Corner anchor free and loss drive feature selection anchor free

CornerNet与feature selection anchor free表格总结

- CornerNet：用左上角、右下角的一组点代替边界框 + 新型的池化方式Corner pooling 。
- Feature selection anchor free：依据anchor-free分支计算loss来在线选择特征图 + anchor-based与anchor-free联合训练才能显著提高性能。
- RepPoints：直接去预测9个representative points（这些顶点并没有明确的语义），然后找出包围这9个点的最紧框去和GT计算loss。然后loss只会回传给对生成这个框有贡献的那些点。
- 对比表格总结链接：
  论文对比总结\CornerNet、Reppoints与Feature Selective Anchor-Free Module for Single-Shot Object Detection对比分析.html

- CornerNet: Detecting Objects as P
  aired Keypoints ECCV.2018

CornerNet

- **主要思想就是把boxes看作左上和右下两个点，这样目标检测就变成了两个关键点的检测。**

- 通过卷积网络生成两个heatmap分别对应左上角和右下角的点，并生成一个embedding vector，属于相同目标的嵌入向量更相似，利用embedding vector来把属于同一类别的点组合起来构成box。

- 论文提出一种针对关键点预测的新池化方式，即corner pooling。

传统方法的不足

- 1. Anchor-based的方法需要anchor覆盖所有的ground truth box并且IOU比较大，所以就需要非常多密集分布的anchor，导致了正负样本的严重不平衡。
- 2. anchor的大小、比例与数量都需要人为的设计，在multi-scale的时候会更加明显，这种人为设定的方式可能不符合物体实际情况。
- 3.作者受到keypoint问题的启发，想到用关键点检测找到top-left和Bottom-right两个点，来准确框出一个目标。

Motivation

- 灵感来源于Newell的多人姿态估计研究中的Bottom-Up的思想，利用embedding后的角点的距离区分左上角和右下角的corner是否属于一个类别，从而确定anchor。
- 文章的Corner Pooling来确保网络得到足够的信息，这比一般的临近位置pooling更加有效。
- 本文中，在CornerNet，卷积神经网络预测两份热力图，来代表不同类别的角点位置，一份代表左上角，一份代表右下角。同时网络还为每一个检测到的点预测一个特征向量，使得来自同一个物体之间的两个角点的向量之间的距离很小，为了得到更为致密的检测框，网络还输出一个偏移量，以此来轻微地调整角点的位置。

主要框架

- 1. 输入一张图像，经过backbone网络（Hourglass network）后，得到feature map。
- 2. 将feature map同时输入到两个branch，分别用于预测Top-Left Corners和Bottom-right Corners。
- 3. 两个branch都会先经过一个叫Corner Pooling的网络，最后输出三个结果，分别是Heatmaps. Embeddings. Offsets。
- 4. 根据Heatmaps能够得到物体的左上角点和右下角点，根据Offsets对左上角和右下角点位置进行更加精细的微调，根据Embeddings可以将同一个物体的左上角和右下角点进行匹配，得到到最终的目标框。

Corner anchor free and loss drive feature selection anchor free

主要框架

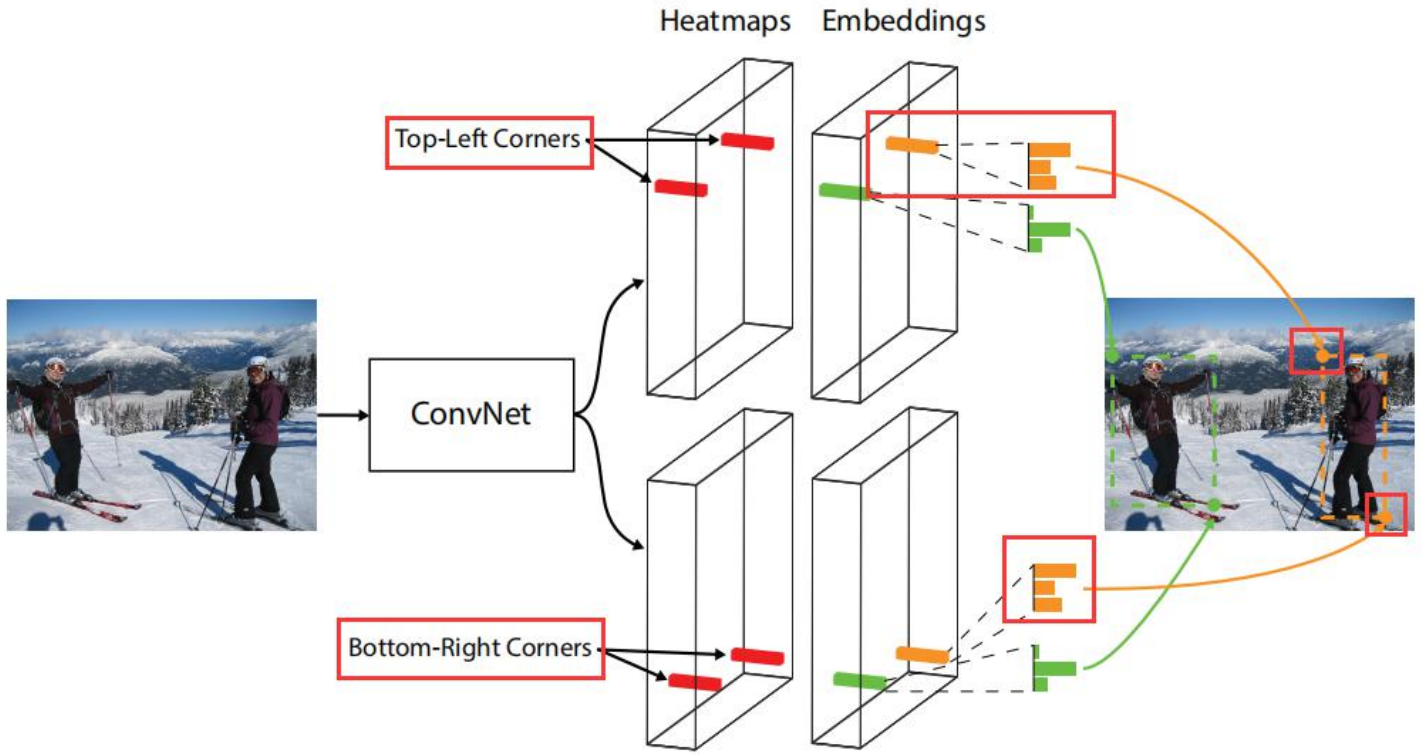


Fig. 1 We detect an object as a pair of bounding box corners grouped together. A convolutional network outputs a heatmap for all top-left corners, a heatmap for all bottom-right corners, and an embedding vector for each detected corner. The network is trained to predict similar embeddings for corners that belong to the same object.

Detecting Corners

- 每个heatmaps集合的形式都是C*H*W，其中C代表的是检测目标的类别数，H和W则代表heatmap的分辨率。
- *P(cij)*代表在预测的特征图上位置*(i,j)*处类别*c*的得分，*y(cij)*是增加了非标准高斯函数的实际标注特征图上对应位置对应类别的得分。
- 输出：设计一个focal loss公式计算 (i, j)像素是类别c的loss

$$y_{cij} = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$L_{det} = \frac{-1}{N}\sum_{c=1}^{C}\sum_{i=1}^{H}\sum_{j=1}^{W}\begin{cases}(1-p_{cij})^{\alpha}\log(p_{cij}) & \text{if } y_{cij}=1\\ (1-y_{cij})^{\beta}(p_{cij})^{\alpha}\log(1-p_{cij}) & \text{otherwise}\end{cases}$$

x和y表示的是以（i , j ）为坐标原点的相对坐标。y_{cij}在真值位置值为1，真值附近一定范围内的值不为0，与真值位置的距离越大，其值逐渐衰减。

α和β是一组超参（作者分别设定2，4 ），用来控制图像中每个点对loss的贡献程度。

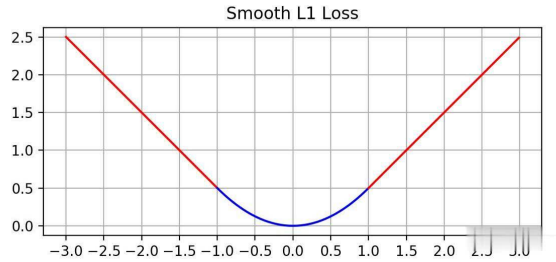
Corner adjustment

- 考虑到特征图与原图之间的分辨率差距，将特征图上某位置映射回原图通常会带来位置精度损失，而这对预测框和标注框之间的重叠比影响较大，因此，网络输出位置偏移量在角点映射回原图之前来轻微地调整其位置。

$$o_k = \left(\frac{x_k}{n} - \left\lfloor \frac{x_k}{n} \right\rfloor, \frac{y_k}{n} - \left\lfloor \frac{y_k}{n} \right\rfloor\right)$$

- loss计算采用Smooth1 Loss：从两个方面限制梯度：当预测框与GT差别过大时，梯度值不至于过大；当预测框与GT差别很小时，梯度值足够小。

$$L_{off} = \frac{1}{N}\sum_{k=1}^{N} SmoothL1Loss(o_k, \hat{o}_k)$$



Grouping Corners

- 主要思想是，在生成角点heatmap时，同时生成一个embedding vector， loss具有两种，L_{pull}和L_{push}；如果一个左上角角点和右下角角点属于同一个物体，那么他们的embeddng vector之间的距离就应该非常小，根据这个距离来对Corner点进行匹配。
- Embedding本质是用一个低维的向量表示一个物体，可以是一个词，或是一个商品。这个embedding向量的性质是能使距离相近的向量对应的物体有相近的含义。比如 Embedding(复仇者联盟)和Embedding(钢铁侠)之间的距离很接近，与Embedding(乱世佳人)的距离就会远一些。

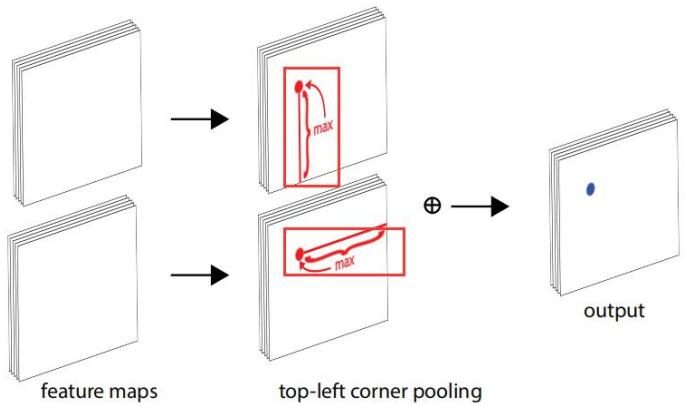
$$L_{pull} = \frac{1}{N}\sum_{k=1}^{N}\left[\left(e_{t_k} - e_k\right)^2 + \left(e_{b_k} - e_k\right)^2\right],$$

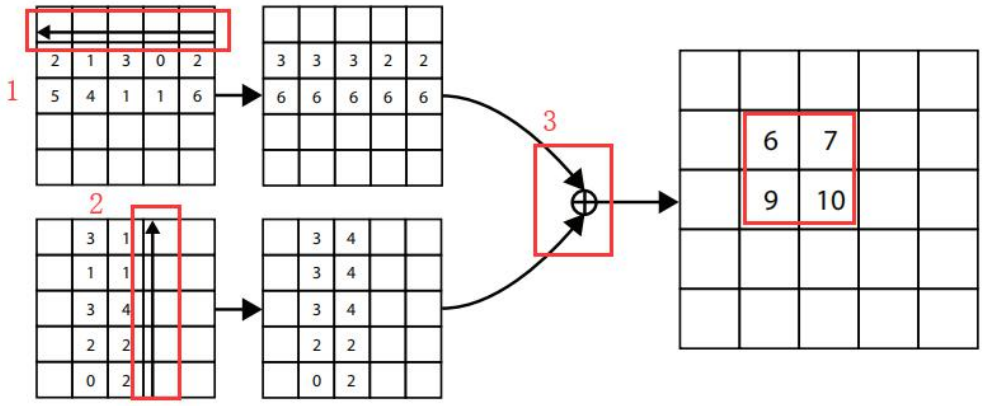
$$L_{push} = \frac{1}{N(N-1)}\sum_{k=1}^{N}\sum_{j=1,j\neq k}^{N}\max(0, \Delta - |e_k - e_j|)$$

左式中e_{tk}和e_{bk}表示图像中第k个目标的top-left点和bottom-right点的embedding向量，N表示物体类别的数量。 $e_k = (e_{t_k} + e_{b_k})/2$表示两个embedding向量的均值。

Corner Pooling

- 下图是top-left corner的 Corner Pooling过程。在水平方向，从最右端开始往最左端遍历，每个位置的值都变成从最右到当前位置为止，出现的最大的值。在垂直方向上，从下到上扫描填充遇到的最大的值。同理，bottom-right corner的Corner Pooling最左端往最右端. 最上往最下遍历。
- 通过Corner Pooling的方式能够一定程度上体现出当前点出发的射线是否与物体向交。





使用总loss驱动Corner对选择

- 完整的training loss是：

$$L = L_{det} + \alpha L_{pull} + \beta L_{push} + \gamma L_{off}$$

- α = β = 0.1，γ = 1

- 在训练期间，设置网络的输入分辨率511×511,导致输出分辨率为128×128。为了减少过拟合，采用标准的数据增强技术，包括随机水平翻转、随机缩放、随机裁剪和随机色彩抖动，其中包括调整图像的亮度，饱和度和对比度。 最后，将PCA应用于输入图像。

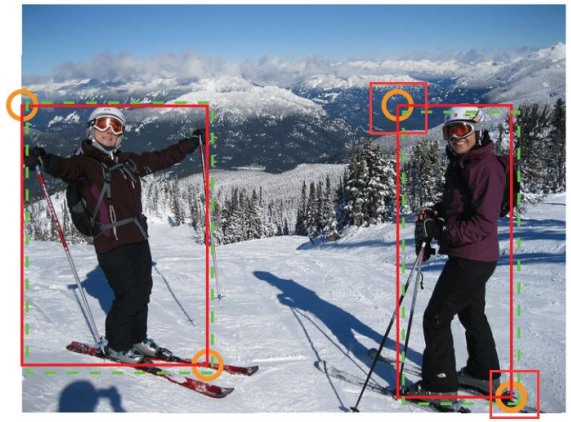
实验

- conner pooling的消融比实验，可以看出添加conner pooling（第二行）对能提升2.0mAP，在目标尺度比较大的数据中提升3.6。

	AP	AP^{50}	AP^{75}	AP^{s}	AP^{m}	AP^{l}
w/o corner pooling	36.5	52.0	38.8	17.5	38.9	49.4
w/ corner pooling	38.4	53.8	40.9	18.6	40.5	51.8
improvement	+2.0	+2.1	+2.1	+1.1	+2.4	+3.6

实验

- 对于每一个角点，只有一个正样本（实际标注），其他的位置都是负样本。训练时，对于在正样本半径内的负样本减少惩罚，而不是同等地给予惩罚。这是因为一对离实际标注很近的角点，同样能够产生一个与实际标注框重叠比很高的检测框。
- Table2是关于对不同位置负样本采取不同权重的损失函数的效果。第一行是不采用惩罚策略；第二行是采用固定半径值的效果，提升2.7；第三行是采用基于目标计算得到的半径值的效果，再次提升2.8。



	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l
w/o reducing penalty	32.9	49.1	34.8	19.0	37.0	40.7
fixed radius	35.6	52.5	37.7	18.7	38.5	46.0
object-dependent radius	38.4	53.8	40.9	18.6	40.5	51.8

实验

- 下表中具有沙漏网络的CornerNet的性能比具有FPN的CornerNet的性能高出8.2%，而具有沙漏网络的anchor-based的探测器的性能高出5.5%。
- 结果表明，骨干网的选择很重要，沙漏网络对于CornerNet的性能至关重要。

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l
FPN (w/ ResNet-101) + Corners	30.2	44.1	32.0	13.3	33.3	42.7
Hourglass + Anchors	32.9	53.1	35.6	16.5	38.5	45.0
Hourglass + Corners	38.4	53.8	40.9	18.6	40.5	51.8

Corner anchor free and loss drive feature selection anchor free

实验

- 通过多尺度评估，CornerNet的AP达到42.2%，在现有的一阶段方法中处于最先进水平，并且与两阶段方法相比具有竞争力。

Method	Backbone	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l	AR^1	AR^{10}	AR^{100}	AR^s	AR^m	AR^l
Two-stage detectors													
DeNet (Tychsen-Smith and Petersson, 2017a)	ResNet-101	33.8	53.4	36.1	12.3	36.1	50.8	29.6	42.6	43.5	19.2	46.9	64.3
CoupleNet (Zhu et al., 2017)	ResNet-101	34.4	54.8	37.2	13.4	38.1	50.8	30.0	45.0	46.4	20.7	53.1	68.5
Faster R-CNN by G-RMI (Huang et al., 2017)	Inception-ResNet-v2 (Szegedy et al., 2017)	34.7	55.5	36.7	13.5	38.1	52.0	-	-	-	-	-	-
Faster R-CNN+++ (He et al., 2016)	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9	-	-	-	-	-	-
Faster R-CNN w/ FPN (Lin et al., 2016)	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2	-	-	-	-	-	-
Faster R-CNN w/ TDM (Shrivastava et al., 2016)	Inception-ResNet-v2	36.8	57.7	39.2	16.2	39.8	52.1	31.6	49.3	51.9	28.1	56.6	71.1
D-FCN (Dai et al., 2017)	Aligned-Inception-ResNet	37.5	58.0	-	19.4	40.1	52.5	-	-	-	-	-	-
Regionlets (Xu et al., 2017)	ResNet-101	39.3	59.8	-	21.7	43.7	50.9	-	-	-	-	-	-
Mask R-CNN (He et al., 2017)	ResNeXt-101	39.8	62.3	43.4	22.1	43.2	51.2	-	-	-	-	-	-
Soft-NMS (Bodla et al., 2017)	Aligned-Inception-ResNet	40.9	62.8	-	23.3	43.6	53.3	-	-	-	-	-	-
LH R-CNN (Li et al., 2017)	ResNet-101	41.5	-	-	25.2	45.3	53.1	-	-	-	-	-	-
Fitness-NMS (Tychsen-Smith and Petersson, 2017b)	ResNet-101	41.8	60.9	44.9	21.5	45.0	57.5	-	-	-	-	-	-
Cascade R-CNN (Cai and Vasconcelos, 2017)	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2	-	-	-	-	-	-
D-RFCN + SNIP (Singh and Davis, 2017)	DPN-98 (Chen et al., 2017)	45.7	67.3	51.1	29.3	48.8	57.1	-	-	-	-	-	-
One-stage detectors													
YOLOv2 (Redmon and Farhadi, 2016)	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4
DSOD300 (Shen et al., 2017a)	DS/64-192-48-1	29.3	47.3	30.6	9.4	31.5	47.0	27.3	40.7	43.0	16.7	47.1	65.0
GRP-DSOD320 (Shen et al., 2017b)	DS/64-192-48-1	30.0	47.9	31.8	10.9	33.6	46.3	28.0	42.1	44.5	18.8	49.1	65.0
SSD513 (Liu et al., 2016)	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8	28.3	42.1	44.4	17.6	49.2	65.8
DSSD513 (Fu et al., 2017)	ResNet-101	33.2	53.3	35.2	13.0	35.4	51.1	28.9	43.5	46.2	21.8	49.1	66.4
RefineDet512 (single scale) (Zhang et al., 2017)	ResNet-101	36.4	57.5	39.5	16.6	39.9	51.4	-	-	-	-	-	-
RetinaNet800 (Lin et al., 2017)	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2	-	-	-	-	-	-
RefineDet512 (multi scale) (Zhang et al., 2017)	ResNet-101	41.8	62.9	45.7	25.6	45.1	54.1	-	-	-	-	-	-
CornerNet511 (single scale)	Hourglass-104	40.6	56.4	43.2	19.1	42.8	54.3	35.3	54.7	59.4	37.4	62.4	77.2
CornerNet511 (multi scale)	Hourglass-104	42.2	57.8	45.2	20.7	44.8	56.6	36.6	55.9	60.3	39.5	63.2	77.3

启发性或普适性

- 1. 用左上角与右下角的一对corner代替了anchor box。
- 2. 提出了一种新的池化方式用于检测边角：corner pooling。

总结

- 1. CornerNet = Detecting Corners + Grouping Corners + Corner Pooling。
- 2. 用左上角、右下角的一组点代替边界框 + 新型的池化方式Corner pooling 。
- 3. 在CornerNet中，卷积神经网络预测两份热力图，来代表不同类别的角点位置，一份代表左上角，一份代表右下角。同时网络还为每一个检测到的点预测一个特征向量，使得同一个物体之间的两个角点的向量之间的距离很小，为了得到更为致密的检测框，网络输出一个偏移量来轻微地调整角点的位置。

总结

- 1. 怎么检测这个两个点？
  生成keypoint的heatmap，heatmap中响应值最大的位置就是点的位置。

- 2. 怎么知道这两个点所组成的框包含物体的类别？
  Corner响应值最大所在的channel即对应了物体的类别。

- 3. 当图像中有多个物体时，怎么知道哪些点可以组成框？（哪些左上角的点和哪些右下角的点能够组成有效的框）
  生成embedding向量，用向量的距离衡量两个Corner是否可以组成对。

总结

- 4. Loss是什么形式？
  loss总共分了三个部分，一部分是用于定位keypoint点的detecting loss，一个是用于精确定位的offset loss，一个是用于对Corner点进行配对的grouping loss。
- 5. 网络结构是怎么样的？
  使用Hourglass作为backbone，使用Corner Pooling构造了prediction module，用来得到最终的结果。
- 6. 有没有什么比较新奇的东西？
  提出的Corner Pooling。第一次使用检测点的方法检测物体。

我的观点

- 1. 把检测目标框变成一对关键点的问题，即左上角和右下角，这样就消除了锚框的设计麻烦。另外，采用的角点池化（corner pooling）技术帮助CNN更好地定位角点位置。

- 2. 左上角右下角关键点分开预测，速度上计算比较慢，一定程度上也可能忽视了两点之间的关联信息。

Corner anchor free and loss drive feature selection anchor free

- Feature Selective Anchor-Free Module for Single-Shot Object Detection CVPR.2019

FSAF

- FSAF的出发点是让图像中的每个目标实例自主选择最适合的特征层，这样，在feature level选择这一步骤中，就不需要再进行anchor的设置，实现了anchor-free。
- 在train阶段，根据loss自动选择最佳的feature level。被选择的feature level就进行后续的回归和预测。在inference阶段，FSAF可以独立预测，也可以与anchor-based方法相结合。
- feature 选择的依据有原来的instance size变成了instance content，使得每个实例能够自由的选择最优层级来优化网络。

传统方法的不足

- 1.预设的anchor size不是任意的，被束缚住尺寸的anchor有助于后续的统一学习，但是对于形状特殊的目标学习不到，如长细条的画板，电线等。
- 2.anchor的数量是十分密集的，即使用nms或者soft-nms去重，负样本的数量依然非常多，与正样本比例严重失衡，RetinaNet等采取合适的比例参数平衡这个差异。
- 3.Anchor数量巨多，需要每一个都进行IOU计算，耗费巨大的算力，降低了效率，步骤十分繁琐，而这些冗余其实是可以消灭的。

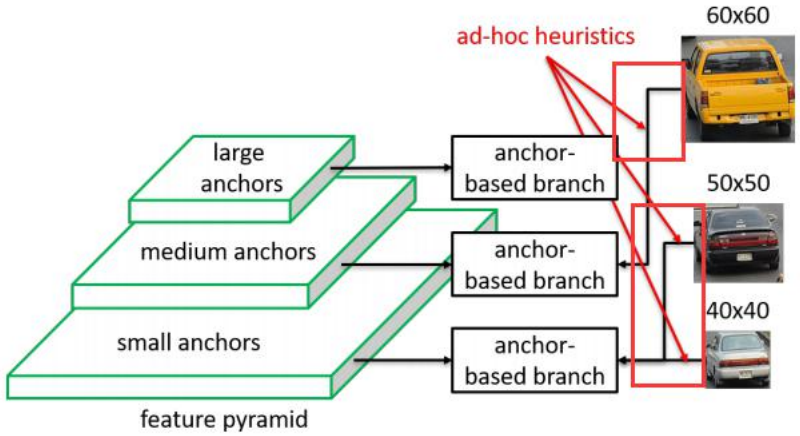
Motivation

- 1. 以FPN为例，一般来说，浅层特征对应小物体，深层特征对应大物体。但是这种大，小的界定是人为进行选择的，下图中，几乎同样角度的车的图像，长宽的差距也是相同的，但是60x60size的图像分配到了medium anchors，而50x50size和40x40size都被分到了small anchors，这是不符合正确认识。

- 2. overlap-based anchor sampling：anchor的密集采样，导致计算量提升，效率下降。

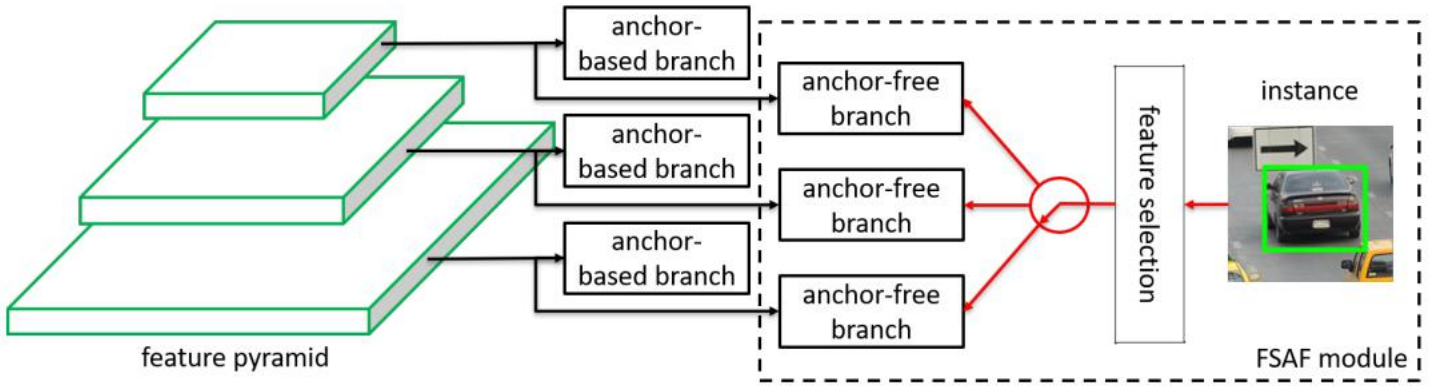
$$l' = \lfloor l_0 + \log_2(\sqrt{wh}/224) \rfloor$$

fpn选择特征层的公式



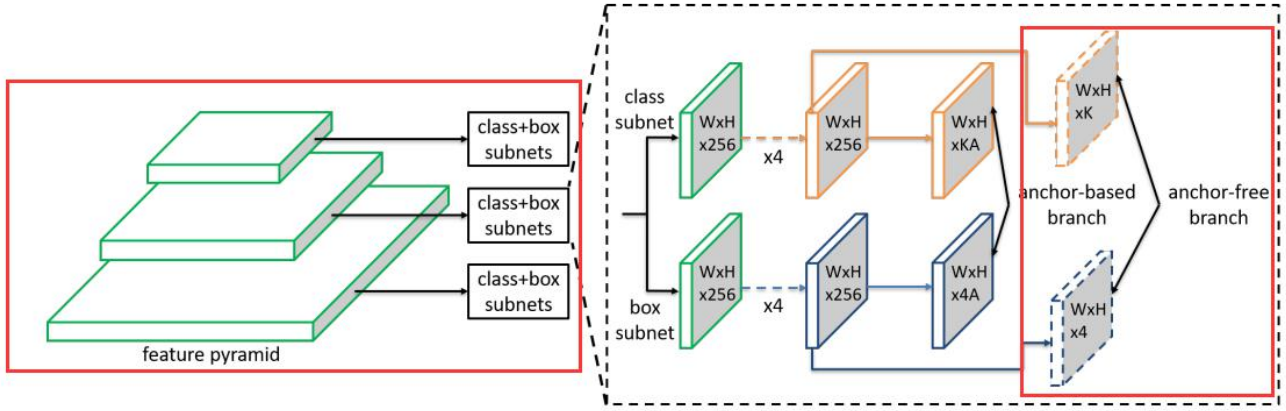
主要框架

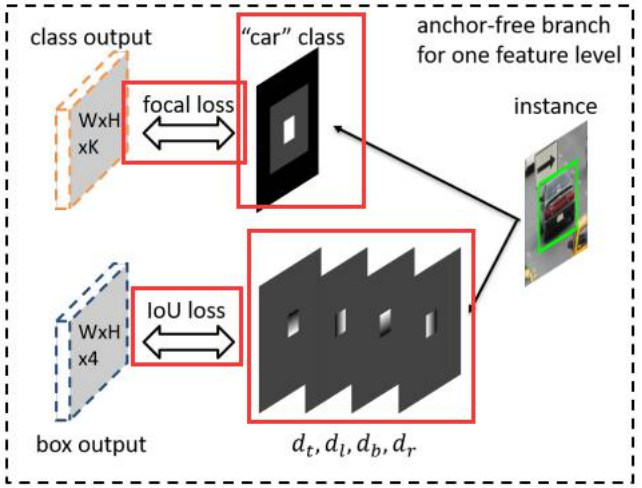
- 1. FSAF主要包含无anchor分支的实现及在线特征选择两部分，本质是anchor free的分支使用在线特征选择。
- 2. 在train时， anchor free的分支添加到特征金字塔的每一层，从而可以以任意层次对box进行编码解码。将每个实例动态的放置在最适合的特征层次上。
- 3. 在inference时，FSAF可以结合anchor based的分支并行的输出预测结果。



主要框架

- 1. 在RetinaNet的输出端，FSAF分别引入两层额外的卷积层，分别用于anchor-free分支的分类及回归。3x3xK的卷积层添加到分类分支. 回归分支之后，后接sigmoid函数，其预测每个位置上K个类别的概率，以anchor-free的方式预测框的偏移量 。
- 2. 白色部分为有效框，灰色部分为忽略框，对应到训练中，白色为正例，区域会填充1；灰色是忽略区域，不参与梯度回传；黑色为负例，区域会填充0。





主要框架

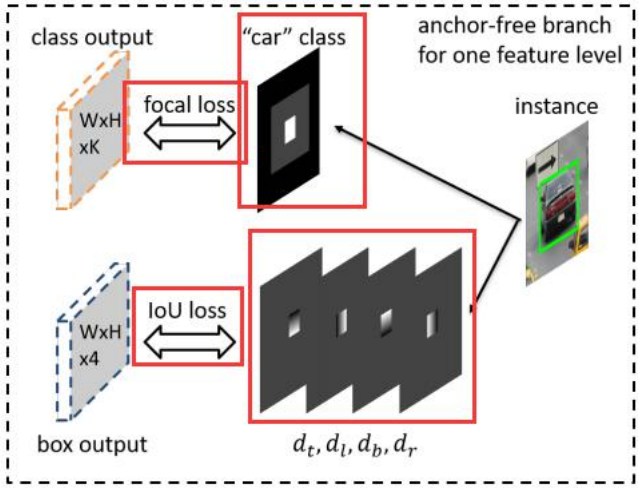
- 在线特征选择机制。每一个实例都经过所有的anchor-free branch来计算平均分类损失和平均回归损失，然后选择损失最小的anchor-free branch对应的损失作为最后的监督该实例的信号。
- 下图中最下面的这路梯度最小，训练阶段则只对最下面的这路进行梯度更新，上面的两路不做任何变化。

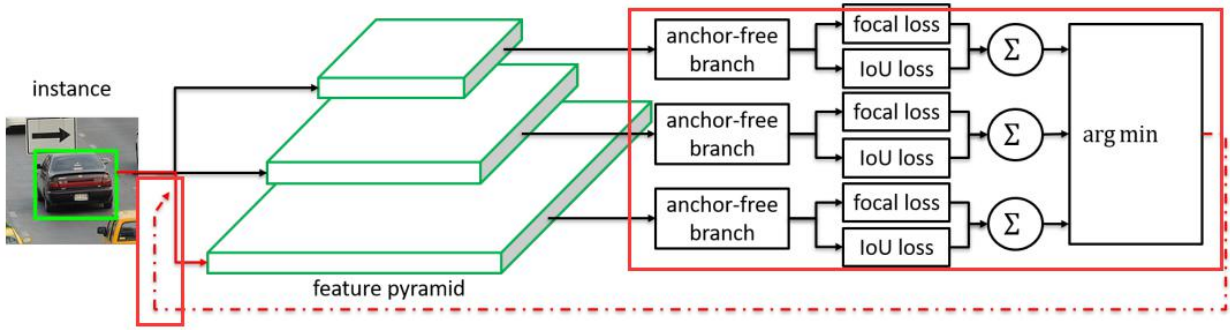
$$L_{FL}^{I}(l) = \frac{1}{N(b_e^l)} \sum_{i,j \in b_e^l} FL(l,i,j)$$

$$L_{IoU}^{I}(l) = \frac{1}{N(b_e^l)} \sum_{i,j \in b_e^l} IoU(l,i,j)$$

$$l^* = \arg\min_l L_{FL}^{I}(l) + L_{IoU}^{I}(l)$$

N（b^l_e）为b^l_e的像素数





Corner anchor free and loss drive feature selection anchor free

实验

- 1. 只使用Heuristic feature selection效果差于只使用Anchor-based branches；只使用Online feature selection 效果持平只使用Anchor based branches; Online feature selection 和 Anchor based branches结合使用效果最好，比基线提升1.5个点。
- 2.基于anchor-based的RetinaNet网络中插入anchor-free模块后，像滑板、飞盘等这种较小. 尺度比较不规则的物体检测效果会更好。

	Anchor-based branches	Anchor-free branches: Heuristic feature selection Eqn. (3)	Anchor-free branches: Online feature selection Eqn. (2)	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet	✓			35.7	54.7	38.5	19.5	39.9	47.5
Ours		✓		34.7	54.0	36.4	19.0	39.0	45.8
			✓	35.9	55.0	37.9	19.8	39.6	48.2
	✓	✓		36.1	55.6	38.7	19.8	39.7	48.9
	✓		✓	**37.2**	**57.2**	**39.4**	**21.0**	**41.2**	**49.7**

Corner anchor free and loss drive feature selection anchor free

实验

- 1. 所有框都是anchor-free的检测结果，绿框表示其和anchor-based选择特征层一致，小目标在大分辨率层，大目标在小分辨率层；
- 2. 红色的表示其和anchor-based选择的特征层不一致。图像中小目标等更难的样本可以检测出来，证明其特征选择的有效。

a: RetinaNet (anchor-based, ResNeXt-101)　　b: Ours (anchor-based + FSAF, ResNet-50)

启发性或普适性

- 1. anchor-free的方法能够在精度上媲美 anchor-based，最大的功劳归于FPN(特征金字塔)与Focal Loss的提出。
- 2. 在每个位置只预测一个框的情况下，FPN 的结构对尺度起到了很好的弥补；
- 3. Focal loss解决正负样本类别不平衡的问题，通过减少容易分类的样本的权重，使模型在训练时更专注难分类的样本，改善模型的优化方向。

总结

- RetinaNet+FSAF = 依据anchor-free分支计算loss来在线选择特征图 + anchor-based与anchor-free联合训练才能显著提高性能

我的观点

- 1. FSAF可以看做是anchor-free系列的尝试，它并没有完全摒弃anchor，而是选择anchor-free分支作为特征层选择的工具，帮助每个实例动态选择最佳特征层，之后还是需要用anchor based计算分类和位置回归。
- 2. 从feature selection角度设计FSAF module来提升性能，从loss角度来看，其提升了梯度反传的效率，有点类似于SNIP，只更新特定scale内物体对应的梯度；但效率比SNIP更高。
- 3. 对提出的想法的性能调优也不简单，例如对重叠区域的处理，对回归范围的限制，如何将 target assign 给不同的 FPN level，head 是否 share 参数试验调优等。

Chapter

存在问题

- 1. 感觉现在的新出的方法特别多，将近期看过的文章作如下总结：
- 2. 总结链接为
  anchor based方法与anchor-free方法总结.xlsx
- 3. anchor-free方法的公式计算部分没有理解得很深刻，需要看源码来深入理解。

Corner anchor free and loss drive feature selection anchor free

anchor based改进

anchor based改进			
角度	论文	mAP	工作阐述
数据层面	Stitcher	41.3(Res-101-FPN)	Stitcher = 利用单张图片中小目标loss的阈值，将大中目标转换中小目标，重新加入训练。(相当于数据增广)
网络层面	FPN	36.2	采用多尺度特征融合(在当前层进行卷积操作之前，将上一层的特征图上采样与当前层的特征图相加，即通过对上一层特征上采样与浅层特征做融合得到深层特征)方式进行预测，这样可保留上一层的一些细节信息，提高准确度；FPN的每一层进行独立预测。
	TridentNet	48.4(TridentNet* + Image Pyramid ResNet-101-Deformable)	对同一物体使用不同大小的感受野来实现数据增广 + 共享权重参数带来对各种尺度适应性
	Adaptive Training Sample Selection	50.7((Multi-scale testing+ResNeXt-64x4d-101-DCN)	ATSS=自适应样本选择(计算所有的IOU的mean (mg)和std dev (vg)，根据自适应阈值tg = mg + vg，并增加条件：center在物体内部，对anchors进行筛选)

Corner anchor free and loss drive feature selection anchor free

anchor free方面

角度	论文	mAP	工作阐述
corner(假free)	CornerNet(鼻祖)	44.6	CornerNet = 用左上角、右下角的一组点代替边界框 + 新型的池化方式Corner pooling。
center point(假free)	CenterNet(CornerNet改进)	47.0(Hourglass-104)	在CornerNet的基础上加入了一支中心点预测，能够组成一个物体的要求不仅仅是两个顶点能匹配，同时这两个顶点定义的框的中心也要有对应的中心点相应，这样就缓解了很多奇怪的误检。CenterNet同时还提出了Center Pooling和Cascade Corner Pooling来改善对于中心点和顶点预测的精度。
key points(真free)	RepPoints(提出使用关键点来表示物体)	46.5	直接去预测9个representative points（这些顶点并没有明确的语义），然后找出包围这9个点的最紧框去和GT计算loss。然后loss只会回传给对生成这个框有贡献的那些点。
feature layer selection(假free)		42.1	RetinaNet+FSAF = 依据anchor-free分支计算loss来在线选择特征图 + anchor-based与anchor-free联合训练才能显著提高性能。

我接下来需要做的

- 1.双边分支网络BBN(few shot)在长尾分布问题上的对比分析。

- 2. 写一份reppoints的代码分析，读懂 reppoints的计算过程 。