yolov1

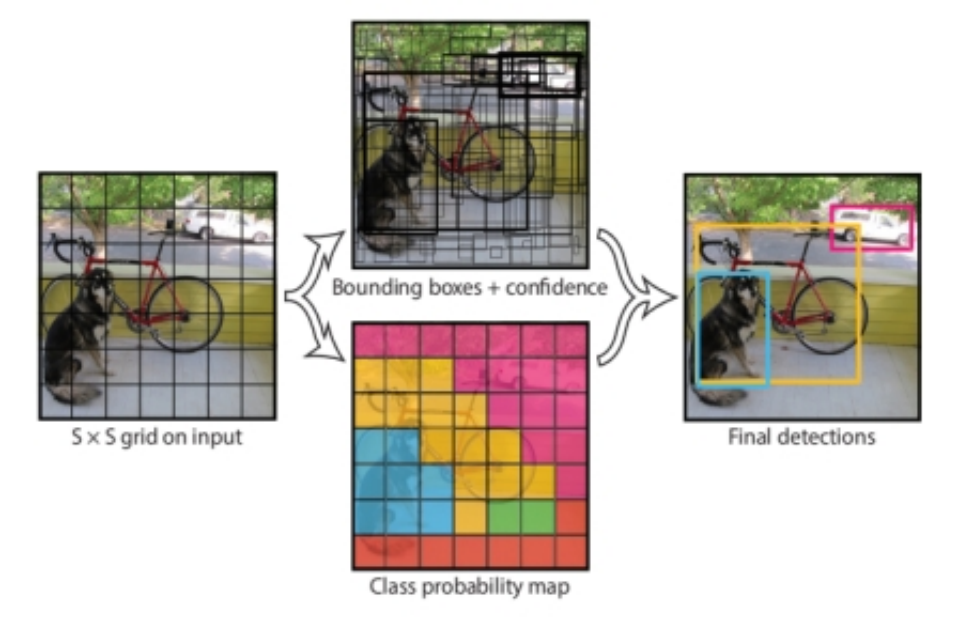
最近准备学习目标检测的相关技术，先从yolo开始学习，打算对yolov1到现如今的v7都进行学习，了解各个版本yolo所使用的相关技术，并且也准备对最近新出的yolo如yolov5、yolox、yolov7等进行测试比较，比较不同版本的yolo的性能如何。

首先从yolo的开山之作yolov1开始学起。

Introduction

Yolo是当前流行的目标检测算法，它不同于R-CNN系列等多阶段目标检测技术，yolov1将目标检测问题看作是回归问题，使用单个神经网络对目标的类别、位置、置信度等进行预测，是一种端到端的学习方法。此种学习方法能够大幅提高模型的运算速度，达到实时检测的要求。

yolo的基本思想是将整张图片划分为S*S的小方格，每个方格负责对中心点落入该方格的目标负责，即该方格负责预测该目标。原论文中将S取为7。

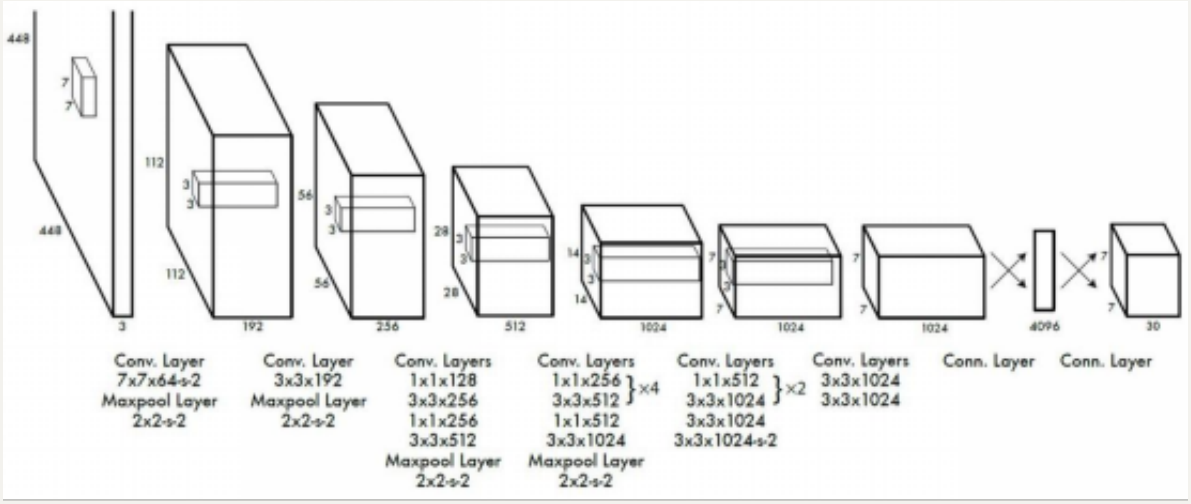


同时每个小方格内包含B个边界框（bounding box），表示边界框的参数有5个，[x,y,w,h,c]，c表示置信度，即该边界框包含目标的可信程度，xywh则表示边界框的位置信息。并且，对于每个小方格也同时输出K个类别概率。K表示总的类别数。所以，对每个小方格需要输出的量有K+B*(4+1)，若取K=20，B=2，则每个方格输出维度为30的向量。对于整张图片，则输出7*7*30的向量。

如果只是看yolo的原理其实很难理解它的流程，如果想要理解的更加深刻，还需要对yolov1的训练流程以及预测流程有清晰的认识（即用代码实现的流程）。在后文中也会详细介绍。

Model

yolov1的网络包含24个卷积层和两个全连接层，其最终输出7*7*30的向量



模型结构

网络的前20层卷积层用于下采样提取图像特征，yolov1的作者将前20层提取出来，加上一个average pooling层和全连接层构成了一个分类网络，并在ImageNet上预训练了一周左右，达到了88%的正确率。预训练的目的是训练模型提取特征的能力，这能够使后续的目标检测任务的训练更加容易，也会收敛的更快。

模型输出解释

1、方格

Yolo的输出位7*\7*30*，相当于将整个图片分为了*77个小方格，30代表了当前这个小方格的预测情况，包括预测框的位置信息、置信度、类别的预测概率。

2、边界框

每个小方格内还包含B个边界框，用于预测坐标的位置(x,y,w,h)，同时还需要预测每个边界框的置信度（该方格包含目标的可信程度），对包含有目标的小方格，还需要预测目标的类别C。当B=2，C=20时，有C+（B*4+B*2）= 30。所以对于每个小方格的预测输出为长度为30的向量，前20位为类别概率，后10位为[c1,c2,x,y,w,h,x,y,w,h]，代表两个预测框的置信度

和坐标预测。其实在这30维向量中对各输出信息的排列顺序没有要求，也可以把20位类别概率放到最后，不过排练的顺序应该和训练过程的损失函数对应起来。

3、[x,y,w,h]说明

x,y,w,h取值均为0-1。x y表示当前预测框的中心点相对于这个小方格的偏移量，而w和h表示预测框的长宽相对于整个图像的大小。比如当前方格所在位置位第一行第二列，即i=1-1=0，j=2-1=1，则方格的x，y在原图像的位置为，(i * 1 / 7 + x)*W，(j * 1 / 7 + y)*H。W和H为原图像的长宽。

为什么x y是相对于小方格的偏移量?

若x y为相对于整个图像的偏移量，则体现不出每个小方格负责预测中心点落入其中的目标，因为该小方格的预测输出的x，y的中心可能会出现在图像的任意位置。

Loss

yolov1的损失函数定义为： 可以分为3类：坐标损失、置信度损失、类别损失

loss function:

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2 \qquad (3)
\end{aligned}
$$

$$\lambda_{coord} = 5 \qquad \lambda_{noobj} = 0.5$$

其中

1_{ij}^{obj}表示第i个小方格的第j个预测框是否含有目标

1_{ij}^{noobj}则表示不含目标

1_{i}^{obj}表示第i个方格是否含有目标

训练过程中某个方格或者某个预测框是否含有目标是根据图像的label来决定的，训练过程中需要根据label计算真实框属于哪个方格，然后根据公式对各个方格计算loss

坐标损失

对预测框坐标与真实框坐标之间的差别做出惩罚。对含有目标的预测框进行坐标的惩罚。

如何界定某个预测框是否含有目标?

每个小方格负责对中心点落入该小方格内的目标进行预测，而该小方格内的两个预测框需要分别计算其与真实框之间的iou，iou大的预测框对该目标负责，即只对iou大的预测框进行坐标惩罚

训练具体流程:

根据label里面的所有真实框信息，找到所用真实框对应的小方格，计算真实框与预测框的iou，iou大的预测框负责预测该目标，即只对iou大的预测框计算坐标损失

计算坐标损失时，宽和高需要开方

若高和宽的计算方式同x和y的计算方式的话，比真实框大或者小同样距离的预测框计算出来的坐标损失相同，相比于小的预测框，我们更希望大的预测框，所以对小的预测框赋予更大的loss，所以对w和h进行开方后，可以降低大预测框的损失值

置信度损失

对于每个含有目标或者不含有目标的预测框都要计算置信度损失，对含有目标的预测框，其置信度的期望值为该预测框与真实框的iou，对于不含目标的预测框，其期望值为0

类别损失

对于含有目标的小方格，需要计算类别损失，类别损失计算方法与普通分类问题计算方法相同，使用均方和误差即可。

模型的coco评判标准

在对模型训练完成之后如何评判模型的性能呢，一般对于目标检测任务，有VOC评判标准和COCO评判标准，当前主要以COCO评判标准为主，而且VOC评判标准也是COCO评判标准的一种特殊情况。

现在对目标检测模型的评价标准一般采用coco评价标准，评价标准中评判模型好坏的量有P（precision）、R（Recall）、AP、mAP

TP(True Positive)

若预测为正例，实际为正例，为TP

FP(False Positive)

若预测为正例，而实际为反例，则为FP，下述相同

TN(True Negative)

FN(False Negative)

Precision

$$P = \frac{TP}{TP + FP}$$

P值反应了模型的所有预测框当中的正确率

Recall

$$R = \frac{TP}{TP + FN}$$

R值为召回率，反应了模型的所有预测框中包含有真实框的比率

一般来说，P值高时，R值就低，P值低时，R值高，所以很难根据单个P-R值来判断模型的好坏，所以引入了AP

AP

对于某一类别而言，在不同的情况下计算当前模型的P、R值，然后以P为纵坐标，R为横坐标画出P-R曲线图，P-R曲线下所包围的面积即为AP值。

在目标检测中，不同的情况指的是设定不同的置信度阈值，对高于阈值的所有预测框保留，低于阈值的舍弃，每次分别对保留的预测框计算P、R值

mAP

对所有类别的AP取均值即为mAP值

coco中AP和mAP

在coco评价标准中，对预测框与真实框的匹配中设定了阈值，对于同类别的真实框与预测框，若预测框与真实框iou大于阈值，则这两个框相互匹配，即为预测成功。Coco评价标准对该阈值取了从0.5到0.95的10个值，对其中的每一个值，都需要按上述步骤计算AP，对10个iou阈值下的AP取均值并对所有类别平均即可得到mAP

目标检测任务中mAP的计算流程

以yolo为例，其输出为7730的向量，每个小方格内包含2个边界框，所以yolov1对每张图片输出772=98个预测框。由于mAP是对所有类别求取的均值，所以需要分别求取每个类别的AP值。所以下述步骤是针对于同类别的预测框和真实框的。所以计算下述步骤前，需要将同类别的真实框和预测框提取出来。

1、 首先需要对输出的预测框进行NMS来筛选掉多余的框，NMS算法不做赘述。此时的置信度阈值选为0.001。

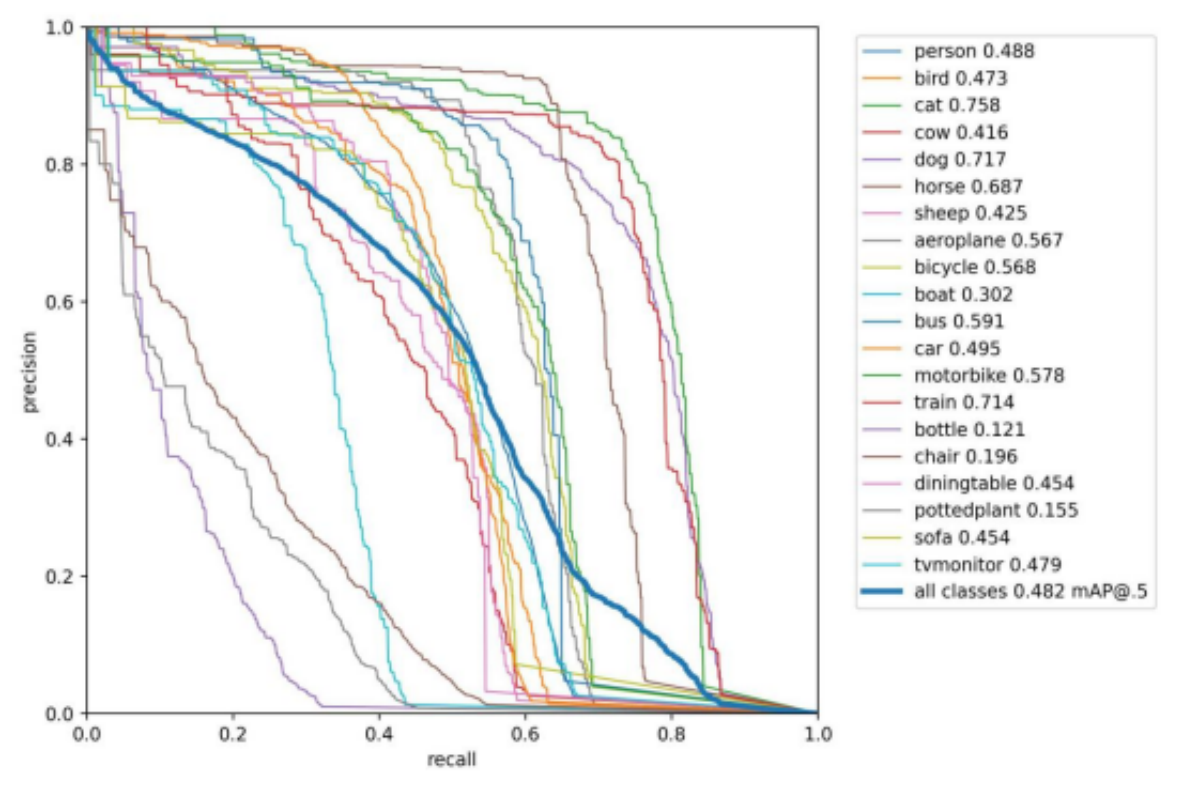
2、 将预测框与真实框进行匹配。流程为对每个真实框计算其与其他所有预测框的IOU，并设定一个IOU阈值（对COCO评判标准来说，为0.5-0.95中的某一个数）。然后选取出IOU大于阈值的所有预测框，并选取其中置信度最大的与真实框相匹配。每个真实框只能与一个预测框匹配。最后统计匹配情况，例如有x个真实框匹配成功，则TP = x，有y个真实框没有匹配成功，则FN = y，有z个预测框没有参与匹配，则FP = z。对整个数据集都进行这样的操作，并且也需要对10个IOU阈值都计算对应的TP、FP、FN，在目标检测问题中，TN没有被用到。

3、 对每个预测框做一个表格，有10个数据，分别表示该预测框在当前IOU阈值下是否匹配成功，是为1，否则为0

4、 对所有预测框进行置信度大小排序，同时第3点的表格也需要按照置信度从大到小排序，根据置信度的值设定置信度阈值（比如对整个数据集有多少个不同的置信度值就设定多少个阈值），根据阈值每次保留置信度大于该阈值的预测框，然后根据第2点计算TP、FP、FN。然后计算P、R，此时得到P-R曲线上的一点，同理对每个置信度阈值做相同操作即可画出每个类别的P-R曲线。

5、 计算P-R曲线下的面积即为AP，对所有类别取均值即为mAP。

下图为在IOU = 0.5时的mAP示意图



总结:

yolov1是yolo系列的开山之作，使用端到端的训练方法，在保证模型速度的同时也取得了不错的预测性能。

但是yolov1也存在缺陷，比如yolov1使用全连接层来预测坐标位置和置信度，导致模型参数量还比较大。同时yolov1的一个方格只允许预测一个目标，导致对多目标紧邻的检测任务效果差，yolo将图片分为7*7个方格，每个方格所对应的感受野也比较大，导致yolo对小目标的检测效果差。而且yolov1对每张图片只预测98个边界框，数量太少，导致模型查全率recall不高。