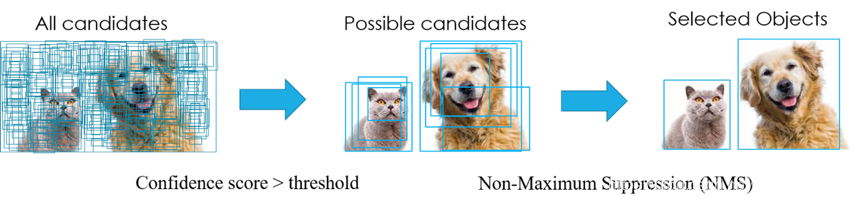
**目标检测算法中检测框合并策略（NMS）**

**1.传统NMS算法**

**1.1NMS介绍**

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。



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。

**1.2 NMS算法过程**

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。

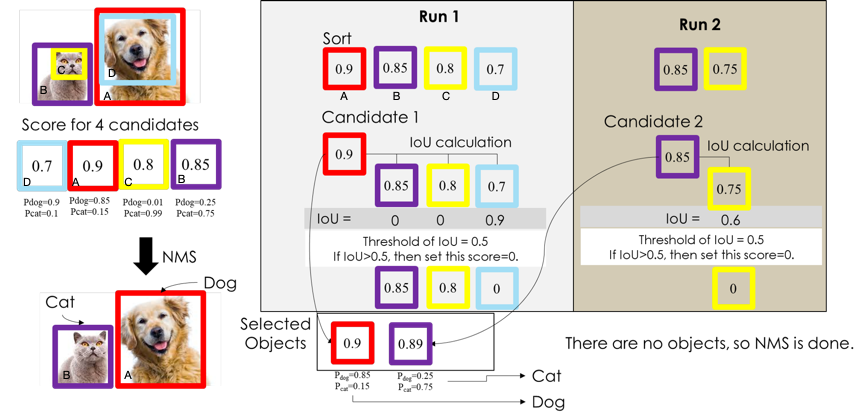
1)根据候选框的类别分类概率做排序，假如有4个 BBox ，其置信度A>B>C>D。

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BCD与A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。



如果是two stage算法，通常在选出BBox时，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。

**1.3 优缺点分析**

**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**。

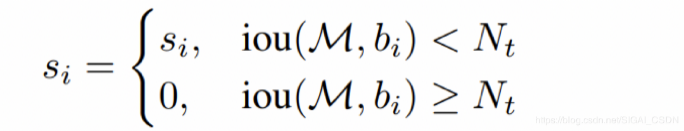
3)NMS一般只能使用CPU计算，无法使用GPU计算。

**2、Soft-NMS**

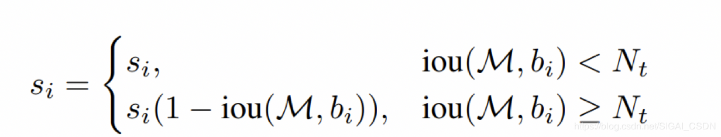
**2.1 算法过程**

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.

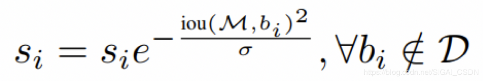
经典的NMS算法将IOU大于阈值的窗口的得分全部置为0，可表述如下:



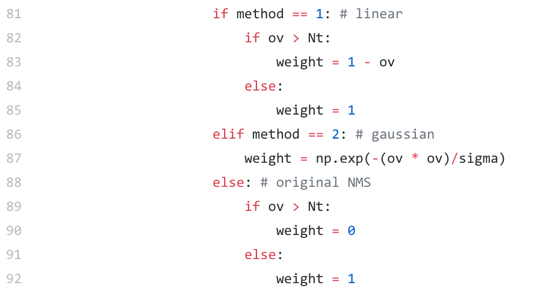
论文置信度重置函数有两种形式改进，一种是**线性加权**的：



一种是**高斯加权**的：

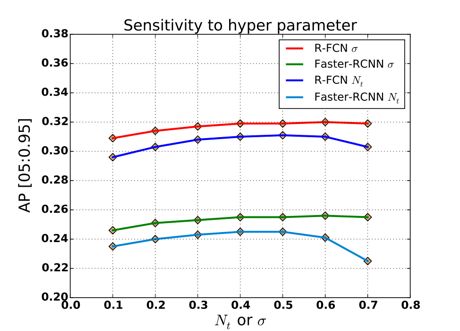


**2.2 代码实现**



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-IoU,高斯加权时引入sigma参数，而原始NMS算法时，直接取0或1。

**2.3 性能分析**



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**。

**2.4 优缺点分析**

**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**

**3.1算法思想**

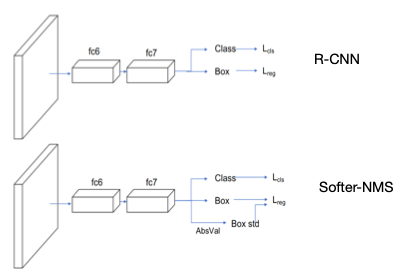
论文的motivation来自于NMS时用到的score仅仅是分类置信度得分，不能反映Bounding box的定位精准度，即**分类置信度和定位置信非正相关的**。NMS只能解决分类置信度和定位置信度都很高的，但是对其它三种类型：“分类置信度低-定位置信度低”，“分类置信度高-定位置信度低”，“分类置信度低-定位置信度高”都无法解决。

论文首先假设Bounding box的是高斯分布，ground truth bounding box是狄拉克**delta分布**（即标准方差为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。

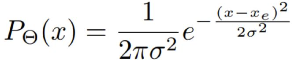
论文提出的**KL loss**，即为最小化Bounding box regression loss，即Bounding box的高斯分布和ground truth的狄拉克delta分布的KL散度。直观上解释，KL Loss使得Bounding box预测呈高斯分布，且与ground truth相近。将**包围框预测的标准差看作置信度**。

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**。



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：



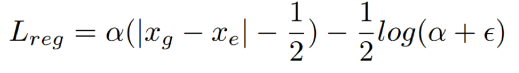
ground truth符合delta分布：

C:\Users\12544\AppData\Roaming\Tencent\Users\1254468141\QQ\WinTemp\RichOle\Y%}0(UZI4%WFDO0`2GISWNP.png

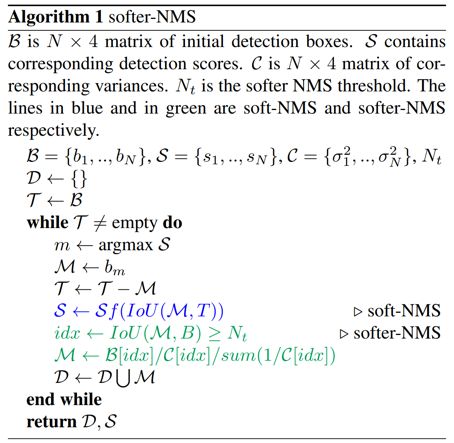
KL散度可以表示为：

C:\Users\12544\AppData\Roaming\Tencent\Users\1254468141\QQ\WinTemp\RichOle\Y1O[EHCO3KF~{5SPX_]1QS0.png

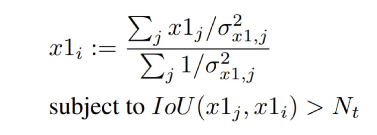
推倒过程详见原论文，重点看看作者推倒的KL loss：



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**？

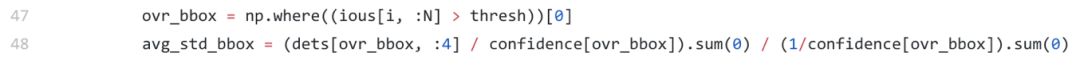


如图 10所示Softer-NMS的实现过程，其实很简单，预测的四个顶点坐标，分别对IoU＞Nt的预测加权平均计算，得到新的4个坐标点。第i个box的x1计算公式如下（j表示所有IoU＞Nt的box）：

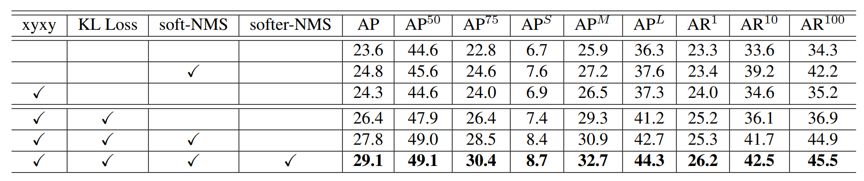


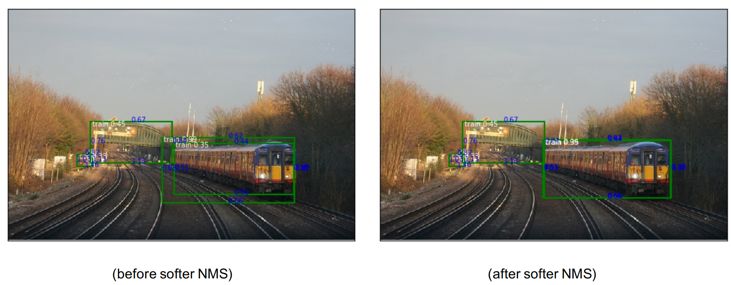
考虑特殊情况，可以认为是**预测坐标点之间求平均值**。

**3.3 代码分析**



**3.4 性能分析**





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。

**3.5  优缺点分析**

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**

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。

**4.1 算法思想**

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。

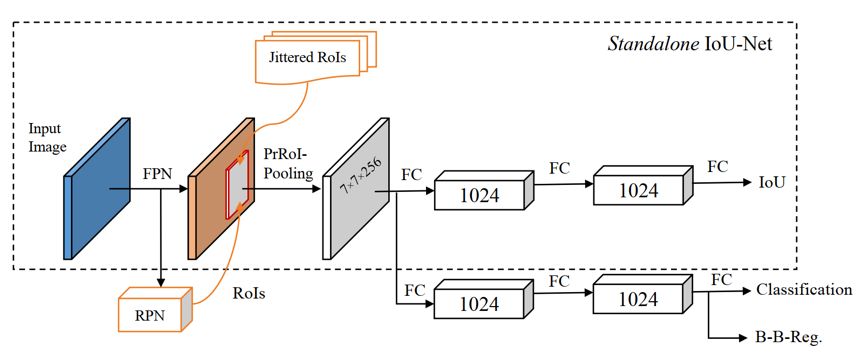
（2）缺乏定位置信度使得被广泛使用的边界框回归方法缺少**可解释性**或可预测性。

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。

**4.2 模型框架**



如图 13所示IoU-Net 的完整架构，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。

**4.3 IoU-guided NMS**

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操作是以预测的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。

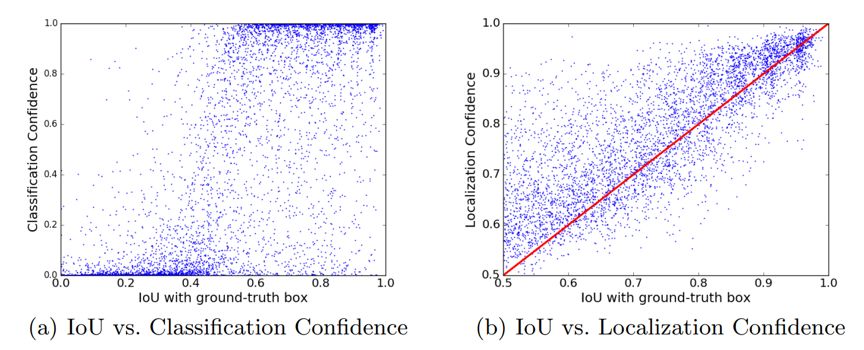
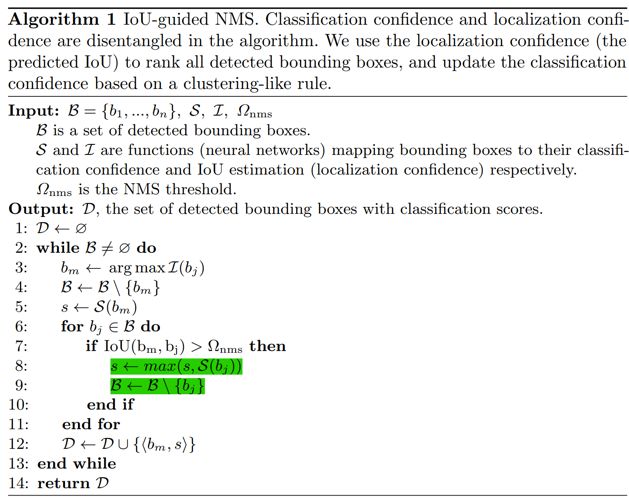


图 14 分类置信度、定位置信度和IOU with ground truth相关性



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，Ωnms是NMS阈值。

选择当前定位置信度最高的box ，记为bm，和其他box计算IOU，当大于阈值Ωnms时，取这些box最大的classification confidence，记为s，那么就**<bm,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**。

**4.4  PrRoI Pooling**

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。

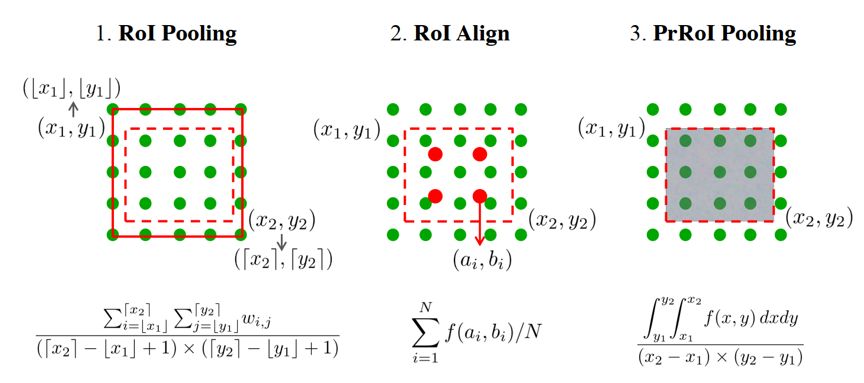
RoI Pooling：在Fast R-CNN首先提出，将特征图上的RoI划分成固定数目的网格区域(如7\*7)，对每个网格区域的边界点坐标（算出来可能是非整数）进行**量化**后再对各区域网格**执行池化操作**。

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。



**4.5  bbox refinement**

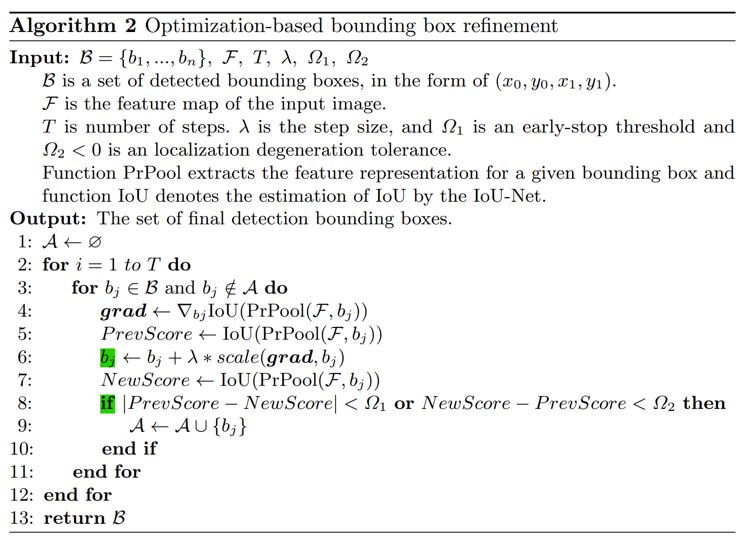
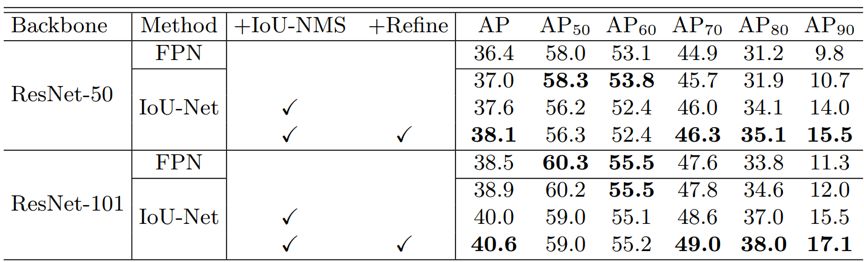


图 17 bbox refinement算法过程

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 bj的更改是通过梯度上升的方式更新定位得分，而停止条件是Ω1（提前停止阈值）和Ω2（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。

**4.6  性能分析**

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 NMS 和边界框修正进一步提升表现。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。



**4.7  优缺点分析**

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. Relation Networks for Object Detection**

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。

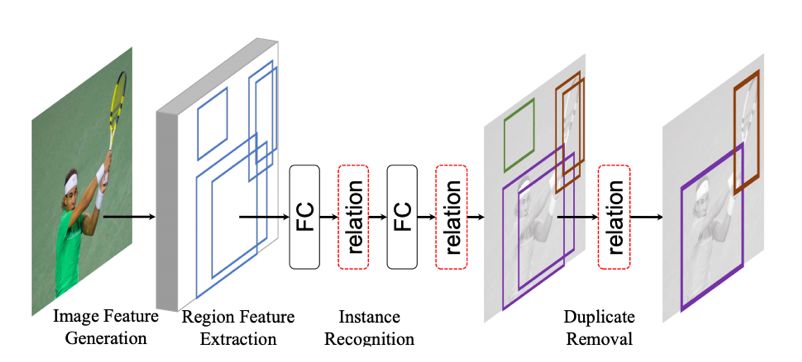


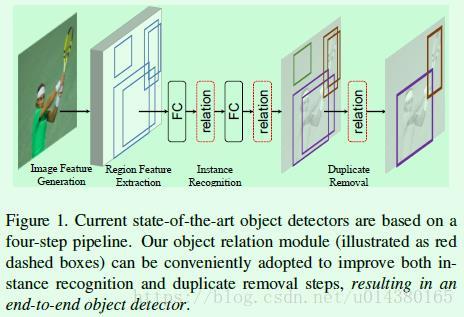
图 19 Relation Networks网络架构

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。

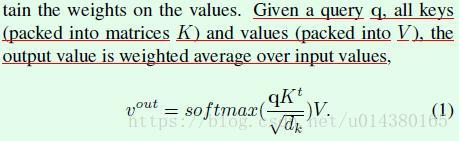
object relation module来描述object之间的关系，从而以attention的形式附加到原来的特征上最后进行回归和分类，另外这篇文章的一个亮点是同时将这种**attention机制引入nms操作**中，不仅实现了真正意义上的end-to-end训练，而且对于原本的检测网络也有提升。

值得注意的是object relation module和网络结构的耦合度非常低，同时输出的维度和输入的维度相同，因此可以非常方便地插入到其他网络结构中，而且可以叠加插入。

在这篇文章中，**object relation module**主要放在两个全连接层后面和NMS模块，如Figure1中的红色虚线框所示。在Figure1中，作者将目前目标检测算法分为4步：1、特征提取主网络；2、得到ROI及特征（RPN网络就包含在其中）；3、基于ROI做边框回归和object分类；4、NMS处理，去除重复框。从作者的分步情况和源码可以清晰地看出，这篇文章主要是基于Faster RCNN系列算法（Faster RCNN，FPN等）引入object relation module。



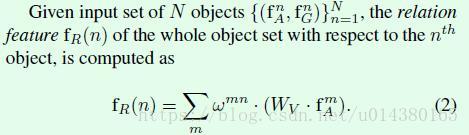
接下来介绍object relation module。因为这篇文章是借鉴了attention is all you need中的attention思想，因此先来了解下那篇文章中的主要公式。在attention is all you need这篇文章中介绍了一个基本的attention模块：scaled dot-product attention，如公式1所示。



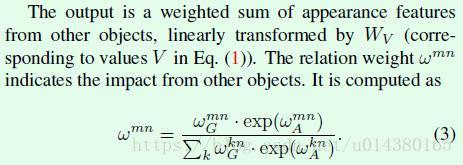
接下来定义两个重要的变量。假设输入中有N个object，那么N个object的两种特征集合如下所示，**fA是常规的图像特征（appearance feature），fG是位置特征（geometric feature）**。这两种特征是后续算法的基础。

这里写图片描述

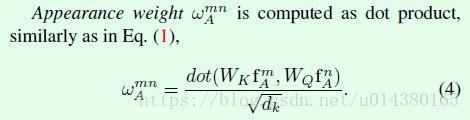
那么受公式1的启发，可以用公式2来刻画第n个object和所有object之间的关系特征。其中fAm表示第m个object的图像特征。**Wv是一个线性变换操作，在代码中用1\*1的卷积层实现**。**wmn是用来描述不同object之间的关系权重**（relation weight），该变量通过后面的一系列公式可以得到。计算得到的关系特征fR将和原有的图像特征fA融合并传递给下一层网络，这就完成了attention过程。那么公式2和公式1有什么联系呢？简单讲，公式2中的Wv对应公式1中的V，公式2中的wmn对应公式1中的softmax()。



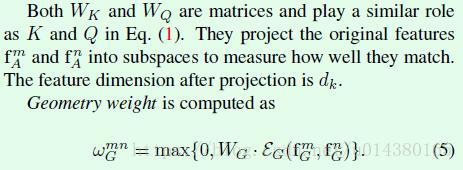
公式2中的wmn通过下面的公式3实现。从公式形式可以看出是一个归一化操作，因此在源码中也是通过softmax层实现（和公式1中的softmax操作对应），只不过对输入中的wG先做了一个log操作，然后将log(wG)+wA作为softmax的输入就能得到公式3。提前说一下该公式中的两个重要变量**wG**和**wA**分别表示object的**位置特征权重**（geometric weight）和**object的图像特征权重**（appearance weight），后面通过公式4和5分别得到。这里我直接用wG代替wGmn，用wA代替wAmn，一方面是因为实际代码实现中对这些变量的实现都是基于所有object统一得到，另一方面是书写简单。



**公式4是计算公式3中的wA变量，也就是图像特征权重**。在代码中，WKfA通过全连接层来实现，WK就是全连接层的参数，fA就是object特征；同理WQfA也是通过全连接层实现，WQ是全连接层参数。最后再做一个scale操作。



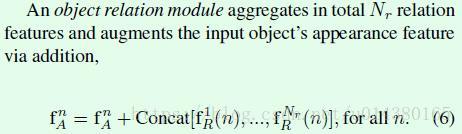
**公式5是计算公式3中的wG变量，也就是位置特征权重。**首先对fG做了坐标上的变换，如下面截图，主要进行**尺度归一化和log操作**，可以增加尺度不变性，不至于因为数值变化范围过大引起训练的发散。可以看出坐标变换公式和目标检测算法中的回归目标构造非常相似，最大的不同点在于对x和y做了log操作，原因在于这里要处理的xm与xn、ym与yn之间的距离要比目标检测算法中的距离远，因为目标检测算法中的距离是预测框和roi之间的距离，而这里是不同预测框（或者说是不同roi）之间的距离，因此加上log可以避免数值变化范围过大。**EG操作主要是将4维的坐标信息embedding成高维的坐标信息（比如默认是64维），借鉴的是attention is all you need这篇文章的思想，主要由一些cosine函数和sine函数构成。**接着，将WG和embedding特征相乘，这里**WG也是通过全连接层实现的**。最后的max操作类似relu层，主要目的在于对位置特征权重做一定的限制。wG的引入是本文和attention is all you need这篇文章较大的不同点。



fG的坐标变换公式：

这里写图片描述

在得到一个**关系特征fR**后，这篇文章最后的做法是**融合多个（Nr个，默认是16）关系特征后再和图像特征fA做融合**，如公式6所示。这里有个细节是关于关系特征的融合方式，这里采用的**concat**，主要原因在于计算量少，**因为每个fR的通道维度是fA的1/Nr倍，concat后的维度和fA相同**。假如用element-wise add的方式进行融合，那么每个fR的维度和fA都一样，这样的加法计算量太大。



综上，可以用下面的Algorithm 1来概括前面提到的公式算法，源码中的实现基本上和Algorithm 1相同，这在前面介绍公式的时候已经详细介绍了，这里就不再赘述。注意几个参数的默认值：dk默认是64，dg默认是64。

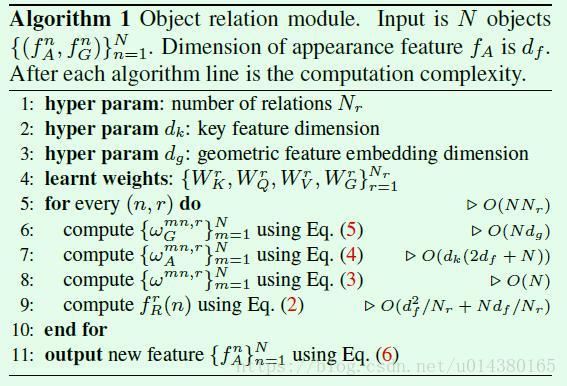
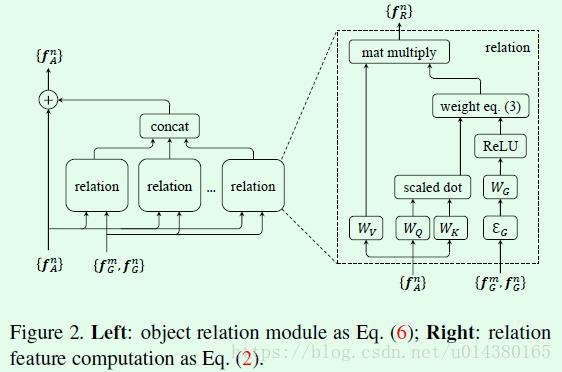
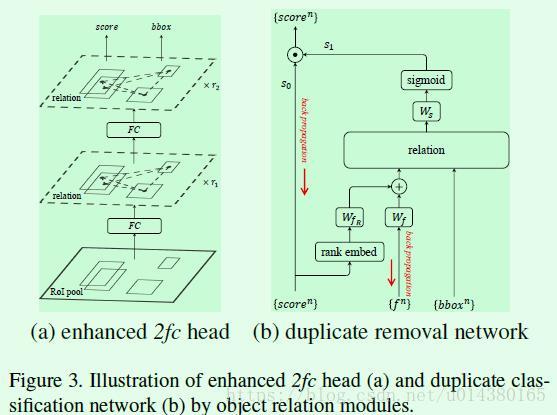
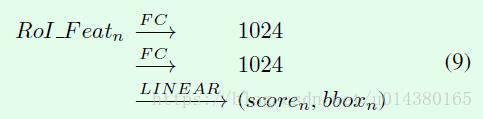


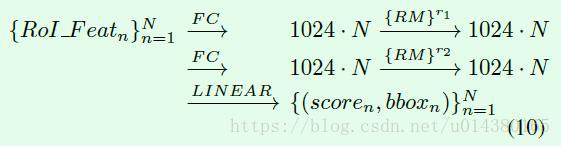
Figure2是用图表来描述Algorithm 1的算法过程。左图是整体上的attention模块和图像特征fA的融合；右图是attention模块的详细构建过程，和前面的公式2~5对应。



介绍完object relation module，接下来就是怎么应用在目标检测算法中了。Figure3是object relation module插入目标检测算法的示意图，**左图是插入两个全连接层的情况，在全连接层之后会基于提取到的特征和roi的坐标构建attention，然后将attention加到特征中传递给下一个全连接层，再重复一次后就开始做框的坐标回归和分类。**右图是插入NMS模块的情况，**图像特征通过预测框得分的高低顺序和预测框特征的融合得到**，**然后将该融合特征与预测框坐标作为relation模块的输入得到attention结果**，最后将**NMS当作是一个二分类过程**，并基于relation模块**输出特征计算分类概率**。



下面以object relation module插入全连接层为例，了解下这篇文章大致是怎么做的。首先是原有的全连接层形式如公式9所示：   


这篇文章直接在每个全连接层后面接RM（relation module），输出的维度和输入维度一致，如公式10所示。RM右上角的r1和r2表示数量，源码中默认是1，因为RM不改变输入的维度，所以RM的数量可以任意添加。   


**介绍完relation module在检测模块（全连接层）的嵌入后，另一部分重要的内容是relation module在NMS模块的嵌入。**

NMS的思想简单讲就是根据将和当前预测得分最高的框的IOU大于某个阈值的框过滤。因此这其实是一个二分类的问题，也就是针对每个ground truth，只有一个预测为该类的框是对的，其他预测为该类的框都是错的。

具体实现参考前面的Figure3(b)，其中s0是预测框的socre，这个socre对应公式9和10中最后一行的输出socre；s1是二分类的结果（1表示对的；0表示错的，要移除）。那么这个二分类怎么做呢?主要分成3步：1、先获取输入score的排序信息（index），然后对index做embedding，并将embedding后的index和图像特征（f）通过一些全连接层（维度从1024降为128）后进行融合，融合后的特征作为relation module的输入之一。2、预测框的坐标作为relation module的另一个输入，经过relation module后得到关系特征。3、关系特征经过一个线性变换（Figure3(b)中的Ws）后作为sigmoid的输入得到分类结果，这样就完成了**预测框的二分类**，且基于的特征引入了attention。

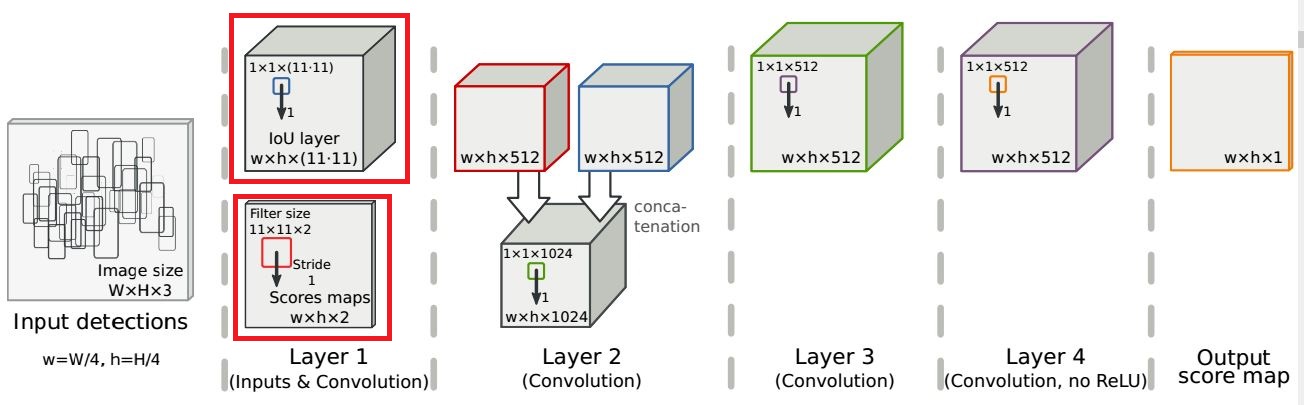
因此将NMS纳入end-to-end训练后，**整体网络的损失函数不仅包含原来的坐标回归和分类损失函数，还包含NMS的二分类损失函数**。

**6. ConvNMS**

**1) Overview**

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**。

传统的NMS在判决融合的时候，只利用到了2个信息：Score 和 IOU ，即每个框的得分和框与框之间的重叠比例。   
文章用神经网络去实现NMS，所利用的同样也是这2个信息。如下图的整个流程图：



**2) Score map的生成**

假设我们的图片是W\*H的，在这里我们以16\*16大小的图片为例，我们对他生成w\*h大小的score map，其中w=W/4，h=H/4，也就是说score map上的一个点对应原图中4\*4的区域。然后假设我们得到了一个物体的bounding box，其置信度为0.8，我们将其中心点的位置记录在score map中，因为他的**中心点**落在第（3，2）个score map区域中，因此该区域置信度为0.8；接着我们又得到一个新的bounding box，其置信度为0.9，且中心点也落在了(3,2)的区域，根据NMS保留大置信度的特点，我们对该区域的置信度进行更新，也就是说(3,2)这个区域的置信度变成了0.9。注意在更新置信度的时候，我们也保留了原始bounding box的原始位置信息，以便最终的索引。

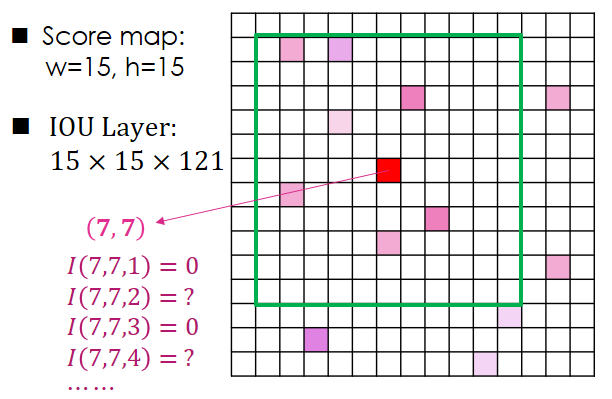
到此，我们已经得到了w×h×1的score map.

文章提到，卷积神经网络的线性操作和relu无法模仿NMS中对score进行排序的操作，因此，文章先用传统的NMS处理一遍bounding boxes, 然后**再生成一张同样大小的score map，**记作S(T)，T为NMS的阈值。

最终，我们得到了**w×h×2的score map**. 记作S(1,T)

**3) IoU layer**

NMS的另一个重要概念是通过重叠面积来筛选候选框。这一步在convnet中是这么做的:对于score map上的每个点，我们已经记录了其是否包含bounding box以及该box的置信度。那么我们以每个点为中心，选取其11\*11的邻域，计算这121个点保存的bounding box与中心点的**IoU**，然后保存成**121维**的向量用于描述该点的IoU特性。例如，以下图这个15\*15的score map为例，对于(7,7)这个点来说，我们取其11\*11的邻域，然后计算邻域中每个点所包含的bounding box和(7,7)的bounding box的IoU，比如领域中左上角第一个点不包含bounding box，则特征I(7,7,1)=0,第二个点包含了bounding box(图中有颜色的小方格代表包含bounding box，白色的则没有包含)，则我们计算该点的**bounding box和(7,7)的bounding box的IoU**,并存入到I(7,7,2)中。以此类推，每个点都有121维的特征，则整个score map输出的特征为15\*15\*121.



**4)网络解析**

我们再来看一下网络结构，注意2点：

IOU层的kernel size 为1，stride也为1。

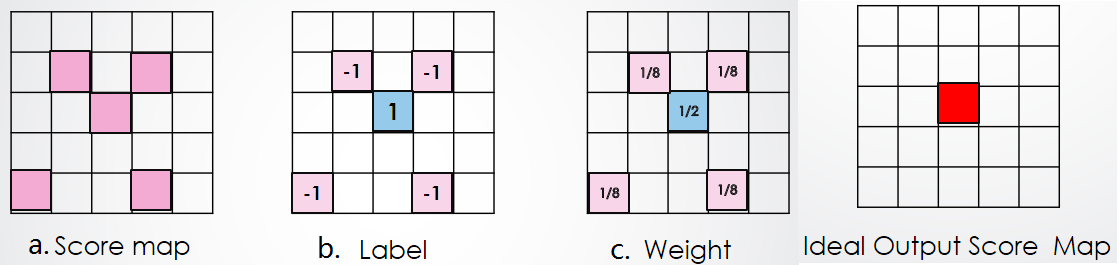
Score map层的kernel size 为11×11，这是为了呼应IOU层；stride 为1 ，pad 为5，这是为了获得和输入同样尺寸的输出，进行卷积操作，改变输出通道。

Layer 2将之前的2个**输出拼接**，之后所有的卷积都是1×1。最终的输出仍然是一个尺寸一致的score map。

**5)输出及Loss**

理想的输出是一个同输入尺寸完全一致的score map 图，在该图中，每一个目标只拥有一个score，相应地也只对应了一个bounding box。

因此训练的目标就是**保留一个，抑制其它**。如下图：



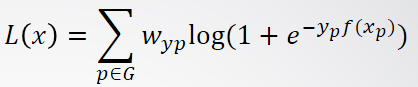
(1)上图a的score map 是我们的输入，由图易知，这里面一共有5个有效的score，则也对应着5个bounding box。

(2)假设**5个bounding box都是同一个目标的检测结果**。则我们的训练目的则是保留最好的一个，抑制其余4个。

为此，我们首先分配标签：**5个bounding box 中满足与ground truth 的IOU大于0.5且得分最高的box作为正样本**，其余均为负样本，如上图b所示。

(3)显然正负样本的数量严重不均衡，因此计算loss之前，我们要分配一下权重用于权衡这种失衡。权重的分配很简单，如图c所示，正样本的权重总和与负样本的权重总和相等。

(4)上右侧的图为理想的输出。综上，我们的Loss Function就可以很容易得出了（类似于[**pixel**](https://www.baidu.com/s?wd=pixel&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)**级别的分类**）：



其中，**p属于G，表示score map 中有值的点**。