一位算法工程师从30+场秋招面试中总结出的超强面经——目标检测篇（含答案）

原创 CV开发者都爱看的 极市平台 2021-05-20 22:04:00 手机阅读

收录于话题
#CV面经

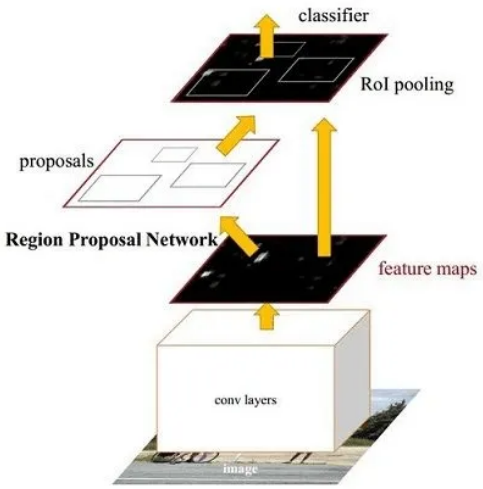
作者丨灯会
来源丨极市平台
编辑丨极市平台

极市导读

作者灯会为21届中部985研究生，凭借自己整理的面经，去年在腾讯优图暑期实习，七月份将入职百度cv算法工程师。在去年灰飞烟灭的算法求职季中，经过30+场不同公司以及不同部门的面试中积累出了CV总复习系列，此为目标检测篇。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

Faster-Rcnn网络

1.faster RCNN原理介绍，要详细画出图



Faster R-CNN是一种两阶段（two-stage）方法,它提出的RPN网络取代了选择性搜索（Selective search）算法后使检测任务可以由神经网络端到端地完成。在结构上，Faster RCNN将特征抽取(feature extraction)，候选区域提取（Region proposal提取），边框回归（bounding box regression），分类（classification）都整合在了一个网络中，使得综合性能有较大提高，在检测速度方面尤为明显。

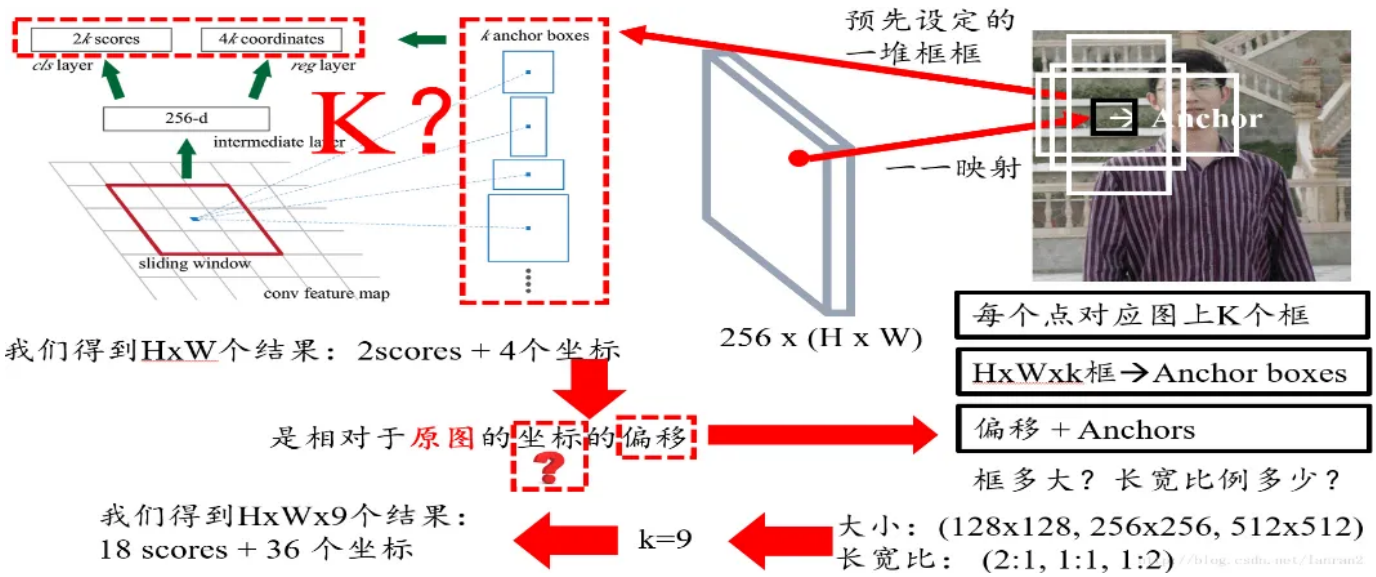
2.RPN（Region Proposal Network）网络的作用、实现细节

RPN网络的作用： RPN专门用来提取候选框，一方面RPN耗时少，另一方面RPN可以很容易结合到Fast RCNN中，成为一个整体。

RPN网络的实现细节：一个特征图（Faster RCNN的公共Feature Map）经过sliding window处理，得到256维特征，对每个特征向量做两次全连接操作，一个得到2个分数，一个得到4个坐标{然后通过两次全连接得到结果2k个分数和4k个坐标[k指的是由锚点产生的K个框(K anchor boxes)]}

2个分数，因为RPN是提候选框，还不用判断类别，所以只要求区分是不是物体就行，那么就有两个分数，前景（物体）的分数，和背景的分数； 4个坐标是指针对原图坐标的偏移，首先一定要记住是原图；

预先设定好共有9种组合，所以k等于9，最后我们的结果是针对这9种组合的，所以有H x W x 9个结果，也就是18个分数和36个坐标。



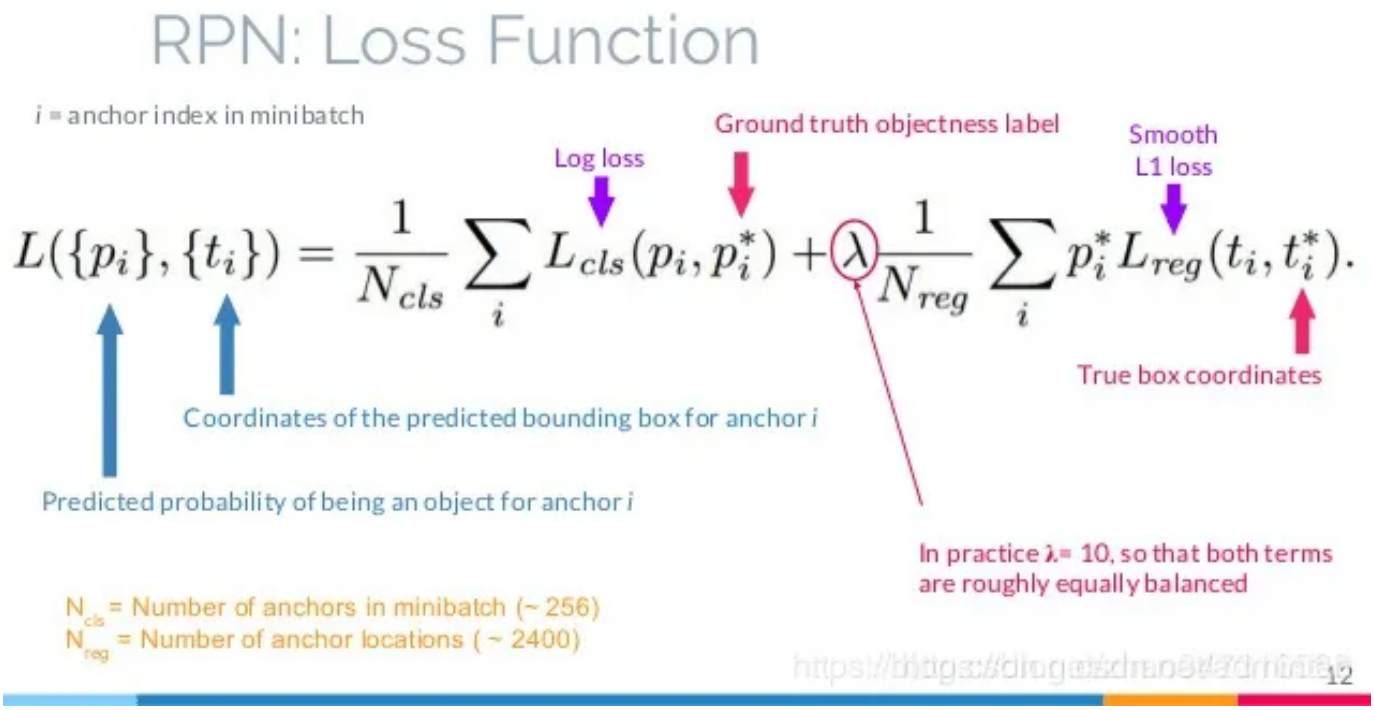
写一下RPN的损失函数(多任务损失:二分类损失+SmoothL1损失)

训练RPN网络时，对于每个锚点我们定义了一个二分类标签（是该物体或不是）。

以下两种情况我们视锚点为了一个正样本标签时：

1.锚点和锚点们与标注之间的最高重叠矩形区域

2.或者锚点和标注的重叠区域指标（IOU）>0.7



$$L(p_i,t_i) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i,t_i^*)$$

RPN损失中的回归损失部分输入变量是怎么计算的?（注意回归的不是坐标和宽高，而是由它们计算得到的偏移量）

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

ti 和 ti* 分别为网络的预测值和回归的目标

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \quad t_w = \log(w/w_a), \quad t_h = \log(h/h_a)$$
$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \quad t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a),$$

在训练RPN时需要准备好目标t*。它是通过ground-truth box（目标真实box）和anchor box（按一定规则生成的anchor box）计算得出的，代表的是ground-truth box与anchor box之间的转化关系。用这个来训练rpn，那么rpn最终学会输出一个良好的转化关系t。而这个t，是predicted box与anchor box之间的转化关系。通过这个t和anchor box，可以计算出预测框box的真实坐标。

RPN中的anchor box是怎么选取的?

滑窗的中心在原像素空间的映射点称为anchor，以此anchor为中心，生成k(paper中default k=9, 3 scales and 3 aspect ratios/不同尺寸和不同长宽比)个proposals。三个面积尺寸（128^2，256^2，512^2），然后在每个面积尺寸下，取三种不同的长宽比例（1:1,1:2,2:1）

为什么提出anchor box?

主要有两个原因：一个窗口只能检测一个目标、无法解决多尺度问题。

目前anchor box尺寸的选择主要有三种方式：人为经验选取、k-means聚类、作为超参数进行学习

为什么使用不同尺寸和不同长宽比？ 为了得到更大的交并比(IOU)。

3.说一下RoI Pooling是怎么做的？有什么缺陷？有什么作用

RoI Pooling的过程就是将一个个大小不同的box矩形框，都映射成大小固定（w * h）的矩形框

具体操作：（1）根据输入image，将ROI映射到feature map对应位置（2）将映射后的区域划分为相同大小的sections（sections数量与输出的维度相同）；（3）对每个sections进行max pooling操作；

这样可以从不同大小的方框得到固定大小的相应的feature maps。值得一提的是，输出的feature maps的大小不取决于ROI和卷积feature maps大小。ROI pooling 最大的好处就在于极大地提高了处理速度。（在Pooling的过程中需要计算Pooling后的结果对应到feature map上所占的范围，然后在那个范围中进行取max或者取average。）

优点： 1.允许我们对CNN中的feature map进行reuse；2.可以显著加速training和testing速度；3.允许end-to-end的形式训练目标检测系统。

缺点： 由于 RoIPooling 采用的是最近邻插值（即INTER_NEAREST），在resize时，对于缩放后坐标不能刚好为整数的情况，采用了粗暴的舍去小数，相当于选取离目标点最近的点，损失一定的空间精度。

两次整数化（量化）过程：1.region proposal的xywh通常是小数，但是为了方便操作会把它整数化。2.将整数化后的边界区域平均分割成 k x k 个单元，对每一个单元边界进行整数化。 //经过上述两次整数化，此时的候选框已经和最开始回归出来的位置有一定的偏差，这个偏差会影响检测或者分割的准确度

怎么做的映射: 映射规则比较简单，就是把各个坐标除以“输入图片与feature map的大小的比值”

ROI Pooling与ROI Align(Mask R-CNN)的区别

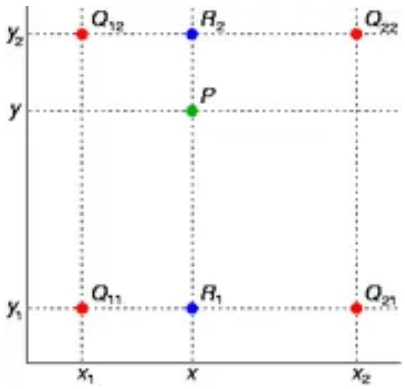
ROI Align: ROI Align的思路很简单：取消量化操作，使用双线性内插的方法获得坐标为浮点数的像素点上的图像数值,从而将整个特征聚集过程转化为一个连续的操作;1.遍历每一个候选区域，保持浮点数边界不做量化。2.将候选区域分割成k x k个单元，每个单元的边界也不做量化。3.在每个单元中计算固定四个坐标位置，用双线性内插的方法计算出这四个位置的值，然后进行最大池化操作。

区别: ROI Align舍去了近似像素取整数的量化方法，改用双线性插值的方法确定特征图坐标对应于原图中的像素位置.ROI Align很好地解决了ROI Pooling操作中两次量化造成的区域不匹配(mis-alignment)的问题。

对于检测图片中大目标物体时，两种方案的差别不大，而如果是图片中有较多小目标物体需要检测，则优先选择RoiAlign，更精准些。

RoI Align中双线性插值计算像素值的具体方法

在数学上，双线性插值是有两个变量的插值函数的线性插值扩展，其核心思想是在两个方向分别进行一次线性插值。



假如我们想得到未知函数 f 在点 P = (x, y) 的值，假设我们已知函数 f 在 Q11 = (x1, y1)、Q12 = (x1, y2), Q21 = (x2, y1) 以及 Q22 = (x2, y2) 四个点的值。最常见的情况，f就是一个像素点的像素值。首先在 x 方向进行线性插值，得到

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \text{ where } \quad R_1 = (x, y_1),$$
$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad \text{where} \quad R_2 = (x, y_2).$$

然后在 y 方向进行线性插值，得到

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2)$$

综合起来就是双线性插值最后的结果：

$$f(x,y) \approx \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y) + \frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y)$$
$$+\frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1) + \frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).$$

由于图像双线性插值只会用相邻的4个点，因此上述公式的分母都是1。

每个采样点的特征值由其相邻的4个整型特征点的像素值通过双线性差值得到。

最近邻插值法(图像的内插):在原图中最近得像素点赋值给新的像素点

4.说一下非极大值抑制（NMS）（non maximum suppression） NMS实现细节 手写NMS代码

用处：本质是搜索局部极大值，抑制非极大值元素。

原理:NMS为非极大值抑制，用来抑制检测时冗余的框。

大致算法流程为：1.对所有预测框的置信度降序排序2.选出置信度最高的预测框，确认其为正确预测，并计算他与其他预测框的IOU 3.根据2中计算的IOU去除重叠度高的，IOU>threshold阈值就删除 4.剩下的预测框返回第1步，直到没有剩下的为止

（需要注意的是：Non-Maximum Suppression一次处理一个类别，如果有N个类别，Non-Maximum Suppression就需要执行N次。）

假设两个目标靠的很近，则会识别成一个bbox，会有什么问题，怎么解决?

当两个目标靠的非常近时，置信度低的会被置信度高的框抑制掉，从而两个目标靠的非常近时会被识别成一个bbox。为了解决这个问题，可以使用softNMS（**基本思想**：用稍低一点的分数来代替原有的分数，而不是直接置零）

5.Faster R-CNN是如何解决正负样本不平衡的问题?

限制正负样本比例为1:1，如果正样本不足，就用负样本补充，这种方法后面研究工作用的不多。通常针对类别不平衡问题可以从调整样本数或修改loss weight两方面去解决，常用的方法有OHEM、OHNM、class balanced loss和Focal loss。

Faster RCNN怎么筛选正负anchor

我们给两种锚点分配一个正标签：（i）具有与实际边界框的重叠最高交并比（IoU）的锚点，

（ii）具有与实际边界框的重叠超过0.7 IoU的锚点。IoU比率低于0.3，我们给非正面的锚点分配一个负标签。

6.faster-rcnn中bbox回归用的是什么公式，说一下该网络是怎么回归bbox的?

$$\begin{aligned}
t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}} \\
t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}) \\
t^*_{\mathrm{x}} &= (x^* - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t^*_{\mathrm{y}} = (y^* - y_{\mathrm{a}})/h_{\mathrm{a}} \\
t^*_{\mathrm{w}} &= \log(w^*/w_{\mathrm{a}}), \quad t^*_{\mathrm{h}} = \log(h^*/h_{\mathrm{a}})
\end{aligned}$$

其中x,y,w,h分别为bbox的中心点坐标，宽与高。x, x_a, x^*分别是预测box、anchor box、真实box。

前两行是预测的box关于anchor的offset与scales，后两行是真实box与anchor的offset与scales。那回归的目的很明显，即使得t_i, t_i^*尽可能相近。回归损失函数利用的是Fast-RCNN中定义的smooth L1函数，对外点更不敏感：

$$\mathrm{L}_{\mathrm{reg}}(t_i, t_i^*) = \sum_{i \in \{x,y,w,h\}} \mathrm{smooth}_{\mathrm{L1}}(t_i - t_i^*)$$

in which

$$\text{smooth }_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

损失函数优化权重W，使得测试时bbox经过W运算后可以得到一个较好的平移量（offsets）与尺度（scales），利用这个平移量（offsets）与尺度（scales）可在原预测bbox上微调，得到更好的预测结果。

为什么要做Bounding-box regression?

边框回归用来微调候选区域/框，使微调后的框更Ground Truth更接近。

7.简述faster rcnn的前向计算过程 简述faster rcnn训练步骤

输入一张待检测图片->vgg16网络conv layers提取整张图片的特征，输出feature map分别输入到RPN和Fast RCNN网络开头->RPN网络得出region proposal，将这些候选框信息送入到Fast RCNN网络开头->利用候选框在之前送到的feature map提取特征，并通过ROI Pooling层得到规定大小的feature map->将这些feature map送入Fast RCNN网络中进行分类和回归坐标，最终得到需检测物体的坐标。

简述faster rcnn训练步骤

第一步，训练RPN，该网络用ImageNet预训练的模型初始化，并端到端微调，用于生成region proposal；

第二步，训练Fast R-CNN，由imageNet model初始化，利用第一步的RPN生成的region proposals作为输入数据，训练Fast R-CNN一个单独的检测网络，这时候两个网络还没有共享卷积层；

第三步，调优RPN，用第二步的fast-rcnn model初始化RPN再次进行训练，但固定共享的卷积层，并且只微调RPN独有的层，现在两个网络共享卷积层了；

第四步，调优Fast R-CNN,由第三步的RPN model初始化fast-RCNN网络，输入数据为第三步生成的proposals。保持共享的卷积层固定，微调Fast R-CNN的fc层。这样，两个网络共享相同的卷积层，构成一个统一的网络。

8. Faster rcnn有什么不足的地方吗？如何改进?

改进：1.更好的特征网络ResNet等；2.更精确的RPN：可以使用FPN网络架构来设计RPN网络3.更好的ROI分类方法：比如ROI分别在conv4和conv5上做ROI-Pooling，合并后再进行分类，这样基本不增加计算量，又能利用更高分辨率的conv4；4.使用softNMS代替NMS;

比较FasterRCNN在RCNN系列中的改进点 RPN提取RP

综合问题

1.简要阐述一下One-Stage、Two-Stage模型

One-Stage检测算法，没有selective search产生region proposal的阶段，直接产生物体的类别概率和位置坐标，经过单次检测即可直接获得最终的检测结果。相比Two-Stage有更快的速度。代表网络有YOLO v1/v2/v3/9000,SSD,Retina-Net.（two-stage算法中的roi pooling会对目标做resize, 小目标的特征被放大，其特征轮廓也更为清晰，因此检测也更为准确）

Two-Stage检测算法将检测问题划分成两个阶段，首先是获取region proposal进行位置精修和分类阶段。相比于One-Stage,精度高，漏检率也低，但是速度较慢，代表网络有Fast rcnn，Faster rcnn，mask rcnn等。

Two-Stage和One-Stage的异同（回答的是Two-Stage先对前景背景做了筛选，再进行回归，回归效果比较好，**准度高但是相比较慢**，One-Stage是直接对特征上的点进行直接回归，优点是**速度快**，因为用了多层特征图出框可能小目标效果比较好一点（个人看法），缺点是因为正负样本失衡导致效果较差，要结合难例挖掘。）

one stage在哪些具体方面检测精度不高（ROI+default box的深层理解）（one-stage算法对小目标检测效果较差，如果所有的anchor都没有覆盖到这个目标，那么这个目标就会漏检。）

Faster rcnn的两阶段训练和end-to-end训练的不一样（回答的是就是把RPN和二阶段拆开训，然后追问RPN在ENDTOEND中怎么回传，答TOTALLoss中有一阶段和二阶段的LOSS，只是回传影响的部分不一样。）

目标检测的发展历程，从传统到深度（传统部分回答的算子结合分类器分类，简单说了一下缺陷，深度部分说了RCNN,FAST,FASTER,SSD,YOLO,FPN,MASK RCNN,Cascade RCNN，都简单的介绍了一下）

传统目标检测：主线：区域选择->特征提取->分类器

传统的做目标检测的**算法基本流程**如下：1. 使用不同尺度的滑动窗口选定图像的某一区域为候选区域；2. 从对应的候选区域提取如Harr HOG LBP LTP等一类或者多类特征；3. 使用Adaboost SVM 等分类算法对对应的候选区域进行分类，判断是否属于待检测的目标。

缺点：1）基于滑动窗口的区域选择策略没有针对性，时间复杂度高，窗口冗余2）手工设计的特征对于多样性的变化没有很好的鲁棒性

2.YOLOV1、YOLOV2、YOLOV3复述一遍 YOLOv1到v3的发展历程以及解决的问题。

YOLO系列算法是一类典型的one-stage目标检测算法，其利用anchor box将分类与目标定位的回归问题结合起来，从而做到了高效、灵活和泛化性能好。

YOLOv1：YOLOv1的核心思想就是利用整张图作为网络的输入，直接在输出层回归 bounding box（边界框） 的位置及其所属的类别。

YOLOv1的基本思想是把一副图片，首先reshape成448×448大小（由于网络中使用了全连接层，所以图片的尺寸需固定大小输入到CNN中），然后将划分成SxS个单元格（原文中S=7），以每个格子所在位置和对应内容为基础，来预测检测框和每个框的Confidence以及每个格子预测一共C个类别的概率分数。

创新点：1. 将整张图作为网络的输入，直接在输出层回归bounding box的位置和所属的类别2. 速度快，one stage detection的开山之作

损失函数设计细节：YOLOv1对位置坐标误差，IoU误差，分类误差均使用了均方差作为损失函数。**激活函数**（最后一层全连接层用线性激活函数，其余层采用leak RELU）

缺点：1. 首先，每个单元格只预测2个bbox，然后每个单元格最后只取与gt_bbox的IOU高的那个最为最后的检测框，也只是说每个单元格最多只预测一个目标。**2.** 损失函数中，大物体 IOU 误差和小物体 IOU 误差对网络训练中 loss 贡献值接近（虽然采用求平方根方式，但没有根本解决问题）。因此，对于小物体，小的 IOU 误差也会对网络优化过程造成很大的影响，从而降低了物体检测的定位准确性。**3.** 由于输出层为全连接层，因此在检测时，YOLO 训练模型只支持与训练图像相同的输入分辨率的图片。**4.** 和two-stage方法相比，没有region proposal阶段，召回率较低

YOLOv2：YOLOv2又叫YOLO9000，其能检测超过9000种类别的物体。相比v1提高了训练图像的分辨率；引入了faster rcnn中anchor box的思想，对网络结构的设计进行了改进，输出层使用卷积层替代YOLO的全连接层，联合使用coco物体检测标注数据和imagenet物体分类标注数据训练物体检测模型。相比YOLO，YOLO9000在识别种类、精度、速度、和定位准确性等方面都有大大提升。**相比于v1的改进：** 1.Anchor: 引入了Faster R-CNN中使用的Anchor，作者通过在所有训练图像的所有边界框上运行k-means聚类来选择锚的个数和形状(k = 5，因此它找到五个最常见的目标形状) 2. 修改了网络结构，去掉了全连接层，改成了全卷积结构。3. 使用Batch Normalization可以从model中去掉Dropout，而不会产生过拟合。4. 训练时引入了世界树（WordTree）结构，将检测和分类问题做成了一个统一的框架，并且提出了一种层次性联合训练方法，将ImageNet分类数据集和COCO检测数据集同时对模型训练。

YOLOv3：YOLOv3总结了自己在YOLOv2的基础上做的一些尝试性改进，有的尝试取得了成功，而有的尝试并没有提升模型性能。其中有两个值得一提的亮点，一个是使用残差模型，进一步加深了网络结构；另一个是使用FPN架构实现多尺度检测。

改进点：1.多尺度预测 （类FPN）：每种尺度预测3个box, anchor的设计方式仍然使用聚类，得到9个聚类中心。2.更好的基础分类网络（类ResNet）和分类器 darknet-53。3.用逻辑回归替代softmax作为分类器。

（1）**yolo的预测框是什么值（x,y,w,h）**

（2）**YOLOv2中如何通过K-Means得到anchor boxes**

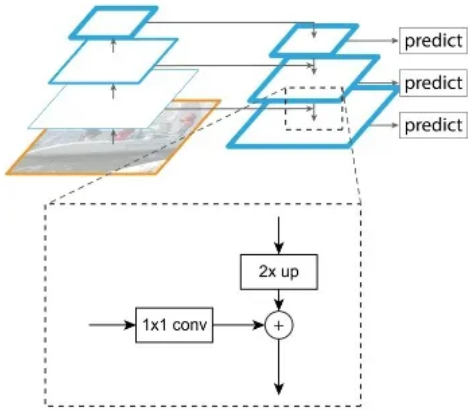
卷积神经网络具有平移不变性，且anchor boxes的位置被每个栅格固定，因此我们只需要通过k-means计算出anchor boxes的width和height即可

（3）**YOLOv3框是怎么聚出来的？ YOLOv3有没有很致命的问题？**

yolov3通过聚类的方式自定义anchor box的大小，在一定程度上，这可以提高定位的准确率。**缺点：** 识别物体位置精准性差，召回率低（在每个网格中预测两个bbox这种约束方式减少了对同一目标的多次检测）（4）**YOLO系列anchor的设计原理，kmeans的原理，anchor距离如何度量，如何改进**k-means原理：K-means算法是很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。该算法认为簇是由距离靠近的对象组成的，因此把得到紧凑且独立的簇作为最终目标。

由于从标记文件的width，height计算出的anchor boxes的width和height都是相对于整张图片的比例（w=anchor_width*****input_width/downsamples、h=anchor_height*input_height/downsamples）

3.简要阐述一下FPN网络具体是怎么操作的 FPN网络的结构



FPN网络直接在Faster R-CNN单网络上做修改，每个分辨率的 feature map 引入后一分辨率缩放两倍的 feature map 做 element-wise 相加的操作。通过这样的连接，每一层预测所用的 feature map 都融合了不同分辨率、不同语义强度的特征，融合的不同分辨率的 feature map 分别做对应分辨率大小的物体检测。这样保证了每一层都有合适的分辨率以及强语义（rich semantic）特征。同时，由于此方法只是在原网络基础上加上了额外的跨层连接，在实际应用中几乎不增加额外的时间和计算量。

FPN的特征融合为什么是相加操作呢?

假设两路输入来说，如果是通道数相同且后面带卷积的话，add等价于concat之后对应通道共享同一个卷积核。FPN里的金字塔，是希望把分辨率最小但语义最强的特征图增加分辨率，从性质上是可以用add的。如果用concat，因为分辨率小的特征通道数更多，计算量是一笔不小的开销。所以FPN里特征融合使用相加操作可以理解为是为了降低计算量。

阐述一下FPN为什么能提升小目标的准确率

低层的特征语义信息比较少，但是目标位置准确；高层的特征语义信息比较丰富，但是目标位置比较粗略。原来多数的object detection算法都是只采用顶层特征做预测。FPN同时利用低层特征高分辨率和高层特征的高语义信息，通过融合这些不同特征层的特征达到预测的效果。并且预测是在每个融合后的特征层上单独进行的。所以可以提升小目标的准确率。

基于FPN的RPN是怎么训练的？（在FPN的每个预测层上都接一个RPN子网，确定RPN子网的正负anchor box样本，再计算各预测层上RPN的anchor box分类和回归损失，利用BP将梯度回传更新权值）

4.简要阐述一下SSD网络

SSD网络的特点是对不同尺度下的feature map中的每一个点都设置一些default box,这些default box有不同的大小和横纵比例，对这些default box进行分类和边框回归的操作。SSD的核心是对固定设置的default box（不同尺度feature map中每一个空间位

置都设置一组default box，这里只考虑空间位置，不考虑feature的通道个数）计算属于各类物体的概率以及坐标调整的数值。这个计算方式是对每层的feature map做卷积操作，卷积核设定为3*3，卷积核的个数是与default box个数相关。

优点：SSD的优点是运行速度超过yolo，精度在一定条件下超过faster rcnn。**缺点**是需要人工设置先验框（prior box）和min_size，max_size和长宽比（aspect_ratio）值，网络中default_box的基础大小和形状不能直接通过学习获得，而是需要手工设置，虽然使用了图像金字塔的思路，但是对小目标的recall（召回率）依然一般

简述SSD网络前向是如何计算的

1数据增强，获取训练样本，将训练数据统一resize到固定尺寸；2.通过卷积网络获取feature map：①使用的卷积网络，前半部分使用基础分类网络获取各层的feature map，这部分称为base network。②下一步计算的输入，就是上述的不同尺寸的feature map；3.通过卷积操作，从特征图中获取检测信息。①此处实现方式与yolo类似；②与Faster R-CNN类似，在特征图中每个点新建若干固定尺寸的anchor。检测信息包括每个anchor的信息。主要包括：confidence（代表这个anchor中是否存在物体）、分类信息以及bbox信息。

SSD的致命缺点，如何改进

SSD主要缺点：SSD对小目标的检测效果一般，作者认为小目标在高层没有足够的信息

对小目标检测的改进可以从下面几个方面考虑：1. 增大输入尺寸2. 使用更低的特征图做检测(比如S3FD中使用更低的conv3_3检测)3. FPN(已经是检测网络的标配了)

5.简要阐述一下RetinaNet

RetinaNet的作者对one-stage检测器准确率不高的问题原因进行探究，发现主要问题在于正负类别不均衡，提出Focal Loss来解决类别不平衡问题。目的是通过减少易分类样本的权重，从而使得模型在训练时更注重于难分类的样本。RetinaNet=ResNet+FPN+Two sub-network+Focal Loss; RetinaNet由backbone网络和两个子任务网络组成，backbone网络负责计算feature map，子任务网络一个负责目标分类，一个负责bbox回归，网络的loss使用Focal loss。

阐述一下ssd和retinanet的区别

SSD的基础网络是VGG，且SSD在使用多层feature map时只是简单的在不同层的feature map上放default box，并没有真正将低维度特征和高维度特征进行融合。且SSD网络中使用的控制正负样本数量比的方法是难样本挖掘方法，loss是分类+回归的loss。而RetinaNet网络的基础网络是resnet+FPN，是真正将低维度的特征和高维度的特征进行了特征融合后再来做检测的。且控制正负样本的方法是使用Focal Loss。

6. faster rcnn和yolo，ssd之间的区别和联系

1.针对之前RCNN系列selective search的方法导致算法没有实时性，所以faster rcnn提出RPN网络来取代之前的方法，可以理解为fasterrcnn=fast rcnn+rpn网络，且rpn网络和fast rcnn的分类，回归网络共用特征提取层，这样使得引入RPN网络不会增加太多计算量。整体流程为先使用RPN网络找出可能存在object的区域，再将这些区域送入fast rcnn中进一步定位和分类。所以faster rcnn是典型的Two stage算法。因为faster rcnn中包含了两次定位，所以其精度一般高于YOLO和SSD算法，所以速度一般慢于YOLO和SSD。

2.YOLO算法的特点是将检测问题转换成回归问题，即YOLO直接通过回归一次既产生坐标，又产生每种类别的概率。YOLO中将每张图分成7*7的网格，每个网格默认可能属于2个object，即在一张图片上提取98个region proposal，相比于faster rcnn使用Anchor机制提取20k个anchor再从中提取最终的300个region proposal，所以faster rcnn的精度比YOLO要高，但是由于需要处理更多region proposal，所以faster rcnn的速度要比YOLO慢。

3.SSD相比于faster rcnn使用了多层网络特征，而不仅仅使用最后一层feature map。SSD还借鉴了YOLO算法中将检测任务转换为回归任务的思想，且SSD也借鉴了faster rcnn中的anchor机制，只是SSD的anchor不是每个位置的精调，而是类似于YOLO那样在feature map上分割出网格，在网格上产生anchor。但是SSD和YOLO不需要selective search步骤，所以SSD和YOLO同属于One-Stage算法。

阐述一下Mask RCNN网络，这个网络相比于Faster RCNN网络有哪些改进的地方

Mask rcnn网络是基于faster rcnn网络架构提出的新的目标检测网络。该网络可以在有效地完成目标检测的同时完成实例分割。Mask RCNN主要的贡献在于如下：1.强化了基础网络：通过ResNeXt-101+FPN用作特征提取网络，达到state-of-the-art的效果。2.ROIAlign替换之前faster rcnn中的ROI Pooling，解决错位（Misalignment）问题。3.使用新的Loss Function：Mask RCNN的损失函数是分类，回归再加上mask预测的损失之和。总结来说，mask rcnn的主要贡献就是采用了ROI Align以及加了一个mask分支。

7.分析一下SSD,YOLO,Faster rcnn等常用检测网络对小目标检测效果不好的原因

SSD，YOLO等单阶段多尺度算法，小目标检测需要较高的分辨率，SSD对于高分辨的低层特征没有再利用，而这些层对于检测小目标很重要。按SSD的设计思想，其实SSD对小目标应该有比较好的效果，但是需要重新精细设计SSD中的default box，比如重新设计min_sizes参数，扩大小default box的数量来cover住小目标。但是随着default box数量的增加，网络速度也会降低。YOLO网络可以理解为是强行把图片分割成7*7个网格，每个网格预测2个目标，相当于只有98个anchor，所以不管是小目标，还是大目标，YOLO的表现都不是很理想，但是由于只需处理少量的anchor，所以YOLO的速度上有很大优势。

Faster rcnn系列对小目标检测效果不好的原因是faster rcnn只用卷积网络的最后一层，但是卷积网络的最后一层往往feature map太小，导致之后的检测和回归无法满足要求。甚至一些小目标在最后的卷积层上直接没有特征点了。所以导致faster rcnn对小目标检测表现较差。

8.手写计算IOU代码

有两个框，设第一个框的两个关键点坐标：（x1,y1）(X1,Y1)，第二个框的两个关键点坐标：（x2,y2）(X2,Y2)。以大小写来区分左上角坐标和右下角坐标。首先，要知道两个框如果有交集，一定满足下面这个公式：max(x1,x2)<=min(X1,X2) && max(y1,y2)<=min(Y1,Y2)！！！！

9.讲一下目标检测优化的方向

【可以从数据集下手，提升特征表征强度（backbone下手，加深加宽或者换卷积方式），RPN下手（级联，FPN，IOU NET），LOSS（行人检测领域有些问题，如重叠，可以靠修改loss提升准确度）。】

10.anchor设置的意义：

其实就是多尺度的滑动窗口

11.如果只能修改RPN网络的话，怎么修改可以提升网络小目标检出率

①修改RPN网络的结构，比如引入FPN结构，利用多层feature map融合来提高小目标检测的精度和召回；②针对小目标重新精细设计Anchor的尺寸和形状，从而更好地对小目标进行检测；

12.如何理解concat和add这两种常见的feature map特征融合方式

两者都可以理解为整合特征图信息。concat是通道数的增加;add是特征图相加，通道数不变。 add是描述图像的特征下的信息量增多了，但是描述图像的维度本身并没有增加，只是每一维下的信息量在增加，这显然是对最终的图像的分类是有益的。而concatenate是通道数的合并，也就是说描述图像本身的特征数（通道数）增加了，而每一特征下的信息是没有增加。 concat每个通道对应着对应的卷积核。 而add形式则将对应的特征图相加，再进行下一步卷积操作，相当于加了一个先验：对应通道的特征图语义类似，从而对应的特征图共享一个卷积核（对于两路输入来说，如果是通道数相同且后面带卷积的话，add等价于concat之后对应通道共享同一个卷积核）。因此add可以认为是特殊的concat形式。但是add的计算量要比concat的计算量小得多。

13.阐述一下如何检测小物体

小目标难以检测的原因：分辨率低，图像模糊，携带的信息少。

①借鉴FPN的思想，在FPN之前目标检测的大多数方法都是和分类一样，使用顶层的特征来进行处理。虽然这种方法只是用到了高层的语义信息，但是位置信息却没有得到，尤其在检测目标的过程中，位置信息是特别重要的，而位置信息又是主要在网络的低层。因此FPN采用了多尺度特征融合的方式，采用不同特征层特征融合之后的结果来做预测。

②要让输入的分布尽可能地接近模型预训练的分布。先用ImageNet做预训练，之后使用原图上采样得到的图像来做微调，使用微调的模型来预测原图经过上采样的图像。该方法提升效果比较显著。

③采用多尺度输入训练方式来训练网络；

④借鉴Cascade R-CNN的设计思路，优化目标检测中Two-Stage方法中的IOU阈值。检测中的IOU阈值对于样本的选取是至关重要的，如果IOU阈值过高，会导致正样本质量很高，但是数量会很少，会出现样本比例不平衡的影响；如果IOU阈值较低，样本数量就会增加，但是样本的质量也会下降。如何选取好的IOU，对于检测结果来说很重要。⑤采用分割代替检测方法，先分割，后回归bbox来检测微小目标。

14.阐述一下目标检测任务中的多尺度

输入图片的尺寸对检测模型的性能影响相当明显，事实上，多尺度是提升精度最明显的技巧之一。在基础网络部分常常会生成比原图小数十倍的特征图，导致小物体的特征描述不容易被检测网络捕捉。通过输入更大、更多尺寸的图片进行训练，能够在一定程度上提高检测模型对物体大小的鲁棒性，仅在测试阶段引入多尺度，也可享受大尺寸和多尺寸带来的增益。

检测网络SSD中最后一层是由多个尺度的feature map一起组成的。FPN网络中采用多尺度feature map分层融合，分层预测的方法可以提升小目标的检测效果。

阐述一下如何进行多尺度训练

多尺度训练可以分为两个方面:一个是图像金字塔，一个是特征金字塔

1、人脸检测的MTCNN就是图像金字塔，使用多种分辨率的图像送到网络中识别，时间复杂度高，因为每幅图都要用多种scale去检测。2、FPN网络属于采用了特征金字塔的网络，一次特征提取产生多个feature map即一次图像输入完成，所以时间复杂度并不会增加多少3、faster rcnn多个anchor带来的多种尺寸的roi可以算muti scale思想的应用。

15.如果有很长，很小，或者很宽的目标，应该如何处理目标检测中如何解决目标尺度大小不一的情况 小目标不好检测，有试过其他的方法吗？比如裁剪图像进行重叠

小目标不好检测的两大原因：1）数据集中包含小目标的图片比较少，导致模型在训练的时候会偏向medium和large的目标。2）小目标的面积太小了，导致包含目标的anchor比较少，这也意味着小目标被检测出的概率变小。

改进方法： 1）对于数据集中含有小目标图片较少的情况，使用过度采样（oversample）的方式，即多次训练这类样本。2）对于第二类问题，则是对于那些包含小物体的图像，将小物体在图片中复制多分，在保证不影响其他物体的基础上，人工增加小物体在图片中出现的次数，提升被anchor包含的概率。3）使用FPN；4）RPN中anchor size的设置一定要合适，这样可提高proposal的准确率。5）对于分辨率很低的小目标，我们可以对其所在的proposal进行超分辨率，提升小目标的特征质量，更有利于小目标的检测。

16.检测的框角度偏移了45度，这种情况怎么处理

RRPN也是基于Faster R-CNN，引入RPN，它对比CTPN加入了旋转信息。CTPN只能检测水平文本，而RRPN可以检测任意方向的文本，因为CTPN的提议框是水平的，而RRPN的提议框带有旋转角度。为什么提出旋转的提议框呢？因为水平提议框在检测倾斜文本的时候会带有一些冗余（非文本部分）

参考文献

- Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.
- Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.
- Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

参考链接

- https://zhuanlan.zhihu.com/p/137735486
- https://zhuanlan.zhihu.com/p/137736076
- https://blog.csdn.net/wshdkf/article/details/80456588
- https://www.cnblogs.com/eilearn/p/9061814.html
- https://zhuanlan.zhihu.com/p/56405179
- https://bbs.cvmart.net/topics/1310

本文亮点总结

1.NMS算法流程为：1.对所有预测框的置信度降序排序2.选出置信度最高的预测框，确认其为正确预测，并计算他与其他预测框的IOU 3.根据2中计算的IOU去除重叠度高的，IOU>threshold阈值就删除 4.剩下的预测框返回第1步，直到没有剩下的为止

2.Mask RCNN主要的贡献在于如下：1.强化了基础网络：通过ResNeXt-101+FPN用作特征提取网络，达到state-of-the-art的效果。2.ROIAlign替换之前faster rcnn中的ROI Pooling，解决错位（Misalignment）问题。3.使用新的Loss Function：Mask RCNN的损失函数是分类，回归再加上mask预测的损失之和。

实操教程：PyTorch vs LibTorch：网络推理速度谁更快？ | 只用两行代码，我让Transformer推理加速了50倍 | PyTorch AutoGrad C++层实现

算法技巧（trick）：深度学习训练tricks总结（有实验支撑） | 深度强化学习调参Tricks合集 | 长尾识别中的Tricks汇总（AAAI2021）

最新CV竞赛：2021 高通人工智能应用创新大赛 | CVPR 2021 | Short-video Face Parsing Challenge | 3D人体目标检测与行为分析竞赛开赛，奖池7万+，数据集达16671张！

阅读原文