Efficient Featurized Image Pyramid Network for Single Shot Detector

Yanwei Pang[1]*, Tiancai Wang[1]*, Rao Muhammad Anwer[2], Fahad Shahbaz Khan[2,3], Ling Shao[2]

[1]School of Electrical and Information Engineering, Tianjin University
[2]Inception Institute of Artificial Intelligence, UAE
[3]Computer Vision Laboratory, Department of Electrical Engineering, Linköping University, Sweden
[1]{pyw,wangtc}@tju.edu.cn, [2]{rao.anwer,fahad.khan,ling.shao}@inceptioniai.org

Abstract

Single-stage object detectors have recently gained popularity due to their combined advantage of high detection accuracy and real-time speed. However, while promising results have been achieved by these detectors on standard-sized objects, their performance on small objects is far from satisfactory. To detect very small/large objects, classical pyramid representation can be exploited, where an image pyramid is used to build a feature pyramid (featurized image pyramid), enabling detection across a range of scales. Existing single-stage detectors avoid such a featurized image pyramid representation due to its memory and time complexity. In this paper, we introduce a light-weight architecture to efficiently produce featurized image pyramid in a single-stage detection framework. The resulting multi-scale features are then injected into the prediction layers of the detector using an attention module. The performance of our detector is validated on two benchmarks: PASCAL VOC and MS COCO. For a 300×300 input, our detector operates at 111 frames per second (FPS) on a Titan X GPU, providing state-of-the-art detection accuracy on PASCAL VOC 2007 testset. On the MS COCO testset, our detector achieves state-of-the-art results surpassing all existing single-stage methods in the case of single-scale inference.

1. Introduction

Generic object detection is one of the fundamental problems in computer vision, with numerous real-world applications in robotics, autonomous driving and video surveillance. Recent advances in generic object detection have been largely attributed to the successful deployment of convolutional neural networks (CNNs) in detection frameworks. Generally, deep object detection approaches can be roughly divided into two categories: two-stage [13, 14, 16, 29] and single-stage detectors [19, 28, 5]. In two-stage approaches, object proposals are first generated and later classified and regressed. Single-stage approaches, on the other hand, directly regress the default anchors into detection boxes by sampling grids on the input image. Single-stage object detectors are generally computationally efficient but inferior in detection accuracy compared to their two-stage counterparts [18].

Among single-stage methods, the Single Shot Multibox Detector (SSD) [28] has recently been shown to provide an optimal tradeoff between speed and detection accuracy. The standard SSD utilizes a VGG-16 architecture as the base network and adds further convolutional (conv) feature layers to the end of the truncated base network. In SSD, independent predictions are made by layers of varying resolutions, where shallow or former layers contribute to predicting small objects while deep or later layers are devoted to detecting large objects. Despite its success, SSD struggles to handle large scale variations across object instances. In particular, the detection performance of SSD on small objects is far from satisfactory [18], which is likely due to the limited information in shallow or former layers.

Multiple solutions have been proposed in the literature to alleviate the issues stemming from scale variations. Feature pyramid is an essential component in many recognition systems, forming the basic ground for a standard solution [1]. Building feature pyramids from image pyramids (featurized image pyramids) has long been pursued and employed in many classical hand-crafted methods [11, 9]. Modern deep object detectors also typically employ some forms of pyramid representation, even though the CNNs used in these approaches are robust to scale variation.

For two-stage methods, earlier works [29, 13] advocated the use of single scale features (see Fig. 1(c)). In contrast, recent two-stage methods [24] have investigated feature pyramid to obtain more accurate detection (see Fig. 1(b)).

*Equal contribution

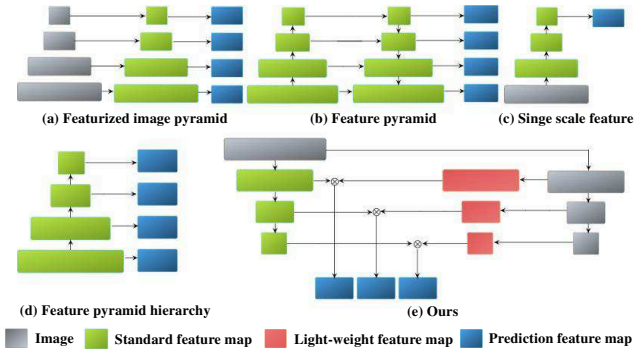


Figure 1. Comparison of our approach with different architectures for multi-scale object detection. (a) Image pyramid for building feature pyramid where features are constructed from images of various scales independently. (b) Feature pyramid network employed in [24] combining features in a layer by layer top-down fusion scheme. (c) Single scale features for faster detection utilized in Fast and Faster R-CNN [13, 29]. (d) Pyramidal feature hierarchy employed in standard SSD where feature pyramid is constructed by a CNN [28]. (e) Our architecture is accurate like (a) but efficient due to the proposed light-weight convolutional block (Sec. 3.2) and integrated with (d).

Here, the objective is to leverage high-level semantics by up-sampling low-resolution feature maps and fusing them with high-resolution feature maps. However, such an approach is still sub-optimal for very small and large sized objects [32]. For very small objects, even a large up-sampling factor cannot match the typical resolution (224×224) of pre-trained networks. Consequently, the high-level semantic features generated by the feature pyramid network will still not be adequate for very small object detection and vice versa. Further, such an approach is computationally expensive due to the layer-by-layer fusion of many layers.

In the case of single-stage methods, SSD exploits multiple CNN layers in a pyramidal feature hierarchy, producing feature maps of varying spatial resolutions (see Fig. 1(d)). However, trading spatial resolution at the cost of high-level semantic information can affect the performance. In this work, we aim to improve the accuracy of SSD without sacrificing its hallmark speed. We re-visit the classical image pyramid approach (see Fig. 1(a)), where feature maps of varying scales are generated by applying a CNN on each of the image scales separately, in a single-stage detection framework. However, the standard image pyramid based feature representation (featurized image pyramids) is slow since each image scale is passed through a deep CNN to extract scale-specific feature maps, thereby making its usage impractical for a high-speed SSD.

Contributions: We introduce a light-weight featurized image pyramid network (LFIP) to produce a multi-scale feature representation. Within the LFIP network (see Fig. 1(e)), an input image is first iteratively downsampled to construct an image pyramid hierarchy, which is then fed to a shallow convolutional block, resulting in a feature pyramid where each level of an image pyramid is featurized. Multi-scale features from the feature pyramid are then combined with the standard SSD features, in an attention module, in order to boost the discriminative power. Furthermore, we introduce a forward fusion module to integrate features from both the former and current layers.

We perform extensive experiments on two benchmarks: PASCAL VOC and MS COCO. Our detector provides superior results on both datasets compared to existing single-stage methods. Further, our approach provides significantly improved results on small objects achieving an absolute gain of 7.4% in average precision (AP) on MS COCO small set, compared with the baseline SSD.

2. Baseline Detector: SSD

We base our approach on the SSD [28] which employs a VGG-16 architecture as the backbone network. Given an input image I of size 300×300, the SSD uses $conv4\_3$ layer with feature size 38×38 and FC_7 (converted into a conv layer) with feature size 19×19 from the original VGG-16 architecture. It truncates the last fully connected layer of the VGG-16 network and further adds a series of progressively smaller conv layers: $conv8\_2$, $conv9\_2$, $conv10\_2$ and $conv11\_2$, with feature size 10×10, 5×5, 3×3 and 1×1, respectively, at the end for detection. The detector adopts a pyramidal hierarchical structure where shallow layers (i.e. $conv4\_3$) predict small object instances and deep layers (i.e. $conv8\_2$) detect large object instances. In this way, each of the aforementioned layers are used for score predictions and offsets, over a predefined set of bounding boxes. The score predictions are performed by $3\times3\times N$ filter dimensions, where N is the number of channels. Consequently, non-maximum suppression (NMS) is applied to obtain final detection scores. We refer to [28] for details.

As mentioned above, the standard SSD localizes objects in a pyramidal hierarchy by exploiting multiple CNN layers, with each layer designated to detect objects of a specific scale. This implies that small object instances are detected using former layers with small receptive fields, while deep layers with large receptive fields are used to localize large object instances. However, the SSD struggles to accurately detect small object instances due to limited information in shallow layers, compared to deep layers [18]. Fu *et al*. [12] proposed to use deconvolution layers to introduce large-scale context and a better feature extraction network (ResNet) to improve the accuracy. Cao *et al*. [4] also investigated the problem of small object detection and introduced contextual information to the SSD. However, these approaches improve SSD at the cost of reduction in speed. Further, the additional contextual information may introduce unnecessary background noise, resulting in a deterioration of accuracy in some cases. Zhang *et al*. [34] extended

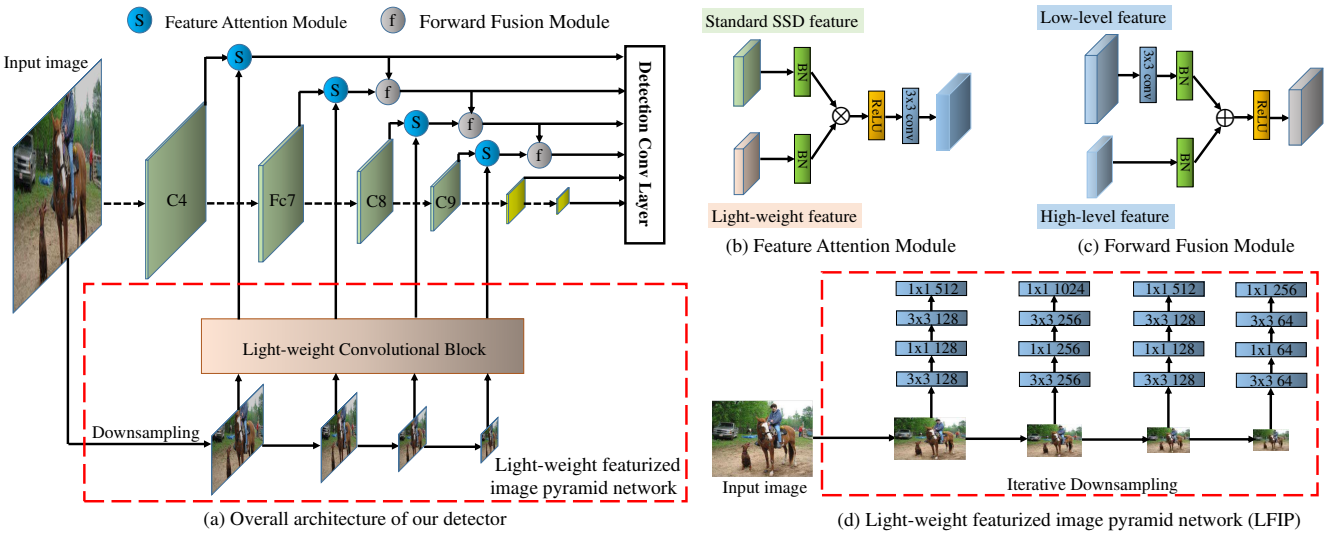


Figure 2. (a) Overall architecture of our single-stage object detector. Our approach extends the SSD with a light-weight featurized image pyramid network (LFIP) whose architecture is shown in (d). Within the LFIP network, an input image is first iteratively downsampled to construct an image pyramid hierarchy. The image pyramid hierarchy is then input to a shallow convolutional block which produces a feature pyramid by featurizing each level of the image pyramid. The resulting feature pyramid is then injected into the standard SSD prediction layers using an attention module shown in (b). We also introduce a forward fusion module to integrate the modulated features from both the former and current layers, shown in (c).

the standard SSD by integrating a semantic segmentation branch. Instead, we re-visit the classic approach of building feature pyramid from image pyramid without sacrificing the hallmark speed of the SSD.

3. Method

Here, we first describe the overall architecture of our approach and introduce an alternative feature extraction strategy, utilized in our light-weight featurized image pyramid network module. Afterwards, we introduce feature attention and forward fusion modules. The overall architecture of our detector, named LFIP-SSD, is illustrated in Fig. 2(a). LFIP-SSD comprises of two main parts: the standard SSD network and the proposed light-weight featurized image pyramid network (LFIP) to produce a feature pyramid representation. As in [28], we employ VGG-16 as the backbone and add a series of progressively smaller conv layers. Different to the standard SSD, LFIP contains an iterative downsampling and a light-weight convolutional block. Features from the LFIP are then injected into the standard SSD layers using an attention module. The resulting features of the current layer are then fused with their former layer counterpart in a forward fusion module.

3.1. Feature Extraction Strategy

Conventional object detection frameworks typically extract features either from a VGG-16 or ResNet-50, in a repeated stack of convolutional blocks and max-pooling operations (see Fig. 3(a)). Though the resulting features are

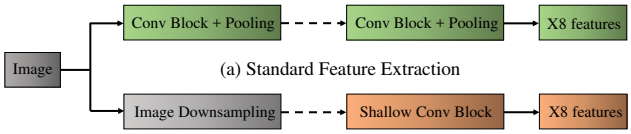


Figure 3. Comparison of our feature extraction strategy, employed in the LFIP network, with its standard SSD counterpart. (a) Standard SSD feature extraction: convolution block together with repeated stride and max-pooling operations to generate features. Here, "X8" shows that downsmapling is performed with a stride of 8. (b) proposed Feature Extraction in LFIP: the input image is first downsampled to the target size and then a shallow convolution block is used to extract features.

semantically strong, they tend to lose discriminative information that likely contributes towards accurate object classification. As an alternative, we introduce an efficient feature extraction strategy (see Fig. 3(b)). In our strategy, an input image is first downsampled, either by interpolation or a pooling operation, to the desired target size of different SSD prediction layers. These downsampled images are then passed through a shallow convolutional block. Compared to the deep CNNs in the traditional image pyramid network, our shallow convolutional block provides the computational efficiency required for high-speed detection, while enhancing discriminative information for multi-scale detection.

3.2. Light-weight Featurized Image Pyramid

As discussed earlier, standard featurized image pyramids are inefficient since each image scale is passed through a

deep CNN to extract scale-specific feature maps. High-speed single-stage detectors therefore tend to avoid such a featurized image pyramid representation. Here, we propose a simple, yet effective solution to efficiently construct a light-weight featurized image pyramid (LFIP) representation. As shown in Fig. 2(d), the LFIP network comprises of an iterative downsampling part and a light-weight convolutional block. Given an input image I, an image pyramid I_p is first constructed through iterative downsapling:

$$I_p = \{i_1, i_2, \ldots, i_n\} \tag{1}$$

where n denotes the number of image pyramid levels. Image scales in the pyramid are selected to match the sizes of standard SSD prediction layer maps, such as $conv4\_3$. Afterwards, each of the image scales is passed through a shallow convolutional block to generate the multi-scale light-weight feature maps:

$$S_p = \{s_1, s_2, \ldots, s_n\} \tag{2}$$

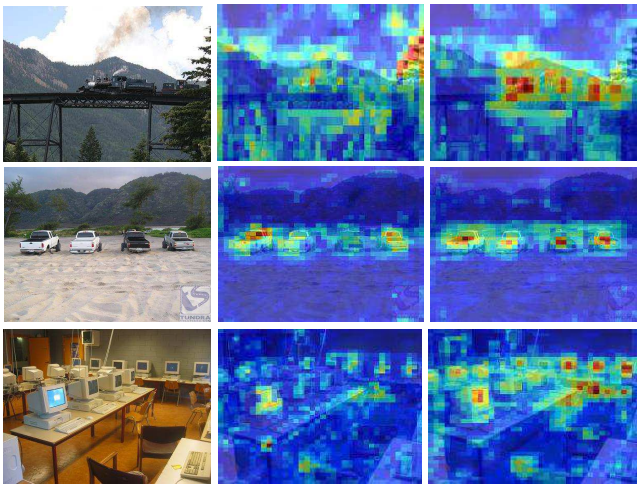
where s_1 denotes the light-weight features for the $conv4\_3$ layer of standard SSD network and s_n represents the last features generated for the $conv9\_2$ layer of the SSD network. The shallow convolutional block includes one 3×3 convolution layer and a bottleneck block, as in [17], but without the identity shortcut. The identity shortcut has been skipped due to the shallow nature of our convolutional block. The conv layers in our shallow convolutional block vary in the number of channels to match the resulting light-weight featurized image pyramids with that of standard SSD feature maps.

3.3. Feature Attention Module

Here, we describe how the light-weight featurized image pyramid, generated from our LFIP network, is injected into the standard SSD prediction layers. We introduce a feature attention module (FAM), as shown in Fig. 2(b). First, both the light-weight featurized image pyramid and standard SSD feature maps are passed through a BatchNorm (BN) layer for normalization. We consider different ways to fuse the normalized feature set: concatenation, element-wise sum and element-wise product. We found that element-wise product provides the best performance. Consequently, we employ a ReLU activation and a 3×3 conv layer to generate modulated features. For an input image I, standard SSD features f_k from the k^{th} SSD prediction layer are combined with the corresponding light-weight LFIP features s_k as:

$$m_k = \varphi_k(\beta(f_k) \otimes \beta(s_k)) \tag{3}$$

where m_k are the modulated features after fusion, $\varphi_k(.)$ denotes the operation including the serial ReLU and 3×3 $conv$ layer, and $\beta(.)$ denotes the BN operation. As shown in Fig. 4, our modulated features enhance the discriminative power of standard SSD features.



(a) Input image (b) SSD feature (c) Modulated feature

Figure 4. Comparison of feature maps obtained from the $conv4\_3$ layer in standard SSD and our modulated features after feature attention module.

3.4. Forward Fusion Module

To further enhance the spatial information, we introduce a simple forward fusion module (FFM) to integrate modulated features from both the former and current layers (Fig. 2(c)). We employ the FFM module for layers from FC_7 to $conv9\_2$. Within FFM, each previous layer is first pass through a 3×3 conv layer to achieve the same size as the current layer. Afterwards, former m_{k-1} and current m_k modulated features are passed through BatchNorm (BN) and combined using an element-wise sum operation. This is followed by a ReLU operation to produce the final prediction d_k as:

$$d_k = \gamma(\phi_k(m_{k-1}) \oplus \beta(m_k)) \tag{4}$$

where $\phi_k(.)$ denotes the operation including the serial 3x3 $conv$ and BN layers, $\beta(.)$ is the BN operation, and γ is the ReLU activation.

4. Experiments

We validate our approach by conducting experiments on two datasets: PASCAL VOC and MS COCO. We first introduce the two datasets and then discuss the implementation details of our proposed detector. We compare our detector with state-of-the-art object detection methods from the literature and also provide a comprehensive ablation study on the PASCAL VOC 2007 dataset.

4.1. Datasets

PASCAL VOC [10]: The PASCAL VOC dataset consists of 20 different object categories. For this dataset, the training is performed on a union of the VOC 2007 trainval set with 5k images and the VOC 2012 trainval set with 11k images. Evaluation is carried out on the PASCAL VOC 2007 test with 5k images. Object detection accuracy is measured in terms of mean Average Precision *mAP*.

MS COCO [26]: The MS COCO dataset consists of 160k images with 80 object categories. The dataset contains 80k training, 40k validation and 40k test-dev images. For MS COCO, training is performed on 120k images from the trainval set and evaluation is carried out on the test-dev set. We follow the standard MS COCO protocol for evaluation, where the overall performance, average precision AP, is measured by averaging over multiple IOU thresholds, ranging from 0.5 to 0.95.

4.2. Implementation Details

All experiments are conducted using VGG-16 [31], pre-trained on ImageNet [30], as the backbone. Our full training and testing code is built on Pytorch and will be publicly available. We follow the similar settings as the baseline SSD [28] for model initialization and optimization. The warm-up strategy is adopted for the first six epochs. The learning rate is first set to 2×10^{-3} and gradually decreases to 10^{-4} and 10^{-5} at 150 and 200 epochs, respectively for the PASCAL VOC dataset. In the case of MS COCO, the same learning rate values (used in the PASCAL VOC) decreases at 90 and 120 epochs. Following [28], we use the same loss function, scales and aspect ratios of the defaults boxes and the data augmentation method. We set the weight decay to 0.0005 and the momentum to 0.9. The batch-size is set to 32 for both datasets. A total number of 250 and 160 epochs are performed for the PASCAL VOC and MS COCO datasets, respectively. The FLOPs of VGG backbone and LFIP are 1.6G and 0.9G, respectively. The FLOPs of LFIP mainly come from convolution operations followed by element-wise multiplication and addition.

4.3. Pascal VOC 2007

We first compare our detector with the baseline SSD and other existing single-stage detectors. For a fair comparison, we use the same settings for both our detector and the baseline SSD. Tab. 1 shows the comparison, in terms of speed and detection accuracy, of our detector with both the baseline SSD and several other single-stage detection methods. The baseline SSD achieves a detection mAP score of 77.2 while running at 120 FPS. Among existing single-stage detectors, RefineDet [33] and DES [34] provide detection accuracies of 80.0 and 79.7 mAP while running at 40 and 77 FPS, respectively. Our detector provides an optimal trade-off between detection accuracy and speed by providing a detection accuracy of 80.4 mAP while running at 111 FPS.

Methods	Backbone	input size	mAP	FPS
SSD [28]	VGG-16	300×300	77.2	120
R-SSD [20]	VGG-16	300×300	78.5	35
RUN [23]	VGG-16	300×300	79.2	40
ESSD [35]	VGG-16	300×300	79.4	25
DSSD [12]	ResNet-101	321×321	78.6	9.5
DES [34]	VGG-16	300×300	79.7	76.8
WeaveNet [6]	VGG-16	320×320	79.7	50
RefineDet [33]	VGG-16	320×320	80.0	40.3
Ours	VGG-16	300×300	80.4	111

Table 1. Speed and performance comparisons of our method with existing single-stage detectors on PASCAL VOC 2007 test set. For all detectors, the input image size is approximately $\sim$ 300x300. For a fair comparison, the speed for all detectors is measured on a single Titan X GPU (Maxwell architecture). The best two results are shown in red and blue. Our detector improves the detection accuracy by 3.2% in mAP over the baseline SSD. Further, our detector provides an optimal trade-off between detection accuracy and speed compared to existing single-stage detectors.

State-of-the-art Comparison: Here, we compare our detector with state-of-the-art single and two-stage detection methods. Tab. 2 shows a per-class comparison for varying input image sizes. Generally, two-stage object detection methods [29, 7] take a large image as input ($\sim 1000 \times 600$) compared to their single-stage counterparts. Among two-stage object detectors, CoupleNet [36] with multi-scale testing provides improved performance with 82.7 mAP. Among single-stage methods, RefineDet [33] achieves a detection accuracy of 81.8 when using an input image of size ($\sim 512 \times 512$). With the same input image size, our detector achieves similar detection accuracy while providing a 2.7-fold speedup compared to RefineDet [33].

Runtime Analysis: Fig. 5 shows the accuracy vs speed comparison of our detector with state-of-the-art single and two-stage methods, on the VOC 2007 test set. All detection speeds are measured on a single Titan X GPU (Maxwell architecture). Our detector processes an image at 111 FPS whereas the baseline SSD runs at 120 FPS. Among existing methods, the two-stage CoupleNet [36] provides superior detection results with a speed of 8 FPS. Our detector provides a 13-fold speedup compared to CoupleNet [36].

4.4. Ablation Study on PASCAL VOC 2007

We conduct an ablation study to validate the effectiveness of different modules proposed in our detector. We analyze the impact on detection performance of different downsampling strategies, various convolutional block depths and the light-weight multi-scale features.

Downsampling Strategies: We investigate three commonly used downsampling strategies to construct the image pyramid: bilinear interpolation, max pooling and average pooling. Tab. 3 (left) shows the comparison when using

Methods	Backbone	input size	mAP	speed	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Two-Stage Detector:																								
Faster-RCNN [29]	VGG-16	1000 × 600	73.2	7.0	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
Faster-RCNN [17]	ResNet-101	1000 × 600	76.4	5.0	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
ION [2]	VGG-16	1000 × 600	76.5	1.2	79.2	79.2	77.4	69.8	55.7	85.2	84.2	89.8	57.5	78.5	73.8	87.8	85.9	81.3	75.3	49.7	76.9	74.6	85.2	82.1
HyperNet [22]	VGG-16	1000 × 600	76.3	0.9	77.4	83.3	75.0	69.1	62.4	83.1	87.4	87.4	57.1	79.8	71.4	85.1	85.1	80.0	79.1	51.2	79.1	75.7	80.9	76.5
R-FCN [7]	ResNet-101	1000 × 600	80.5	9.0	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
CoupleNet MS [36]	ResNet-101	1000 × 600	82.7	8.2	85.7	87.0	84.8	75.5	73.3	88.8	89.2	89.6	69.8	87.5	76.1	88.9	89.0	87.2	86.2	59.1	83.6	83.4	87.6	80.7
Single-Stage Detector:																								
SSD [28]	VGG-16	300 × 300	77.5	120.0	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	84.0	79.4	52.3	77.9	79.5	87.6	76.8
RON [21]	VGG-16	384 × 384	75.4	15.0	86.5	82.9	76.6	60.9	55.8	81.7	80.2	91.1	57.3	81.1	60.4	87.2	84.8	84.9	81.7	51.9	79.1	68.6	84.1	70.3
DSSD [12]	ResNet-101	321 × 321	78.6	9.5	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
RefineDet [33]	VGG-16	320 × 320	80.0	40.3	83.9	85.4	81.4	75.5	60.2	86.4	88.1	89.1	62.7	83.9	77.0	85.4	87.1	86.7	82.6	55.3	82.7	78.5	88.1	79.4
Ours	VGG-16	300 × 300	80.4	111.0	84.0	85.8	78.2	75.3	60.8	88.6	87.6	87.9	63.3	83.8	78.9	86.0	87.7	88.6	81.9	56.8	80.8	80.5	88.2	79.1
SSD [28]	VGG-16	512 × 512	79.5	60.0	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
DES [34]	VGG-16	512 × 512	81.7	31.7	87.7	86.7	85.2	76.3	60.6	88.7	89.0	88.0	67.0	86.9	78.0	87.2	87.9	87.4	84.4	59.2	86.1	79.2	88.1	80.5
DSSD [12]	ResNet-101	513 × 513	81.5	5.5	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
RefineDet [33]	VGG-16	512 × 512	81.8	20.1	88.7	87.0	83.2	76.5	68.0	88.5	88.7	89.2	66.5	87.9	75.0	86.8	89.2	87.8	84.7	56.2	83.2	78.7	88.1	82.3
Ours	VGG-16	512 × 512	81.8	53.0	86.6	88.2	81.7	76.1	66.6	89.0	89.2	86.1	66.5	87.3	79.2	85.3	88.7	87.5	84.2	57.9	83.7	79.8	87.4	82.9

Table 2. Per-class state-of-the-art comparison on the PASCAL VOC 2007 dataset. All detection methods are trained on the union of VOC2007 and VOC2012 trainval and tested on VOC2007 test. When comparing with single-stage detectors, our number is marked in red and blue if it is the best two in the column. Our two detection methods have exactly the same settings except having different input sizes (300×300 and 512×512). Our detector achieves promising results and provides a good trade-off between detection accuracy and speed, compared to state-of-the-art approaches in literature.

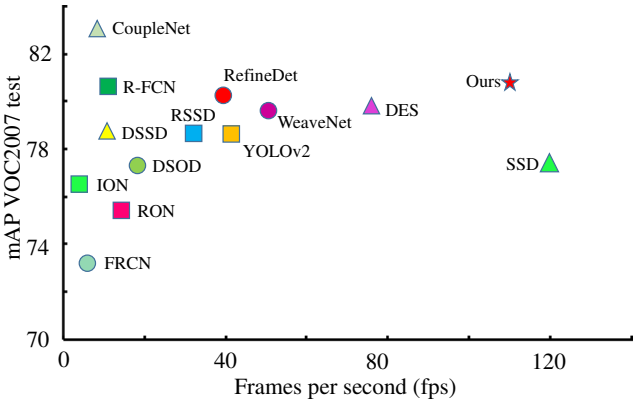


Figure 5. Accuracy vs speed comparison on the PASCAL VOC 2007 test set. For fair comparison, all detectors are trained on the VOC 2007+2012 trainval and the speed is measured on a single Titan X GPU. For two-stage detectors, an input image size of $\sim 1000\times600$ is used. All single-stage detectors use an input image size of $\sim 300\times300$ except YOLOv2 (544×544). Our detector achieves a 9-fold speedup compared to the two-stage CoupleNet.

Methods	mAP	Methods	mAP
Bilinear interpolation	79.8	Progressive decrement	79.6
Average pooling	79.9	Constant depth	80.0
Max pooling	**80.0**	Progressive increment	**80.2**

Table 3. Analyzing the impact of different downsampling strategies when constructing the image pyramid (left). Here, we consider bilinear interpolation, average pooling and max pooling downsampling strategies. We also analyze the impact of network depth on the shallow convolutional block (right).

Add-on	SSD					Ours
conv 4_3		✓	✓	✓	✓	✓
conv 7			✓	✓	✓	✓
conv 8_2				✓	✓	✓
conv 9_2					✓	
with FFM						✓
mAP	77.2	79.4	79.9	80.2	79.9	**80.4**

Table 4. Ablation results on PASCAL VOC 2007 dataset with multi-scale Light-Weight Feature fusion at convolutional features at different stages of SSD model.

different downsampling strategies. The results show that changing the downsampling strategies has negligible effect on the overall detection results, though max pooling provides the best performance of 80.0 mAP.

Shallow Convolutional Block Depth: Here, we analyze different depths of the shallow convolutional block. We consider three different strategies: constant depth, progressive incrementation and progressive decrementation. For constant depth, same number of convolution layers are used for different levels in the shallow convolutional block. In progressive incrementation, we progressively increase the depth of the shallow convolutional block for the corresponding deeper prediction layers. In progressive decrementation, we progressively decrease the depth of the shallow convolutional block for the corresponding deeper prediction layers. Tab. 3 (right) shows the impact of using different depth strategies for the shallow convolutional block. The progressive incrementation provides the best results.

Impact of LFIP on SSD Prediction Layers: Here, we analyze the impact of our LFIP representation on the standard SSD. We perform an experiment by systematically injecting the LFIP representation at different stages of the stan-

Methods	Backbone	Input size	Time(ms)	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Two-Stage Detector:									
Faster [29]	VGG-16	$\sim 1000 \times 600$	147	24.2	45.3	23.5	7.7	26.4	37.1
Faster-FPN [24]	ResNet-101-FPN	$\sim 1000 \times 600$	240	36.2	59.1	39.0	18.2	39.0	48.2
R-FCN [7]	ResNet-101	$\sim 1000 \times 600$	110	29.9	51.9	-	10.8	32.8	45.0
Deformable R-FCN [8]	ResNet-101	$\sim 1000 \times 600$	125	34.5	55.0	-	14.0	37.7	50.3
Mask-RCNN [15]	ResNeXt-101-FPN	$\sim 1280 \times 800$	210	39.8	62.3	43.4	22.1	43.2	51.2
Cascade R-CNN [3]	ResNet-101-FPN	$\sim 1280 \times 800$	141	42.8	62.1	46.3	23.7	45.5	55.2
Single-Stage Detector:									
SSD [28]	VGG-16	300×300	12	25.1	43.1	25.8	6.6	25.9	41.4
DSSD [12]	ResNet-101	321×321	-	28.0	46.1	29.2	7.4	28.1	47.6
RefineDet [33]	VGG-16	320×320	24.8	29.4	49.2	31.3	10.0	32.0	44.4
RFBNet [27]	VGG-16	300×300	15	30.3	49.3	31.8	11.8	31.9	45.9
Ours	VGG-16	300×300	14	30.0	48.8	31.7	10.9	32.8	46.3
SSD [28]	VGG-16	512×512	28	28.8	48.5	30.3	10.9	31.8	43.5
DSSD [12]	ResNet-101	513×513	182	33.2	53.3	35.2	13.0	35.4	**51.1**
RefineDet [33]	VGG-16	512×512	41.5	33.0	54.5	35.5	16.3	36.3	44.3
RetinaNet [25]	ResNet-101-FPN	$\sim 832 \times 500$	90	34.4	53.1	36.8	14.7	**38.5**	49.1
RFBNet [27]	VGG-16	512×512	33	34.4	55.7	36.4	17.6	37.0	47.6
Ours	VGG-16	512×512	29	**34.6**	**55.8**	**36.8**	**18.3**	38.2	47.1

Table 5. State-of-the-art comparison on MS COCO test-dev set. When using 300×300 and 512×512 input image sizes, our detector improves the overall detection performance by 4.9% and 5.8% in AP, respectively, compared to the baseline SSD.

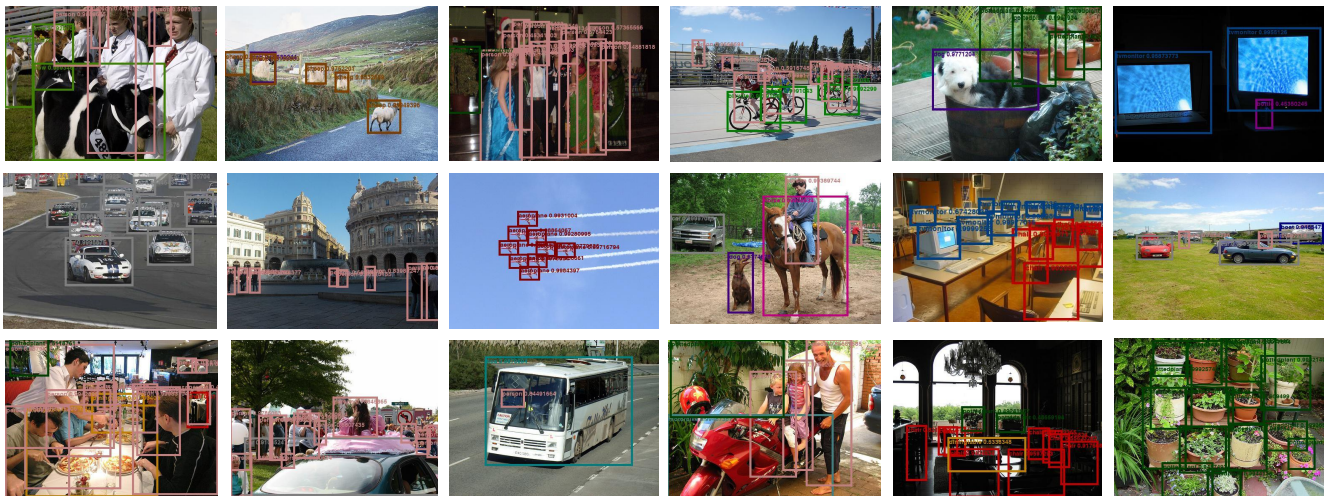


Figure 6. Qualitative results of our detector on the PASCAL VOC 2007 test set (corresponding to 81.8 mAP). The model was trained on all the train and validation datasets in VOC 2007 and VOC 2012. Each color is related to an object category.

dard SSD. Tab. 4 shows the detection results when injecting the LFIP representation at different stages of the standard SSD. A large gain (2.2%) in mAP is achieved when integrating the LFIP representation at the $conv4\_3$ level. Further improvements in detection performance are achieved up to $conv8\_2$ level. The performance slightly deteriorates when further an additional LFIP representation is inserted at the $conv9\_2$ level, which is likely due to the low-resolution of the $conv9\_2$ features. The overall performance is further improved when integrating the forward fusion module, leading to an accuracy of 80.4 mAP. We further validate our approach with ResNet-50 backbone. For a 300×300 input, our approach achieves an absolute gain of 2.8% mAP over the baseline SSD. Fig. 6 shows example detections on the VOC 2007 test set with our detector.

4.5. MS COCO

Here, we evaluate the performance of our detector on the MS COCO dataset. Tab. 5 shows the performance of our detector on MS COCO test-dev set. When using a 300×300 input image, our detector improves the overall detection performance by 4.9% in AP compared to the baseline SSD. While two-stage detectors provide better detection accuracy, they are slow and generally require more

Figure 7. Qualitative results, especially on images with small objects, of our detector on the COCO test-dev (corresponding to 34.6 mAP).

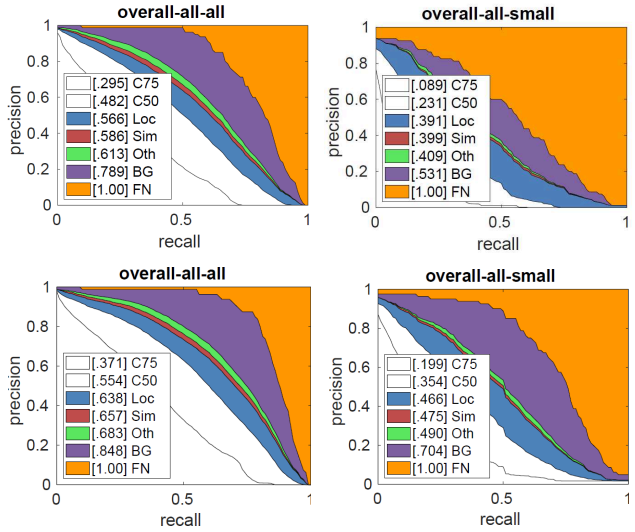


Figure 8. Error analysis for the performance of the baseline SSD (top row) and our detector (bottom row) across all categories, on the overall and the small sized objects subsets. The plots in each sub-image describe a series of precision recall curves using different evaluation settings [26] and the area under each curve is shown in brackets in the legend. Our detector provides consistent improvements over the baseline SSD.

than 100ms to process an image. For a 512×512 input, we follow a similar strategy as in [27] by up-sampling the $conv7$ feature maps and concatenating with $conv4\_3$ after applying our LFIP module. Both RetinaNet [25] and RFBNet [27] obtain an AP score of 34.4. Our detector (512×512 input size) achieves an AP of 34.6% while being relatively faster (29 ms), compared to RFBNet (33 ms). Further, our approach provides a 3-fold speedup over RetinaNet [25].

Qualitative Analysis: The MS COCO dataset is especially suitable to evaluate the performance on small sized objects since approximately 41% of its objects are small (area $< 32^2$) [26]. The dataset can be further divided into large, medium, and small sized objects. We analyze the performance of our detector using the error analysis provided by [26]. Fig. 8 shows the error analysis plots of the baseline SSD (top row) and our detector (bottom row) for overall and small sized objects. The plots in each sub-image describe a series of precision recall curves using different evaluation settings, as outlined in [26]. We show the area under each curve in brackets in the legend. For the baseline SSD (top row), the overall AP at IoU=.75 is .295 and perfect localization is likely to increase the AP to .566. Eliminating background false positives would increase the performance to .789 AP. In the case of our detector (bottom row), the overall AP at IoU=.75 is .371 and perfect localization is likely to increase the AP to .638. Further, eliminating background false positives would increase the performance to .848 AP. When analyzing the performance on small sized objects, the improvement achieved by our detector is more prominent. Our detector increases the overall AP at IoU=.75 from .089 to .199 and eliminating background false positives would increase the performance from .531 to .704. Fig. 7 shows example detections with our detector on the COCO test-dev.

5. Conclusion

We introduced a light-weight architecture to efficiently construct featurized image pyramids. We introduced a shallow convolutional block that takes as input the image pyramid and produces feature pyramid. Multi-scale features from the feature pyramid are then combined with standard SSD features, in an attention module. We further introduced a forward fusion module to integrate the modulated features from both former and current prediction layers. Experiments on two benchmarks clearly demonstrated that our approach provides superior detection accuracy at high speed.

Acknowledgment This work was supported by the National Natural Science Foundation of China (Grant No. 61632018)

References

[1] E. Adelson, C. Anderson, J. Bergen, P. Burt, and J. Ogden. Pyramid methods in image processing. *RCA engineer*, 1984. 1

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016. 6

[3] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 7

[4] G. Cao, X. Xie, W. Yang, Q. Liao, G. Shi, and J. Wu. Feature-fused SSD: Fast detection for small objects. *arXiv:1709.05054*, 2017. 2

[5] J. Cao, Y. Pang, and X. Li. Triply supervised decoder networks for joint detection and segmentation. In *CVPR*, 2019. 1

[6] Y. Chen, J. Li, B. Zhou, J. Feng, and S. Yan. Weaving multi-scale context for single shot detector. *arXiv:1712.03149*, 2017. 5

[7] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 5, 6, 7

[8] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017. 7

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1

[10] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 5

[11] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010. 1

[12] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. Berg. Dssd: Deconvolutional single shot detector. *arXiv:1701.06659*, 2017. 2, 5, 6, 7

[13] R. Girshick. Fast r-cnn. In *ICCV*, 2015. 1, 2

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[15] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask R-CNN. In *ICCV*, 2017. 7

[16] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014. 1

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 6

[18] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017. 1, 2

[19] R. Girshick J. Redmon, S. Divvala and A. Farhadi. Look once: Unified, real-time object detection. In *CVPR*, 2016. 1

[20] J. Jeong, H. Park, and N. Kwak. Enhancement of ssd by concatenating feature maps for object detection. *arXiv:1705.09587*, 2017. 5

[21] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen. Ron: Reverse connection with objectness prior networks for object detection. In *CVPR*, 2017. 6

[22] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *CVPR*, 2016. 6

[23] K. Lee, J. Choi, J. Jeong, and N. Kwak. Residual features and unified prediction network for single stage detection. *arXiv:1707.05031*, 2017. 5

[24] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 2, 7

[25] T.-Yi. Lin, P. Goyal, R. Girshick, K. He, and P. Dollr. Focal loss for dense object detection. In *ICCV*, 2017. 7, 8

[26] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5, 8

[27] S. Liu and D. Huang. Receptive field block net for accurate and fast object detection. In *ECCV*, 2018. 7, 8

[28] W. Liu, D. Anguelov, C. Szegedy D. Erhan, and S. Reed. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1, 2, 3, 5, 6, 7

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2, 5, 6, 7

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. In *IJCV*, 2015. 5

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *NIPS*, 2014. 5

[32] B. Singh and L. Davis. An analysis of scale invariance in object detection - snip. In *CVPR*, 2018. 2

[33] S. Zhang, L. Wen, X. Bian, Z. Lei, and Stan Z. Li. Single-shot refinement neural network for object detection. In *CVPR*, 2018. 5, 6, 7

[34] Z. Zhang, S. Qiao, C. Xie, W. Shen, B. Wang, and A. L. Yuille. Single-shot object detection with enriched semantics. *CVPR*, 2017. 2, 5, 6

[35] L. Zheng, C. Fu, and Y. Zhao. Extend the shallow part of single shot multibox detector via convolutional neural network. *arXiv:1801.05918*, 2018. 5

[36] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu. Couplenet: Coupling global structure with local parts for object detection. In *ICCV*, 2017. 5, 6