LADet:A Light-weight and Adaptive Network for Multi-scale Object Detection

Jiaming Zhou ENCOREZJM@BUAA.EDU.CN
Yuqiao Tian TIANYUQIAO@BUAA.EDU.CN
Weicheng Li LIWEICHENG@163.COM
Rui Wang WANGRUI@BUAA.EDU.CN
Zhongzhi Luan ZHONGZHI.LUAN@BUAA.EDU.CN
Depei Qian DEPEIQ@BUAA.EDU.CN
School of Computer Science and Engineering, Beihang University, Beijing 100191, China

Editors: Jiaming Zhou

Abstract

Scale variation is one of the most significant challenges for object detection task. In comparison with previous one-stage object detectors that simply make feature pyramid network deeper without consideration of speed, we propose a novel one-stage object detector called LADet, which consists of two parts, Adaptive Feature Pyramid Module(AFPM) and Light-weight Classification Function Module(LCFM). Adaptive Feature Pyramid Module generates complementary semantic information for each level feature map by jointly utilizing multi-level feature maps from backbone network, which is different from the top-down manner. Light-weight Classification Function Module is able to exploit more type of anchor boxes without a dramatic increase of parameters because of the utilization of interleaved group convolution. Extensive experiments on PASCAL VOC and MS COCO benchmark demonstrate that our model achieves a better trade-off between accuracy and efficiency over the comparable state-of-the-art detection methods.

Keywords: Object detection, Feature pyramid, Scale variation

1. Introduction

Scale variation is one of the most significant challenges for object detection task. In recent years, deep neural networks have achieved great success Ren et al. (2015); Zhao et al. (2018); Redmon and Farhadi (2018) in object detection. Recent object detectors with high accuracy can be categorized into two-stage detectors and single-stage detectors. The two-stage detectors firstly generate a sparse set of proposals, and then each proposal is further classified and regressed. The one-stage detectors detect objects by regular and dense sampling over locations, scales and aspect ratios.

The main advantage of one-stage approach is its high computational efficiency while the detection accuracy is often lower than the two-stage detectors. Previous study shows that the higher-level semantic features and denser scale coverage are keys of improving the accuracy of one-stage detectors which detect objects by enumerating a large number of anchor boxes. Therefore, Some recent methods suggest that feature pyramidal representations and careful anchor boxes design can boost the performance of one-stage approach.

Feature pyramid approach exploits different scale feature maps to detect multi-scale objects. To the best of our knowledge, the Single Shot Detector (SSD) Liu et al. (2016) is one of the first at-

tempts on utilizing such an approach. SSD directly generates predictions from multiple feature maps with different scales to naturally handle objects of various sizes. However, the low-level semantic information from the shallow feature maps limit the ability of classification and regression of SSD. To compensate the absence of semantics in shallow feature map, Scale-Transferrable Detection Network(STDN) Zhou et al. (2018) exploit scale-transfer module to the last block of DenseNet Huang et al. (2017) to generate high-resolution feature maps with higher-level semantic information. FPN Lin et al. (2017a), DSSD Fu et al. (2017) and RON Kong et al. (2017) augment a top-down pathway and lateral connections to incorporate strong semantic information in high-level features. However, only the low-level feature map can benefit from the top-down pathway with stronger semantic information. High-level feature maps have large valid receptive field because of the large downsampling factors, which is good for image classification but restricts the object location ability. M2Det Zhao et al. (2018) firstly uses Feature Fusion Module to fusing feature maps of the backbone and then generates a group of multi-scale features through a set of Thinned U-shape Modules. In addition, SFAM aggregates the features into the multi-level feature pyramid. Although M2Det obtains strong semantic information from different scale feature maps, the intensive computation compromises the efficiency of the detector.

Most of the previous state-of-the-art one stage detectors are based on enumerating anchor boxes, which need careful design. A set of anchor boxes are usually chosen by handcraft Ren et al. (2015) or statistical methods Redmon and Farhadi (2017, 2018) like clustering. MetaAnchor Yang et al. (2018) takes advantage of weight prediction to obtain a dynamic anchor function, which is more robust in comparison with anchor settings. In order to cover objects of different shapes, the predefined anchor boxes should have multiple aspect ratios. YOLOv3 Redmon and Farhadi (2018) exploits 3 anchor box with different aspect ratios at each pyramid feature map for detection while RetinaNet Lin et al. (2017b) has 9 anchors for denser scale coverage. However, the more anchor boxes we use, the more dramatic increase of parameters in anchor functions, especially if the number of classes is large.

In order to address the previous issues without impairing the efficiency of the detectors, we develop an Adaptive Feature Pyramid Module(AFPM) and Light-weight Classification Function Module(LCFM). On one hand, AFPM is designed to overcome the disadvantage of the top-down pathway structure and enable the multi-level feature fusion to be more flexible. Breaking the traditional top-down pathway structure, AFPM is a new attempt to build feature pyramid network and has the potential to generate multi-level feature maps with higher-level semantic information. In AFPM, we first rescale the multi-level feature pyramids extracted from a backbone network to the same resolution and then feed them into a channel-wise attention module to extract complementary semantic information separately, which will be added to the original feature pyramids. On the other hand, to utilize to feature maps generated by AFPM for multi-scale objects detection, we develop LCFM and it is composed of two group convolution layers, which have structure-sparse kernels. An intermediate permutation operation is exploited between the two convolution in order to approximate the original dense or high-rank kernel. The basic motivation of LCFM is to reconstruct the classification function with a sequence of sparse kernels, which can reduce the number of parameters.

To evaluate the effectiveness of AFPM and LCFM, we embedded these two modules into RefineDet Zhang et al. (2018) to yield a novel end-to-end one-stage object detector we call LADet. The proposed network achieves remarkable performance with an mAP of 81.4% on PASCAL VOC benchmark and mmAP of 33.6% on MS COCO benchmark at speed of 20.8 FPS.

The main contributions of this paper are summarized as follows:

- We propose the Adaptive Feature Pyramid Module to enhance the feature pyramids by generating complementary semantic information;
- We propose the Light-weight Classification Function Module which can be exploited to reduce the number of parameters of classification subnet;
- LADet achieves a better trade-off between accuracy and speed over the comparable state-of-the-art detectors.

2. Related Work

Previous works have paid much attention to scale variation problem of object detection and researchers have made great efforts on it. As for feature map, feature pyramid structure are widely used to embed high-level semantic information into multi-scale feature maps. To further utilize the enhanced feature maps, many state-of-the-art one-stage detectors use more anchor boxes to produce predictions of classification scores and bounding boxes.

2.1. Feature Pyramid

Feature pyramid is one of the tactics to deal with scale variation. In the task of object detection, it is wildly acknowledged that keeping the valid receptive fields in consistence with object sizes is the key to improving the detection accuracy. However, the valid receptive fields are different according to the depth of feature maps and fixed in a certain layer when the neural network is specified. Therefore, the feature pyramid network take advantage of the multi-level feature maps to detects objects of different scales. SSD Liu et al. (2016) and MR-CNN Gidaris and Komodakis (2015) directly utilize multi-scale feature maps from different layers. FPN Lin et al. (2017a), YOLOv3 Redmon and Farhadi (2018), RetinaNet Lin et al. (2017b) and RefineDet Zhang et al. (2018) fuse the deep and shallow layers in a top-down manner nearby to enhance the semantic information of the low-level feature maps. M2Det Zhao et al. (2018) augments a Thin U-shape module to yield feature maps with different scales and then feeds them into the detection subnet.

2.2. Anchor box

Most of the previous works model anchor boxes via enumeration. After, one-stage detectors detect objects by regular and dense sampling on strong semantic feature maps which are extracted from backbone network and feature pyramid network. Taking the different valid receptive fields of multi-level feature maps into consideration, SSD uses anchors at three aspect ratios at each pyramid level. For denser scale coverage, RetinaNet adds anchors of sizes $\{1, 2^{1/3}, 2^{2/3}\}$ of the original set of 3 aspect ratio anchors at each level. However, the number of anchor boxes have negative impacts on the number of parameters and computational efficiency due to the enumeration strategy. To address this issue, YOLOv2 and YOLOv3 propose a statistical method like clustering which is good for choosing high-quality anchor boxes. RetinaNet reduces the number of parameters by sharing weights across all pyramid levels at classification subnet and box regression subnet.

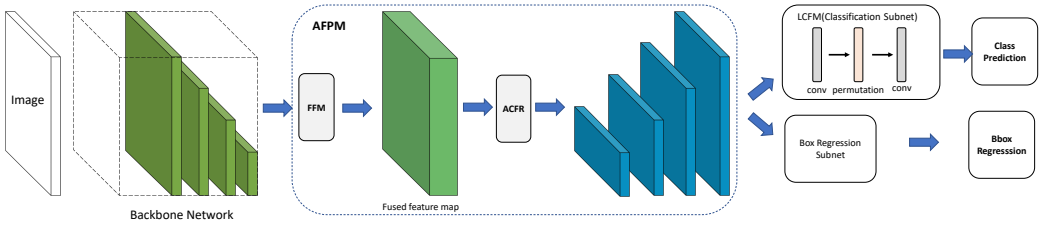


Figure 1: An overview of our proposed model. Multi-level feature maps extracted from the backbone network are fed into AFPM to generate pyramid feature map with complementary semantic information, and then the LCFM and the box regression subnet produce classification scores and dense bounding boxes based on the enhanced feature maps.

3. Proposed Method

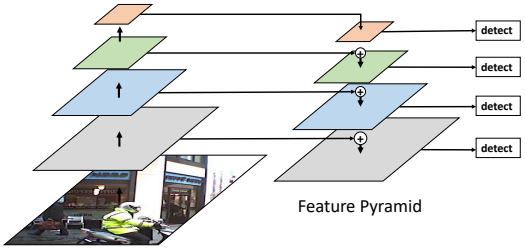
The overall architecture of the proposed method is shown in Figure 1. Firstly, multi-level feature maps are extracted from the backbone network and then fed into AFPM to refine the feature maps by generating complementary semantic information. Finally, dense bounding boxes and classification scores are produced based on the learned feature maps, followed by the non-maximum suppression(NMS) to produce the final results. AFPM is consist of two steps, i.e. Feature Fusion and Adaptive Channel-wise Feature Refinement. Feature Fusion applies scale normalization operation across all levels to obtain super-resolution feature maps, which will be concatenated together. Adaptive Channel-wise Feature Refinement yields attention maps for each pyramid level via fully-connected layers. LCFM is composed of two group convolution layers, which have structure-sparse kernels. An intermediate permutation operation is exploited between the two convolution in order to approximate the original dense or high-rank kernel. More details about the two core modules and network configurations are introduced in the following.

3.1. Adaptive Feature Pyramid Module

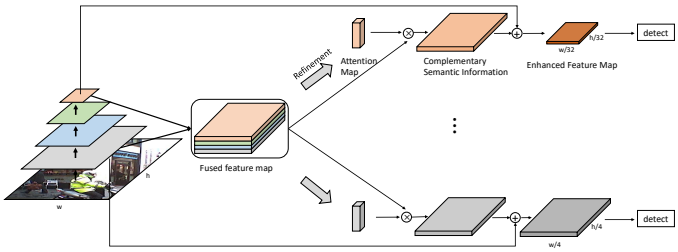
Scale variance is one of the fundamental challenges of object detection task. Most of the previous feature pyramid structure is based on the top-down pathway, which is beneficial to the low-level feature maps but not the high-level feature maps. A drawback of this approach is its unidirectional flow of semantic information while the high-level feature maps need detail semantic information to alleviate the problem of blurry boundary for large object localization. To address this problem, we propose AFPM to generate complementary semantic information from the fusion feature map to enhance the semantic information for each pyramid level. The comparison between top-down style and AFPM is shwon Figure 2.

Feature Fusion Module. To fuse the feature maps across all pyramid levels, we have to normalize the multi-level feature maps with same resolution. Inspired by the super-resolution approach Shi et al. (2016), we develop a scale normalization operation to rescale pyramid feature maps with different sample factor. Supposed that the shape of the input feature map is $(C \cdot r^2) \times H \times W$, where r is the sample factor. Scale normalization is an operation of periodic rearrangement of elements at the same spatial space of r^2 channels, which is mathematically formulated as follows:

$$FM^{HR}_{c,y,x} = FM^{LR}_{\left\lfloor \frac{c}{r^2} \right\rfloor, y+\left\lfloor \frac{mod(c,r^2)}{r} \right\rfloor, x+mod(mod(c,r^2),r)} \tag{1}$$



(*a*) top-down style



(*b*) AFPM

Figure 2: Structure Comparison between top-down style and AFPM.

Where FM^{HR} are high-resolution feature maps and FM^{LR} are low-resolution feature maps. Compared with up-sampling operation, the output feature maps only have $1/r^2$ times of orginal channels. Otherwise, the number of channels will dramatically increase after concatenation. To fuse the super-resolution feature maps, we concatenate the multiple outputs of scale normalization.

Adaptive Channel-wise Feature Refinement. The next step is to generate complementary semantic information for each pyramid level, which is called Adaptive Channel-wise Feature Refinement. At each level, We firstly feed the fused feature map $ffm_{c,x,y}$ to a global average pooling layer and Z_c^l denotes the output tensor. The channel-wise attention map AM_c^l is produced by two fully-connected layers. Eq.4 will generate the complementary semantic information via a channel-wise multiplication between AM_c^l and $ffm_{c,x,y}$, and $fr_{c,x,y}^l$ denotes the final enhanced feature map.

$$Z_c^l = GAP(ffm_{c,x,y}) \tag{2}$$

$$AM_c^l = \sigma(W_2^l \delta(W_1^l Z_c^l)) \tag{3}$$

$$fr_{c,h,w}^l = AM_c^l \otimes ffm_{c,x,y} \tag{4}$$

Let l denotes the pyramid level, the two fully-connected layers W_2^l and W_1^l are implemented by two $1 * 1$ convolutional layers with batch normalization, δ refers to the ReLU function, σ is the sigmoid activation, $\otimes$ denotes the channel-wise multiplication. Finally, several convolution layers which kernel sizes are $3 * 3$ and strides are set to 2 will be applied to the $fr_{c,h,w}^l$ for down-sampling, followed by an element-wise addition with the original feature map at lenvel l.

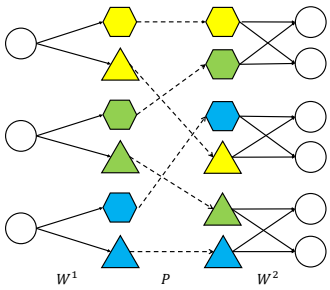


Figure 3: Illustration of Light-weight Classification Function Module

3.2. Light-weight Classification Function Module

The classification function module predicts the probability of object presence at each spatial position for each of the anchor boxes and object classes. For the purpose of improving the detection accuracy, one-stage detectors use multiple anchors boxes at each spatial position to cover boxes of various scales and aspect ratios. With more anchor boxes are utilized, the average IoU between predefined anchor boxes and ground truth bounding boxes grows higher, which is beneficial to training the detectors. The classification function module is a $k * k$ conv layer with $C * A$ filters attached to the pyramid feature maps. Here C denotes the number classes, A denotes the number of anchor boxes and k refers to the kernel size. Therefore, the more anchor boxes we use, the more dramatic increase of parameters in anchor functions, especially if the number of classes is large.

To overcome this problem, we propose a Light-weight Classification Function Module to build a lightweight and efficient module. Ideally, the increase of parameters and computational complexity should be smooth when the number of anchor boxes is growing. Additionally, the new module should keep dense connectivity between input and output channels, which can increase the expressive capacity of networks. Bearing the above two demands in mind, we take advantage of Thin Feature Map Li et al. (2017) and Interleaved Low-Rank Group Convolutions Sun et al. (2018) to design the LCFM. As shown in Figure 3, we divide the $3*3$ conv layer into two group convs, whose kernel sizes are $1 * 3$ and $3 * 1$. The LCFM can be formulated as follows:

$$y = W^2 P W^1 x \tag{5}$$

where $x = [\mathrm{x}_1^c, \mathrm{x}_2^c, \cdots, \mathrm{x}_{classes}^c]$ is the input feature map, $x_i^c \in \mathbb{R}^{\frac{c_{in}}{classes} \times H \times W}$. The classes prediction output is denoted as $y \in \mathbb{R}^{A \times classes \times H \times W}$ and P refers to the permutation matrices. Two structured-sparse kernels($W^1 \in \mathbb{R}^{(A \times classes) \times \frac{c_{in}}{classes} \times 1 \times 3}$ and $W^2 \in \mathbb{R}^{(A \times classes) \times A \times 3 \times 1}$) are mathematically formulated as:

$$W^i = \begin{bmatrix} W_1^i & 0 & 0 & 0 \\ 0 & W_2^i & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & W_{classes}^i \end{bmatrix} \tag{6}$$

Here W_g^i(i=1 or 2) denotes the kernel matrix over the corresponding channels in the gth branch. This method adopts two sparse kernel as an approximation of the original dense or high-rank kernel, which can reduce parameters of classification function module and improve computational efficiency.

Method	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster Ren et al. (2015)	VGG-16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
ION Bell et al. (2016)	VGG-16	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87.0	54.4	80.6	73.8	85.3	82.2	82.2	74.4	47.1	75.8	72.7	84.2	80.4
Faster He et al. (2016)	Residual-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
MR-CNN Gidaris and Komodakis (2015)	VGG-16	78.2	80.3	84.1	78.5	70.8	68.5	88.0	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85.0	76.4	48.5	76.3	75.5	85.0	81.0
R-FCN Dai et al. (2016)	Residual-101	80.5	79.9	87.2	81.5	72.0	69.8	86.8	88.5	89.8	67.0	88.1	74.5	89.8	90.6	79.9	81.2	53.7	81.8	81.5	85.9	79.9
SSD300 Liu et al. (2016)	VGG-16	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5	77.0	86.1	87.5	83.9	79.4	52.3	77.9	79.5	87.6	76.8
SSD512 Liu et al. (2016)	VGG-16	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79.0	86.6	80.0
DSSD321 Fu et al. (2017)	Residual-101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5	78.7	86.7	88.7	86.7	79.7	51.7	78.0	80.9	87.2	79.4
DSSD513 Fu et al. (2017)	Residual-101	81.5	86.6	86.2	82.6	74.9	62.5	89.0	88.7	88.8	65.2	87.0	78.7	88.2	89.0	87.5	83.7	51.1	86.3	81.6	85.7	83.7
STDN300 Zhou et al. (2018)	DenseNet-169	78.1	81.1	86.9	76.4	69.2	52.4	87.7	84.2	88.3	60.2	81.3	77.6	86.6	88.9	87.8	76.8	51.8	78.4	81.3	87.5	77.8
STDN321 Zhou et al. (2018)	DenseNet-169	79.3	81.2	88.3	78.1	72.2	54.3	87.6	86.5	88.8	63.5	83.2	79.4	86.1	89.3	88.0	77.3	52.5	80.3	80.8	86.3	82.1
STDN513 Zhou et al. (2018)	DenseNet-169	80.9	86.1	89.3	79.5	74.3	61.9	88.5	88.3	89.4	67.4	86.5	79.5	86.4	89.2	88.5	79.3	53.0	77.9	81.4	86.6	85.5
RefineDet512 Zhang et al. (2018)∗	VGG-16	79.7	88.2	86.1	79.3	74.1	69.2	88.0	87.6	88.1	64.6	86.5	72.1	84.3	88.6	85.6	83.1	49.8	80.2	76.5	86.2	76.4
LADet	VGG-16	80.4	87.5	86.2	80.4	74.9	69.4	87.3	88.7	88.8	64.4	86.4	75.2	84.6	87.4	85.3	84.0	52.7	80.6	78.9	87.7	77.4
LADet	DenseNet-169	81.4	87.6	85.5	81.4	73.5	71.6	87.2	88.9	88.8	67.8	86.4	77.0	85.7	89.1	88.0	84.8	54.8	82.6	81.4	87.0	78.9

Table 1: Detection results on PASCAL VOC2007 dataset. $*$ refers to our implementation.

3.3. Network Configuration

We embed our proposed methods into RefineDet to yield a novel one-stage object detector called LADet. We choose DenseNet169 and VGG16 as the backbone networks. As for DenseNet-169, we append a max pooling layer and two conv layers to the end of the basic network and extract four feature at different scales, including dense block[2,3,4] and the last output feature map. As for VGG-16, we adopt the modification according to RefineDet. Firstly, we convert fc6 and fc7 of VGG-16 to convolution layers conv fc6 and conv fc7 via subsampling parameters. Since conv4_3 and conv5_3 have different feature scales compared to other layers, we use L2 normalization to scale the feature norms in conv4_3 and conv5_3 to 10 and 8, then learn the scales during backpropagation. The input size is set to 512×512. We replace the TCB module of RefineDet to our propoased AFPM, which is aimed to generate multi-level feature maps. As for detection, the classification subnet is replaced by LCFM and the bounding box regression subnet remains the same.

4. Experiments

In this section, we introduce the implementation details and experiment setup first. And then we present the detection result on PASCAL VOC and MS COCO dataset. Finally, we explore the effectiveness of AFPM and LCFM.

4.1. Implementation Details and Experiment Setup

Experiments are conducted on three datasets: PASCAL VOC 2007, PASCAL VOC 2012 and MS COCO. The PASCAL VOC Everingham et al. (2010) and MS COCO Lin et al. (2014) datasets include 20 and 80 object categories respectively. For PASCAL VOC, LADet is trained on the PASCAL VOC2007 and PASCAL VOC2012 trainval sets and tested on the VOC 2007 test set. For MS COCO, following the protocol in MS COCO, we use the trainval35k set for training, which is a union of 80k images from train split and a random 35 subset of images from the 40k image val split. To compare with state-of-the-art methods, we report COCO AP on the test-dev split. We conduct the training process on 4 NVIDIA 1080TI GPUs, CUDA 8.0 and CUDNN 7.0. the batch size is set to 32(8 each for 4 GPUs). Before training the whole network, the backbone networks are supposed to be pre-trained on the ImageNet 2012 dataset Russakovsky et al. (2015). By default, we start training with warm-up strategy for 600 iterations, initialize the learning rate as 2×10^{-3}, and decay it to 2×10^{-4} and 2×10^{-5} at 300 epochs and 350 epochs, and stop at 400 epochs. At each

Method	Train Data	Input Size	Backbone	FPS	Avg. Precision, IoU: 0.5	0.75	0.5:0.95	Avg. Precision, Area: S	M	L
two-stage										
OHEM++ Shrivastava et al. (2016)	trainval	$\sim 1000 \times 600$	VGG-16	7.0	45.9	26.1	25.5	7.4	27.7	40.3
Faster Ren et al. (2015)	trainval	$\sim 1000 \times 600$	VGG-16	7.0	42.7	-	21.9	-	-	-
R-FCN Dai et al. (2016)	trainval	$\sim 1000 \times 600$	ResNet-101	9.0	51.9	-	29.9	10.8	32.8	45.0
CoupleNet Zhu et al. (2017)	trainval35k	$\sim 1000 \times 600$	ResNet-101	8.2	54.8	37.2	34.4	13.4	38.1	50.8
one-stage										
RON384 Kong et al. (2017)	trainval	384×384	VGG-16	15.0	49.5	27.1	27.4	-	-	-
SSD300 Liu et al. (2016)	trainval35k	300×300	VGG-16	43.0	43.1	25.8	25.1	6.6	25.9	41.4
SSD512 Liu et al. (2016)	trainval35k	512×512	VGG-16	22.0	48.5	30.3	28.8	10.9	31.8	43.5
SSD513 Fu et al. (2017)	trainval35k	513×513	ResNet-101	-	50.4	33.1	31.2	10.2	34.5	49.8
DSSD321 Fu et al. (2017)	trainval35k	321×321	ResNet-101	9.5	46.1	29.2	28.0	7.4	28.1	47.6
DSSD513 Fu et al. (2017)	trainval35k	513×513	ResNet-101	5.5	53.3	35.2	33.2	13.0	35.4	51.1
STDN300 Zhou et al. (2018)	trainval	300×300	DenseNet-169	41.5	45.6	29.4	28.0	7.9	29.7	45.1
STDN513 Zhou et al. (2018)	trainval	513×513	DenseNet-169	28.6	51.0	33.6	31.8	14.4	36.1	43.4
CornerNet Law and Deng (2018)	trainval35k	512×512	Hourglass	4.4	57.8	45.3	40.5	20.8	44.8	56.7
RetinaNet Lin et al. (2017b)	trainval35k	∼832×500	ResNet-101	11.1	53.1	36.8	34.4	14.7	38.5	49.1
M2Det Zhao et al. (2018)	trainval35k	512×512	ResNet-101	15.8	59.4	41.7	38.8	20.5	43.9	53.4
RefineDet Zhang et al. (2018)	trainval35k	512×512	VGG-16	22.3	54.5	35.5	33.0	16.3	36.3	44.3
LADet	trainval35k	512×512	VGG-16	20.8	58.3	39.2	33.6	16.6	37.8	45.2

Table 2: Detection results on MS COCO test-dev

pyramid level we use anchors at three aspect ratios $\{1:2, 1:1, 2:1\}$. We implement LADet in Pytorch[1] and code is available at *https://github.com/nob87208/LADet*.

4.2. Pascal VOC

Table 1 shows our results on the PASCAL VOC2007 dataset. The results of RefineDet512 are produced by our implementation and other statistical results stem from references.

We compare our methods with state-of-the-art detectors. It is noteworthy that, our proposed detector with backbone DenseNet169 achieves 81.4% mAP, surpassed most one-stage detector, e.g., SSD512, STDN513, RefineDet512, etc. We use RefineDet as our baseline because our detectors are based on RefineDet architecture. When using the VGG16 as backbone, the accuracy of our proposed detector is 0.7% higher than that of RefineDet512. Especially, the improvement of accuracy reaches 1.4% in comparison with the original RefineDet512 when using the DenseNet169 as backbone. The extra 0.5% improvement is contributed by AFPM and LCFM when we compare the results between STDN513 and LADet(DenseNet-169 backbone). The accuracy of M2Det surpass LADet on MS COCO, but we do not compare our detectors with M2Det on PASCAL VOC2007 because the results are not made public and the source code is unaccessible. What's more, the accuracy is slightly lower than DSSD513 when the backbone is DenseNet-169. We think the reason may be that Resdidual-101 has more network parameters than DenseNet-169, and thus the basic feature maps have stronger semantic information. However, this may decrease the detection efficiency as well.

4.3. MS COCO

To further validate our approach, we test our detectors on MS COCO benchmark and results are summarized in Table 2. It is important to note that the FPS statistics stem from references Zhao

1. https://pytorch.org/

Method	Backbone	Neck	input size	mAP	FPS
RetinaNet	DenseNet169	fpn	∼1000 ×600	73.7	10.1
	DenseNet169	afpm	∼1000×600	74.1	7.4
RefineDet	VGG16	tcb	512×512	79.7	22.3
	VGG16	afpm	512×512	80.4	20.9

Table 3: Effect of Adaptive Feature Pyramid Module

et al. (2018). Similarly, we focus on the comparison with RefineDet512. Our proposed method is 0.6% better than RefineDet512 and achieves better detection accuracies for the objects of all scales. However, M2Det achieves a higher AP than our detector but its efficiency is 5 FPS slower. Generally, LADet achieves a better trade-off between accuracy and efficiency when speed is more important.

4.4. The Exploration of Submodule

Since LADet consists of two modules Adaptive Feature Pyramid Module and Light-weight Classification Function Module, we need to evaluate each of its effectiveness to the overall performance.

4.4.1. EFFECT OF ADAPTIVE FEATURE PYRAMID MODULE

In order to demonstrate the effectiveness of AFPM, we replace the original feature pyramid structures of RetinaNet and RefineDet to AFPM. The results of Table 3 are produced by test the detectors on PASCAL VOC2007, with a single NVIDIA 1080TI GPU and the batch size is 1. To reduce the impact of input size, input images are resized to $\sim 1000 \times 600$ for ReinaNet-based detectors and 512×512 for RefineDet-based detectors. It can be seen from lines 1 and 2 that mAP increases 0.4% for RetinaNet. Additionally, we observe a noticeable mAP gain from 79.7% to 80.4% for RefineDet. What's more, AFPM boost the accuracy of both ReinaNet-based detectors and RefineDet-based detectors without impairing the efficiency.

4.4.2. EFFECT OF LIGHT-WEIGHT CLASSIFICATION FUNCTION MODULE

Similarly, we conduct experiments on RetinaNet and RefineDet to validate the effectiveness of Light-weight Classification Function Module. To the best of our knowledge, previous works use $3 * 3$ conv layer Lin et al. (2017b) or $1 * 1$ conv layer Redmon and Farhadi (2018) to predicts the probability of object. Thus, we compare LCFM to these two classification function module on PASCAL VOC2007 benchmark. As shown in the third column in Table 4, we notice that the original classification function module of RetinaNet has 35.5 times parameters than those of LCFM. Especially, LCFM has no harm to the accuracy of RetinaNet though it has much fewer number of parameters. Furthermore, we apply different number of anchor boxes to RetinaNet and RefineDet. Comparing the results in lines 8, 10 and 12 of Table 5, we can notice that the accuracy increases from 79.9% to 80.7%, which verifies that denser scale coverage is keys of improving the accuracy of one-stage detectors. What's more, taking results in lines 1 and 2 as an example, LCFM can achieve the same accuracy as the original classification function module. Especially, we observe a slightly mAP gain from 79.9% to 80.1% in comparison with lines 7 and 8. We think the reason may be that fewer parameters avoid the detector from overfitting.

Method	Function Module	Backbone	Params(times)	mAP
RetinaNet	3*3 conv	DenseNet-169	$\times$ 35.5	73.7
RetinaNet	1*1 conv	DenseNet-169	$\times$ 4	72.5
RetinaNet	lcfm	DenseNet-169	$\times$ 1	73.4

Table 4: Comparison between LCFM and other classification function module. Benefit from LCFM, we reduce the number of parameters without impairing accuracy.

Method	Backbone	Anchor type	function module	Anchor boxes	mAP
RetinaNet	DenseNet-169	1 sc * 3 ar	ori	16320	71.5
	DenseNet-169	1 sc * 3 ar	lcfm	16320	71.4
	DenseNet-169	2 sc * 3 ar	ori	32640	72.8
	DenseNet-169	2 sc * 3 ar	lcfm	32640	72.3
	DenseNet-169	3 sc * 3 ar	ori	48960	73.7
	DenseNet-169	3 sc * 3 ar	lcfm	48960	73.4
RefineDet	VGG16	1 sc * 3 ar	ori	16320	79.7
	VGG-16	1 sc * 3 ar	lcfm	16320	79.9
	VGG-16	1 sc * 5 ar	ori	32640	79.9
	VGG-16	1 sc * 5 ar	lcfm	32640	80.1
	VGG-16	1 sc * 7 ar	ori	48960	80.0
	VGG-16	1 sc * 7 ar	lcfm	48960	80.7

Table 5: Effect of Multiple anchor boxes setting. *1 sc $*$ 3 ar* means we use 1 anchor box scale and at each scale has 3 aspect ratios for each pyramid level.

4.5. Conclusions

In this work, we propose LADet to address the problem of scale variance in the task of object detection. Compared with the traditional top-down pathway structure, AFPM has the ability to enhance the semantic information for each pyramid feature map in a nonlinear way, which is more flexible. Our study provides a different way to generate pyramid feature maps, which has the potential to build a better detector to deal with the scale variation problem. To further improve the multi-scale object detection accuracy, we exploit more anchor boxes with the usage of LCFM. At 20.8 FPS, our proposed network get 81.4% mAP on PASCAL VOC2007 and 33.6% AP on MS COCO, achieving a better trade-off between accuracy and efficiency over the comparable state-of-the-art detection methods.

Acknowledgments

This work was supported by National Key R&D Program of China under grant No.2017YFB0202202 and by NSFC under grant No.61732002 and 61202425.The corresponding author of this paper is Rui Wang.

References

Sean Bell, C Lawrence Zitnick, Kavita Bala, and Ross Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2874–2883, 2016.

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

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2): 303–338, 2010.

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

Spyros Gidaris and Nikos Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1134–1142, 2015.

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.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Tao Kong, Fuchun Sun, Anbang Yao, Huaping Liu, Ming Lu, and Yurong Chen. Ron: Reverse connection with objectness prior networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5936–5944, 2017.

Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018.

Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017.

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.

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, 2017a.

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, 2017b.

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.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

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.

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

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016.

Ke Sun, Mingjie Li, Dong Liu, and Jingdong Wang. Igcv3: Interleaved low-rank group convolutions for efficient deep neural networks. *arXiv preprint arXiv:1806.00178*, 2018.

Tong Yang, Xiangyu Zhang, Zeming Li, Wenqiang Zhang, and Jian Sun. Metaanchor: Learning to detect objects with customized anchors. In *Advances in Neural Information Processing Systems*, pages 318–328, 2018.

Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4203–4212, 2018.

Qijie Zhao, Tao Sheng, Yongtao Wang, Zhi Tang, Ying Chen, Ling Cai, and Haibin Ling. M2det: A single-shot object detector based on multi-level feature pyramid network. *arXiv preprint arXiv:1811.04533*, 2018.

Peng Zhou, Bingbing Ni, Cong Geng, Jianguo Hu, and Yi Xu. Scale-transferrable object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 528–537, 2018.

Yousong Zhu, Chaoyang Zhao, Jinqiao Wang, Xu Zhao, Yi Wu, and Hanqing Lu. Couplenet: Coupling global structure with local parts for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4126–4134, 2017.