目标检测文献综述

1.RCNN：[Rich feature hierarchies for accurate object detection and semantic segmentation](file:///Users/lianlian/Documents/machinelearning/neuralNetwork/CV/Girshick_Rich_Feature_Hierarchies_2014_CVPR_paper.pdf)

key point: （1） 使用CNN进行自底向上的定位和分割物体

（2） 标签数据少时，使用监督预训练网络之后再对网络进行fine tune

步骤：（1）生成与目标种类无关的region proposals

（2）CNN网络提取特征

（3）针对每个类的线性SVM二分类器

对每个步骤进行详细说明：

region proposals：使用selective search搜索rp，提取两千个region proposals

提取特征：采用CNN提取目标区域中图像的特征，首先第一步需要将调整图像尺寸使其能够输入到CNN中，采用了直接对图像进行缩放的方式，输入尺寸为227\*227

检测：讲每个候选区域的特征输入到SVM中进行检测，得到该类的得分，之后根据IOU值使用极大值抑制的方法去除掉IOU大于某个阈值的region，为什么不直接用神经网络softmax层做分类而是用SVM？因为SVM和CNN对正负样本的定时方式不同，CNN在训练时对标注比较宽松，SVM则会比较严格，只有当bounding box将物体都包进去了才将它标位物体类别。在论文中好像有提到CNN中IOU阈值设为0.5，SVM阈值设为0.3.

问题：相当于每检测一幅图像需要输入2000个region图像数据用CNN对它做特征提取，大部分时间都花在这一步，之后再对每个region做21个svm二分类，不过是线性运算较快。

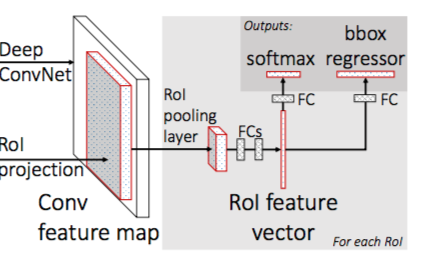
2.fast-rcnn: [Fast R-CNN](file:///Users/lianlian/Documents/machinelearning/neuralNetwork/CV/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf)

借鉴了SPPnet，速度方面进行了改进，同时增加了精度，RCNN速度慢的问题在于没有共享从同一幅图中提取出来的特征，中间进行了多次重复计算，SPPnet最早提出共享计算的方法，和RCNN一样，SPP依然一开始对图像进行selective search，选出两千个候选框，然后用卷积网络对输入图像进行特征提取，得到feature maps，之后根据某种对应方式得到feature maps 上对应的候选框，然后再对候选框进行SVM分类，和RCNN相比，共享了CNN的计算结果。

Fast-RCNN开始同样是将输入图像和选择性搜索得到的ROI输入到CNN中，提取得到feature maps，然后对每个候选框经过ROI pooling layer和全连接层，得到一固定长度的特征向量，然后将每个ROI的特征向量分别输入一个softmax层得到分类结果和一个regressor得到检测区域的回归结果。

和SPPnet最大的不同在于ROI pooling layer：

每一个RoI都有一个四元组（r,c,h,w）表示，其中（r，c）表示左上角，而（h，w）则代表高度和宽度。这一层使用最大池化（max pooling）来将RoI区域转化成固定大小的H\*W的特征图。假设一个RoI的窗口大小为h\*w,则转换成H\*W之后，每一个网格都是一个h/H \* w/W大小的子网，利用最大池化将这个子网中的值映射到H\*W窗口即可。



之后的训练步骤使用了多任务损失函数来计算整体的损失，是一个end-to-end的过程。

3.faster-cnn：[Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks](5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf)

fast-rcnn实现了特征参数的贡献，对proposals的计算成为限制物体检测的主要瓶颈，二faster-rcnn主要贡献在于使用RPN网络生成region proposals ，大大减少了计算proposals的损耗。

依据在于CNN提取出来的特征图可以用来生成region proposals



RPN如何实现：1.图片输入网络后，经过一系列conv+relu得到feature map，输出51\*39\*256特征

2.计算anchor ：在feature对每个特征点预测多个region proposals，具体做法是，把每个特征点映射回原图的感受野的中心作为基准点，围绕基准点选取k\*k个anchor

3.划分正负样本：对每个标定的ground truth区域，选与其重叠比例最大的anchor作为正样本，对剩余的anchor，若重叠比例大于0.7，同样作为正样本，若anchor与任意一个ground truth重叠区域均小于0.3，记作负样本，因此，一个ground truth可能具有多个正样本。之后对非正非负的anchor和跨越图像边界的anchor弃去不用

4.对每幅图随机取256个anchor，正负比例1：1对RPN进行训练。得到RPN

RPN和fast-rcnn如何共享参数：

1.RPN和fast-rcnn叠代训练，先用训练好的RPN输出proposals输入训练fast-rcnn。再用fast-rcnn训练得到的参数初始化RPN，重新训练RPN。之后再叠代循环训练

2.前面共享cnn输入的特征，之后由RPN生成proposals，再将得到的proposals输入到RoI pooling层和fast-rcnn对接（用bbox对接？使用cls）

3.4-step-Alternating Training：

（1） 用ImageNet模型初始化，独立训练一个RPN网络

（2） 仍然用ImageNet模型初始化，但是使用上一步RPN网络产生的proposal作为输入，训练一个Fast-Rcnn网络，至此，两个网络每一层的参数完全不共享

（3） 使用第二步的Fast-RCNN网络参数初始化一个新的RPN网络，但是把RPN，Fast-RCNN共享的卷积层的learning rate置为0，也就是不更新，仅仅更新RPN特有的网络层，重新训练，此时，RPN和FAST-RCNN已经共享了所有公共卷积层；

（4）仍然固定共享的那些网络层，把fast-rcnn特有的网络层也加入进来，形成一个unified network，继续训练，fine tunefast-rcnn特有的网络层，此时，网络已经实现我们设想的网络内部预测proposal并实现检测功能的目标。

可在github上查看流程：<https://github.com/rbgirshick/py-faster-rcnn/tree/master/models/pascal_voc/VGG16/faster_rcnn_alt_opt>

关于faster-rcnn的扩展包括使用多特征融合的方式早呢更加对小物体的检测精度，在hypernet等一些网络中有用到。

Mask-Rcnn还没有研究

基于回归的物体检测模型

1.YOLO

将物体检测看做一个单纯的回归问题，直接从输入的图像像素得到检测框和所属物体的概率

YOLO的优点有：1.快，实时检测，并且相较其他的实时检测系统精度高

2.能够利用到上下文信息，能够更好的检测出背景

3.YOLO泛化性能更强

缺点在于：

1.精度差

2.定位不够准确

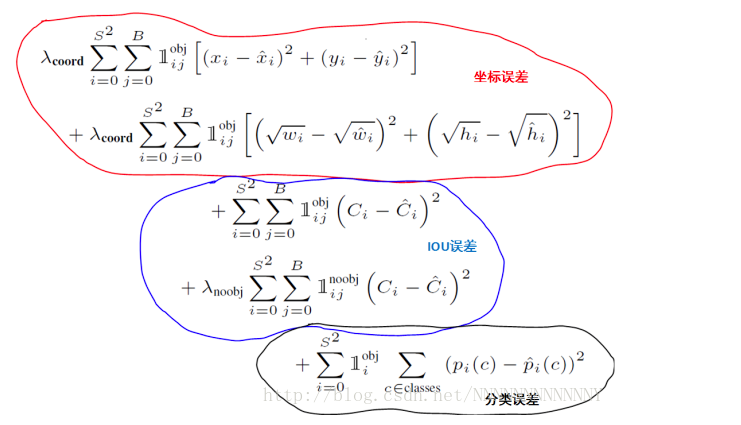
3.recall精度差

大致流程：

1.将输入图像分为S\*S个网格，如果一物体的中心落在网格内，这个网格则负责对这个物体的检测

2.每个网格预测B个bounding box和每个BB的confidence，confidence包括BB中包含这个物体的置信度和BB的预测精度。Confidence=Pr(object)\*IOU

3.所以每个BB需要预测五个值，x,y,w,h,c。每个网格需要预测C个类的条件概率，即预测当前被负责物体的种类Pr(class|object)

loss函数：

不过YOLO模型还存在很多缺点：

1.每个网格只能负责一个物体的识别

2.loss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，不太合理。

3.YOLO采用多个下采样层，学习到的特征不够精细，影响了检测效果。

2.YOLO v2

精度的改进

（1）使用batch normalization

（2）使用448\*448高分辨率的imagenet数据对网络进行微调以适应高分辨输入

（3）借鉴faster-rcnn中的anchor思想对物体进行检测，提高定位精度

（4）对anchor的尺寸进行K均值聚类，找出一个较好的anchor值