R-CNN

1. 目标

现存的物体检测的方法可以识别出图像上的所占比例较大的物体，而对于较小物体的识别并不理想，整个论文是利用R-CNN来提高较小物体识别的准确性的（Region proposal + CNN， 这并不是一个算法计算层面的改进，跟我看到题目理解的有些冲突）

1. 问题以及解决方案

在文章中提出了两个问题：

1. Localizing objects with a deep network

传统的： to build a sliding-window detector, then to judge these region by order.

R-CNN: generates around 2000 category-independent region proposals and then extracts a fixed-length feature vector from each proposal region using CNN, and then classifies each region with category-specific linear SVMs.

1. Training a high-capacity model with only a small quantity of annotated detection data.

R-CNN: To use unsupervised pre-training, followed by supervised fine-tuning. （也就是说，先进行一个无监督的训练根据一个较大的知识库，然后在训练结果的基础上进行有监督的调优根据一个较小的知识库，使用的数据集分别是ILSVRC 和 PASCAL）

1. 流程
2. region proposals

(selective search 大概2000个regions) 文中好像并没有提到，基本思路：

1. 使用一种分割手段将图像分割成小区域
2. 查看现有区域合并可能性最高的两个区域。重复直到整张图像合并成一个区域
3. 输出所有曾经存在过的区域，称为候选区。

由于此过程相对独立可以使用任何算法实现。



1. Feature extracts
2. 预处理

先在每个建议框周围加上16个像素值为建议框像素平均值的边框，将数据转换成输入格式的数据，化成同一尺寸 227×227

1. （CNN特征提取）

先将所有建议框像素减去该建议框像素平均值后，再依次将每个227×227的建议框输入CNN网络获取4096维的特征，2000个建议框的CNN特征组合成2000x4096维矩阵；

在训练CNN时，对训练数据标定宽松，ss方法取得的proposal即使只包含部分目标区域，我们也将其标定为特定物体的类别。但这种宽松条件下的cNN只能用来做特征提取

3、特征送入SVM分类器判断是否属于该类

1） 将2000×4096维矩阵与N个SVM的权制组成的二维矩阵相乘（由于SVM是二分类，有N个类别，所以有N个SVM），获得了2000xN，表示的是每个proposal region 是某个类别的得分。

2） 分别对上述2000×N维矩阵中每一列即每一类进行非极大值抑制剔除重叠建议框，得到该列即该类中得分最高的一些建议框；

为什么要进行非极大值抑制？什么叫非极大值抑制？

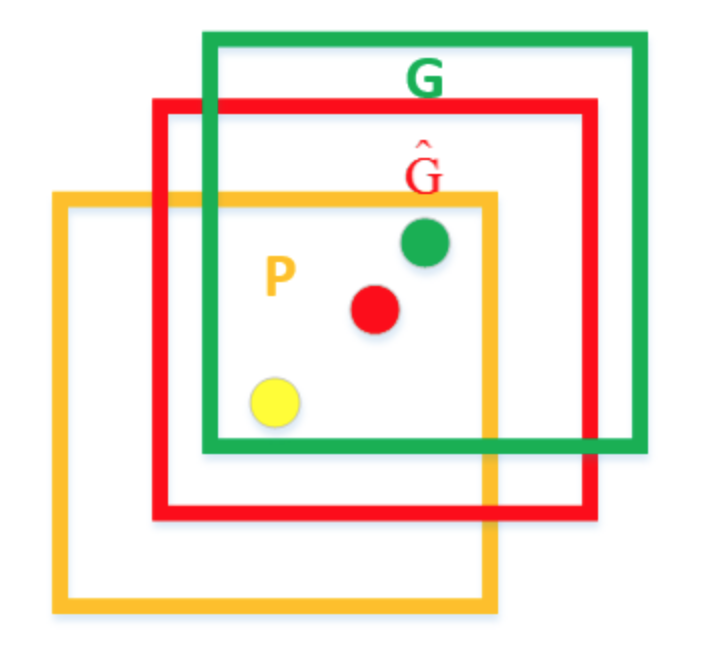
在2000xN 表示每个proposal 是每个物体的得分情况，这个时候，一个物体可能被多个proposal 包围着，这个时候需要去除得分较低的以减少重叠。

1. 每列从大到小排序
2. 从每列最大的得分建议框开始，分别与该列后面的得分建议框进行IoU计算，若IoU>阈值，则剔除得分较小的建议框，否则认为图像中存在多个同一类物体；
3. 从每列次大的得分建议框开始，重复步骤2
4. 重复步骤3直至遍历完所有的proposal

4、使用回归器修正proposal的位置

1）分别用20个回归器对上述20个类别中剩余的建议框进行回归操作，最终得到每个类别的修正后的得分最高的bounding box

采用回归就是为了对proposal的位置进行校正（在他的附录？？？里面提到了具体的实现，我是找的中文的看到）



如上图，黄色为 region proposal， 绿色为实际窗口，红色表示回归后的预测模型，当Region Proposal与实际窗口的IoU>0.6时可以认为是线性变换（？？？？不知道为啥），现在就是要找到P 到 的一个线性变换（注意指的是点点点），这就是一个最小二乘解决线性回归的问题啦。（宝宝不想推公式，把过程看了一遍，标红，以后自己推一遍）

1. 方法的优缺点和运行速度

但是，

1. 正常都可以想到，在提取proposal时由于有大量的重叠，需要进行大量的重复计算，所以计算量大，无法做到实时。而且提取特征时也有大量的冗余操作，这里我想到的是是不是可以保存共享CNN特征
2. 而且由于全连接的原因，输入proposal 的时候需要做resize，这里是不是还会造成图像的畸变

文中给出了

计算Region Proposals和features平均所花时间：13s/image on a GPU；53s/image on a CPU。而在训练大量的图片和大量的冗余和重复的操作时，必定会浪费大量的时间，在运行时间上可以说是很慢了。

Fast R CNN

一、RCNN存在的问题

（先看到了时间对比，先放这！！！！）Fast R-CNN trains the very deep VGG16 network 9× faster than R-CNN, is 213× faster at test-time, and achieves a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains VGG16 3× faster, tests 10× faster, and is more accurate.

在文章开始，竟然告诉我了RCNN的缺点。

1. Training is a multi-stage pipeline. R-CNN first fine- tunes a ConvNet on object proposals using log loss. Then, it fits SVMs to ConvNet features. These SVMs act as object detectors, replacing the softmax classifier learnt by fine-tuning. In the third training stage, bounding-box regressors are learned.

2. Training is expensive in space and time. For SVM and bounding-box regressor training, features are ex- tracted from each object proposal in each image and written to disk. With very deep networks, such as VGG16, this process takes 2.5 GPU-days for the 5k images of the VOC07 trainval set. These features re- quire hundreds of gigabytes of storage.

3. Object detection is slow. At test-time, features are extracted from each object proposal in each test image. Detection with VGG16 takes 47s / image (on a GPU).

总结来说就是，RCNN的测试、训练速度都很慢，操作冗余繁琐，而且占用大量的磁盘空间（这是我没有想到的）

运行时间，给出了详细的每张图片47s

二、SPP-NET

为了采取共享计算的方法，他是先进行了整个图像的卷积特征的提取，另外也省掉了resize的操作，但是在CNN最后一个卷积层与其后的全连接层间加入了一个spp-layer，避免了resize的操作。

Spp-layer 就是为了产生固定大小的feature map， 以匹配全连接层。

缺点：

1. Like R-CNN, training is a multi-stage pipeline that involves extracting features, fine-tuning a network with log loss, training SVMs, and finally fitting bounding-box regressors.
2. Features are also written to disk. But unlike R-CNN, the fine-tuning algorithm proposed in [11] cannot update the convolutional layers that precede the spatial pyramid pooling. Unsurprisingly, this limitation (fixed convolutional layers) limits the accuracy of very deep networks.
3. Fast R CNN 过程

1、任意size图片输入CNN网络，经过若干卷积层与池化层，得到特征图；这样一次提取所有的CNN特征，避免了重复操作

2、在任意size图片上采用selective search算法提取约2k个建议框；

文章中对建议框是否2000更为合适进行了试验。

文中对鱼尺度不变性问题，如何在同一个训练好的网络里识别不同大小的两个物体，提及到了两种处理方式：brute-force（单一尺度）和image pyramids（多尺度）单一尺度直接在训练和测试阶段将image定死为某种scale，直接输入网络训练就好，然后期望网络自己能够学习到scale-invariance的表达；多尺度在训练阶段随机从图像金字塔【缩放图片的scale得到，相当于扩充数据集】中采样训练，测试阶段将图像缩放为金字塔中最为相似的尺寸进行测试；

可以看出，多尺度应该比单一尺度效果好。作者在5.2节对单一尺度和多尺度分别进行了实验，该实验证明了深度神经网络善于直接学习尺度不变形，对目标的scale不敏感。

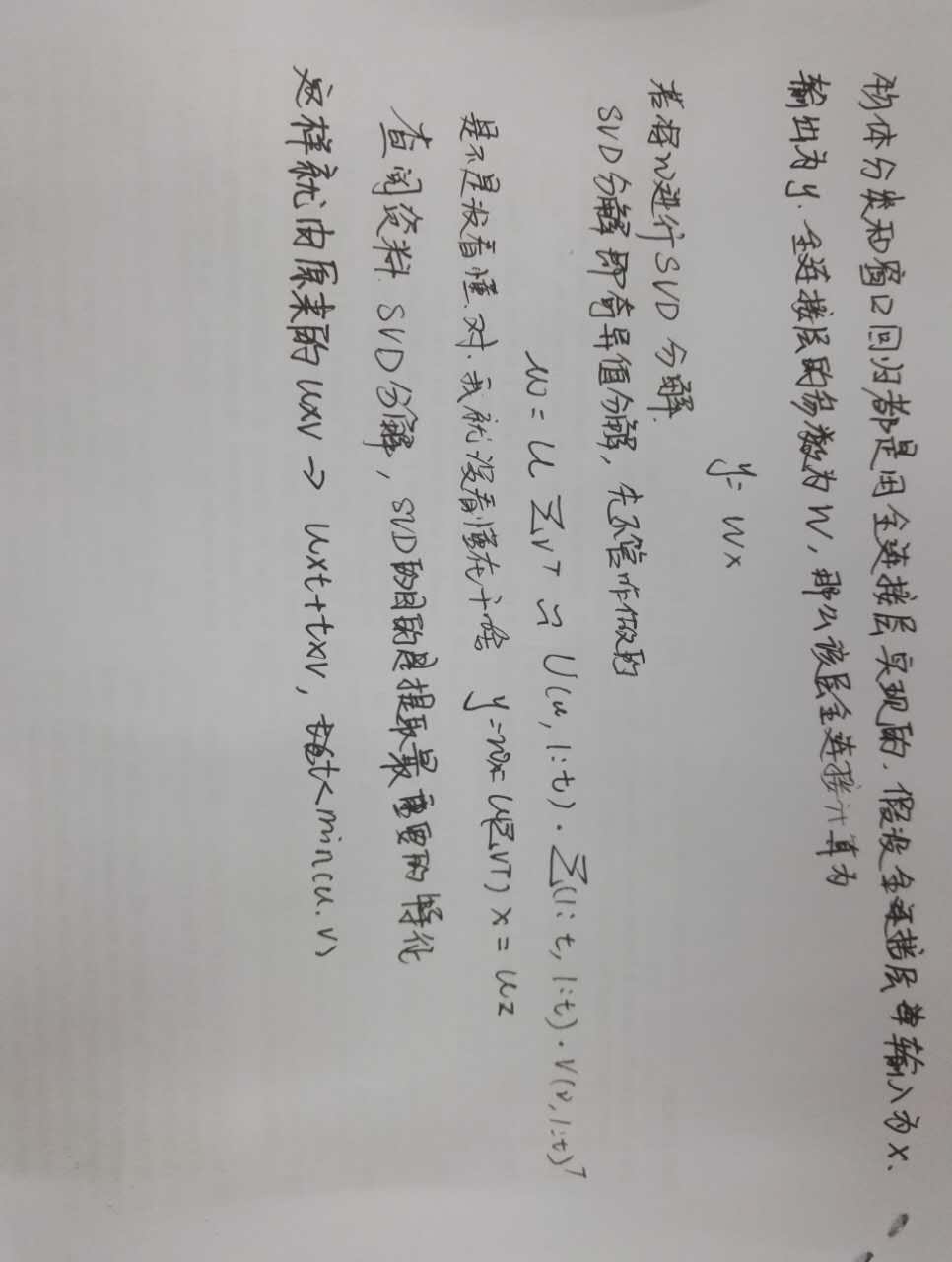
3、根据原图中建议框到特征图映射关系，在特征图中找到每个建议框对应的特征框【深度和特征图一致】，并在RoI池化层中将每个特征框池化到H×W【VGG-16网络是7×7】的size；

ROI（region of interest） 池化层这里没有看懂，只知道他对输入做了一个标准化的转换，具体细节模糊。可能是因为对CNN没有理解吧。但是在文中提到了假设建议框对应特征图中的特征框大小为h×w，将其划分H×W个子窗口，每个子窗口大小为h/H×w/W， then max-pooling the values in each sub-window into the corresponding output grid cell ？？？？？这是啥意思？ 我找了一些文献（然后对每个子窗口采用max pooling下采样操作）还是不知道在干嘛， 反正是将大小不一的特征框转化成了大小统一的数据输入下一层。

4、固定H×W【VGG-16网络是7×7】大小的特征框经过全连接层得到固定大小的特征向量；

5、第4步所得特征向量经由各自的全连接层【由SVD分解实现】，分别得到两个输出向量：一个是softmax的分类得分，一个是Bounding-box窗口回归；

在目标检测的任务中，selective search 提取的约2000的proposals，RoI池化层后的全连接层需要进行约2k次的全连接的计算，SVD就是用来分解加速全连接层计算，怎么做的呢？（SVD需要继续深入理解）



为什么不用SVM呢？文中给了试验对比，这样的话也将所有的特征存储在现存中不用占用磁盘空间了。

6、对每一类进行非极大值抑制剔除重叠建议框，得到该列即该类中得分最高的一些建议框

四、方法的优缺点和运行速度

运行速度：

文中给出了运行时间的对比：

1. PASCAL VOC 2007训练集上，使用VGG-16【L for large】网络Fast R-CNN训练时间为9.5h，同等条件下R-CNN需要84h，快8.8倍；
2. PASCAL VOC 2007测试集上，使用VGG-16【L for large】网络不采用SVD Fast R-CNN测试时间为0.32s/image【不包括候选区域提取时间】，同等条件下R-CNN需要47.0s/image，快146倍；采用SVD测试时间为0.22s/image【不包括候选区域提取时间】，快213倍；
3. PASCAL VOC 2007测试集上，使用VGG-16【L for large】网络不采用SVD Fast R-CNN mAP为66.9%，同等条件下R-CNN mAP为66.0%；Fast R-CNN采用SVD mAP为66.6%。

Fast R-CNN还消除了数百GB的磁盘存储，因为它不缓存特征。

缺点：

1. 显然速度是提高了很多，但是selective search算法，会浪费大部分的时间的，虽然节约了特征提取的时间。

怎样解决这个proposal的提取，我猜faster R CNN 应该会解决这个问题了吧。

总结：这篇文章读起来有点难理解了，主要过了一遍算法的流程，里面的思想并没有深入理解，主要是一些算法公式不熟悉，而且在CNN各层理解的不够透彻，打算抽时间推推公式还有CNN，然后再回来读一次。

Faster R CNN

终于到第三篇了，有点高估自己读论文的速度了……

看到了RPN，果然用CNN 再去解决这个问题，CNN大法好（果断要过一遍CNN）

1. 问题

问题很简单了，就是region proposal的获取时间

解决办法就是构造两个额外的卷积层来实现RPN（不要问我咋实现，我也很纳闷，我要去仔细看CNN了）

第一个层把每个卷积映射位置编码为一个短的（例如256-d）特征向量

第二个层在每个卷积映射位置，输出这个位置上多种尺度和长宽比的k个区域建议的objectness得分和回归边界（k=9是典型值）

1. 过程
2. 首先向CNN网络输入任意大小的图片
3. 经过CNN网络前向传播至最后共享的卷积层，一方面得到供RPN网络输入的特征图，另一方面继续前向传播至特有卷积层，产生更高维特征图
4. 使用供RPN网络输入的特征图，经过RPN网络得到proposal和区域到得分并对区域得分采用非极大值抑制【阈值为0.7】，输出其Top-N【文中为300】得分的区域建议给RoI池化层
5. 第2步得到的高维特征图和第3步输出的区域建议同时输入RoI池化层，提取对应区域建议的特征；
6. 第4步得到的区域建议特征通过全连接层后，输出该区域的分类得分以及回归后的bounding-box

RPN网络？？？

只画RPN网络部分，滑动窗口实在卷积层特征图上进行的，维度较原始图像降低了很多倍

1. 首先套用ImageNet上常用的图像分类网络，文中试验了ZF和VGG-16，在最后一个共享的卷积层输出的卷积特征
2. 对于1）中特征图，用n×n【论文中设计为3×3，n=3看起来很小，但是要考虑到这是非常高层的feature map，其size本身也没有多大，因此9个矩形中，每个矩形窗框都是可以感知到很大范围的】的滑动窗口在特征图上滑动扫描【代替了从原始图滑窗获取特征】，每个滑窗位置通过卷积层1映射到一个低维的特征向量【ZF网络：256维；VGG-16网络：512维，低维是相对于特征图大小W×H，typically~60×40=2400】后采用ReLU，并为每个滑窗位置考虑k种【论文中k=9】可能的参考窗口【论文中称为anchors，见下解释】，这就意味着每个滑窗位置会同时预测最多9个区域建议【超出边界的不考虑】，对于一个W×H的特征图，就会产生W×H×k个区域建议；
3. 步骤2）中的低维特征向量输入两个并行连接的卷积层2：reg窗口回归层【位置精修】和cls窗口分类层，分别用于回归区域建议产生bounding-box【超出图像边界的裁剪到图像边缘位置】和对区域建议是否为前景或背景打分，这里由于每个滑窗位置产生k个区域建议，所以reg层有4k个输出来编码【平移缩放参数】k个区域建议的坐标，cls层有2k个得分估计k个区域建议为前景或者背景的概率

Anchors？？？

表示RPN网络中对特征图滑窗时每个滑窗位置所对应的原图区域中9种可能的大小，相当于模板，对任意图像任意滑窗位置都是这9中模板

作者在文中表示采用Anchors这种方法具有平移不变性，就是说在图像中平移了物体，窗口建议也会跟着平移。同时这种方式也减少了整个模型的size，输出层512×(4+2)×9=2.8×10^4个参数【512是前一层特征维度，(4+2)×9是9个Anchors的前景背景得分和平移缩放参数】，而MultiBox有1536×（4+1）×800=6.1×10^6个参数，而较小的参数可以在小数据集上减少过拟合风险。

学习区域建议的损失函数？？？？

这一堆损失函数吧，也真难为人了，不写了，假装他定义的好

三种共享网络的训练方式？？？？

1. 交替训练
2. 近似联合训练
3. 联合训练

文中提到了一种实用的4步训练算法，通过交替优化来学习共享的特征

**第一步**，依上述训练RPN，该网络用ImageNet预训练的模型初始化，并端到端微调用于区域建议任务。

**第二步**，利用第一步的RPN生成的建议框，由Fast R-CNN训练一个单独的检测网络，这个检测网络同样是由ImageNet预训练的模型初始化的，这时候两个网络还没有共享卷积层。

**第三步**，用检测网络初始化RPN训练，但我们固定共享的卷积层，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**，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。

三、方法的优缺点和运行速度

文中通过对比试验，表明，运行速度很快，效率很好。

区域建议的步骤几乎是无损耗的。使一个一致的，基于深度学习的目标检测系统以5-17 fps的速度运行。学到的RPN也改善了区域建议的质量，进而改善整个目标检测的准确性。

我不能明确的想到此时优缺点和影响运行速度的因素了，但是吧，觉得吧，肯定速度还会优化的

|  |  |  |  |
| --- | --- | --- | --- |
|  | 使用方法 | 缺点 | 改进 |
| R-CNN | 1. SS提取region proposals 2. CNN提取特征 3. SVM进行分类 4. BB进行回归 | 1. 训练步骤繁琐 2. 运行时间慢 3. 占用大量空间 4. 我认为resize也会又问题的 | 1、最后一个卷积层后加了一个ROI pooling layer；  2、损失函数使用了multi-task loss（多任务损失）函数，将边框回归直接加到CNN网络中训练。分类Fast R-CNN直接用softmax替代R-CNN用的SVM进行分类。 |
| Fast-R-CNN | 1. SS提取region proposals 2. CNN提取特征 3. Softmax进行分类 4. 多任务损失函数进行边框回归 | 1、SS耗费大量的时间 | RPN进行proposal的提取 |
| Faster-R-CNN | 1、RPN提取region proposals   1. CNN提取特征 2. Softmax进行分类 3. 多任务损失函数进行边框回归 | 获取region proposal，再对每个proposal分类计算量还是比较大（这个我是百度了，不太清楚） |  |

ps：读文章的过程中，百度看了许多中文的，透彻理解起来确实有些费劲费劲费劲！！！！只能尽力的提炼出自己的理解和流程。我是认为，如果能自己实现一遍可能会很好的理解，但是实现也是一个难题，我先不做了，待完成事项。对于老师说的重点评估方法的优缺点和运行速度，评估起来也就是在RCNN的缺点改进过程中体现出来的速度的提升，已在下面简单写了自己的理解想法，如果需要详细的设计什么实验的话，希望听从老师的建议。