Feature pyramid network is the most frequently used method to solve the scale problem. Next, we will revisit existing feature pyramid networks.

Feature Pyramid Networks: CNN-based object detection methods usually exploit multi-scale feature pyramid networks [10], [11], [12], [13], [14], [15] to tackle the scale problem in object detection. FPN [10] introduces a top-down path to combine low-resolution feature maps of deep layers and high-resolution feature maps of shallow layers. PAFPN [11] introduces a feature pyramid network consisting of a top-down path and a bottom-up path. BiFPN [16] introduces a repeated and weighed dual-path (top-down path and bottom-up path) feature pyramid network. The work of [15], [13] were proposed to aggregate multi-scale features first and then re-assign features to different scales. Liu *et al.* [20] proposed to aggregate multi-scale features by learning point-wise adaptive spatial weights. AugFPN [19] improves multi-scale feature learning by introducing three components (consistent supervision, residual feature augmentation, and soft RoI selection). The works of [18] and [17] were developed to design feature pyramid networks by the neural architecture search (NAS). SEPC [44] aggregates multi-scale features by the deformable 3-D convolution. Most of these feature pyramid networks aggregate multi-scale features by element-wise operations. It would lead to the lack of long-range dependencies. To solve this problem, we propose to employ the latent space by learning the projection, then conduct feature fusion in the latent space by the graph convolution networks (GCN), and finally reverse-project the features from the latent space to the original grid space.

Modeling Long-range Dependencies: Self-attention mechanisms [45], [21], [23], [46] are usually used to model long-range dependencies. Modeling long-range dependencies has achieved great success in various computer vision tasks [47], [48], [49], [50]. Non-Local neural networks [21] models global context by pixel-level pairwise relations. However, computing pixel-level pairwise relations is time-consuming. Several methods [45], [22] were proposed to accelerate the speed of non-local neural networks. In CCNet [45], long-range dependencies are modeled by using an efficient criss-cross attention module. Several methods [24], [25], [51] utilize the graph neural networks to model long-range dependencies. LatentGNN [25] was proposed to accelerate the graph neural networks by introducing a latent space. LMP [52] is an effective video object detection method to enrich feature representations by capturing the local and relative longer-range relationships. All these models are usually utilized on single-level features, so that long-range dependencies across multi-level features,

which are critical for multi-scale object detection, cannot be captured. In our LFPN, the inner-scale fusion module and the cross-scale fusion module are employed to enhance the feature representations in the latent space by capturing both inner-scale and cross-scale long-range dependencies.

III. Proposed Approach

We propose a Latent Feature Pyramid Network (LFPN) to enhance the feature presentations by mining both inner-scale and cross-scale long-range dependencies. Firstly, it projects features from the grid space to the latent space, then conducts feature fusion in the latent space by the graph convolution networks, and finally reverse-projects features from the latent space back to the original grid space. LFPN is lightweight and easy to implement, and can be seamlessly integrated into various feature pyramid networks, backbones, and object detectors. Our LFPN can improve detection accuracy on different sized object by exploring the relations between different sized objects.

Next, we will review the existing feature pyramid networks and then detail the design of our LFPN.

A. Existing Feature Pyramid Networks

In this subsection, we review existing feature pyramid networks for object detection, and analyze the limitations of them.

Modern object detection networks consist of three parts: the backbone, feature pyramid network, and detection head network. Feature pyramid network located between the backbone and the detection head network is used to fuse multi-level features for detecting different sized objects. We denote the features from the backbone by $X_i \in \mathbb{R}^{H_i \times W_i \times C_i}, i = 1, 2, ..., L$, where H_i and W_i are the height and width of i-th features, L represents the number of feature layers. The output of the feature pyramid network is denoted by: $Y_i = f(X_i)$, where $f(\bullet)$ represents the feature fusion function.

The output of FPN [10] which is one of the most popular feature pyramid networks can be computed as: $Y_i = \sum_{j \geq i}^{L} X_j$. The state-of-the-art feature pyramid networks BiFPN [16] and ASFF [20] introduce adaptive weights to conduct feature fusion, the output of these feature pyramid networks can be computed as: $Y_i = \sum_{j}^{L} S_j \odot X_j$, where S_j represents learnable weights and $\odot$ represents element-wise multiplication. These feature pyramid networks that conduct feature fusion in the grid space achieve remarkable performance in object detection. However, there are limitations in the element-wise fusion. Because convolution operation can only extract features in local regions, the long-range dependencies are deficient. Moreover, element-wise operations still conduct on the local regions, and so the relations between different regions cannot be used to enhance the feature representations by employing such element-wise operations. We find that enhancing features in the feature pyramid by introducing long-range dependencies is an effective manner. Features of different levels have different semantic information and are usually complementary. Inspired by this observation, we enhance the feature representations by capturing long-range dependencies between different level features. To this end, we propose a latent feature pyramid network which enhances feature representations by modeling the inner-scale and cross-scale long-range dependencies. In the next section, we would detail our proposed latent feature pyramid network (LFPN).

B. Latent Feature Pyramid Network

The proposed latent feature pyramid network is shown in Fig. 2. It first projects features into the latent space by learning a projection matrix, then fuses features in the latent space, and finally reverse-projects features from the latent space back to the original grid space. This process is implemented by four modules: projection (PR), inner-scale fusion (ISF), cross-scale fusion (CSF), and reverse-projection (RPR). In the next subsection, we would detail the four parts of our LFPN.

Projection (PR): As described in the Sec. I, conducting feature fusion in the grid space leads to the lack of long-range dependencies. To address this issue, we propose to find the latent space which can model long-range dependencies. For this purpose, we need to learn a projection matrix to map the features $X_i \in \mathbb{R}^{H_i \times W_i \times C_i}$ to a low-dimensional latent space. The projection matrix is denoted by $P_i \in \mathbb{R}^{H_i \times W_i \times D_i}$, where D_i represents the number of dimension in the latent space. Then the input features go through a convolution layer with group normalization and then are reshaped to the size of $(H_i \times W_i) \times C_i$. The projection matrix is reshaped to the size of $D_i \times (H_i \times W_i)$. Next the projected features can be computed as: $Y_i^p = P_i \otimes X_i \in \mathbb{R}^{D_i \times C_i}$, where $\otimes$ stands for matrix multiplication. In our LFPN, we compute the project matrix by using a 2-D convolution operation with group normalization [53]. The size of the convolution kernel is 1×1, and the numbers of input and output channels are C_i and D_i, respectively. The projection is conducted for every level features. At last, we obtain L projected features $Y_i^p \in \mathbb{R}^{D_i \times C_i}, i = 1, 2, .., L$.

Inner-scale fusion (ISF): The input features of the ISF are obtained by the projection module. For each level, the input features in the latent space consist of D_i feature vectors that have C_i channels. Each feature vector is projected from the original grid space. Fig. 2 shows the process of the inner-scale fusion (ISF) module. In the ISF module, we enhance the feature representations by utilizing the graph convolution networks (GCN). The ISF aims at capturing inner-scale relations between different feature vectors through a graph convolution layer. We treat each feature vector as a separate node of $\mathcal{V}_{ISF}$ of a graph $\mathcal{G}_{ISF} = (\mathcal{V}_{ISF}, \mathcal{A}_{ISF})$, where $\mathcal{A}_{ISF}$ denotes the adjacency matrix capturing the neighborhood relationship of different nodes. The graph convolution operation is used to improve the features Y_i^p as follows:

$$\hat{Y_i^p} = (I - A_{ISF}) Y_i^p W_{ISF}, \tag{1}$$

where $\hat{Y_i^p}$ is the enhanced feature, $W_{ISF} \in \mathbb{R}^{C_i \times C_i}$ is the learnable parameter matrix, $A_{ISF} \in \mathbb{R}^{D_i \times D_i}$ is the adjacency matrix, and I is the identity matrix. The matrix $(I - \mathcal{A}_p)$ is used to conduct Laplacian smoothing [54] for propagating the node features over the graph. In our implementation, the skip connection is used to implement it. The adjacency matrix and

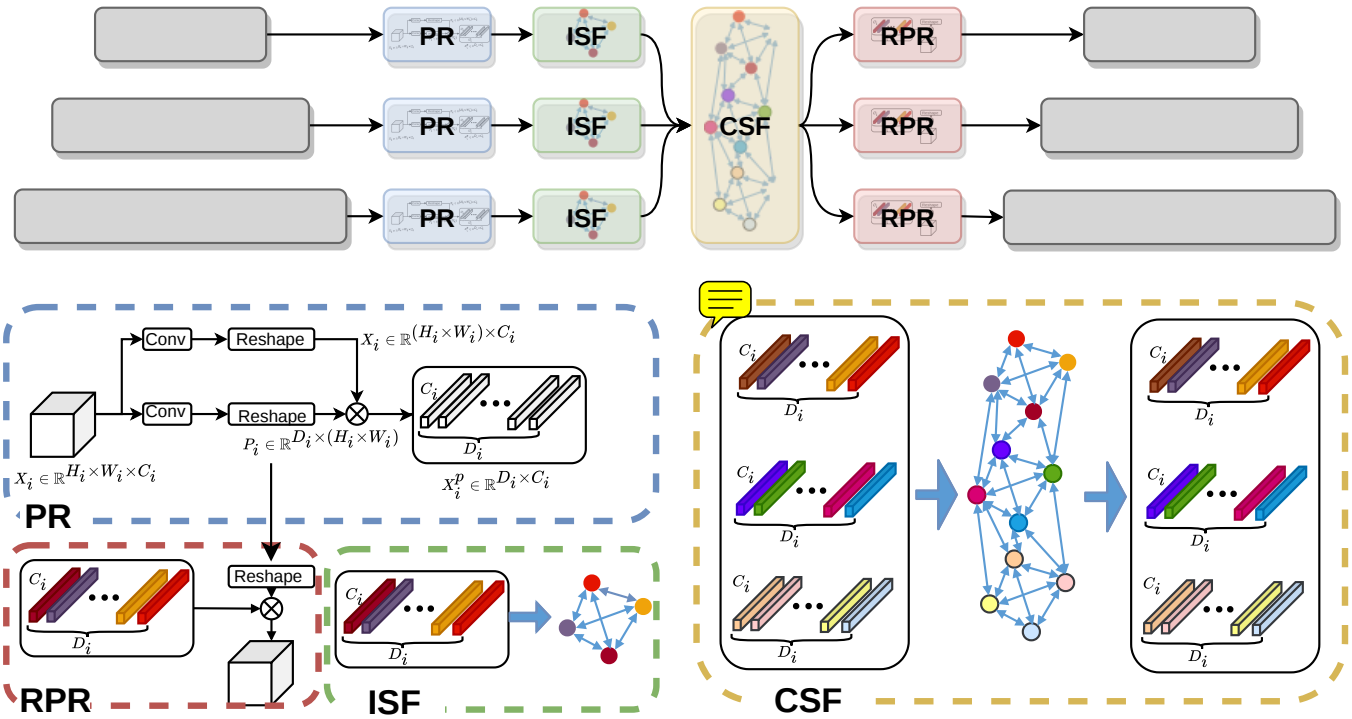


Fig. 2. The overall network architecture of our latent feature pyramid network (LFPN). It comprises four modules: projection (PR), inner-scale fusion (ISF), cross-scale fusion (CSF), reverse-projection (RPR). $\otimes$ represent matrix multiplication. The projection module projects features from the grid space into the latent space. The inner-scale fusion module captures inner-scale long-range dependencies by the graph convolution networks. The cross-scale fusion module enhances feature representations by modeling cross-scale long-range dependencies. The reverse-projection module projects enriched features from the latent space back to the grid space.

the parameter matrix are randomly initialized and updated by end-to-end learning.

Cross-scale fusion (CSF): The cross-scale fusion module is depicted in Fig. 2. The input of the CSF is the enhanced feature $\hat{Y_i^p}$ obtained by the ISF module. First, we concatenate L enhanced features $\hat{Y_i^p}, i = 1, 2, .., L$ and obtain the combined features $Y_{CSF}^p \in \mathbb{R}^{(\sum_l^L D_l) \times C}$. The combined feature representations are enhanced by considering cross-scale long-range dependencies. For instance, considering a scenario where a person uses a mobile phone, due to the objects with different sizes are detected in different level features, the person and the mobile phone are detected in different level features. Intuitively, modeling the relations between the person and the mobile phone can help detect them. Our CSF module aims at capturing long-range dependencies between multi-level features through a graph convolution layer. Similar to the ISF module, the enhanced combined features $\hat{Y_{CSF}^p}$ can be obtained by:

$$\hat{Y_{CSF}^p} = (I - A_{CSF}) Y_{CSF}^p W_{CSF}, \tag{2}$$

where $W_{CSF} \in \mathbb{R}^{C \times C}$ is the learnable parameter matrix, $\mathcal{A}_{CSF} \in \mathbb{R}^{(\sum_l^L D_l) \times (\sum_l^L D_l)}$ is the adjacency matrix of a graph $\mathcal{G}_{CSF} = (\mathcal{V}_{CSF}, \mathcal{A}_{CSF})$, and I is the identity matrix. Analogous to the ISF module, the adjacency matrix and the parameter matrix are end-to-end trainable.

Afterwards, we split the combined features to obtain L enriched features $\tilde{Y_i^p} \in \mathbb{R}^{D_i \times C_i}$ which have the same shape as: $\hat{Y_i^p} \in \mathbb{R}^{D_i \times C_i}$. The resulting enriched features are then utilized as an input to the reverse-projection module.

Reverse-projection (RPR): Here, we reverse-project the features from the latent space back to the original grid space. The reverse-projection matrix $P_i^r \in \mathbb{R}^{(H_i \times W_i) \times D_i}$ is computed by transposing the projection matrix $P_i \in \mathbb{R}^{D_i \times (H_i \times W_i)}$ obtained in the projection module. The reverse-projected features can be computed as: $Y_i^e = P_i^r \otimes \tilde{Y_i^p} \in \mathbb{R}^{(H_i \times W_i) \times C_i}$, where $\otimes$ represents matrix multiplication. Next, we reshape the re-projected features to the size of $H_i \times W_i \times C_i$ and use a 1×1 convolution layer with group normalization [53] to make the spatial size and the channel number of the feature Y_i^e the same as that of the input feature of LFPN X_i.

Finally, we add the enriched feature representation Y_i^e to the input feature representation of LFPN. The output of LFPN can be computed as:

$$Y_i = Y_i^e + X_i, \tag{3}$$

where X_i and Y_i are the input feature representations of LFPN, respectively.

C. Applications of LFPN

Our proposed LFPN is a generic feature pyramid network which can be seamlessly integrated into various feature pyramid networks. Next, we choose FPN [10] as an example to detail the process of integrating the LFPN into other feature pyramid networks.

TABLE I
COMPARISON (IN AVERAGE PRECISION) OF OUR LFPN WITH THE BASELINE (THE SSD [8]) WITH VGG16 [4] ARE CHOSEN AS THE BASELINE). BESIDE OUR FINAL LFPN, WE ALSO SHOW THE PERFORMANCE OF OUR ISF AND CSF ALONE. FOR A FAIR COMPARISON, WE USE THE SAME TRAINING AND TESTING SETTINGS. WE OBSERVE A CONSISTENT IMPROVEMENT IN DETECTION ACCURACY DUE TO PROGRESSIVELY INTEGRATING ISF AND CSF.

	AP	AP_{50}	AP_{75}
Baseline	25.6	43.8	26.3
Baseline + ISF	26.4	45.6	26.5
Baseline + CSF	26.5	46.3	27.0
Our LFPN: Baseline + ISF +CSF	26.8	46.8	27.0

We combine FPN and our proposed LFPN by a sequential arrangement. The output of FPN is taken as the input of the LFPN. The output features Y_i can be computed as:

$$Y_i = f_{LFPN}(f_{FPN}(X_i)), \tag{4}$$

where f_{FPN} and f_{LFPN} represent the feature fusion function of FPN and LFPN, respectively.

IV. EXPERIMENTS

A. Datasets and Evaluation Metrics

Datasets: We perform experiments on two object detection benchmarks: MS COCO [55] and Pascal VOC [56]. MS COCO is a challenging dataset for object detection. It includes 80 object categories and consists of training, validation, and test sets. There are around 118k, 5k, and around 20k images in the training, validation, and test sets, respectively. Pascal VOC is a popular dataset for object detection. It contains 20 categories and includes more than 20k images.

Evaluation Metrics: For the MS COCO dataset, we report the COCO-style Average Precision (AP) consisting of AP (AP with averaged across IoU thresholds from 0.5 to 0.95, with an interval of 0.05), AP_{50} (AP with IoU threshold 0.5), AP_{75} (AP with IoU threshold 0.75), AP_S (AP of small sized objects), AP_M (AP of medium sized objects), and AP_L (AP of large sized objects). For the Pascal VOC dataset, we report results using the mean Average Precision (mAP) with the IoU threshold set to 0.5.

B. Implementation Details

We use a PyTorch [57]-based object detection toolbox MMDetection[1] [58] for all experiments. The network is trained on two NVIDIA GPUs and a mini-batch comprises 2 images per GPU.

For the MS COCO dataset, we perform training on the *train2017* set and report the results on the *val2017* set and the *test-dev* set for comparison, respectively. It is noting that the *test-dev* set is withheld and results are obtained by uploading the evaluation server. If the detector is SSD, we resize the input image as 300×300. For other detectors, we resize the input images to make their shorter edge be 800 pixels and their long edge be less than 1333 pixels, and adopt $1\times$ training scheme.

[1]https://github.com/open-mmlab/mmdetection

TABLE II
ANALYZING THE IMPACTS OF DIFFERENT NUMBERS OF DIMENSION IN THE LATENT SPACE ON THE MS COCO *val2017* SET.

# Dimension	AP	AP_{50}	AP_{75}
64	26.7	46.7	27.0
128	26.8	46.8	27.0
256	26.8	47.0	26.9

TABLE III
COMPARISON (IN RUNNING TIME AND DETECTION ACCURACY) WITH NON-LOCAL NETWORK [21] ON THE MS COCO *val2017* SET. ALL METHODS ADOPT THE SINGLE-STAGE OBJECT DETECTOR RETIANET AND RESNET50.

	Speed (FPS)	AP
FPN [10] + Non-local [21]	8.8	37.0
FPN [10] + LFPN	13.3	37.5

For the Pascal VOC dataset, we perform training on the combined training and validation sets of Pascal VOC 2007 and Pascal VOC 2012. The results are reported on the Pascal VOC 2007 test set. The scale of the input image is set to 600 pixels on the short edge, and the long edge must be less than 1000 pixels. We adopt $2\times$ training scheme for all the experiments on the Pascal VOC dataset.

For both datasets, other training and test settings are set to be the same as in MMDetection [58].

C. MS COCO

The impacts of the ISF and the CSF: Here we analyze the impacts of our ISF module and our CSF module. We choose the SSD [8] with VGG16 as the baseline. Tab. I gives the impacts of integrating our ISF and CSF into the baseline. For a fair comparison, we use the same experimental settings for all experiments in Tab. I. The baseline obtains an accuracy of 25.6% AP. The introduction of the ISF improves the overall detection performance from 25.6% to 26.4% in AP. Similar findings are from the CSF. Our final LFPN combining both the ISF and the CSF provides a significant gain of 1.2% in AP over the baseline. The significant improvement in AP demonstrates that both the ISF module and the CSF module are effective and could improve the detection accuracy. It demonstrates that capturing the long-range dependencies of both of the inner-scale feature representations and the cross-scale feature representations is critical to improve the detection accuracy.

Dimension in Latent Space: Here we analyze the impacts of different numbers of dimension in the latent space. We consider three numbers: 64, 128, and 256. For a fair comparison, except the number of dimension in the latent space, other architectures and experimental settings are kept the same. Tab. II gives the impacts of using different numbers of dimension in the latent space. Considering the trade-off between the detection accuracy and the speed, we choose 128 as the number of dimension in the latent space.

Comparison with the non-local network [21]: Here we conduct an experiment to compare our LFPN and the non-local network. Tab. III gives the results. The non-local network

TABLE IV
COMPARISON (IN TERMS OF AVERAGE PRECISION) WITH DIFFERENT FEATURE PYRAMID NETWORKS ON THE MS COCO *val2017* AND *test-dev* SET. NOTE THAT THE DETECTORS AND THE BACKBONES OF ALL METHODS ARE RETINANET AND RESNET50. OUR LFPN ACHIEVES CONSISTENT IMPROVEMENTS ON ALL OF THE FPN [10], PAFPN [11], AUGFPN [19], AND NAS-FPN [18]

	val2017						*test-dev*					
	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
FPN [10]	36.5	55.4	39.1	20.4	40.3	48.1	36.9	56.2	39.3	20.5	39.9	46.3
FPN [10]+LFPN	37.5	57.3	40.3	21.2	41.2	49.1	37.8	57.9	40.2	21.3	40.6	47.5
PAFPN [11]	37.0	56.3	39.6	21.2	41.0	48.0	37.3	56.8	39.8	21.3	40.6	45.6
PAFPN + LFPN	38.0	57.7	40.7	22.2	42.2	49.3	38.3	58.2	40.8	21.5	41.3	47.5
NAS-FPN [18]	39.3	57.3	41.9	23.0	43.4	50.2	39.2	57.3	42.1	22.2	42.2	48.6
NAS-FPN [18] + LFPN	39.8	57.7	42.6	23.2	43.9	51.5	39.7	57.9	42.5	22.4	43.0	48.7
AugFPN [19]	37.9	57.5	40.6	21.8	41.7	50.1	38.2	58.4	40.8	22.0	41.0	48.1
AugFPN [19]+LFPN	38.5	58.4	41.3	22.5	42.1	51.4	38.7	59.1	41.2	22.0	41.5	48.7

TABLE V
DETECTION ACCURACY WITH DIFFERENT BACKBONES INCLUDING RESNET50, RESNET101 AND RESNEXT32×4D ON THE MS COCO *val2017* SET AND THE *test-dev* SET. FOR A FAIR COMPARISON, THE DETECTORS OF ALL METHODS ARE RETINANET. R50: RESNET50. R101: RESNET101. X101: RESNEXT32×4D. CONSISTENCE IMPROVEMENTS ARE OBTAINED ON VARIOUS BACKBONES.

		val2017						*test-dev*					
Backbone	Feature Pyramid	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
R50 [5]	FPN [10]	36.5	55.4	39.1	20.4	40.3	48.1	36.9	56.2	39.3	20.5	39.9	46.3
R50 [5]	FPN [10]+LFPN	37.5	57.3	40.3	21.2	41.2	49.1	37.8	57.9	40.2	21.3	40.6	47.5
R101 [5]	FPN [10]	38.5	57.6	41.0	21.7	42.8	50.4	39.0	58.6	41.7	21.9	42.2	49.3
R101 [5]	FPN [10] + LFPN	39.7	59.5	42.6	23.2	43.5	51.7	40.0	60.3	42.8	22.6	43.2	50.5
X101 [6]	FPN [10] [10]	39.9	59.6	42.7	22.3	44.2	52.5	40.4	60.3	43.1	23.0	43.7	50.6
X101 [6]	FPN [10] + LFPN	40.7	61.3	43.2	24.5	45.1	53.3	41.2	61.9	44.3	24.3	44.7	50.9

is added behind each level features of FPN [10]. For a fair comparison, all methods in Tab. III employ the RetinaNet and ResNet50 as the detector and backbone, respectively. Moreover, the experimental settings are kept the same. It can be observed that compared with the non-local network, our LFPN obtains superior detection accuracy with faster speed. In the next, we would detail the reason as follows.

(1) The non-local network can only captures the long-range dependencies on the single scale features. On the contrary, our LFPN can model long-range dependencies on the inner-scale and cross-scale feature representations, simultaneously. The long-range dependencies on the cross-scale feature representations explore the relationship between multi-scale objects to improve the detection accuracy of different sized objects.

(2) The non-local network needs to compute the similarity between every pixels, which costs a lot of computational costs. On the contrary, our LFPN projects the features into the latent space from the grid space. The dimension in the latent space is smaller than the number of pixels, which leads that the computation complexity of our LFPN is smaller than the non-local network. Therefore, our LFPN is faster than the non-local network.

Results on different feature pyramid networks: Here, we combine our proposed latent feature pyramid network (LFPN) with different feature pyramid networks. The results are given in Tab. IV. For a fair comparison, all the methods utilize the same object detector (RetinaNet [33]) and the same backbone (ResNet50 [5]). Compared with FPN [10] that is one of the most popular feature pyramid methods, combining our LFPN and FPN obtains gains of 1.0% and 0.9% in AP on the *val2017* set and the *test-dev* set, respectively. Compared with a bidirectional feature pyramid network (PAFPN) [11], our LFPN boosts the AP score from 37.0% to 38.0% and obtains a 1.0% AP improvement for small sized objects, 1.2% AP improvement for medium sized objects, and 1.3% AP improvement for large sized objects on the *val2017* set, which demonstrates the effectiveness of our LFPN. The network architecture search (NAS)-based method NAS-FPN [18] achieves AP scores of 39.3% and 39.2% on the *val2017* set and the *test-dev* set, respectively. Our LFPN with NAS-FPN provides a superior detection accuracy compared with NAS-FPN with AP scores of 39.8% and 39.7% on the *val2017* and *test-dev* sets, respectively. (The detection accuracy of NAS-FPN is slightly below in the original paper, the reason is that the data augmentation is not used.) Compared with the Aug-FPN [19], our LFPN can obtain absolute gains of 0.6% and 0.5% on the *val2017* and *test-dev* set, respectively. In addition, our LFPN can improve the detection accuracy with all sized objects. It can be concluded that our proposed LFPN achieves consistent improvements on different feature pyramid networks.

TABLE VI
DETECTION ACCURACY WITH DIFFERENT BACKBONES INCLUDING SWIN-TRANSFORMER-TINY [59] (SWIN-T) AND RESNET50 WITH GCNET (R50-GCNET) [22] ON THE MS COCO *val2017* SET. FOR A FAIR COMPARISON, ALL METHODS ADOPT THE SAME SINGLE-STAGE OBJECT DETECTOR RETINANET. OUR METHOD PROVIDES CONSISTENCE IMPROVEMENTS ON VARIOUS BACKBONES.

Backbone	Feature Pyramid	AP
Swin-T [59]	FPN	40.5
Swin-T [59]	FPN+LFPN	41.2
R50-GCNet [22]	FPN	38.2
R50-GCNet [22]	FPN+LFPN	38.8

Results on different backbones: Here, we evaluate the effects

TABLE VII
COMPARISON (IN TERMS OF AVERAGE PRECISION) WITH OTHER DETECTORS ON THE MS COCO *val2017* SET AND THE *test-dev* SET. FOR A FAIR COMPARISON, WE USE THE SAME EXPERIMENTAL SETTINGS WHEN COMPARING WITH EACH METHOD. OUR LFPN ACHIEVES CONSISTENT IMPROVEMENTS ON VARIOUS OBJECT DETECTORS. NOTE THAT THE BACKBONES OF ALL METHODS ARE RESNET50.

	val2017						*test-dev*					
	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet [33]	36.5	55.4	39.1	20.4	40.3	48.1	36.9	56.2	39.3	20.5	39.9	46.3
RetinaNet+LFPN	37.5	57.3	40.3	21.2	41.2	49.1	37.8	57.9	40.2	21.3	40.6	47.5
FreeAnchor [60]	38.7	57.3	41.5	21.0	42.0	51.3	38.9	57.8	41.6	21.3	41.5	49.0
FreeAnchor + LFPN	39.4	59.1	41.9	23.0	43.2	50.6	39.9	59.8	42.7	22.6	42.8	49.1
GuidedAnchoring [61]	37.0	56.8	39.4	21.0	40.2	50.4	36.6	56.4	39.2	19.6	39.0	46.9
GuidedAnchor + LFPN	38.0	58.1	40.7	22.0	41.6	51.1	38.4	58.9	41.1	21.4	41.2	48.8
SABL [62]	37.7	56.2	40.4	21.4	41.6	49.4	38.0	56.7	40.4	21.1	41.1	47.4
SABL + LFPN	38.9	58.0	41.5	22.5	42.8	50.6	39.1	58.6	41.7	22.1	42.1	48.8
PISA [63]	36.9	55.7	39.9	20.5	40.4	48.7	37.4	56.5	40.3	20.3	40.4	47.3
PISA+ LFPN	38.1	57.7	41.2	22.0	41.8	50.5	38.4	58.3	41.4	21.6	41.3	48.4
Faster RCNN [30] w FPN	37.4	58.1	40.4	21.2	41.0	48.1	37.7	58.7	40.8	21.7	40.6	46.7
Faster RCNN w FPN+LFPN	38.7	60.4	41.9	23.5	42.5	49.0	38.7	60.6	41.8	22.6	41.7	47.3
Cascade RCNN [31] w FPN	40.3	58.6	44.0	22.5	43.8	52.9	40.6	59.2	44.0	23.0	43.4	51.1
Cascade RCNN w FPN+LFPN	41.0	60.3	44.6	23.6	45.0	52.9	41.4	60.8	44.9	24.2	44.4	51.5

TABLE VIII
COMPARISON OF THE PROPOSED LFPN WITH OTHER FEATURE PYRAMID NETWORKS IN TERMS OF RUNNING TIME AND MODEL PARAMETERS. FOR A FAIR COMPARISON, THE DETECTORS OF ALL METHODS ARE RETINANET, AND THE SPEED OF ALL METHODS IS MEASURED ON A SINGLE NVIDIA QUADRO P6000 GPU. R50: RESNET50. R101: RESNET101. X101: RESNEXT32×4D.

Backbone	Feature Pyramid	# Par.(M)	Speed(FPS)	AP
R50	FPN [10]	37.7	14.0	36.5
R50	FPN [10]+LFPN	38.8	13.3	37.5
R101	FPN [11]	56.7	10.8	38.5
R101	FPN [10]+LFPN	57.8	10.5	39.7
X101	FPN [11]	56.4	8.2	39.9
X101	FPN [10]+LFPN	57.5	8.1	40.7

of different backbones on our LFPN in terms of average precision. We conduct experiments on three different backbones: ResNet50 [5], ResNet101 [5], and ResNeXt32×4d [6]. For a fair comparison, we choose the same object detector (RetinaNet) and the same feature pyramid network (FPN) as the baseline. We report the results in Tab. V. Our LPFN can obtain absolute gains of 1.0%, 1.2%, and 0.8% on ResNet50, ResNet101 and ResNeXt32×4d, respectively. It can be observed that our LFPN can be successfully used in various backbones.

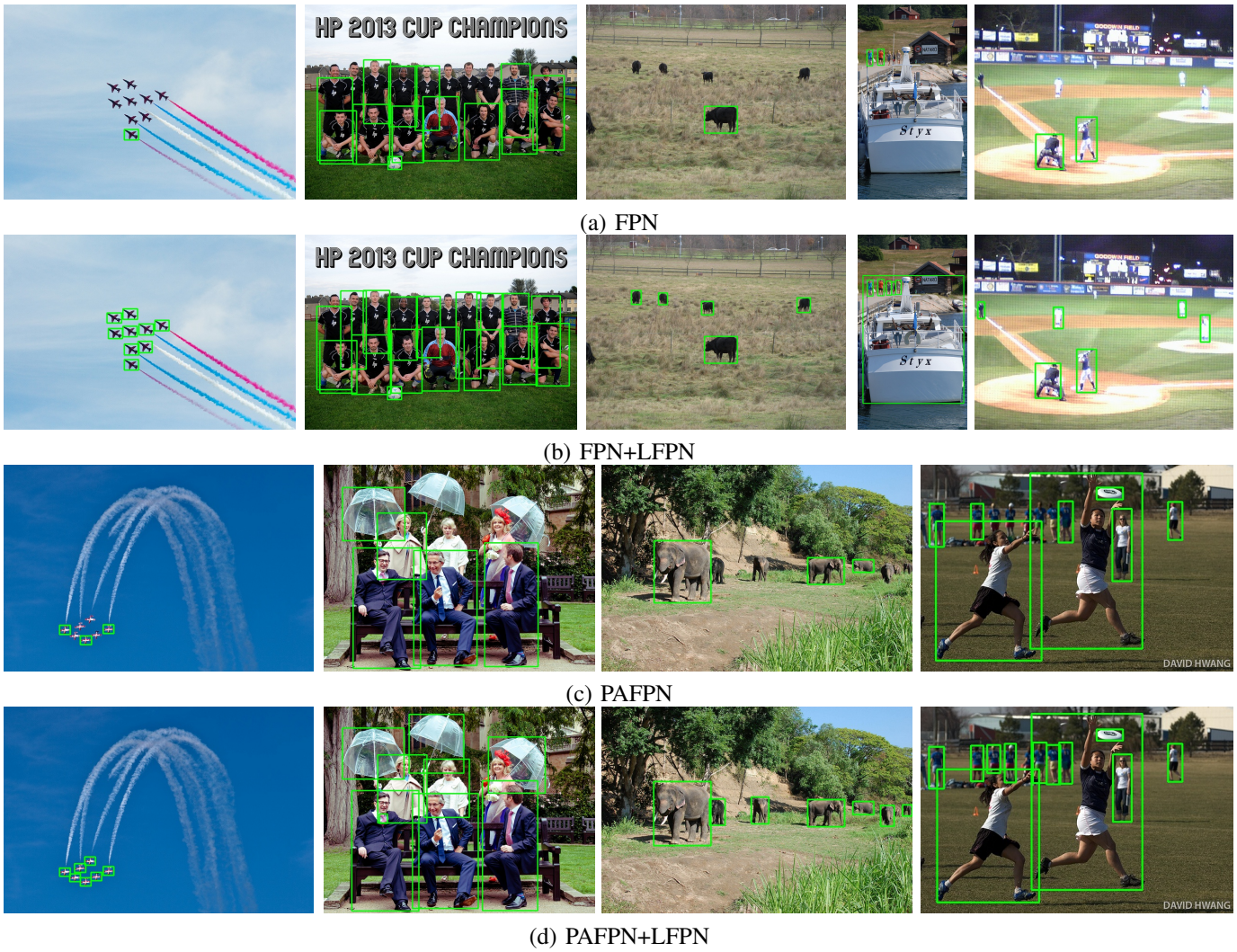
In addition, two self-attention mechanisms-based backbones (Swin-Transformer [59] and GCNet [22]) are set to be the backbones to validate the effectiveness of our LFPN. The results are given in Tab. VI. Swin-Transformer is a recently proposed transformer-based method which could capture the long-range dependencies. GCNet is a light-weight network which can model the global context. It can be observed that our LPFN can obtain absolute gains of 0.7% and 0.6% on Swin-Transformer-Tiny and ResNet with GCNet, respectively. Although Swin-Transformer and GCNet can model the long-range dependencies, our LFPN can still improve the detection accuracy. The main reason is that our LFPN can capture the long-range dependencies across multi-level feature representations to further improve the detection accuracy.

Results on different detectors: We integrate our LFPN into various different detectors including RetinaNet [33], FreeAnchor [60], Guided Anchoring [61], SABL [62], and PISA [63]. We download the models of these detectors from MMDetection Model Zoo[2] and test them on the *val2017* set and the *test-dev* set, respectively. The results are given in Tab. VII. All results are obtained by utilizing ResNet50 and FPN. GuidedAnchoring [61] where predefined dense anchors are replaced with predicted anchors, obtains AP scores of 37.0% and 36.6% on the *val2017* set and the *test-dev* set, respectively. Integrating our LFPN into GuidedAnchoring obtains 1.0% and 1.8% AP improvements on the *val2017* set and the *test-dev* set, respectively. Side-Aware Boundary Localization (SABL) [62] where a novel bounding box regression manner is employed, achieves AP scores of 37.7% and 38.0% on the *val2017* set and the *test-dev* set, respectively. Our LFPN with SABL provides a superior detection accuracy of AP scores of 38.9% and 39.1% compared with SABL on the *val2017* and *test-dev* sets, respectively. In addition, compared with SABL, we observe that our LFPN with SABL obtains consistent gains of 1.1%, 1.2%, and 1.2% on small, medium and large sized objects. PrIme Sample Attention (PISA) [63] is aimed at selecting prime samples in training process and obtains AP scores of 36.9% and 37.4% on the *val2017* and *test-dev* sets, respectively. Our approach combined with PISA outperforms the PISA with AP scores of 38.1% and 38.4% on the *val2017* and *test-dev* sets, respectively.

Moreover, our LFPN can be integrated into two two-stage object detectors (Faster RCNN [30] and Cascade RCNN [31]). Faster R-CNN with FPN achieves an AP score of 37.4% and 37.7% on the *val2017* and *test-dev* sets, respectively. Faster RCNN with our LFPN obtains an absolute improvement of 1.3% and 1.0%, respectively. Cascade R-CNN with FPN achieves AP scores of 40.3% and 40.6% on the *val2017* and *test-dev* sets, respectively. Cascade R-CNN with our LFPN obtains superior results with AP scores of 41.0% and 41.4% on the *val2017* and *test-dev* sets, respectively.

We find that integrating our LFPN into different detectors

[2]https://github.com/open-mmlab/mmdetection/blob/master/docs/model_zoo.md



(a) FPN

(b) FPN+LFPN

(c) PAFPN

(d) PAFPN+LFPN

Fig. 3. Qualitative comparison on the MS COCO *test-dev*. For a fair comparison, the backbone and detector are ResNet50 and RetinaNet, respectively. Green boxes indicate detector prediction.

leads to significant improvements by around 1% on both *val2017* and *test-dev* sets. In addition, our LFPN can improve the detection accuracy from small sized objects to large sized objects. It can be concluded that integrating our LFPN into different detectors can improve the detection performance of all detectors. The experimental results demonstrate the effectiveness and the generality of our proposed LFPN.

Speed and model parameters: We also evaluate the speed and the model size of our LFPN. Three backbones including ResNet50, ResNet101 and ResNeXt32×4d are chosen to conduct this experiment. The experimental results are reported in Tab. VIII. For a fair comparison, all results are obtained on a single NVIDIA Quadro P6000 GPU using the same test settings. The detection performance in Tab. VIII is tested on the *val2017* set. When using a strong backbone (ResNeXt32×4d), our LFPN boosts the AP score from 39.9% to 40.7% while only reduces the speed by 0.1 FPS and increases 1.1M parameters. Therefore, it can be concluded that the proposed LFPN results in better detection accuracy at the cost of slight computational burden. In addition, when the backbone is ResNet50, the inference time of the whole detector is 75.2 milliseconds(ms), where the LFPN costs 6.8 ms. It demonstrates that our LFPN takes only a fraction of inference time. In addition, we provide the inference time of each part in LFPN. Specifically, the inner-scale fusion module (ISF) takes 2.7 ms, and the cross-scale fusion module (CSF) takes 0.5 ms.

Qualitative comparison: Fig. 3 shows the qualitative detection comparisons between existing feature pyramid networks (*e.g.*, FPN and PAFPN) and our LFPN combining with these feature pyramid networks. It can be observed that our LFPN can stably improve the detection results from small sized objects to large sized objects.

D. PASCAL VOC

We conduct experiments on the popular object detection dataset PASCAL VOC. For a fair comparison, all experiments use the same settings. Tab. IX gives the comparison of our LFPN with other feature pyramid networks in terms of detection accuracy. The popular feature pyramid network FPN [10] with ResNet50 achieves a detection accuracy mAP of 79.6%.

TABLE IX
OBJECT DETECTION PERFORMANCE COMPARISON ON THE PASCAL VOC 2007 TEST SET. FOR ALL METHODS, WE USE THE SAME EXPERIMENTAL SETTINGS FOR A COMPARISON. OUR LFPN ACHIEVES CONSISTENT IMPROVEMENT IN mAP OVER POPULAR FEATURE PYRAMID NETWORK FPN [10] AND PAFPN [11]. IN ADDITION, OUR LFPN OBTAINS CONSISTENT GAINS ON BOTH RESNET50 AND RESNET101.

Detector	Backbone	Feature Pyramid	mAP
RetinaNet [33]	ResNet50	FPN [10]	79.6
RetinaNet [33]	ResNet50	FPN+LFPN	80.8
RetinaNet [33]	ResNet50	PAFPN [11]	79.9
RetinaNet [33]	ResNet50	PAFPN+LFPN	81.1
RetinaNet [33]	ResNet101	FPN [10]	80.7
RetinaNet [33]	ResNet101	FPN+LFPN	81.5
RetinaNet [33]	ResNet101	PAFPN [11]	80.8
RetinaNet [33]	ResNet101	PAFPN+LFPN	81.6

TABLE X
ANALYZING THE IMPACTS OF ISF MODULE AND CSF MODULE ON THE PASCAL VOC 2007 TEST SET.

Methods	mAP
Baseline	79.6
Baseline + CSF	80.5
Baseline + ISF	80.1
Our LFPN: Baseline +ISF +CSF	80.8

Integrating our LFPN into FPN with ResNet50 provides superior detection accuracy with a mAP score of 80.8%. When using the same backbone ResNet101, compared with FPN, our LFPN boosts the detection accuracy mAP from 80.7% to 81.5%. Similarly, a consistent improvement in detection accuracy is obtained over the popular feature pyramid network: PAFPN [11] with both ResNet50 and ResNet101, when using the same settings. Experimental results on the Pascal VOC dataset demonstrate the effectiveness and the generality of our LFPN.

Ablation study on Pascal VOC: Here, we conduct the ablation studies on the Pascal VOC dataset to validate the effectiveness of our proposed modules. We analyze the impacts of the ISF module and the CSF module and the repeated LFPNs.

The impacts of ISF and CSF: Here, we analyze our LFPN on the Pascal VOC dataset by demonstrating the impacts of integrating our proposed ISF module and CSF module. Tab. X gives the detection results. The baseline method is RetinaNet with ResNet50-FPN. For a fair comparison, we use the same experimental settings for all experiments. Our proposed CSF module improves the detection performance by 0.9% in mAP. A gain (0.5%) in mAP is achieved when integrating the ISF module. Further, our LFPN that integrates both ISF module and CSF module achieves an absolute gain of 1.2% in terms of mAP over the baseline. It demonstrates that both the ISF module and the CSF module are effective and could improve the detection accuracy.

Performance on repeated LFPNs: Here we conduct an experiment by utilizing repeated LFPNs. The LFPN repeated by 3 times achieves a mAP score of 80.8% on the Pascal VOC 2007 test set, which does not bring improvements on detection accuracy. It can be concluded that the repeated LFPNs are useless to improve the detection performance and our LFPN is strong enough with no need to stack it repeatedly.

V. CONCLUSION

We have proposed a latent feature pyramid network (LFPN) for object detection that can be integrated into any feature pyramid network seamlessly. Our LFPN projects features from the grid space to the latent space, then conducts inner-scale and cross-scale feature fusion in the latent space through the graph convolution networks (GCN), and finally reverse-projects the features from the latent space back to the grid space. Our LFPN can improve the detection accuracy by modeling the long-range dependencies on inner-scale and cross-scale feature representations. Experiments on two challenging detection datasets clearly demonstrate that our approach can improve the detection accuracy on all scales of objects with negligible loss of the inference speed.

REFERENCES

[1] H. Qiu, H. Li, Q. Wu, F. Meng, L. Xu, K. N. Ngan, and H. Shi, "Hierarchical context features embedding for object detection," *IEEE Trans. Multimedia*, 2020.
[2] L. Chen, H. Zheng, Z. Yan, and Y. Li, "Discriminative region mining for object detection," *IEEE Trans. Multimedia*, 2020.
[3] S. Wu, Y. Xu, B. Zhang, J. Yang, and D. Zhang, "Deformable template network (dtn) for object detection," *IEEE Trans. Multimedia*, 2021.
[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Adv. Neural Inform. Process. Syst.*, 2014.
[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.
[6] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.
[7] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.
[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[9] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Eur. Conf. Comput. Vis.*, 2016.
[10] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection." in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.
[11] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.
[12] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, "Ron: Reverse connection with objectness prior networks for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.
[13] T. Kong, F. Sun, C. Tan, H. Liu, and W. Huang, "Deep feature pyramid reconfiguration for object detection," in *Eur. Conf. Comput. Vis.*, 2018.
[14] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Parallel feature pyramid network for object detection," in *Eur. Conf. Comput. Vis.*, 2018.
[15] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn: Towards balanced learning for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.
[16] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.
[17] H. Xu, L. Yao, W. Zhang, X. Liang, and Z. Li, "Auto-fpn: Automatic network architecture adaptation for object detection beyond classification," in *Int. Conf. Comput. Vis.*, 2019.
[18] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Nas-fpn: Learning scalable feature pyramid architecture for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[19] C. Guo, B. Fan, Q. Zhang, S. Xiang, and C. Pan, "Augfpn: Improving multi-scale feature learning for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[20] S. Liu, D. Huang, and Y. Wang, "Learning spatial fusion for single-shot object detection," *arXiv:1911.09516*, 2019.

[21] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[22] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," *arXiv preprint arXiv:1904.11492*, 2019.

[23] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "A^2-nets: Double attention networks," in *Adv. Neural Inform. Process. Syst.*, 2018.

[24] Y. Chen, M. Rohrbach, Z. Yan, Y. Shuicheng, J. Feng, and Y. Kalantidis, "Graph-based global reasoning networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2019.

[25] S. Zhang, S. Yan, and X. He, "LatentGNN: Learning efficient non-local relations for visual recognition," 2019.

[26] R. Girshick, "Fast R-CNN," in *Int. Conf. Comput. Vis.*, 2015.

[27] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014.

[28] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Adv. Neural Inform. Process. Syst.*, 2015.

[31] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[32] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[33] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Int. Conf. Comput. Vis.*, 2017.

[34] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Eur. Conf. Comput. Vis.*, September 2018.

[35] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," in *arXiv preprint arXiv:1904.07850*, 2019.

[36] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Int. Conf. Comput. Vis.*, 2019.

[37] T. Kong, F. Sun, H. Liu, Y. Jiang, and J. Shi, "Foveabox: Beyond anchor-based object detector," *arXiv preprint arXiv:1904.03797*, 2019.

[38] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[39] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[40] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Adv. Neural Inform. Process. Syst.*, 2016.

[41] X. Lu, B. Li, Y. Yue, Q. Li, and J. Yan, "Grid r-cnn," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[42] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, "D2det: Towards high quality object detection and instance segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[43] H. Zhang, H. Chang, B. Ma, N. Wang, and X. Chen, "Dynamic R-CNN: Towards high quality object detection via dynamic training," *arXiv preprint arXiv:2004.06002*, 2020.

[44] X. Wang, S. Zhang, Z. Yu, L. Feng, and W. Zhang, "Scale-equalizing pyramid convolution for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[45] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Int. Conf. Comput. Vis.*, 2019.

[46] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[47] P. Yi, Z. Wang, K. Jiang, J. Jiang, and J. Ma, "Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations," in *Int. Conf. Comput. Vis.*, 2019.

[48] Z. Li, J. Tang, and T. Mei, "Deep collaborative embedding for social image understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2070–2083, 2019.

[49] Z. Li, J. Tang, L. Zhang, and J. Yang, "Weakly-supervised semantic guided hashing for social image retrieval," *International Journal of Computer Vision*, no. 2, 2020.

[50] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Int. Conf. Comput. Vis.*, 2019.

[51] Y. Li and A. Gupta, "Beyond grids: Learning graph representations for visual recognition," in *Adv. Neural Inform. Process. Syst.*, 2018.

[52] Z. Zhu and Z. Li, "Online video object detection via local and mid-range feature propagation," in *Proceedings of the 1st International Workshop on Human-Centric Multimedia Analysis*, 2020, p. 73–82.

[53] Y. Wu and K. He, "Group normalization," in *Eur. Conf. Comput. Vis.*, 2018.

[54] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," *CoRR*, vol. abs/1801.07606, 2018. [Online]. Available: http://arxiv.org/abs/1801.07606

[55] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Eur. Conf. Comput. Vis.*, 2014.

[56] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[57] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[58] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[59] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Int. Conf. Comput. Vis.*, 2021.

[60] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, "FreeAnchor: Learning to match anchors for visual object detection," in *Adv. Neural Inform. Process. Syst.*, 2019.

[61] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[62] W. Jiaqi, Z. Wenwei, C. Yuhang, C. Kai, P. Jiangmiao, G. Tao, S. Jianping, L. C. Change, and L. Dahua, "Side-aware boundary localization for more precise object detection," in *Eur. Conf. Comput. Vis.*, 2020.

[63] Y. Cao, K. Chen, C. C. Loy, and D. Lin, "Prime sample attention in object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

Jin Xie received the Ph.D degree in information and communication engineering from the Tianjin University, Tianjin, China, in 2021. He is currently an associate professor at the Chongqing University. His research interests include machine learning and computer vision, in which he has published 10 papers in CVPR, ICCV, ECCV, IEEE TPAMI, IEEE TIP, IEEE TCSVT, and IEEE TCYB.

Yanwei Pang received the Ph.D. degree in electronic engineering from the University of Science and Technology of China in 2004. Currently, he is a professor at the Tianjin University, China and is also the founding director of the Tianjin Key Laboratory of Brain Inspired Intelligence Technology (BIIT), China. His research interests include object detection and image recognition, in which he has published 150 scientific papers including 40 IEEE Transactions papers and 30 top conferences (*e.g.*, CVPR, ICCV, and ECCV) papers. He is an associate editor of both IEEE T-NNLS and Neural Networks (Elsevier) and a guest editor of Pattern Recognition Letters. He is a senior member of the IEEE.

Jing Nie received the B.S. degree in communication engineering from Tianjin University, Tianjin, China, in 2017, where she is currently pursuing the Ph.D. degree under the supervision of Prof. Y. Pang. Her research interests include object detection and image enhancement.

Jiale Cao received the Ph.D. degree in information and communication engineering from the Tianjin University, Tianjin, China, in 2018. He is currently an associate professor at the Tianjin University. His research interests include object detection and deep learning, in which he has published 20 papers in CVPR, ICCV, ECCV, IEEE TIP, IEEE TCSVT and IEEE TIFS.

Jungong Han is a Full Professor and the Chair in Computer Science with Aberystwyth University, Aberystwyth, U.K. He has published over 180 articles, including more than 40 IEEE TRANSACTIONS and more than 40 A* conference papers. His research interests span the fields of video analysis, computer vision, and applied machine learning.