layout: post
title: "yolo系列文章之yolov1详解"
date: 2020-07-07
description: "目标检测"

tag: 目标检测

论文地址：[You Only Look Once: Unified, Real-Time Object Detection]

主要思想

先前的目标检测研究主要将目标检测问题看作一个分类问题，稍加改动分类器将其应用到目标检测任务中。在做着的工作中，将目标检测问题看作一个回归问题，在空间上分割边界框及其联系的类别概率。单个神经网络在一次评估中直接从完整图像上预测边界框和类别概率。由于整个检测pipeline是一个单一网络，因此可以直接对检测性能进行端到端的优化。

整个的yolo架构非常快，基础的yolo模型可以达到实时的45FPS(每秒45幅图片)，其中一个小版本的Fast yolo可以达到155FPS，对比现在最优的目标检测器，尽管yolo在位置的预测上产生更多的错误，但是相对不太可能在背景上预测物体产生假阳性（false positive）。yolo能够学习到非常通用的表达，而且从自然图像到其他领域（例如艺术品）泛化时，能够得到超出RCNN，DPM等算法的结果。

整体介绍

RCNN目标检测器采用区域建议（ region proposal）的方法，先生成一系列的区域，然后利用分类器，对每个区域进行分类，分类之后，应用后处理来精修框，移除重复的框，基于场景中的其他物体重新给bbox评定得分。这种复杂的pipeline很慢并且很难优化，各部分必须单独训练，训练过程较慢。

YOLO直接将目标检测问题看作一个回归问题，直接利用一个单一的网络，从图像像素直接生成bbox坐标和类别的概率。you only look once就可以直到物体在哪，是什么。

YOLO的优点：

- unified model 使得yolo的速度非常快
- 不像 sliding window和基于region proposal的rcnn系列方法，YOLO在训练与测试时可以看到整张图片，对全局的语义信息进行编码，因此相对可以产生较少的背景的假阳性（false positive）
- YOLO的泛化能力好，在自然图像上训练，在艺术品图像测试可以得到更好的效果

YOLO的缺点：YOLO准确率相对于RCNN系列的文章还有很大的差距，尤其是在小目标的识别上，还有很大的差距

训练测试还有预训练模型均开源：http://pjreddie.com/yolo/

网络结构介绍

YOLO是一个unified network，直接利用一张图片的全局特征来预测所有的bbox与所有的类别得分。

将一幅图片分成`SxS`个栅格， 如果一个物体的中心落到了某个栅格中，那么这个栅格就会用来预测这个物体。

每个栅格预测B个bbox和这些bbox的置信度得分，这些置信度反映了这个模型对于bbox中包含物体以及准确程度，如果在一个栅格中不存在物体那么这个置信度就为0，如果存在，那么置信度为预测框与真实bbox的IOU值。

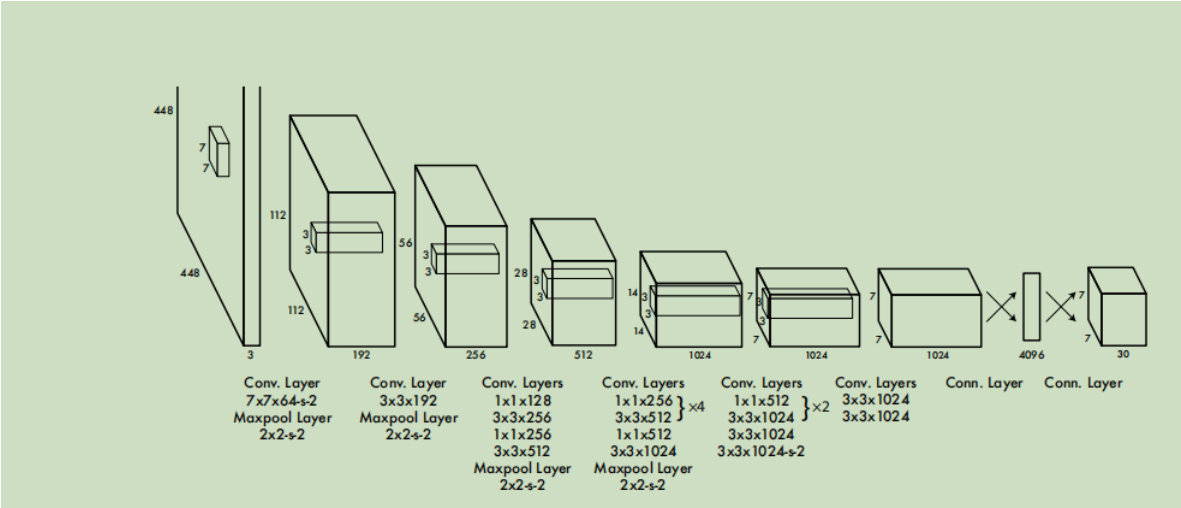
每个bbox包含5个预测值：`x, y, w, h, confidence`，这个`x, y`是bbox的中心，`w, h`为bbox的宽高，confidence为这个bbox与ground truth的IOU。

每个网格单元还预测C个条件类别概率。这些概率以包含目标的网格单元为条件。每个网格单元我们只预测的一组类别概率，而不管边界框的的数量B是多少。

然后利用类别的条件概率乘以bbox的confidence就得到每个bbox的特定类别的置信度得分。

由于grid cell的数目为`SxS`，bbox的数目为B，每个bbox预测`x, y, w, h, confidence`5个值，一共有C个类别，那么最终输出为`SxSx(Bx5+C)`大小的张量。

网络结构图：



在pascal voc数据集上的实验中`S = 7, B = 2, C = 20`，最终的输出为`7x7x30`。

训练过程中使用ImageNet预训练模型，并且使用图像的宽高将预测输出的w和h归一化到0， 1的范围内，同时利用grid cell的位置偏移，将bbox的中心x， y归一化到0，1。最后一层使用线性激活函数，其他层使用leak Relu。

优化了模型输出中的平方和误差。我们使用平方和误差，因为它很容易进行优化，但是它并不完全符合我们最大化平均精度的目标。分类误差与定位误差的权重是一样的，这可能并不理想。另外，在每张图像中，许多网格单元不包含任何对象。这将这些单元格的“置信度”分数推向零，通常压倒了包含目标的单元格的梯度。这可能导致模型不稳定，从而导致训练早期发散。

为了改善这一点，我们增加了边界框坐标预测损失，并减少了不包含目标边界框的置信度预测损失。我们使用两个参数λ_{coord}和λ_{noobj}来完成这个工作。我们设置λ_{coord}=5和λ_{noobj}=0.5。

平方和误差同样给大bbox和小bbox误差赋予了相同的权重。我们的误差指标应该反映出，大的bbox的偏差的重要性不如小bbox。为了部分解决这个问题，我们预测边界框宽度和高度的平方根，而不是宽度和高度。

YOLO对于每个grid cell来预测多个边界框，在训练的时候对于每个cell选择具有最大IOU的bbox来预测目标，这导致边界框预测器之间的专业化。每个预测器可以更好地预测特定大小，方向角，或目标的类别，从而改善整体召回率。

loss function:

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \qquad (3)$$

其中1_i^{obj}表示目标是否出现在网格单元i中，1_{ij}^{obj}表示网格单元i中的第j个边界框预测器"负责"该预测

YOLO的限制

YOLO对边界框的预测加了很强的空间约束，因为每个网格单元只预测两个盒子，只能有一个类别。这个空间约束限制了我们的模型可以预测的邻近目标的数量。我们的模型在密集的物体中表现不好，例如鸟群。

由于我们的模型学习从数据中预测边界框，因此它很难泛化到新的、不常见的长宽比或其他构造的目标。由于采用了多个下采样层，我们的模型也使用相对较粗糙的特征来预测边界框。

最后，当我们训练一个近似检测性能的损失函数时，我们的损失函数会同样的对待小边界框与大边界框的误差。大边界框的小误差通常是良性的，但小边界框的小误差对IOU的影响要大得多。我们的主要错误来源是不正确的定位。

总结

YOLO是一个unified目标检测器，构建简单，并且可以直接在整张图像上进行训练，整个模型可以进行端到端的训练，快速高效。