RetinaNet

Focal Loss for Dense Object Detection

Tsung-Yi Lin Priya Goyal Ross Girshick Kaiming He Piotr Dollár
Facebook AI Research (FAIR)

one-stage网络首次超越two-stage

2017
Computer Vision and Pattern Recognition

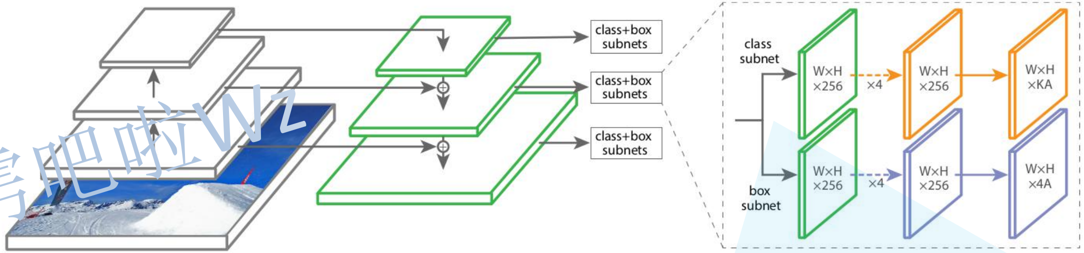


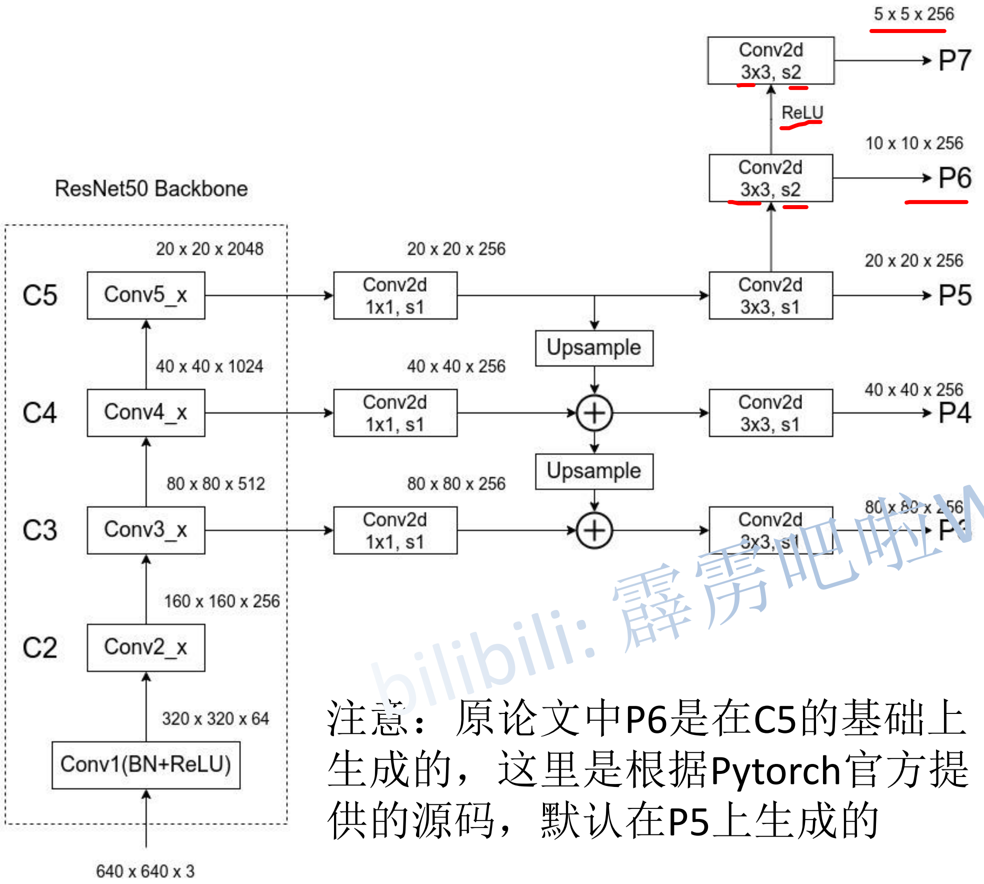
Figure 3. The one-stage **RetinaNet** network architecture uses a Feature Pyramid Network (FPN) [20] backbone on top of a feedforward ResNet architecture [16] (a) to generate a rich, multi-scale convolutional feature pyramid (b). To this backbone RetinaNet attaches two subnetworks, one for classifying anchor boxes (c) and one for regressing from anchor boxes to ground-truth object boxes (d). The network design is intentionally simple, which enables this work to focus on a novel focal loss function that eliminates the accuracy gap between our one-stage detector and state-of-the-art two-stage detectors like Faster R-CNN with FPN [20] while running at faster speeds.

https://arxiv.org/abs/1708.02002

RetinaNet

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

RetinaNet



[2]RetinaNet uses feature pyramid levels P_3 to P_7, where P_3 to P_5 are computed from the output of the corresponding ResNet residual stage (C_3 through C_5) using top-down and lateral connections just as in [20], P_6 is obtained via a 3×3 stride-2 conv on C_5, and P_7 is computed by applying ReLU followed by a 3×3 stride-2 conv on P_6. This differs slightly from [20]: (1) we don't use the high-resolution pyramid level P_2 for computational reasons, (2) P_6 is computed by strided convolution instead of downsampling, and (3) we include P_7 to improve large object detection. These minor modifications improve speed while maintaining accuracy.

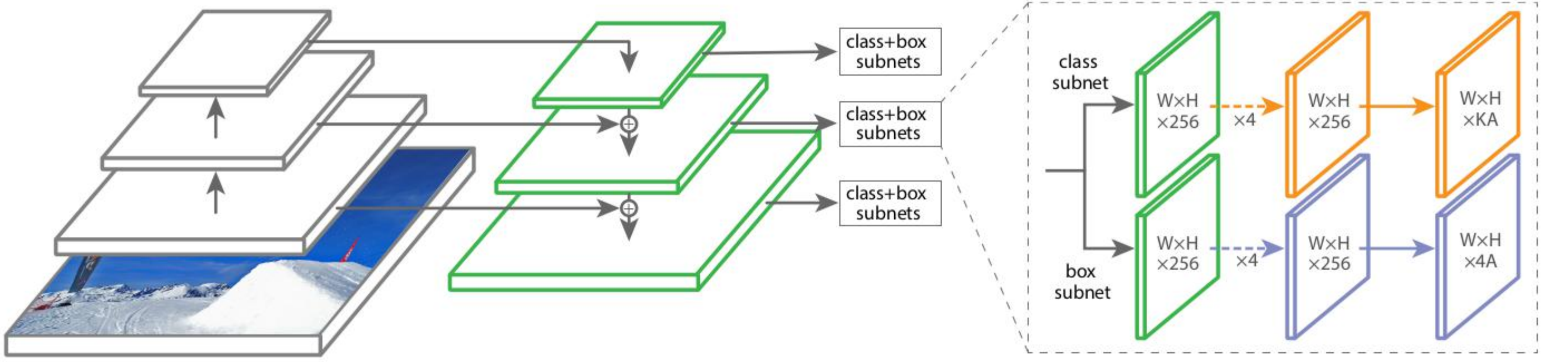
Scale	Ratios
$32\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$	{1:2, 1:1, 2:1}
$64\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$	{1:2, 1:1, 2:1}
$128\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$	{1:2, 1:1, 2:1}
$256\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$	{1:2, 1:1, 2:1}
$512\{2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$	{1:2, 1:1, 2:1}

注意：原论文中P6是在C5的基础上生成的，这里是根据Pytorch官方提供的源码，默认在P5上生成的

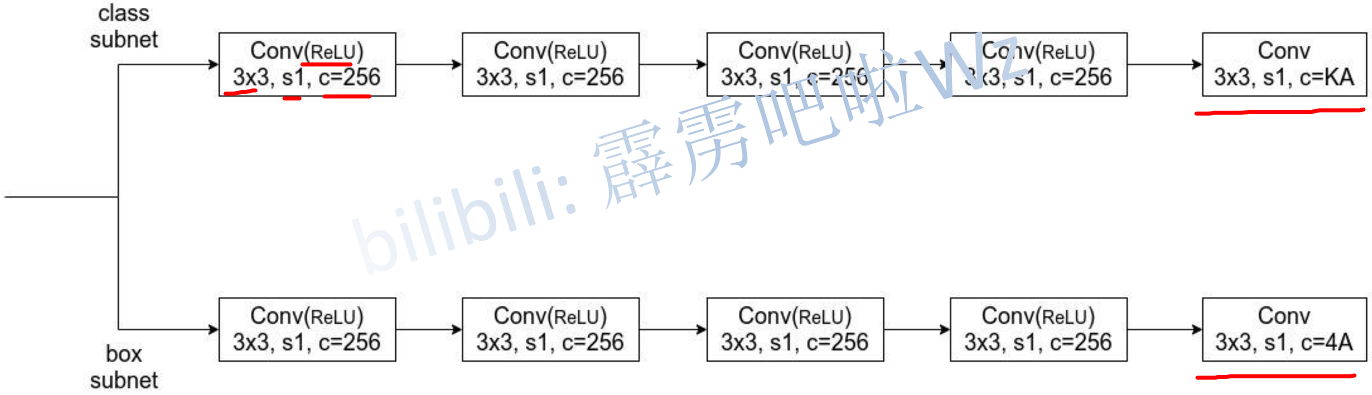
$$512 \times 2^{\frac{2}{3}} \approx 813$$

RetinaNet

权值共享



(a) ResNet (b) feature pyramid net (c) class subnet (top) (d) box subnet (bottom)



RetinaNet

class detection and with adjusted thresholds. Specifically, anchors are assigned to ground-truth object boxes using an intersection-over-union (IoU) threshold of 0.5; and to background if their IoU is in [0, 0.4). As each anchor is assigned to at most one object box, we set the corresponding entry in its length K label vector to 1 and all other entries to 0. If an anchor is unassigned, which may happen with overlap in [0.4, 0.5), it is ignored during training. Box regression targets are computed as the offset between each anchor and its assigned object box, or omitted if there is no assignment.

正负样本匹配

1. IoU >= 0.5 , 正样本
2. IoU < 0.4, 负样本
3. IoU ∈ [0.4, 0.5), 舍弃

RetinaNet

分类损失　　回归损失

$$Loss = \boxed{\frac{1}{N_{POS}} \sum_i L_{cls}^i} + \boxed{\frac{1}{N_{POS}} \sum_j L_{reg}^j}$$

L_{cls} ： Sigmoid Focal Loss

L_{reg} ： L1 Loss

N_{pos} ： 正样本的个数

i ： 所有的正负样本

j ： 所有的正样本

沟通方式

1.github

https://github.com/WZMIAOMIAO/deep-learning-for-image-processing

2.bilibili

https://space.bilibili.com/18161609/channel/index

3.CSDN

https://blog.csdn.net/qq_37541097/article/details/103482003