Scale-Aware Trident Networks for Object Detection

Yanghao Li* Yuntao Chen[1,3]* Naiyan Wang[2] Zhaoxiang Zhang[1,3,4]
[1] University of Chinese Academy of Sciences [2] TuSimple
[3] Center for Research on Intelligent Perception and Computing, CASIA
[4]Center for Excellence in Brain Science and Intelligence Technology, CAS
lyttonhao@gmail.com chenyuntao2016@ia.ac.cn zhaoxiang.zhang@ia.ac.cn winsty@gmail.com

Abstract

*Scale variation is one of the key challenges in object detection. In this work, we first present a controlled experiment to investigate the effect of receptive fields on the detection of different scale objects. Based on the findings from the exploration experiments, we propose a novel Trident Network (TridentNet) aiming to generate scale-specific feature maps with a uniform representational power. We construct a parallel multi-branch architecture in which each branch shares the same transformation parameters but with different receptive fields. Then, we propose a scale-aware training scheme to specialize each branch by sampling object instances of proper scales for training. As a bonus, a fast approximation version of TridentNet could achieve significant improvements without any additional parameters and computational cost. On the COCO dataset, our TridentNet with ResNet-101 backbone achieves state-of-the-art single-model results by obtaining an mAP of **48.4**. Code will be made publicly available.*

1. Introduction

In recent years, deep convolutional neural networks (CNNs) [17, 37, 30] have achieved great success in object detection. Typically, these CNN-based methods can be roughly divided into two types: one stage methods such as YOLO [34] or SSD [30] which directly utilizes feed-forward CNN to predict the bounding boxes of interest, while two stage methods such as Faster R-CNN [37] or R-FCN [10] first generate proposals, and then exploit the extracted region features from CNN for further refinement. However, a central issue in both methods lies in handling scale variation. It is very common that the scale of object instances varies in a wide range, which impedes the detectors, especially for very small or very large objects.

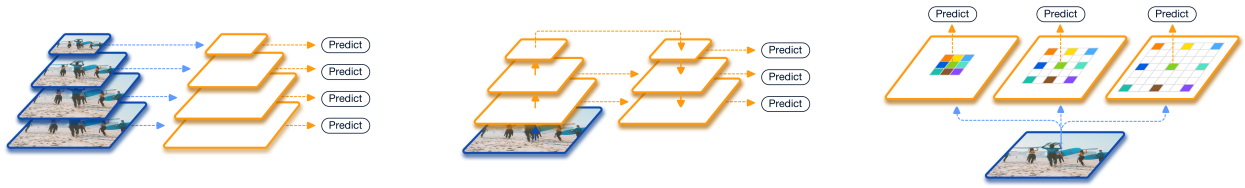
To remedy the scale variation issue, an intuitive way is to leverage multi-scale image pyramids [1], which is popular in both hand-crafted feature based methods [12, 31] and current deep CNN based methods (Figure 1(a)). Strong evidence [22, 29] shows that current standard deep detectors [37, 10] could benefit from multi-scale training and testing. To avoid training objects with extreme scales (small/large objects in smaller/larger scales), SNIP [40, 41] proposes a scale normalization method that selectively trains the objects of appropriate sizes in each image scale. Nevertheless, the increase of inference time makes the image pyramid methods infeasible for practical applications.

The other line of efforts aims to employ in-network feature pyramids to approximate image pyramids with less computation cost. The idea is first demonstrated in [13], where a fast feature pyramid is constructed for object detection by interpolating some feature channels from nearby scale levels. In the deep learning era, the approximation is even easier. SSD [30] utilizes multi-scale feature maps from different layers and detects objects of different scales at each feature layer. To compensate the absence of semantics in low-level features, FPN [26] (Figure 1(b)) further augments a top-down pathway and lateral connections to incorporate strong semantic information in high-level features. However, the representational power for objects of different scales still differ, since their features are extracted on different layers in FPN. This makes feature pyramids an unsatisfactory alternative for image pyramids.

Both image pyramid and feature pyramid methods share the same motivation that models should have different receptive fields for objects of different scales. Despite of the inefficiency, image pyramids fully utilize the representational power of the model to transform objects of all scales equally. In contrast, feature pyramids generate multi-level features thus sacrificing the feature consistency across different scales. The goal of this work is to get the best of two worlds by creating features with a uniform representational power for all scales efficiently.

In this paper, instead of feeding in multi-scale inputs

* Equal Contribution



(a) Image Pyramid (b) Feature Pyramid (c) Trident Network

Figure 1: (a) Using multiple images of several scales as input, image pyramid methods perform feature extraction and object detection independently for each scale. (b) Feature pyramid methods utilize the features from different layers of CNNs for different scales, which is computational friendly. This figure takes FPN [26] as an example. (c) Our proposed Trident Network generates scale-aware feature maps efficiently by trident blocks with different receptive fields.

like image pyramids, we propose a novel network structure to adapt the network for different scales. In particular, we create multiple scale-specific feature maps with the proposed trident blocks as shown in Figure 1(c). With the help of dilated convolutions [43], different branches of trident blocks have the same network structure and share the same parameter weights yet have different receptive fields. Furthermore, to avoid training objects with extreme scales, we leverage a scale-aware training scheme to make each branch specific to a given scale range matching its receptive field. Finally, thanks to weight sharing through the whole multi-branch network, we could approximate the full TridentNet with only one major branch during inference. This approximation only brings marginal performance degradation. As a result, it could achieve significant improvement over the single-scale baseline without any compromise to inference speed. This property makes TridentNet more desirable over other methods for practical uses.

To summarize, our contributions are listed as follows:

- We present our investigation results about the effect of receptive field on objects of different scales. To our best knowledge, we are the first to design controlled experiments to explore receptive field on the object detection task.
- We propose a novel Trident Network to deal with scale variation problem for object detection. Through multi-branch structure and scale-aware training, TridentNet could generate scale-specific feature maps with a uniform representational power.
- We propose a fast approximation method through one major branch with the help of our weight-sharing trident-block design, thus introducing no additional parameters and computational cost during inference.
- We validate the effectiveness of our approach on standard COCO benchmark with thorough ablation studies. Compared to the state-of-the-art methods, our proposed method achieves remarkable performance with **an mAP of 48.4 using a single model with ResNet-101 as backbone**.

2. Related Work

Deep Object Detectors. Deep learning based object detection methods have shown dramatic improvements in both accuracy and speed recently. As one of the predominant detectors, two-stage detection methods [17, 16, 37, 10, 5, 24] first generate a set of region proposals and then refine them by CNN networks. In [17], R-CNN generates region proposals by Selective Search [42] and then classifies and refines the cropped proposal regions from the original image by a standard CNN independently and sequentially. To reduce the redundant computation of feature extraction in R-CNN, SPPNet [19] and Fast R-CNN [16] extract the feature of the whole image once, and then generate region features through spatial pyramid pooling and RoI pooling layers, respectively. The RoI pooling layer is further improved by RoI Align layer [18] to address the coarse spatial quantization problem. Faster R-CNN [37] first proposes a unified end-to-end framework for object detection. It introduces a region proposal network (RPN) that share the same backbone network with the detection network to replace the original standalone time-consuming region proposal methods. To further improve the efficiency of Faster R-CNN, R-FCN [10] constructs a position-sensitive score maps through fully convolutional network to avoid the RoI-wise head network. To avoid additional large score maps in R-FCN, Light-Head R-CNN [24] is designed by using a thin feature map and a cheap R-CNN subnet to build a two-stage detector more efficiently.

On the other hand, one-stage methods which are popularized by YOLO [34, 35, 36] and SSD [30], aim to be more efficient by directly classifying the pre-defined anchors and further refining them using CNNs without the proposal generation step. Based on the multi-layer prediction module in SSD, DSSD [14] introduces additional context information with deconvolutional operators to improve the accuracy. RetinaNet [27] proposes a new focal loss to address the extreme foreground-background class imbalance which stands out as a central issue in one-stage detectors. Inheriting the merits of two-stage approaches, RefineDet [44] proposes an anchor refinement module to first filter the negative

anchor boxes and coarsely adjust the anchor boxes for the next detection module.

Methods for handling scale variation. As the most challenging problem in object detection, large scale variation across object instances hampers the accuracy of detectors. Multi-scale image pyramid [22, 29, 11] is a common scheme to improve the detection methods, especially for objects of small and large scales. Based on the image pyramid strategy, SNIP [40] proposes a scale normalization method to train objects that fall into the desired scale range for each resolution during multi-scale training. To perform multi-scale training more efficiently, SNIPER [41] only selects context regions around the ground-truth instances and sampled background regions for each scale during training. However, SNIP and SNIPER still suffer from the inevitable increasing of inference time.

Instead of taking multiple images as input, methods using multi-level features alleviate the problems caused by scale variation by using different layers in a CNN. Methods like HyperNet [23] and ION [2] concatenate low-level and high-level features from different layers to generate better feature maps for prediction. Since the features from different layers usually have different resolutions, specific normalization or transformation operators need to be designed before fusing multi-level features. Instead, SSD [30] and MS-CNN [4] perform object detection at multiple layers for objects of different scales without feature fusion. TDM [39] and FPN [26] further introduce a top-down pathway and lateral connections to enhance the semantic representation of low-level features at bottom layers. PANet [29] enhances the feature hierarchies in FPN by additional bottom-up path augmentation and proposes adaptive feature pooling to aggregate features from all levels for better prediction. Rather than using features from different layers, our proposed TridentNet generates scale-specific features through multiple parallel branches, thus endowing our network the same representational power for all objects of different scales.

Dilated convolution [43]. Dilated convolution (aka Atrous convolution [21]) enlarges the convolutional kernel with original weights by performing convolution at sparsely sampled locations, thus increasing the receptive field size without additional parameter cost. Dilated convolution has been widely used in semantic segmentation to incorporate large-scale context information [43, 6, 45, 7]. In object detection field, DetNet [25] designs a specific detection backbone network to maintain the spatial resolution and enlarge the receptive field using dilated convolution. Deformable convolution [11] further proposes a more general convolution operator by learning the 2D offsets adaptively. In our work, we employ dilated convolution in our multi-branch architecture with different dilation rates to adapt the receptive fields for objects of different scales.

3. Investigation of Receptive Field

There are several design factors of the backbone network that may affect the performance of object detectors including downsample rate, network depth and receptive field. Several works have already discussed their impact [4, 25]. The impacts of the first two factors are straightforward: deeper network with low downsample rate may increase the complexity, but benefit the detection task in general. Nevertheless, as far as we know, there is no previous work that studies the impact of receptive field in isolation.

To investigate the effect of receptive field on the detection of different scale objects, we replace some convolution operators in backbone network with the dilated variant [43]. We use different dilation rates to control the receptive field of a network.

Dilated convolution with dilation rate d_s inserts $d_s - 1$ zeros between consecutive filter values, enlarging the kernel size without increasing the number of parameters and computations. Specifically, a dilated 3×3 convolution could have the same receptive field as the convolution with kernel size of $3 + 2(d_s - 1)$. Suppose the downsample rate of current feature map is s with respect to the input image, then a dilated convolution of rate d_s could increase the receptive field of the network by $2(d_s - 1)s$. Thus if we modify n conv layers with d_s dilation rate, the receptive field could be increased by $2(d_s - 1)sn$.

We conduct this pilot experiment on the COCO benchmark [28] with a Faster R-CNN [37] detector. The results are reported in the COCO-style Average Precision (AP) on all objects and objects of small, medium and large sizes, respectively [28]. We use ResNet-50 and ResNet-101 [20] as the backbone network and vary the dilation rate d_s of the 3×3 conv operator between 1 to 3 for the residual blocks in the *conv*4 stage.

Table 1 summarizes the detection results with different dilation rates. We can find that as the receptive field size increases (larger dilation rate), the performance of the detector on small objects drops consistently on both ResNet-50 and ResNet-101. While for large objects, the detector benefits from the increasing receptive fields. The above findings suggest that:

1. The performance on objects of different scales are influenced by the receptive field of a network. The most suitable receptive field is strongly correlated with the scale of objects.

2. Although ResNet-101 has a large enough theoretical receptive field to cover objects of large scale (greater than 96×96 resolution) in COCO, the performance of

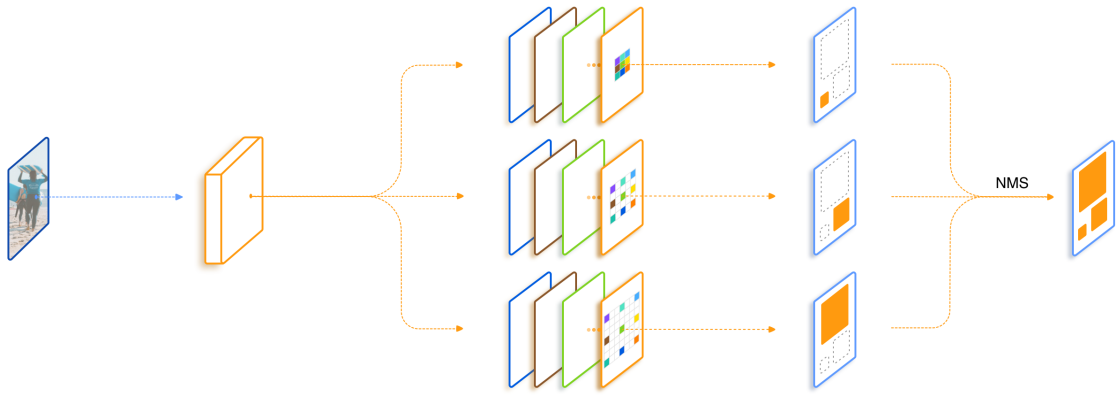


Figure 2: Illustration of the proposed TridentNet. The multiple branches in trident blocks share the same parameters with different dilation rates to generate scale-specific feature maps. Objects of specified scales are sampled for each branch during training. The final proposals or detections from multiple branches will be combined using Non-maximum Suppression (NMS). Here we only show the backbone network of TridentNet. The RPN and Fast R-CNN heads are shared among branches and ignored for simplicity.

Backbone	Dilation	AP	AP_s	AP_m	AP_l
ResNet-50	1	0.332	**0.174**	0.384	0.464
	2	0.342	0.168	**0.386**	0.486
	3	0.341	0.162	0.383	**0.492**
ResNet-101	1	0.379	**0.200**	**0.430**	0.528
	2	0.380	0.191	0.427	**0.538**
	3	0.371	0.181	0.410	**0.538**

Table 1: Object detection results with different receptive fields using Faster R-CNN [37] evaluated on the COCO *minival* dataset [28].

large objects could still be improved when enlarging the dilation rate. This finding shares the same spirit in [32] that the effective receptive field is smaller than the theoretical receptive field. We hypothesize that the effective receptive field of detection networks needs to balance between objects of small and large scales. Increasing dilation rates enlarges the effective receptive field by emphasizing large objects, thus compromising the detection results of small objects.

The aforementioned experiments motivate us to adapt the receptive field for objects of different scales as detailed in the next section.

4. Trident Network

In this section, we describe our scale-aware Trident Network (TridentNet) for object detection. The proposed TridentNet consists of weight sharing trident blocks and a deliberately designed scale-aware training scheme. Finally, We also present the inference details of TridentNet, including a fast inference approximation method.

4.1. Network Structure

Our goal is to inherit the merits of different receptive field sizes and avoid their drawbacks for detection networks. We propose a novel Trident architecture to achieve this as shown in Figure 2. In particular, our method takes a single-scale image as input, and then creates scale-specific feature maps through parallel branches of convolutions with the same parameters but with different dilation rates.

Multi-branch Block We construct TridentNets by replacing some convolution blocks with the proposed *trident blocks* in the backbone network of a detector. A trident block consists of multiple parallel branches in which each shares the same structure with the original convolution block except the dilation rate.

Taking ResNet as an example, for a single residual block in the bottleneck style [20], which consists of three convolutions with kernel size 1×1, 3×3 and 1×1, a corresponding trident block is constructed as multiple parallel residual blocks with different dilation rates for the 3×3 convs. Stacking trident blocks allows us to modulate receptive fields of different branches in an efficient way similar to the pilot experiment in Section 3. Typically, we replace the blocks in the last stage of backbone with trident blocks since larger strides lead to larger difference in receptive fields as needed. Detailed design choices could be referred in Section 5.2.

Weight sharing among branches. An immediate problem of our multi-branch trident block is that it introduces

several times parameters which may potentially incur overfitting. Fortunately, different branches share the same structure (except dilation rates) and thus make weight sharing straightforward. In this work, we share the weights of all branches and their associated RPN and R-CNN heads, and only vary the dilation rate of each branch.

The advantages of weight sharing are three-fold. It reduces the amount of parameters and makes TridentNet need no extra parameters compared with the original detector. It also echoes with our motivation that objects of different scales should go through a uniform transformation with the same representational power. A final point is that transformation parameters could be trained on more object samples from all branches. In other words, the same parameters are trained for different scale ranges under different receptive fields.

4.2. Scale-aware Training Scheme

The proposed trident architecture generates scale-specific feature maps according to the pre-defined dilation rates. However, the degradation in Table 1 caused by scale mismatching (*e.g.* small objects on the branch with too large dilation) still exists for each single branch. Thus, it is natural to detect objects of different scales on different branches. Here, we propose a scale-aware training scheme to improve the scale awareness of each branch and avoid training objects of extreme scales on mismatched branches.

Similar to SNIP [40], we define a valid range $[l_i, u_i]$ for each branch i. During training, we only select the proposals and ground truth boxes whose scales fall in the corresponding valid range of each branch. Specifically, for an Region-of-Interest (ROI) with width w and height h, it is valid for branch i when:

$$l_i \leq \sqrt{wh} \leq u_i. \tag{1}$$

This scale-aware training scheme could be applied on both RPN and R-CNN. In the original RPN design, the RPN head performs object/non-object binary classification and bounding box regression. The targets of classification and regression are defined according to a set of reference boxes (*i.e.* anchors) [37]. In our scale-aware training, we select ground truth boxes which are valid for this branch according to Eq. 1 during anchor label assignment for RPN. Similarly, we sample valid proposals for each branch during the training of R-CNN.

4.3. Inference and Approximation

During inference, we generate detection results for all branches and then filter out the boxes which fall outside the valid range of each branch. We then use NMS or soft-NMS [3] to combine the detection outputs of multiple branches to obtain the final results.

Fast Inference Approximation To further speedup our network, we can use only one major branch as an approximation of TridentNet during inference. In particular, we set its valid range as $[0, \infty]$ to predict objects of all scales. For a three-branch network as in Figure 2, we use the middle branch as our major branch since its valid range covers both large and small objects. In this way, our fast TridentNet incurs no additional time cost compared with a standard Faster R-CNN detector. Surprisingly, we find that this approximation only suffers from a slight performance drop compared with the original TridentNet. Detailed ablation of inference approximation could be referred in Section 5.2.

5. Experiments

In this section, we conduct experiments on the COCO dataset [28]. Following [2, 26], we train models on the union of 80k training images and 35k subset of validation images (*trainval35k*), and evaluate on a set of 5k validation images (*minival*). We also report the final results on a set of 20k test images (*test-dev*). We first describe the implementation details of TridentNets and training settings in Section 5.1. We then conduct thorough ablation experiments to validate the proposed method in Section 5.2. Finally, Section 5.3 compares the results of TridentNets with state-of-the-art methods on the *test-dev* set.

5.1. Implementation Details

We re-implement Faster R-CNN [37] as our baseline method in MXNet [8]. Following other standard detectors [17, 37], the network backbones are pre-trained on the ImageNet [38] and then finetuned on the detection dataset. The input images are resized to have a short side of 800 pixels. Both the baselines and TridentNets are trained in an end-to-end manner. By default, we train in a batch size of 16 on 8 GPUs. We train 12 epochs in total, with learning rate starting from 0.02 and decreased by a factor of 0.1 after the 8th and 10th epoch. We adopt the output of conv4 stage in ResNet [20] as the backbone feature map and conv5 stage as the rcnn head in both baseline and TridentNet. For each image, we sample 128 ROIs per-branch for TridentNet. If not otherwise specified, we utilize three branches as our default TridentNet structure. The dilated rates are set to 1, 2 and 3 in three branches, respectively. When adopting scale-aware training scheme for TridentNet, we set the valid ranges of three branches as $[0, 90]$, $[30, 160]$ and $[90, \infty]$, respectively.

For the evaluation, we report the standard COCO evaluation metric of Average Precision (AP) [28] as well as AP_{50}/AP_{75}. We also report COCO-style AP_s, AP_m and AP_l on objects of small (less than 32×32), medium (from 32×32 to 96×96) and large (greater than 96×96) sizes.

5.2. Ablation Studies

Backbone	Method	Multi-branch	Weight-sharing	Scale-aware	AP	AP_{50}	AP_s	AP_m	AP_l
ResNet-101	(a) Baseline	-	-	-	37.9	58.8	20.0	43.0	52.8
	(b) Multi-branch	✓			39.0	59.7	20.6	43.5	55.1
	(c) TridentNet w/o scale-aware	✓	✓		40.3	61.1	21.8	44.7	**56.7**
	(d) TridentNet w/o sharing	✓		✓	39.3	60.4	21.4	43.8	54.2
	(e) TridentNet	✓	✓	✓	**40.6**	**61.8**	**23.0**	**45.5**	55.9
ResNet-101-Deformable	(a) Baseline	-	-	-	39.9	61.3	21.6	45.0	55.6
	(b) Multi-branch	✓			40.5	61.5	21.9	45.3	56.8
	(c) TridentNet w/o scale-aware	✓	✓		41.4	62.8	23.4	45.9	**57.4**
	(d) TridentNet w/o sharing	✓		✓	40.3	61.6	22.9	45.0	55.0
	(e) TridentNet	✓	✓	✓	**41.8**	**62.9**	**23.6**	**46.8**	57.1

Table 2: Object detection results evaluated on the COCO *minival* set [28]. Starting from our baseline, we gradually add multi-branch design, weight sharing among branches and scale-aware training scheme in our TridentNet for ablation studies.

Components of TridentNet. First, we analyze the importance of each component in TridentNet. The baseline methods (Table 2(a)) are evaluated on ResNet-101 and ResNet-101-Deformable [11] backbones. Then we gradually apply our multi-branch architecture, weight sharing design and scale-aware training scheme.

1. **Multi-branch**. Based on the pilot experiment, Table 2(b) evaluates a straightforward way to combine multiple branches of different dilation rates. This multi-branch variant improves over the baselines on AP for both ResNet-101 (from 37.9 to 39.0) and ResNet-101-Deformable (from 39.9 to 40.5), especially for large objects (2.3/1.2 increase). It indicates that even this simplest multi-branch design could benefit from different receptive fields.

2. **Scale-aware**. Table 2(d) shows the ablation results of adding scale-aware training based on multi-branch (Table 2(b)). It brings additional improvements (0.8/1.0 increase on ResNet-101/ResNet-101-Deformable) for small objects, but drops in performance for large objects. We conjecture that the scale-aware training design prevents each branch from training objects of extreme scales, but may also bring about over-fitting problem in each branch caused by reduced effective samples.

3. **Weight-sharing**. Applying weight sharing on multi-branch (Table 2(c)) and TridentNet (Table 2(e)), we could achieve consistent improvements on both two base networks. It demonstrates the effectiveness of weight-sharing that not only reducing number of parameters, but also improving the performance of detectors. With the help of weight-sharing (Table 2(e)), all branches share the same parameters which are fully trained on objects of all scales, thus alleviating the over-fitting issue in scale-aware training (Table 2(d)).

Branches	AP	AP_{50}	AP_s	AP_m	AP_l
1	33.2	53.8	17.4	38.4	46.4
2	35.9	56.7	**19.0**	40.6	51.2
3	**36.6**	**57.3**	18.3	**41.4**	**52.3**
4	36.5	**57.3**	18.8	**41.4**	51.9

Table 3: Object detection results on the COCO *minival* set [28] using different number of branches on ResNet-50.

Finally, TridentNets achieve significant improvements (2.7/1.9 AP increase) on the two base networks. It also reveals that the proposed TridentNet structure is compatible with methods like deformable convolution [11] which could adjust receptive field adaptively.

Number of branches. We study the choice of the number of branches in TridentNets. Table 3 shows the results using one to four branches. Note that we do not add scale-aware training scheme here to avoid elaborately tuning valid ranges for different number of branches. The results in Table 3 demonstrate that TridentNets consistently improve over the single-branch method (baseline) with 2.7 to 3.4 AP increase. As can be noticed, four branches do not bring further improvement over three branches. Thus, considering the complexity and performance, we choose three branches as our default TridentNet setting.

Stage of Trident blocks. We conduct ablation study on TridentNet to find the best stage to place trident blocks in ResNet. Table 4 shows the results of applying trident blocks in conv2, conv3 and conv4 stages, respectively. The corresponding feature map strides are 4, 8 and 16 for the three stages. Comparing with conv4 stage, TridentNets on conv2 and conv3 stages achieve minor increase over the baseline.

Stage	AP	AP_{50}	AP_s	AP_m	AP_l
Baseline	33.2	53.8	17.4	38.4	46.4
conv2	34.1	54.8	17.1	39.1	48.6
conv3	34.4	55.0	17.5	39.3	49.0
conv4	**36.6**	**57.3**	**18.3**	**41.4**	**52.3**

Table 4: Object detection results evaluated on the COCO *minival* set [28] by replacing conv blocks with trident blocks in different stages of ResNet-50.

This is because the strides in conv2 and conv3 feature maps are not large enough to widen the discrepancy of receptive fields between three branches.

Number of trident blocks. As conv4 stage in ResNet has multiple residual blocks, we also conduct ablation study to explore how many trident blocks are needed for TridentNet. Here we replace different number of residual blocks with trident blocks on conv4 on ResNet-101. The results in Figure 3 shows that when the number of trident blocks grows beyond 10, the performance of TridentNet becomes stable. It indicates the robustness of TridentNet with regards to the number of trident blocks when the disparity of receptive fields between branches is large enough.

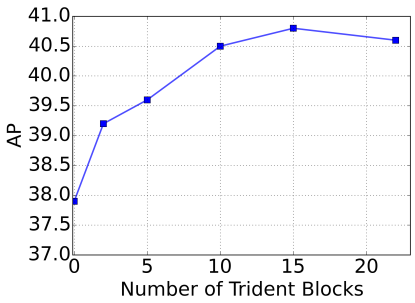


Figure 3: Object detection results on the COCO *minival* set [28] using different number of trident blocks on ResNet-101.

Performance of each branch. Our TridentNet has a multi-branch architecture, here we investigate the performance of each branch. Specifically, we evaluate the performance of each branch independently without scale range constraints. Table 5 shows the results of three single branches and three-branch method. As expected, through our scale-aware training, branch-1 with smallest receptive field achieves good results on small objects, branch-2 works well on objects with middle scale while branch-3 with largest receptive field is good at large objects. Finally, the three-branch method inherits the merits from three single branches and achieves the best results.

Method	Branch No.	AP	AP_{50}	AP_s	AP_m	AP_l
Baseline	-	37.9	58.8	20.0	43.0	52.8
TridentNet	Branch-1	31.5	53.9	22.0	43.3	29.9
	Branch-2	37.8	58.4	18.0	45.3	53.4
	Branch-3	31.9	48.8	7.1	37.9	**56.1**
	3 Branches	**40.6**	**61.8**	**23.0**	**45.5**	55.9

Table 5: Object detection results of each branch in TridentNet evaluated on the COCO *minival* set [28]. The dilation rates of three branches in Trident Network are set as 1, 2 and 3. The results are based on ResNet-101.

Scale-aware Ranges	AP	AP_{50}	AP_s	AP_m	AP_l
(a) Baseline	37.9	58.8	20.0	43.0	52.8
(b) $[0, 90]$, $[30, 160]$, $[90, \infty]$	37.8	58.4	18.0	**45.3**	53.4
(c) $[0, 90]$, $[0, \infty]$, $[90, \infty]$	39.3	60.1	19.1	44.6	56.4
(d) $[0, \infty]$, $[0, \infty]$, $[0, \infty]$	**40.0**	**61.1**	**20.9**	44.3	**56.6**

Table 6: Object detection results of branch-2 in TridentNet evaluated on the COCO *minival* set [28]. All results are based on ResNet-101 and share the same hyper-parameters.

To further reduce the inference time of TridentNet, we expect to use one major branch to approximate the three-branch results. We set the branch-2 as the major branch and investigate the effects of scale ranges in the scale-aware scheme in Table 6. Table 6(c) improves the major branch with 1.5 AP over the default scale-aware range setting (Table 6(b)) by enlarging its scale-aware range to train objects of all scales. Furthermore, enlarging scale-aware ranges for three branches achieves the best performance with 40.0 AP, which is already close to the three-branch results 40.6 AP. We hypothesize this may due to the weight-sharing strategy. Since the weights of the major branch are shared on other branches, it is better to train these weights consistently on objects of all scales for all branches.

5.3. Comparison with State-of-the-Arts

In this section, we evaluate our TridentNet on COCO *test-dev* set and compare with other state-of-the-art methods. In this part, we utilize ResNet-101 as our backbone network and increase the training epoch by $2\times$ ($3\times$ for multi-scale training). It is difficult to directly compare different detectors, since different methods may differ in backbone networks and training/test settings, including image pyramid scheme [40, 41], deformable convolutions [11], large-batch Batch-Normalization [33] and Soft-NMS [3]. Thus we report the results of our methods with two types in Table 7. TridentNet applies our method on Faster R-CNN with ResNet-101 as backbone and TridentNet* is

Method	Backbone	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
SSD513 [30]	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [14]	ResNet-101	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [27]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
Faster R-CNN+++ [20]	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [26]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Cascade R-CNN [5]	ResNet-101-FPN	42.8	62.1	46.3	23.7	45.5	55.2
Faster R-CNN by G-RMI [22]	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Deformable R-FCN [11]	Aligned-Inception-ResNet-Deformable	37.5	58.0	40.8	19.4	40.1	52.5
AttractionNet [15]	VGG16 + Wide ResNet	35.7	53.4	39.3	15.6	38.0	52.7
DCR [9]	ResNet-101-FPN-Deformable	41.7	64.0	45.9	23.7	44.7	53.4
SNIP [40]	ResNet-101-Deformable	44.4	66.2	44.9	27.3	47.4	56.9
SNIPER [41]	ResNet-101-Deformable	46.1	67.0	51.6	29.6	48.9	58.1
TridentNet	ResNet-101	42.7	63.6	46.5	23.9	46.6	56.6
TridentNet*	ResNet-101-Deformable	**48.4**	**69.7**	**53.5**	**31.8**	**51.3**	**60.3**

Table 7: Comparisons of single-model results for different object detection methods evaluated on the COCO *test-dev* set [28].

equipped with these common techniques. For multi-scale training/testing, the shorter side of images are set as [600, 800, 1000, 1200].

The results in Table 7 shows that our TridentNet with single-scale image as input could achieve 42.7 AP on *test-dev* set without bells and whistles. To compare fairly with SNIP and SNIPER, we apply image pyramids scheme (multi-scale training and testing), Soft-NMS, deformable convolutions and large-batch BN on TridentNet*. It further improves the results to 48.4 AP. To our best knowledge, for single models with ResNet-101 as backbone, our result is the best entry among state-of-the-art methods. It shows that our TridentNet is compatible with these common techniques.

Fast approximation. We also evaluate our method under the setting of using major branch for inference. The major branch could achieve 42.2/47.6 AP for TridentNet and TridentNet*, respectively, which only drops slightly compared to the three-branch methods (42.7/48.4). Note that performing inference with major branch introduces no more parameters or computations cost than a standard Faster R-CNN detector.

Compare with FPN. To compare fairly with current state-of-the-art feature pyramid method FPN [26], we utilize the *2fc* head instead of conv5 head for the baseline and TridentNet. Table 8 compares the results of these methods under the same training setting. It could be noted that TridentNet improves significantly over the baseline and FPN on all scales. It demonstrates the effectiveness of our scale specific feature maps, which are generated by TridentNet with the same representation power. Furthermore, as the fast approximation of TridentNet, the major branch variant achieves 41.0 AP which still improves 1.2 AP over the baseline and FPN.

Method	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
2fc Baseline	39.8	61.7	43.0	22.0	44.7	54.4
FPN† [26]	39.8	61.3	43.3	22.9	43.3	52.6
TridentNet	**42.0**	**63.5**	**45.5**	**24.9**	**47.0**	**56.9**

Table 8: Comparisons of detection results on the COCO *minival* set [28]. Following FPN†, all methods are based on ResNet-101 with *2fc* head using 2x training schedule.

6. Conclusion

In this paper, we present a simple object detection method Trident Network to build in-network scale-specific feature maps with the same representational power. A scale-aware training scheme is proposed for our multi-branch architecture to equip each branch with specialized ability for corresponding scales. The fast inference method with the major branch makes TridentNet achieve significantly improvements over baseline methods without any extra parameters and computations. We believe that TridentNets could benefit current object detection and other vision tasks. We would like to explore this direction in the future work.

†Detectron: https://github.com/facebookresearch/Detectron/blob/master/MODEL_ZOO.md

References

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984. 1

[2] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016. 3, 5

[3] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-NMS-improving object detection with one line of code. In *ICCV*, 2017. 5, 7

[4] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*, 2016. 3

[5] Z. Cai and N. Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018. 2, 8

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 3

[7] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 3

[8] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems. In *NIPS Workshop*, 2015. 5

[9] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, and T. Huang. Revisiting rcnn: On awakening the classification power of faster rcnn. In *ECCV*, 2018. 8

[10] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 1, 2

[11] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017. 3, 5, 6, 7, 8

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

[13] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014. 1

[14] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 2, 8

[15] S. Gidaris and N. Komodakis. Attend refine repeat: Active box proposal generation via in-out localization. In *BMVC*, 2016. 8

[16] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 2

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 2, 5

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017. 2

[19] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015. 2

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4, 5, 8

[21] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297. 1990. 3

[22] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017. 1, 3, 8

[23] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *CVPR*, 2016. 3

[24] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Light-head R-CNN: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017. 2

[25] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. DetNet: Design backbone for object detection. In *ECCV*, 2018. 3

[26] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 2, 3, 5, 8

[27] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *ICCV*, 2017. 2, 8

[28] 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 *ECCV*, 2014. 3, 4, 5, 6, 7, 8

[29] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018. 1, 3

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 1, 2, 3, 8

[31] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 1

[32] W. Luo, Y. Li, R. Urtasun, and R. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *NIPS*, 2016. 3

[33] C. Peng, T. Xiao, Z. Li, Y. Jiang, X. Zhang, K. Jia, G. Yu, and J. Sun. MegDet: A large mini-batch object detector. In *CVPR*, 2018. 7

[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1, 2

[35] J. Redmon and A. Farhadi. YOLO9000: Better, faster, stronger. In *CVPR*, 2017. 2

[36] J. Redmon and A. Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[37] 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, 3, 4, 5

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 5

[39] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016. 3

[40] B. Singh and L. S. Davis. An analysis of scale invariance in object detection–SNIP. In *CVPR*, 2018. 1, 3, 5, 7, 8

[41] B. Singh, M. Najibi, and L. S. Davis. SNIPER: Efficient multi-scale training. In *NIPS*, 2018. 1, 3, 7, 8

[42] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013. 2

[43] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2, 3

[44] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Single-shot refinement neural network for object detection. In *CVPR*, 2018. 2

[45] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 3