目标检测算法中检测框合并策略技术综述

陈泰红

陈泰红

研究方向：机器学习、图像处理

物体检测（Object Detection）的任务是找出图像或视频中的感兴趣目标，同时实现输出检测目标的位置和类别，是机器视觉领域的核心问题之一，学术界已有将近二十年的研究历史。随着深度学习技术的火热发展，目标检测算法也从基于手工特征的传统算法转向了基于深度神经网络的检测技术。从最初 2013 年提出的 R-CNN、OverFeat，到后面的 Fast/Faster R-CNN、SSD、YOLO 系列，以及Mask R-CNN、RefineDet、RFBNet等(图 1,完整论文列表参见[1])。短短不到五年时间，基于深度学习的目标检测技术，在网络结构上，从 two stage 到 one stage，从 bottom-up only 到 Top-Down，从 single scale network 到 feature pyramid network，从面向 PC 端到面向移动端，都涌现出许多好的算法技术，这些算法在开放目标检测数据集上的检测效果和性能都很出色。

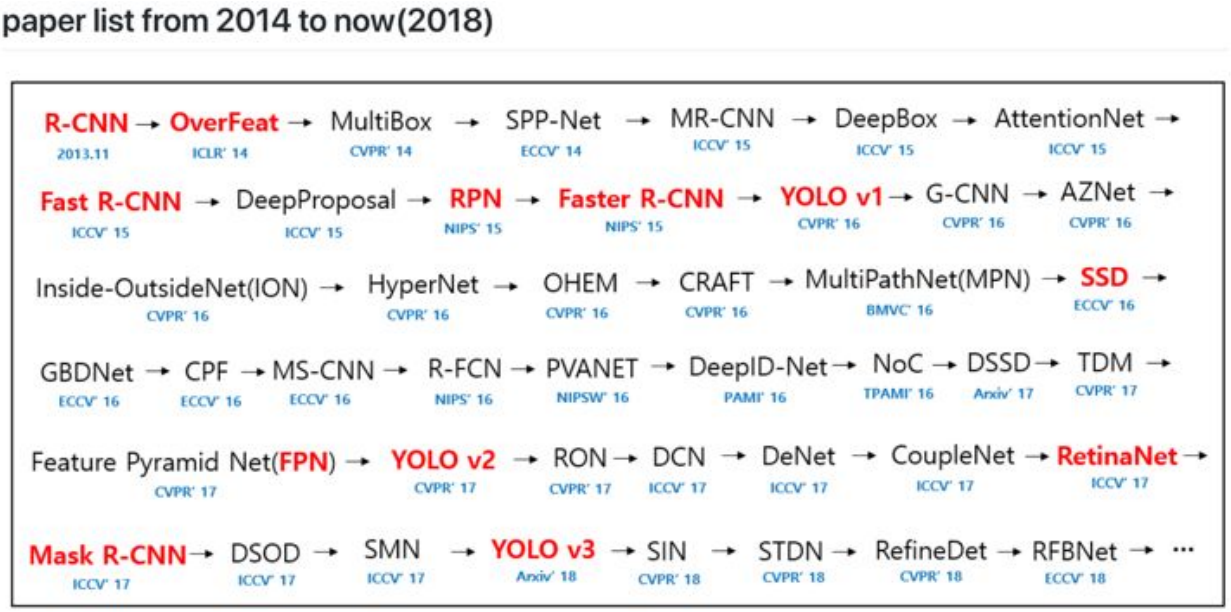


图 1 目标检测领域重要论文

物体检测过程中有很多不确定因素，如图像中物体数量不确定，物体有不同的外观、形状、姿态，加之物体成像时会有光照、遮挡等因素的干扰，导致检测算法有一定的难度。进入深度学习时代以来，物体检测发展主要集中在两个方向：two stage算法如R-CNN系列和one stage算法如YOLO、SSD等。两者的主要区别在于two stage算法需要先生成proposal（一个有可能包含待检物体的预选框），然后进行细粒度的物体检测。而one stage算法会直接在网络中提取特征来预测物体分类和位置。

two stage算法以及部分one stage算法（SSD系列），都需要对Region Proposal去重。比如经典的Faster RCNN算法会生产2000的Region Proposal，如果对所有的检测检测框进行分类和处理，会造成大量无效计算。使用某些算法对检测框去重，是目标检测领域的一个重要方向。

本文主要介绍在目标检测中使用的检测框去重，包括NMS，Soft-NMS,Softer-NMS,以及Relation Netwrok，ConvNMS，NMS Network，Yes-Net等，详细讲述NMS算法的演变和最新进展。

1、传统NMS算法

在目标检测中，常会利用非极大值抑制算法(NMS，non maximum suppression)对生成的大量候选框进行后处理，去除冗余的候选框，得到最佳检测框，以加快目标检测的效率。其本质思想是其思想是搜素局部最大值，抑制非极大值。非极大值抑制，在计算机视觉任务中得到了广泛的应用，例如边缘检测、人脸检测、目标检测（DPM，YOLO，SSD，Faster R-CNN）等。即如图 2所示实现效果，消除多余的候选框，找到最佳的bbox。

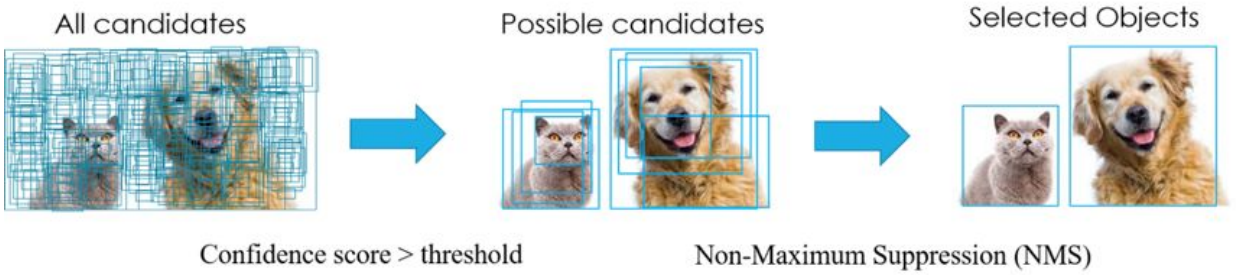


图 2 NMS消除冗余候选框

以图 2为例，每个选出来的Bounding Box检测框（既BBox）用（x,y,h,w,confidence score，Pdog,Pcat）表示，confidence score表示background和foreground的置信度得分，取值范围[0,1]。Pdog,Pcat分布代表类别是狗和猫的概率。如果是100类的目标检测模型，BBox输出向量为5+100=105。

值得注意的是，RCNN有一句话的NMS介绍，Fast-RCNN无任何NMS的解释，Faster有大量篇幅对NMS的效果分析。

NMS主要就是通过迭代的形式，不断的以最大得分的框去与其他框做IoU操作，并过滤那些IoU较大（即交集较大）的框。如图 3图 4所示NMS的计算过程。

1、根据候选框的类别分类概率做排序，假如有4个 BBox ，其置信度A>B>C>D。

2、先标记最大概率矩形框A是算法要保留的BBox；

3、从最大概率矩形框A开始，分别判断ABC与D的重叠度IOU（两框的交并比）是否大于某个设定的阈值(0.5)，假设D与A的重叠度超过阈值，那么就舍弃D；

4、从剩下的矩形框BC中，选择概率最大的B，标记为保留，然后判读C与B的重叠度，扔掉重叠度超过设定阈值的矩形框；

5、一直重复进行，标记完所有要保留下来的矩形框。

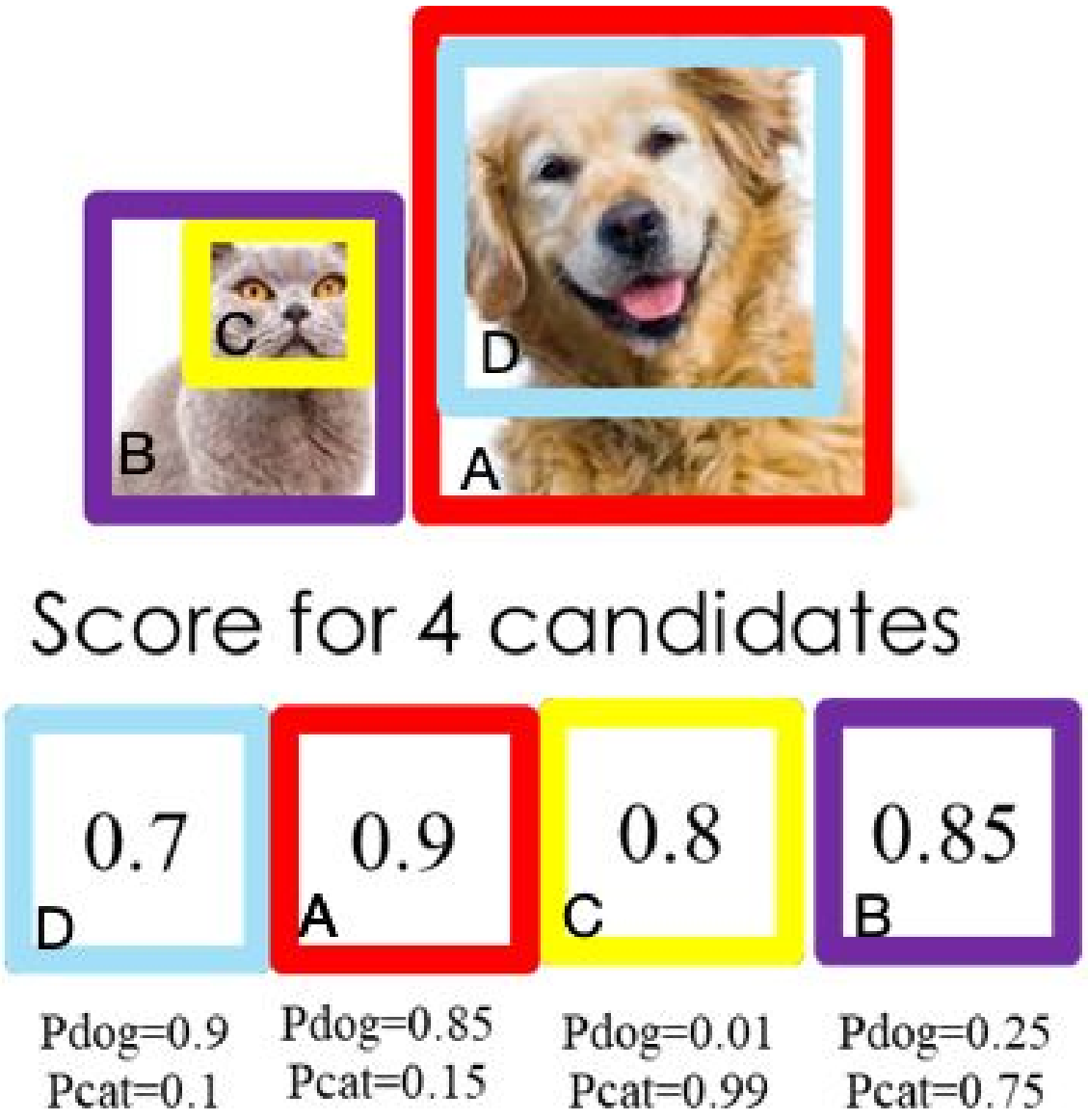


图 3猫和狗两类目标检测

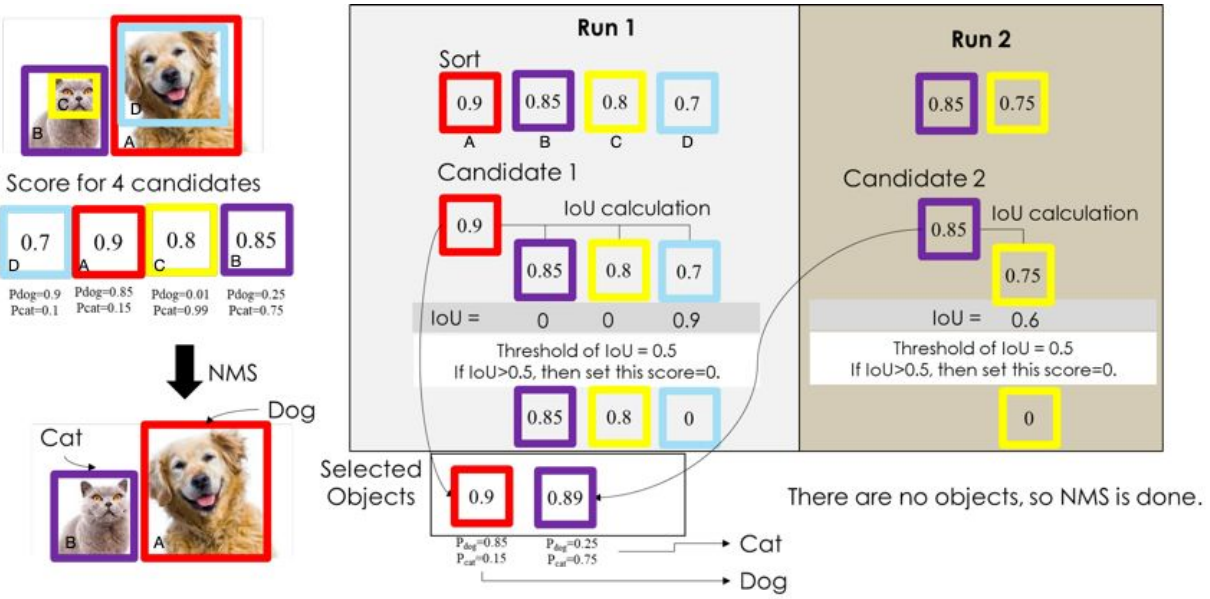


图 4 NMS算法过程

如果是two stage算法，通常在选出BBox有BBox位置(x,y,h,w)和confidence score，没有类别的概率。因为程序是生成BBox，再将选择的BBox的feature map做rescale(一般用ROI pooling)，然后再用分类器分类。NMS一般只能在CPU计算，这也是two stage相对耗时的原因。

但如果是one stage作法，BBox有位置信息(x,y,h,w)、confidence score，以及类别概率，相对于two stage少了后面的rescale和分类程序，所以计算量相对少。

NMS缺点：

1、NMS算法中的最大问题就是它将相邻检测框的分数均强制归零(既将重叠部分大于重叠阈值Nt的检测框移除)。在这种情况下，如果一个真实物体在重叠区域出现，则将导致对该物体的检测失败并降低了算法的平均检测率（average precision, AP）。

2、NMS的阈值也不太容易确定，设置过小会出现误删，设置过高又容易增大误检。

3、NMS一般只能使用CPU计算，无法使用GPU计算。

2、Soft-NMS

这篇ICCV2017的文章，是NMS算法的改进，论文题目很霸气：一行代码改进目标检测，既《Improving Object Detection With One Line of Code》[2]由UMIACS大学提出。

NMS算法是略显粗暴，因为NMS直接将删除所有IoU大于阈值的框。soft-NMS吸取了NMS的教训，在算法执行过程中不是简单的对IoU大于阈值的检测框删除，而是降低得分。算法流程同NMS相同，但是对原置信度得分使用函数运算，目标是降低置信度得分，其伪代码如图 5所示：

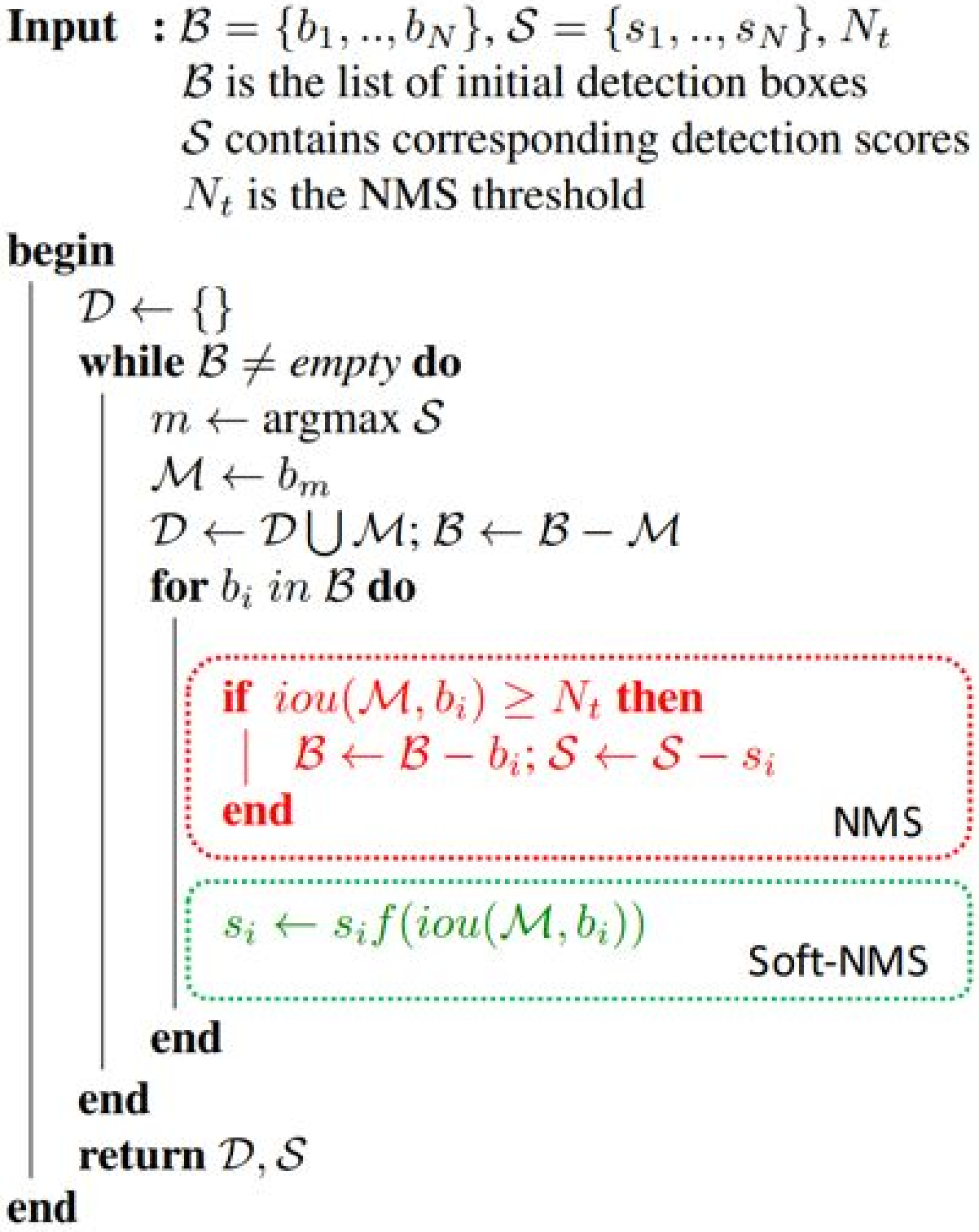


图 5 soft伪代码

b_i为待处理BBox框，B为待处理BBox框集合，s_i是b_i框更新得分，N_t是NMS的阈值，D集合用来放最终的BBox，f是置信度得分的重置函数。 b_i和M的IOU越大，b_i的得分s_i就下降的越厉害。

经典的NMS算法将IOU大于阈值的窗口的得分全部置为0，可表述如下：

$$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ 0, & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases}$$

论文置信度重置函数有两种形式改进，一种是线性加权的：

$$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ s_i(1 - \text{iou}(\mathcal{M}, b_i)), & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases}$$

一种是高斯加权的：

$$s_i = s_i e^{-\frac{\text{iou}(\mathcal{M}, b_i)^2}{\sigma}}, \forall b_i \notin \mathcal{D}$$

论文作者提供 R-FCN 和Faster-RCNN集成soft-NMS的代码实现 https://github.com/bharatsingh430/soft-nms。

其实最重要的改变在soft-nms/lib/nms/cpu_nms.pyx 81行到92行：

```
81                    if method == 1: # linear
82                        if ov > Nt:
83                            weight = 1 - ov
84                        else:
85                            weight = 1
86                    elif method == 2: # gaussian
87                        weight = np.exp(-(ov * ov)/sigma)
88                    else: # original NMS
89                        if ov > Nt:
90                            weight = 0
91                        else:
92                            weight = 1
```

图 6 soft-NMS主要代码实现

可以明显看到soft-NMS最重要是更新weight变量的值。采用线性加权时，更新为1-ov,高斯加权时引入sigma参数，而原始NMS算法时，直接取0或1。

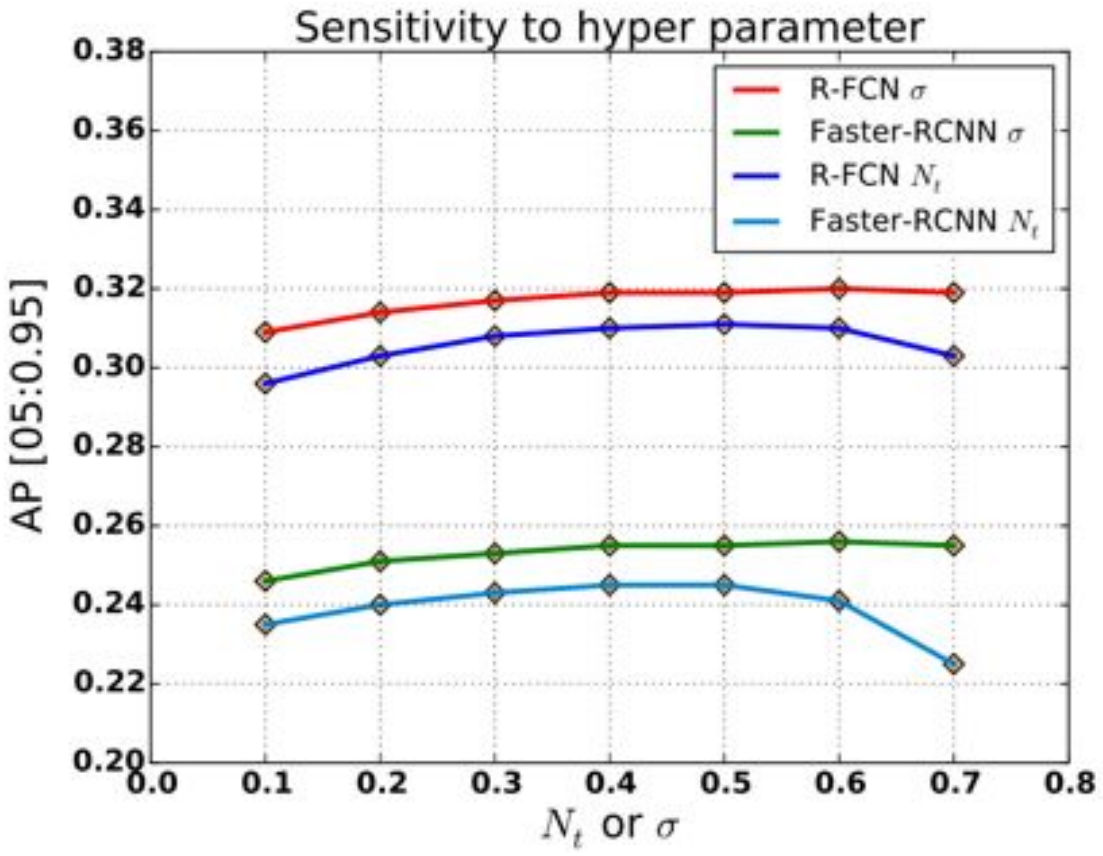


图 7 soft伪代码

论文中对比实验中数据集采用VOC 2007，COCO，基础模型包括R-FCN，Faster-RCNN可以看到性能的变化。但是个人觉得在图 7（原论文中图4）显示δ全范围取值均对AP有明显提升，证明soft-NMS不是依靠猜测某些超参数才有效。

soft-NMS优点：

1、Soft-NMS可以很方便地引入到object detection算法中，不需要重新训练原有的模型、代码容易实现，不增加计算量（计算量相比整个object detection算法可忽略）。并且很容易集成到目前所有使用NMS的目标检测算法。

2、soft-NMS在训练中采用传统的NMS方法，仅在推断代码中实现soft-NMS。作者应该做过对比试验，在训练过程中采用soft-NMS没有显著提高。

3、NMS是Soft-NMS特殊形式，当得分重置函数采用二值化函数时，Soft-NMS和NMS是相同的。soft-NMS算法是一种更加通用的非最大抑制算法。

soft-NMS缺点：

soft-NMS也是一种贪心算法，并不能保证找到全局最优的检测框分数重置。除了以上这两种分数重置函数，我们也可以考虑开发其他包含更多参数的分数重置函数，比如Gompertz函数等。但是它们在完成分数重置的过程中增加了额外的参数。

3、Softer-NMS

卡内基梅隆大学与旷视科技的研究人员提出了一种新的非极大抑制算法Softer-NMS[3]，其方法是在Soft-NMS和NMS基础上改进，也算是一脉相承。论文不是简单的更改CNN结构或调整参数，引入高斯分布，狄拉克delta分布，KL 散度等数学知识，建议首先阅读维基百科/百度百科对相应的概念做基本了解。

论文的motivation来自于NMS时用到的score仅仅是分类置信度得分，不能反映Bounding box的定位精准度，既分类置信度和定位置信非正相关的。NMS只能解决分类置信度和定位置信度都很高的，但是对其它三种类型：“分类置信度低-定位置信度低”，“分类置信度高-定位置信度低”，“分类置信度低-定位置信度高“都无法解决。

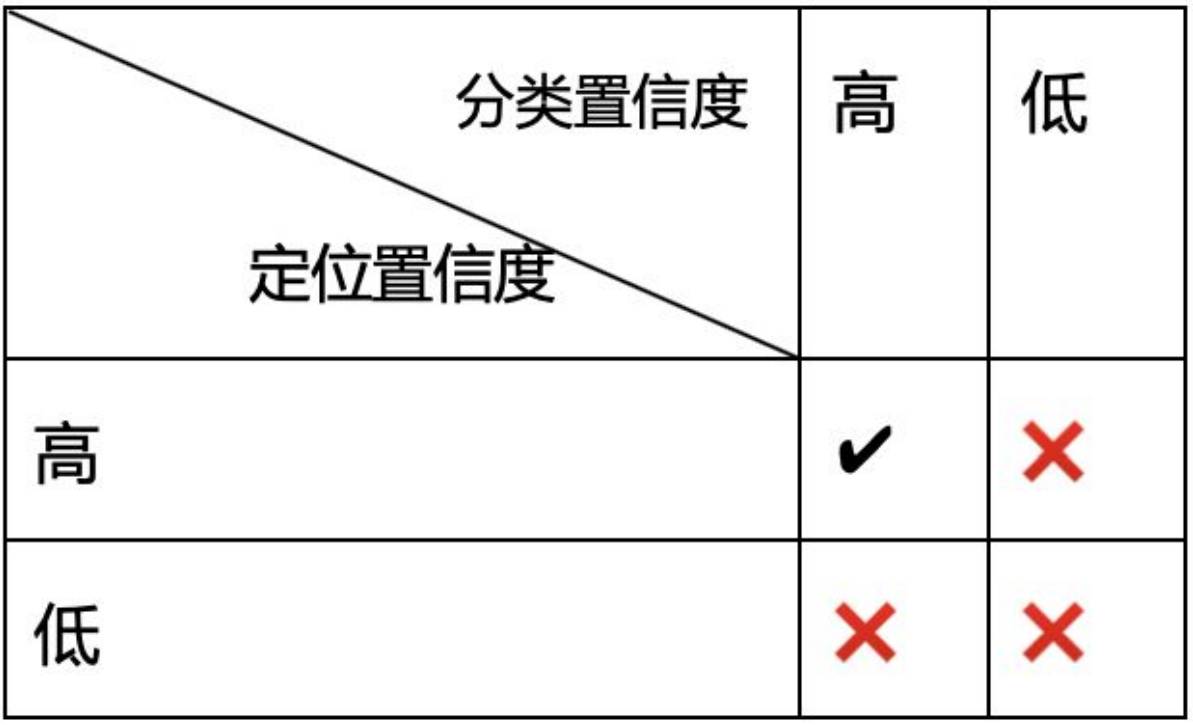


图 8 分类置信度和定位置信度

论文首先假设Bounding box的是高斯分布，round truth bounding box是狄拉克delta分布（既标准方差为0的高斯分布极限）。

KL 散度用来衡量两个概率分布的非对称性度量，KL散度越接近0代表两个概率分布越相似。

论文提出的KL loss，即为最小化Bounding box regression loss，既Bounding box的高斯分布和ground truth的狄拉克delta分布的KL散度。直观上解释，KL Loss使得Bounding box预测呈高斯分布，且与ground truth相近。而将包围框预测的标准差看作置信度。

论文提出的Softer-NMS，基于soft-NMS，对预测标注方差范围内的候选框加权平均，使得高定位置信度的bounding box具有较高的分类置信度。

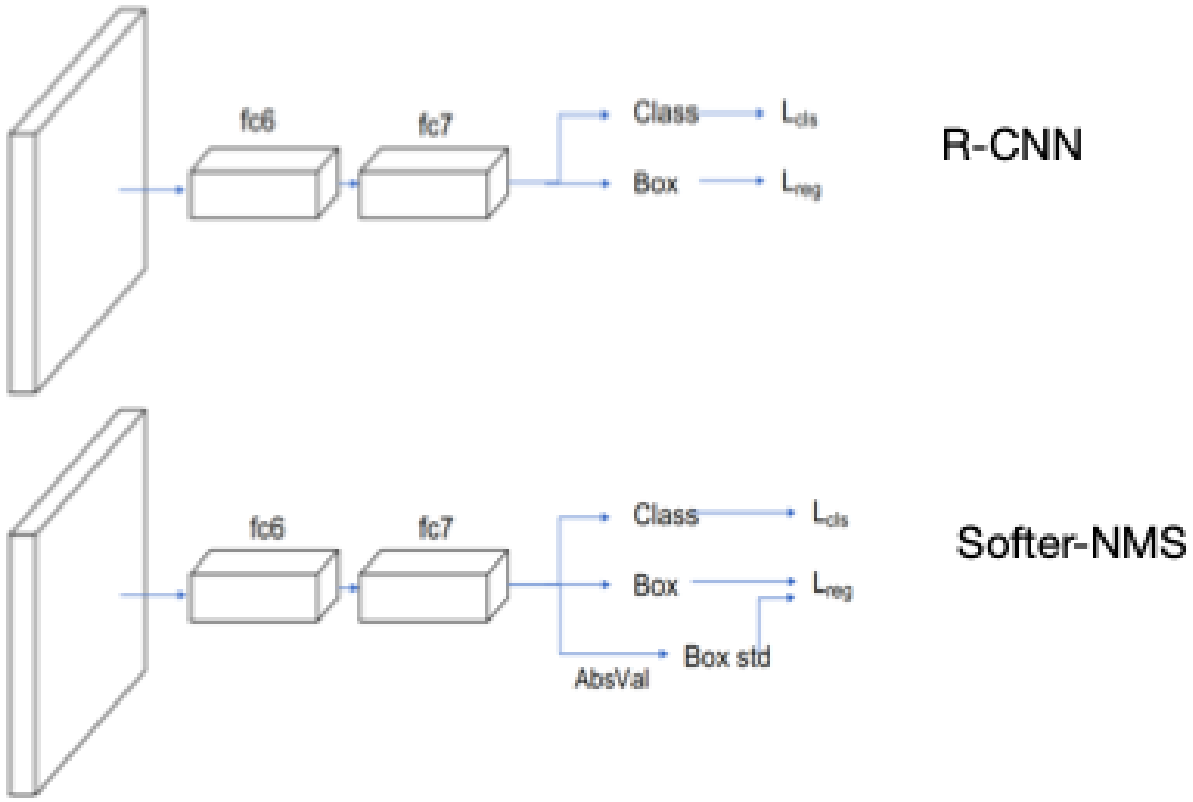


图 9 R-CNN和Softer-NMS异同

如所示Softer-NMS网络结构，与R-CNN不同的是引入absolute value layer（图中AbsVal）,实现标注方差的预测。

论文假设Bounding box为高斯分布：

$$P_\Theta(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-x_e)^2}{2\sigma^2}}$$

ground truth符合delta分布：

$$P_D(x) = \delta(x - x_g)$$

KL散度可以表示为：

$$\hat{\Theta} = \arg\min_\Theta D_{KL}(P_D(x)||P_\Theta(x))$$

推倒过程详见原论文，重点看看作者推倒的KL loss：

$$L_{reg} = \alpha(|x_g - x_e| - \frac{1}{2}) - \frac{1}{2}log(\alpha + \epsilon)$$

是不是和L1正则化很像？是不是预测的Bounding box与ground truth的曼哈顿距离的一维表示？

Algorithm 1 softer-NMS

$\mathcal{B}$ is $N \times 4$ matrix of initial detection boxes. $\mathcal{S}$ contains corresponding detection scores. $\mathcal{C}$ is $N \times 4$ matrix of corresponding variances. N_t is the softer NMS threshold. The lines in blue and in green are soft-NMS and softer-NMS respectively.

$\mathcal{B} = \{b_1, .., b_N\}, \mathcal{S} = \{s_1, .., s_N\}, \mathcal{C} = \{\sigma_1^2, .., \sigma_N^2\}, N_t$
$\mathcal{D} \leftarrow \{\}$
$\mathcal{T} \leftarrow \mathcal{B}$
while $\mathcal{T} \neq$ empty **do**
  $m \leftarrow$ argmax $\mathcal{S}$
  $\mathcal{M} \leftarrow b_m$
  $\mathcal{T} \leftarrow \mathcal{T} - \mathcal{M}$
  $\mathcal{S} \leftarrow \mathcal{S}f(IoU(\mathcal{M}, T))$ ▷ soft-NMS
  $idx \leftarrow IoU(\mathcal{M}, B) \geq N_t$ ▷ softer-NMS
  $\mathcal{M} \leftarrow \mathcal{B}[idx]/\mathcal{C}[idx]/sum(1/\mathcal{C}[idx])$
  $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{M}$
end while
return $\mathcal{D}, \mathcal{S}$

图 10 Softer-NMS实现过程

如图 10所示Softer-NMS的实现过程，其实很简单，预测的四个顶点坐标，分别对IoU＞N_t的预测加权平均计算，得到新的4个坐标点。第i个box的x1计算公式如下（j表示所有IoU＞N_t的box）：

$$x1_i := \frac{\sum_j x1_j/\sigma_{x1,j}^2}{\sum_j 1/\sigma_{x1,j}^2}$$

$$\text{subject to } IoU(x1_j, x1_i) > N_t$$

考虑特殊情况，可以认为是预测坐标点之间求平均值。

softerNMS的开源代码在https://github.com/yihui-he/softerNMS。

在softer-NMS/detectron/core/test.py有Softer-NMS(配置cfg.STD_NMS)， Soft-NMS(配置cfg.TEST.SOFT_NMS.ENABLED)以及NMS的实现。

```
793             if cfg.STD_NMS:
794                 confs_j = confs[inds, j * 4:(j + 1) * 4]
795                 nms_dets, _ = py_nms_utils.soft(dets_j,
796                         confidence=confs_j )
797             elif cfg.TEST.SOFT_NMS.ENABLED:
798                 nms_dets, _ = box_utils.soft_nms(
799                     dets_j,
800                     sigma=cfg.TEST.SOFT_NMS.SIGMA,
801                     overlap_thresh=cfg.TEST.NMS,
802                     score_thresh=0.0001,
803                     method=cfg.TEST.SOFT_NMS.METHOD
804                 )
805             else:
806                 keep = box_utils.nms(dets_j, cfg.TEST.NMS)
807                 nms_dets = dets_j[keep, :]
```

在softer-NMS/detectron/utils/py_cpu_nms.py文件有Softer-NMS的具体实现，加权求平均在47-48行代码实现：

```
47          ovr_bbox = np.where((ious[i, :N] > thresh))[0]
48          avg_std_bbox = (dets[ovr_bbox, :4] / confidence[ovr_bbox]).sum(0) / (1/confidence[ovr_bbox]).sum(0)
```

xyxy	KL Loss	soft-NMS	softer-NMS	AP	AP^{50}	AP^{75}	AP^{S}	AP^{M}	AP^{L}	AR^{1}	AR^{10}	AR^{100}
				23.6	44.6	22.8	6.7	25.9	36.3	23.3	33.6	34.3
		✓		24.8	45.6	24.6	7.6	27.2	37.6	23.4	39.2	42.2
✓				24.3	44.6	24.0	6.9	26.5	37.3	24.0	34.6	35.2
✓	✓			26.4	47.9	26.4	7.4	29.3	41.2	25.2	36.1	36.9
✓	✓	✓		27.8	49.0	28.5	8.4	30.9	42.7	25.3	41.7	44.9
✓	✓	✓	✓	**29.1**	**49.1**	**30.4**	**8.7**	**32.7**	**44.3**	**26.2**	**42.5**	**45.5**

图 11对比交叉实验

图 12 softer-NMS效果对比

论文的实验过程，实验结果可以参考原论文的第四节Experiments。

作者测试了在MS-COCO数据库上的推断延迟，发现Softer-NMS只是轻微增加了一点时间，可以忽略不计。

如图 12所示，论文对预测的坐标4个坐标点具有平均化的效果，使得各个box几乎完全重合。

1、个人认为论文提出的KL loss就是曼哈顿距离，但是通过KL散度去证明，让数学不太好的同学不明觉厉。

2、论文提出的Softer-NMS,本质是对预测的检测框加权求平均，为什么要这样，以及为什么让box高度重叠？个人认为Softer-NMS的理论没有在应该什么的地方深入。

4、IoU-guided NMS

IoU-Net是旷视的另外一篇论文[4]，是ECCV2018接收并做口头报告（清华和北大的学生在旷视实习时完成提交），和Softer-NMS一样，基于 CNN 的目标检测方法存在的分类置信度和定位置信度不匹配的问题。

传统的NMS算法缺失定位置信度，带来了两个缺点：

（1）在抑制重复检测时，由于定位置信度的缺失，分类分数通常被用作给检测框排名的指标。

（2）缺乏定位置信度使得被广泛使用的边界框回归方法缺少可解释性或可预测性。

论文提出的IoU-Net，通过预测检测框和ground truth的IOU实现：

1、IOU-guided NMS，也就是在NMS阶段引入定位得分（localization confidence）作为排序指标而不是采用传统的分类得分。

2、提出基于优化的bbox refinement替换传统的regression-based方法，提高了回归部分的可解释性。论文同时还提出了Precise ROI Pooling（PrROI Pooling），通过积分方式计算ROI特征，使得前向计算的误差进一步降低，同时反向传播时基于连续输入值计算梯度使得反向传播连续可导，相比之下ROI Pooling和ROI Align由于采用量化或几个点插值方式求ROI特征，不可避免地带来一些噪声，而且在反向求导时只对特定输入回传梯度。

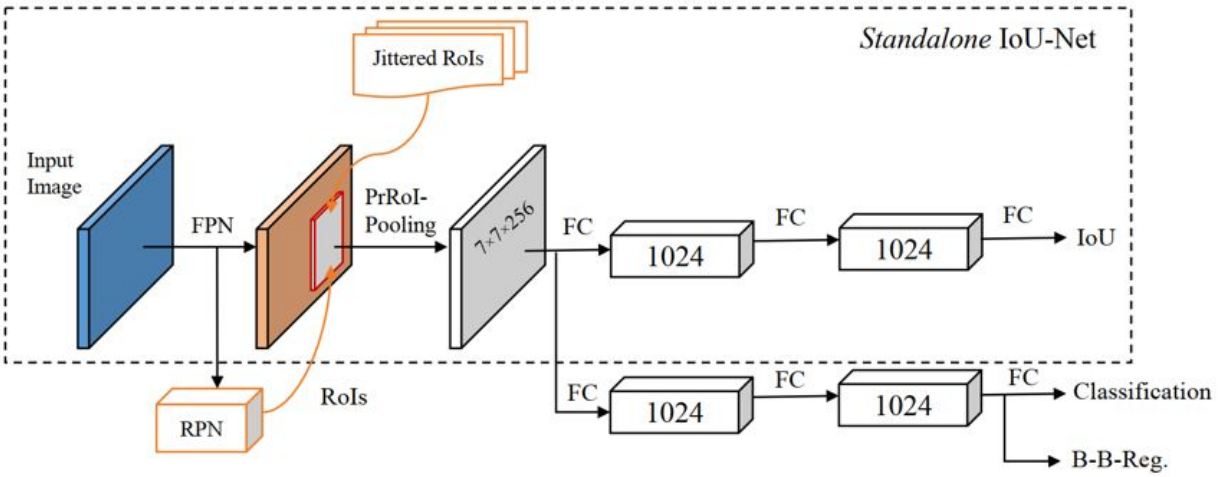
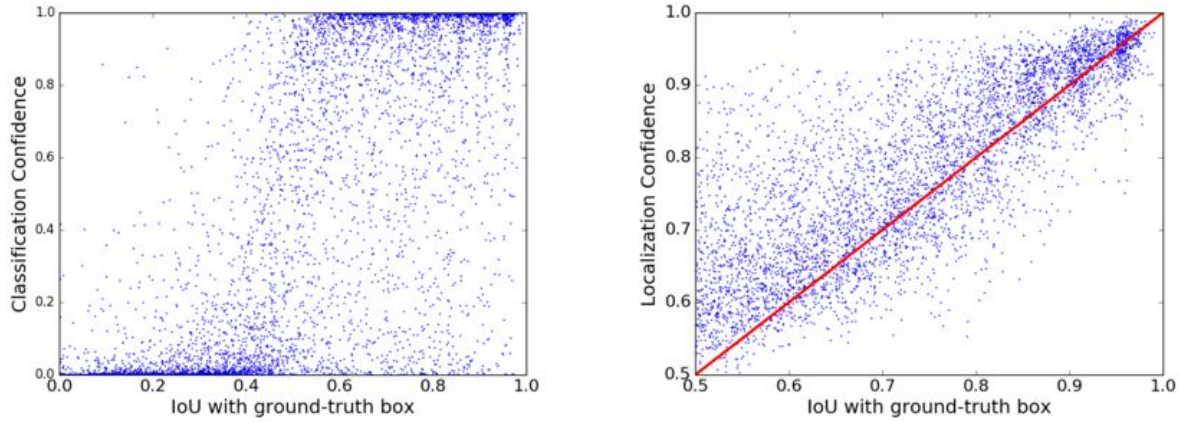


图 13 IoU模型架构

如图 13所示IoU-Net 的完整架构，使用使用了 ResNet-FPN作为骨干网络。输入图像首先输入一个 FPN 骨干网络，经过RPN和PrRoI 池化层，特征分别传送到 IoU 预测器和分类回归分支（虚线框表示标注的IoU-Net实现）。

IoU-Net模型整体上是在FPN网络中嵌入了IOU预测支路，IOU预测支路的监督信息就是ground truth和预测框的IOU值，该支路在结构设计上和FPN网络原有的回归和分类支路类似，另外将ROI pooling替换成PrROI pooling（precise ROI Pooling）。这里虚线框圈起来的部分表示standalone IOU-Net，在后续验证optimation-based bbox refinement算法优势时会将这部分结构应用在已有算法的预测结果上，相当于用IoU的监督信息对预测框做进一步的refinement。

如图 15所示分类置信度和定位置信度，分别和IOU with ground truth可视化分析，从（a）可以直观看出IOU值和分类得分之间没有明显的正相关，但是从（b）可以看出IOU值和回归得分之间有明显的正相关，除了直观上看以外，作者还计算了两张图各自的皮尔逊相关系数（Pearson correlation coefficient），分别是（a）：0.217，（b）：0.617，这也符合直观的视觉感受。这说明用分类得分作为依据判断一个预测框是否准确预测对ground truth是不合理的。因此作者提出了IoU-guided NMS，也就是NMS操作以预测的IOU为依据而不是传统的以预测框的分类得分为依据来解决这个问题。



(a) IoU vs. Classification Confidence (b) IoU vs. Localization Confidence

图 14 分类置信度、定位置信度和IOU with ground truth相关性

Algorithm 1 IoU-guided NMS. Classification confidence and localization confidence are disentangled in the algorithm. We use the localization confidence (the predicted IoU) to rank all detected bounding boxes, and update the classification confidence based on a clustering-like rule.

Input: $\mathcal{B} = \{b_1, ..., b_n\}$, $\mathcal{S}$, $\mathcal{I}$, Ω_{nms}
$\mathcal{B}$ is a set of detected bounding boxes.
$\mathcal{S}$ and $\mathcal{I}$ are functions (neural networks) mapping bounding boxes to their classification confidence and IoU estimation (localization confidence) respectively.
Ω_{nms} is the NMS threshold.
Output: $\mathcal{D}$, the set of detected bounding boxes with classification scores.

$\mathcal{D} \leftarrow \varnothing$
while $\mathcal{B} \neq \varnothing$ **do**
  $b_m \leftarrow \arg\max \mathcal{I}(b_j)$
  $\mathcal{B} \leftarrow \mathcal{B} \setminus \{b_m\}$
  $s \leftarrow \mathcal{S}(b_m)$
  for $b_j \in \mathcal{B}$ **do**
    if $\text{IoU}(\text{b}_\text{m}, \text{b}_\text{j}) > \Omega_{\text{nms}}$ **then**
      $s \leftarrow max(s, \mathcal{S}(b_j))$
      $\mathcal{B} \leftarrow \mathcal{B} \setminus \{b_j\}$
    end if
  end for
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\langle b_m, s \rangle\}$
end while
return $\mathcal{D}$

图 15 IOU-guided NMS算法流程

IOU-guided NMS 和NMS和主要的不同点在于用predicted IoU(localization confidence)来做排序。B表示检测框集合，S是分类置信度函数（CNN实现），I是定位置信度函数，Ω_{nms}是NMS阈值。

选择当前定位置信度最高的box ，记为b_m，和其他box计算IOU，当大于阈值Ω_{nms}时，取这些box最大的classification confidence，记为s，那么就<b_m,s>为当前迭代的结果。简单总结一下，就是将当前拥有最高IoU得分的bbox为聚类中心，然后去找聚类中最高的classification confidence作为聚类中心的classification confidence。

截止发稿日IoU-Net还未公布源码，论文复现有一定难度。

论文提出Precise RoI Pooling（PrRoI Pooling）用于bounding box修订。相比RoI Pooling，RoI Align，都是基于ROI坐标和特征图进行特征提取的过程，但是计算方式不同，如图 16所示。

RoI Pooling：在Fast R-CNN首先提出，将特征图上的RoI划分成固定数目的网格区域(如7*7)，对每个网格区域的边界点坐标（算出来可能是非整数）进行量化后再对各区域网格执行池化操作。

RoI Align：在Mask RCNN首先提出，对特征图上RoI被划分后的各网格区域选取固定数目的等间隔采样点，根据采样点邻近特征值利用双线性插值计算采样点的特征值，然后对采样点的特征值进行池化操作

区别：RoI Pooling是对划分后的各网格区域进行池化，计算区域网格边界点坐标的时候存在量化误差；RoI Align采用对网格区域采样的方式进行池化，且使用了插值计算采样点的特征值，要更精确。

PrROI Pooling：PrROI Pooling采用积分方式计算每个bin的值。这种计算方式和ROI Align最大的区别在于计算一个bin的值时不仅仅考虑该bin中4个插值点的均值，而是将bin中的插值看作是连续的，这样就可以通过对该bin中所有插值点求积分得到该bin所包围点的总和，最后除以面积就得到该bin的值，因此结果更加准确。

个人认为PrROI Pooling是为了在bbox refinement过程方便导数计算。

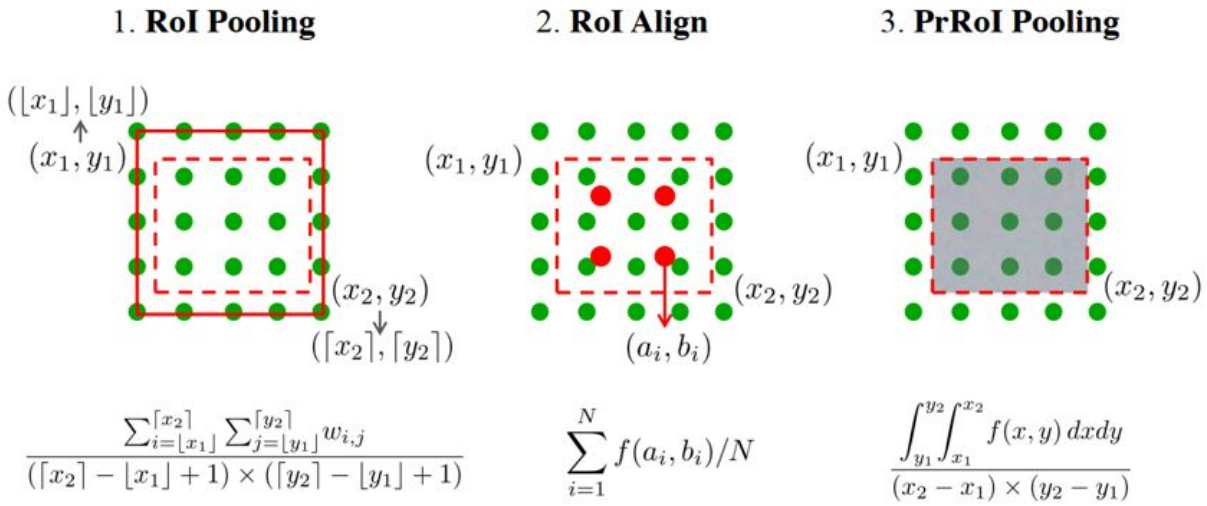


图 16 Pool方式对比

Algorithm 2 Optimization-based bounding box refinement

Input: $\mathcal{B} = \{b_1, ..., b_n\}$, $\mathcal{F}$, T, λ, Ω_1, Ω_2

$\mathcal{B}$ is a set of detected bounding boxes, in the form of (x_0, y_0, x_1, y_1).
$\mathcal{F}$ is the feature map of the input image.
T is number of steps. λ is the step size, and Ω_1 is an early-stop threshold and $\Omega_2 < 0$ is an localization degeneration tolerance.
Function PrPool extracts the feature representation for a given bounding box and function IoU denotes the estimation of IoU by the IoU-Net.

Output: The set of final detection bounding boxes.

$\mathcal{A} \leftarrow \varnothing$
for $i = 1$ *to* T **do**
  for $b_j \in \mathcal{B}$ and $b_j \notin \mathcal{A}$ **do**
    $\boldsymbol{grad} \leftarrow \nabla_{b_j} \text{IoU}(\text{PrPool}(\mathcal{F}, b_j))$
    $PrevScore \leftarrow \text{IoU}(\text{PrPool}(\mathcal{F}, b_j))$
    $b_j \leftarrow b_j + \lambda * scale(\boldsymbol{grad}, b_j)$
    $NewScore \leftarrow \text{IoU}(\text{PrPool}(\mathcal{F}, b_j))$
    if $|PrevScore - NewScore| < \Omega_1$ **or** $NewScore - PrevScore < \Omega_2$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{b_j\}$
    end if
  end for
end for
return $\mathcal{B}$

图 17 bbox refinement算法过程

bbox refinement是IoU-Net最难理解的部分。因为没有论文的源代码，这里仅对算法过程进行说明。在第六行可以看到bbox b_j的更改是通过梯度上升的方式更新定位得分，而停止停止条件是Ω_1（提前停止阈值）和Ω_2（定位方差），故bbox refinement本质上是个无监督的优化算法，不断优化detected box的位置实现。

IoU-Net 可与目标检测框架一起并行地端到端优化。论文将 IoU-Net添加到网络中有助于网络学习更具判别性的特征，这能分别将 ResNet50-FPN 和 ResNet101-FPN 的整体 AP 提升 0.6% 和 0.4%。IoU 引导式 NMS 和边界框修正还能进一步提升表现。论文使用 ResNet101-FPN 得到了 40.6% 的 AP，相比而言基准为 38.5%，提升了 2.1%。IoU-Net的推理速度在显著提升检测水平基础上实现推理速度无明显差异。

Backbone	Method	+IoU-NMS	+Refine	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
ResNet-50	FPN			36.4	58.0	53.1	44.9	31.2	9.8
	IoU-Net			37.0	**58.3**	**53.8**	45.7	31.9	10.7
		✓		37.6	56.2	52.4	46.0	34.1	14.0
		✓	✓	**38.1**	56.3	52.4	**46.3**	**35.1**	**15.5**
ResNet-101	FPN			38.5	**60.3**	**55.5**	47.6	33.8	11.3
	IoU-Net			38.9	60.2	**55.5**	47.8	34.6	12.0
		✓		40.0	59.0	55.1	48.6	37.0	15.5
		✓	✓	**40.6**	59.0	55.2	**49.0**	**38.0**	**17.1**

图 18 IoU-Net在MS-COCO最终实验

1、如图 18所示，IoU-Net的AP提升收益来自于AP70- AP90，在低阈值的时候表现一般。在图 14也可以观察到，在低阈值，定位置信度和IoU with ground truth非正相关。

2、IoU-Net网络架构清晰明了，但是在bounding box优化时不是很清晰。

3、IoU-Net利用定位置信度提出了IoU-Guided NMS及基于优化的bounding box的新算法， MS-COCO 实验结果表明了该方法的有效性和实际应用潜力。

5、其它去重算法

本章简要介绍基于NMS算法的其他创新点。

Relation Network（CVPR 2018 oral）是由MSRA提出。

当下主流的目标检测的方法还是对各个物体进行单独的检测，本文基于Attention，提出了一种object relation module，通过引入不同物体之间的外观和geometry特征做interaction，实现对物体之间relation的建模，提高检测效果，并且将关系模块运用在duplicate remove中，进行可学习的NMS（提出了一种特别的代替NMS的去重模块，可以避免NMS需要手动设置参数的问题），实现了第一个完全end-to-end的目标检测系统。

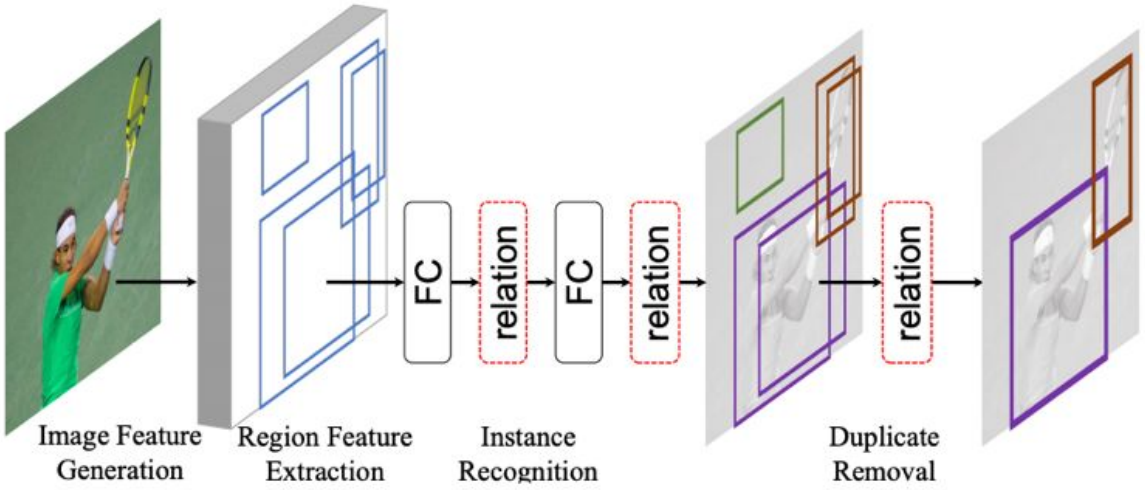


图 19 Relation Networks网络架构

这篇文章对关系的建模还有改进的空间，学出来的所谓“关系”并不清晰，（论文最后声明：our understanding of how relation module works is preliminary and left as future work.欲知后事如何，且听下回分解），更像是把Attention强行套入目标检测系统中。不过可学习的nms是很大的创新。

其主要考虑IoU阈值设定得高一些，则可能抑制得不够充分，而将IoU阈值设定得低一些，又可能多个ture positive被merge到一起。ConvNMS[6]其设计一个卷积网络组合具有不同overlap阈值的greedyNMS结果，通过学习的方法来获得最佳的输出。

考虑目标间具有高遮挡的密集场景，其提出一个新的网络架构来执行NMS。经分析，检测器对于每个目标仅产生一个检测结果有两个关键点是必要的，一是一个loss惩罚double detections以告诉检测器我们对于每个目标仅需一个检测结果，二是相邻检测结果的joint processing以使得检测器具有必要的信息来分辨一个目标是否被多次检测。论文提出Gnet[7]，基于CNN实现NMS网络。

不同于NMS，其主要有两个缺点，一是阈值必须人工设定，而在固定阈值下选择所有目标的输出边框是很难的，二是当检测器利用NMS时其假设输出边框都是独立的，但这些边框很可能是共享一些逻辑关系的。因此Yes-Net考虑利用RNN作为滤波器以得到最好的检测边框[8]，其能提升检测器泛化能力。

以上仅为个人阅读论文以及项目实践过程中的理解，总结和一些思考，观点难免偏差，望读者以怀疑的态度阅读，欢迎交流指正。

参考文献

[1] https://github.com/hoya012/deep_learning_object_detection

[2] Bodla N, Singh B, Chellappa R, et al. Soft-nms—improving object detection with one line of code[C]//Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017: 5562-5570.

[3] He Y, Zhang X, Savvides M, et al. Softer-NMS: Rethinking Bounding Box Regression for Accurate Object Detection[J]. arXiv preprint arXiv:1809.08545, 2018.

[4] Borui Jiang, Ruixuan Luo,et al. Acquisition of Localization Confidence for Accurate Object Detection. arXiv preprint arXiv: 1807.11590, 2018.

[5] Han Hu, Jiayuan Gu,Zheng Zhang,et al.Relation Networks for Object Detection. arXiv preprint arXiv: 1711.11575, 2017.

[6] Jan Hosang, Rodrigo Benenson, Bernt Schiele.A Convnet for Non-maximum Suppression. arXiv preprint arXiv: 1511.06437, 2015.

[7] J. H. Hosang, R. Benenson, and B. Schiele. Learning nonmaximum suppression. In CVPR, pages 6469–6477, 2017

[8] Liangzhuang Ma,Xin Kan,Qianjiang Xiao.An effective Detector Based on Global Information. arXiv preprint arXiv:1706.09180.2017.

欢迎给CVer点赞和转发

▲长按关注我们

欢迎点赞和Ad!