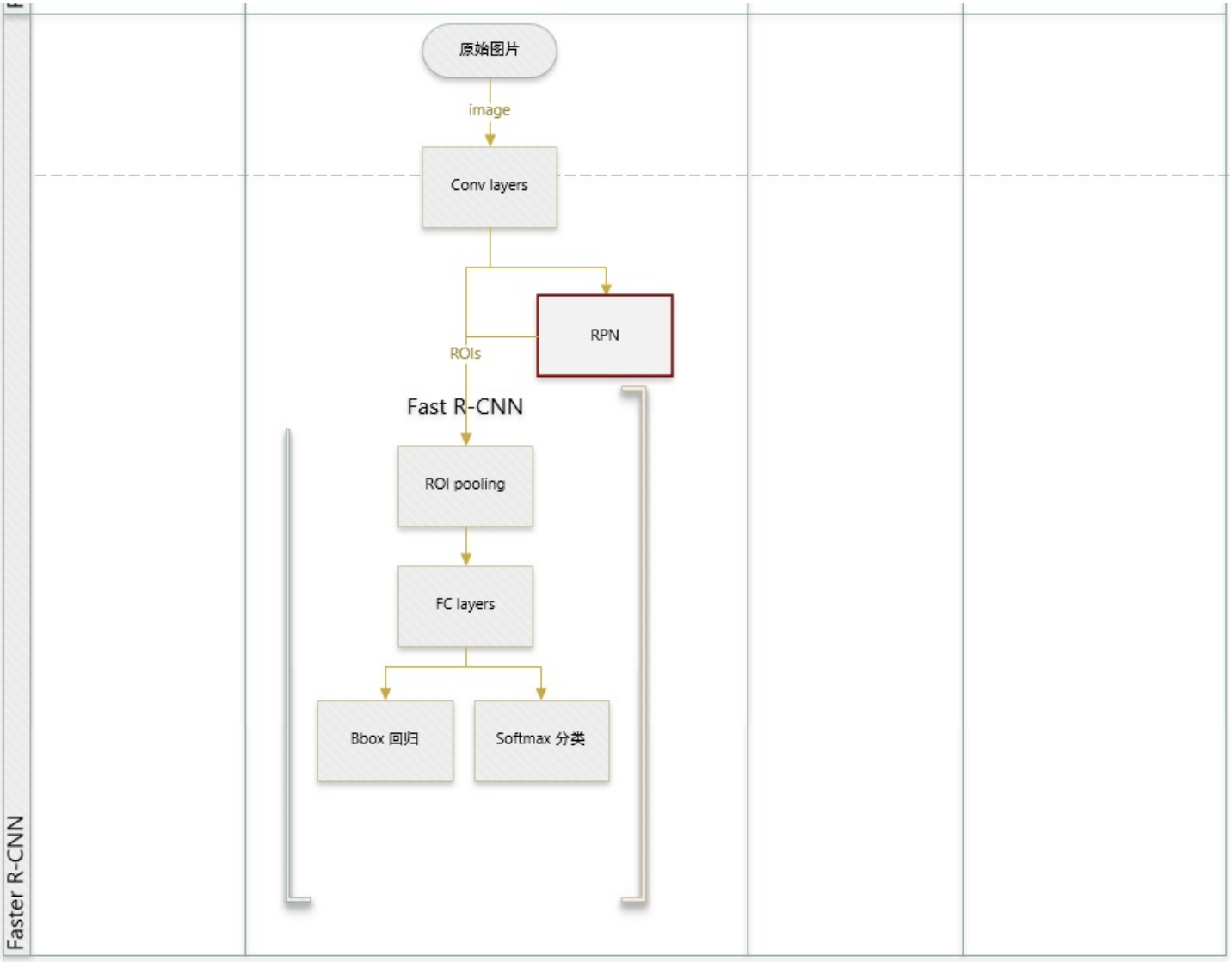
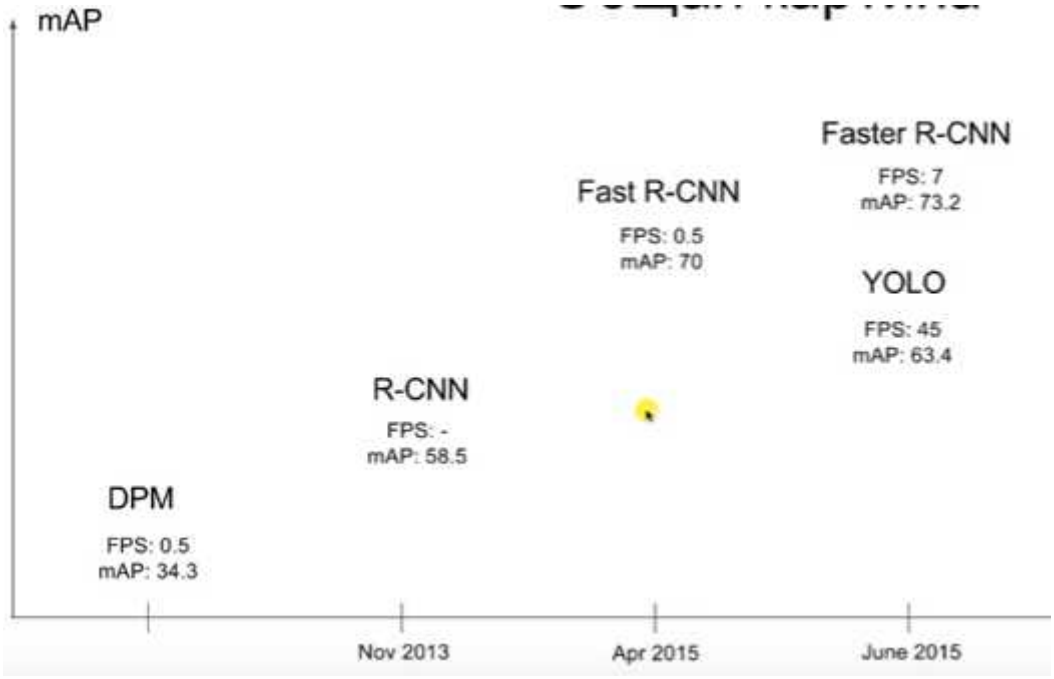


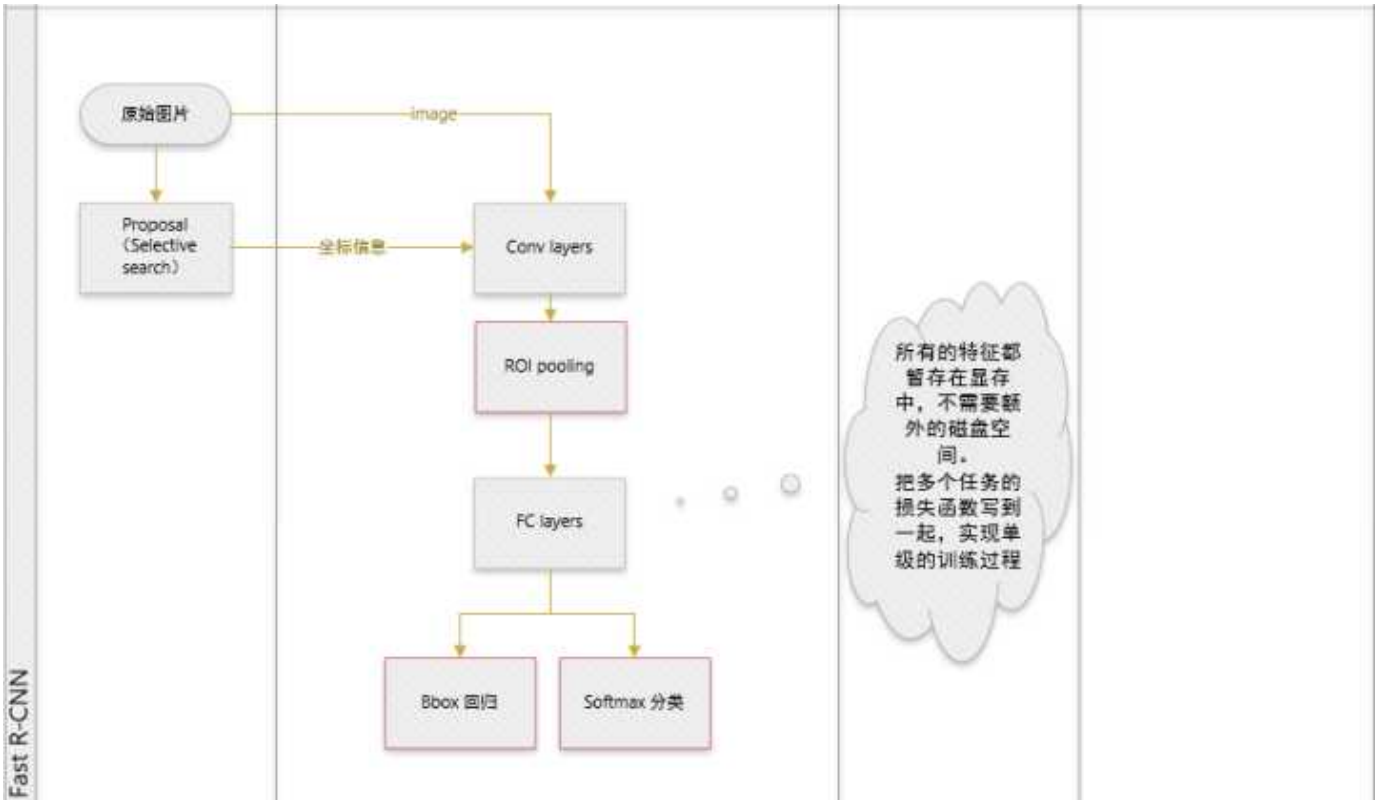
Faster R-CNN

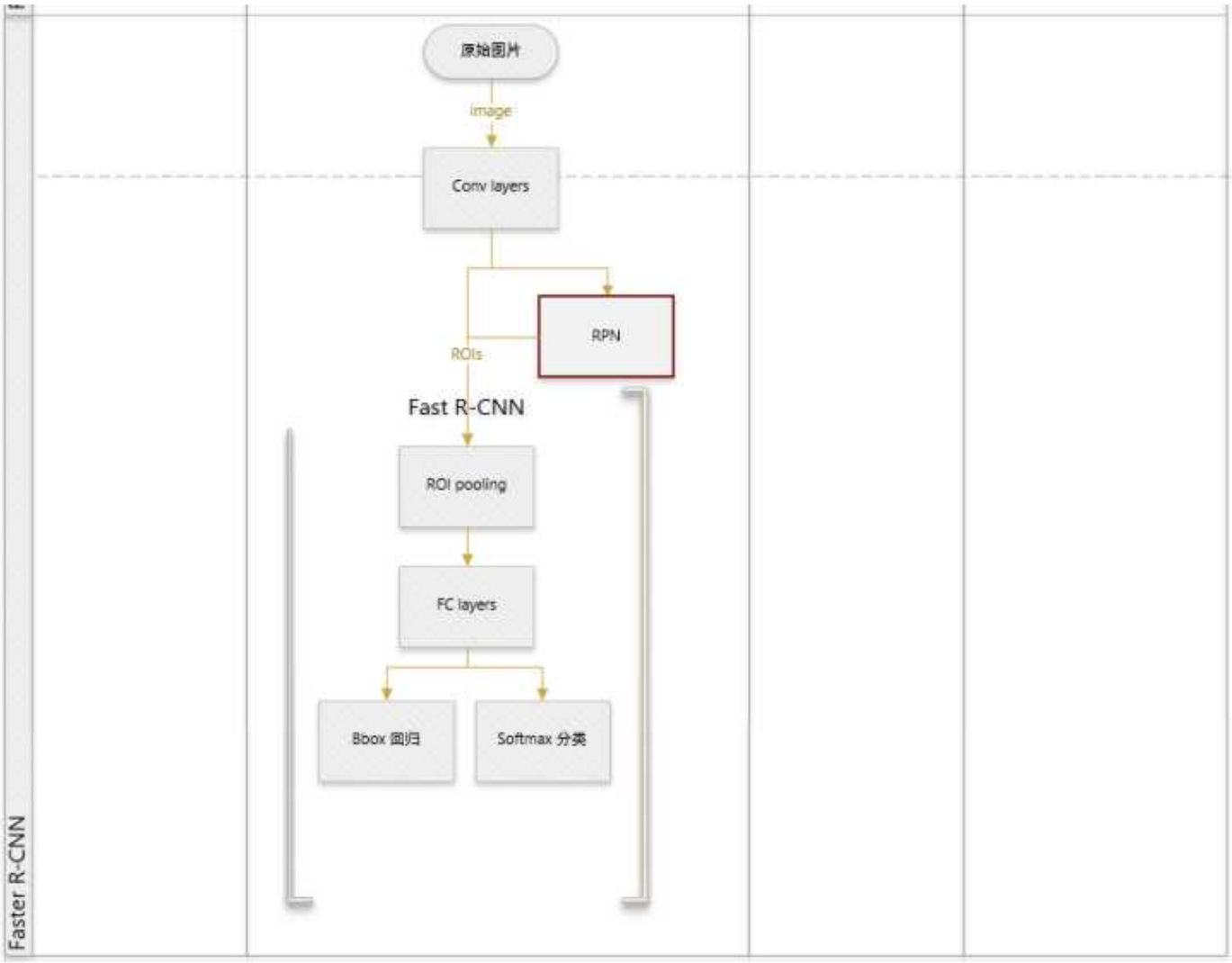
晓雷 · 1 年前

Fast-RCNN基本实现端对端（除了proposal阶段外），下一步自然就是要把proposal阶段也用CNN实现（放到GPU上）。这就出现了Faster-RCNN，一个完全end-to-end的CNN对象检测模型。

论文提出：网络中的各个卷积层特征（feature map）也可以用来预测类别相关的region proposal（不需要事先执行诸如selective search之类的算法），但是如果简单的在前面增加一个专门提取proposal的网络又显得不够优雅，所以最终把region proposal提取和Fast-RCNN部分融合进了一个网络模型 (区域生成网络 RPN层)，虽然训练阶段仍然要分多步，但是检测阶段非常方便快捷，准确率也与原来的Fast-RCNN相差不多，从此，再也不用担心region proposal提取耗时比实际对象检测还多这种尴尬场景了。(faster RCNN可以大致看做“区域生成网络+fast RCNN“的系统，用区域生成网络代替fast RCNN中的Selective Search方法)


mAP
Faster R-CNN
FPS: 7
mAP: 73.2
Fast R-CNN
FPS: 0.5
mAP: 70
YOLO
FPS: 45
mAP: 63.4
R-CNN
FPS: -
mAP: 58.5
DPM
FPS: 0.5
mAP: 34.3
Nov 2013
Apr 2015
June 2015


原始图片
image
Proposal (Selective search)
坐标信息
Conv layers
ROI pooling
FC layers
Bbox 回归
Softmax 分类
所有的特征都暂存在显存中，不需要额外的磁盘空间。
把多个任务的损失函数写到一起，实现单级的训练过程
Fast R-CNN



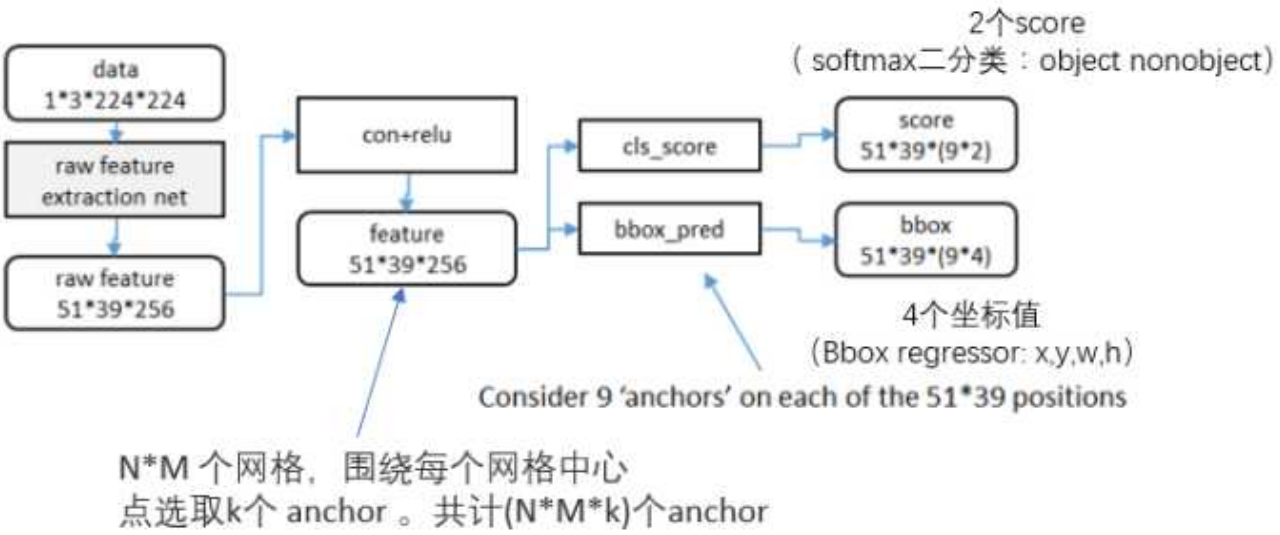
区域生成网络 （ Region Proposal Networks ）

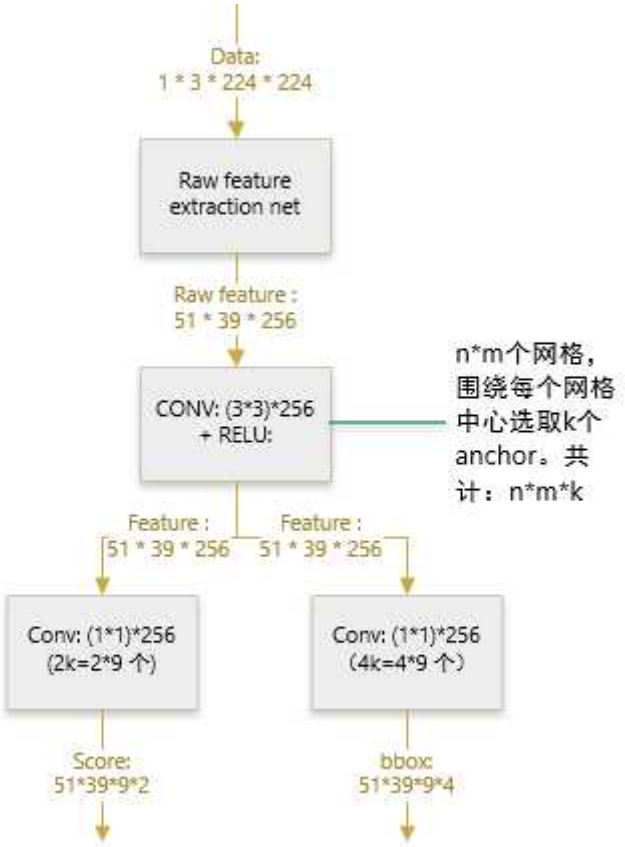
分析:

如何训练出一个网络来替代selective search相类似的功能呢？论文借鉴SPP和ROI中的思想 在feature map中提取proposal。 先通过对应关系把feature map的点映射回原图（参看：原始图片中的ROI如何映射到到feature map?），在每一个对应的原图设计不同的固定尺度窗口（bbox），根据该窗口与ground truth的IOU给它正负标签，让它学习里面是否有object，这样就训练一个网络（Region Proposal Network）。

由于我们只需要找出大致的地方，无论是精确定位位置还是尺寸，后面的工作都可以完成，作者对bbox做了三个固定：固定尺度变化（三种尺度），固定scale ratio变化（三种ratio），固定采样方式（只在feature map的每个点在原图中的对应ROI上采样，反正后面的工作能进行调整）。如此就可以降低任务复杂度。

可以在特征图上提取proposal之后，网络前面就可以共享卷积计算结果（SPP减少计算量的思想）。这个网络的结果就是卷积层的每个点都有有关于k个achor boxes的输出，包括是不是物体，调整box相应的位置。

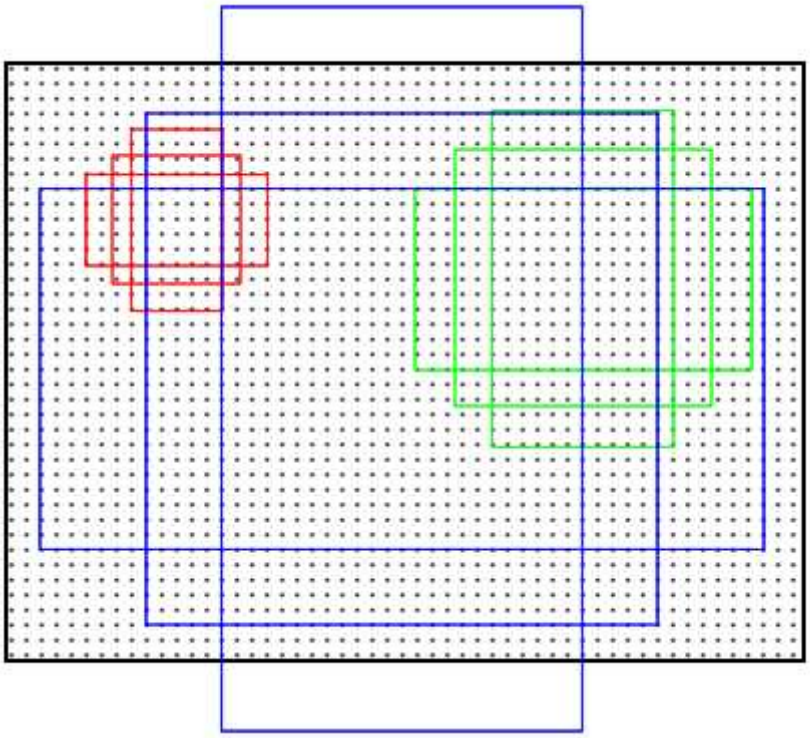


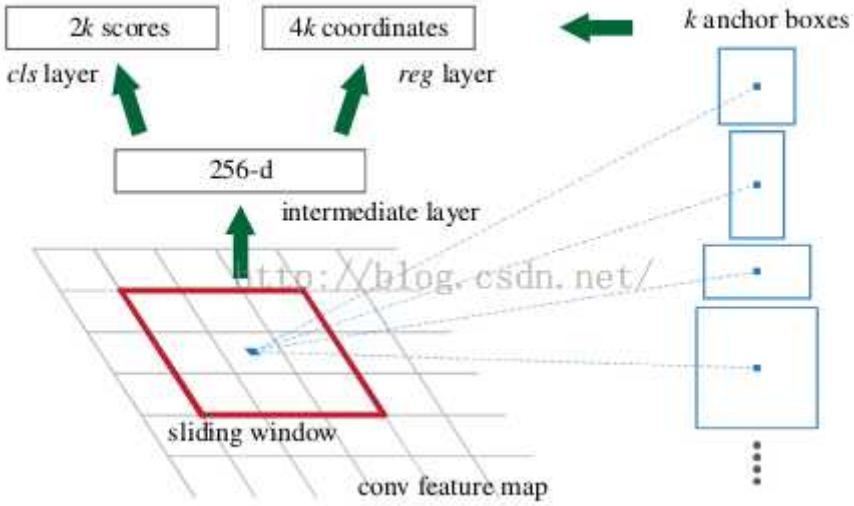


具体过程：

- 得到最终用来预测的feature map：图片在输入网络后，依次经过一系列conv+relu （套用ImageNet上常见的分类网络即可 本论文实验了5层的ZF,16层的VGG-16）得到的feature map，额外添加一个conv+relu层，输出51*39*256维特征（feature map）。准备后续用来选取proposal，并且此时坐标依然可以映射回原图。

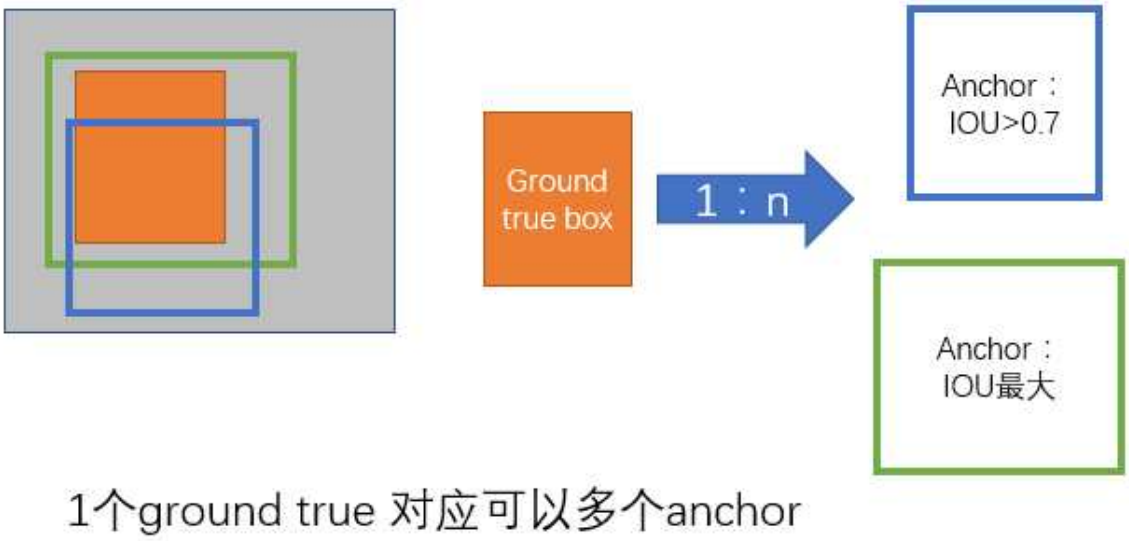
- 计算Anchors： 在feature map上的每个特征点预测多个region proposals。具体作法是：把每个特征点映射回映射回原图的感受野的中心点当成一个基准点，然后围绕这个基准点选取k个不同scale、aspect ratio的anchor。论文中3个scale（三种面积 $\{128^2, 256^2, 521^2\}$），3个aspect ratio({1:1,1:2,2:1})





> - 关于正负样本的划分： 考察训练集中的每张图像（含有人工标定的ground true box）的所有anchor（N*M*k）
>
> - a. 对每个标定的ground true box区域，与其重叠比例最大的anchor记为 正样本 (保证每个ground true 至少对应一个正样本anchor)

- b. 对a)剩余的anchor，如果其与某个标定区域重叠比例大于0.7，记为正样本（每个ground true box可能会对应多个正样本anchor。但每个正样本anchor 只可能对应一个grand true box）；如果其与任意一个标定的重叠比例都小于0.3，记为负样本。



1个ground true 对应可以多个anchor

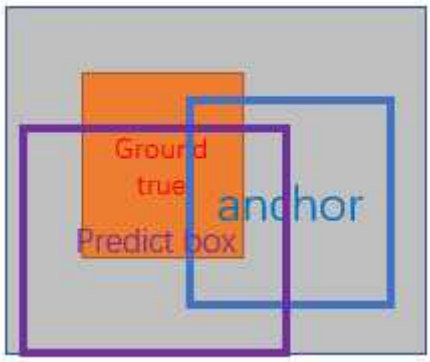
- c. 对a),b)剩余的anchor，弃去不用。
- d. 跨越图像边界的anchor弃去不用

- 定义损失函数：对于每个anchor，首先在后面接上一个二分类softmax，有2个score 输出用以表示其是一个物体的概率与不是一个物体的概率（p_i），然后再接上一个bounding box的regressor 输出代表这个anchor的4个坐标位置（t_i），因此RPN的总体Loss函数可以定义为：

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

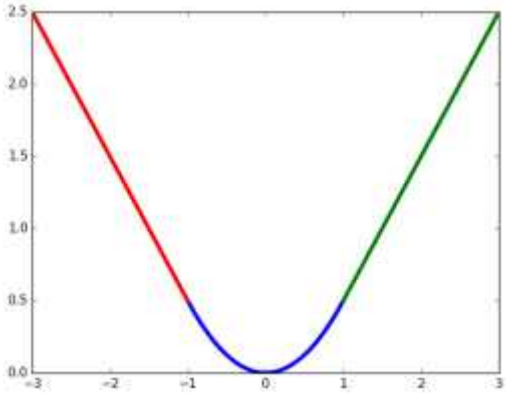
  - i表示第i个anchor，当anchor是正样本时 p_i^* =1，是负样本则=0。 t_i^* 表示 一个与正样本anchor 相关的ground true box 坐标 （每个正样本anchor 只可能对应一个ground true box： 一个正样本anchor 与某个grand true box对应，那么该anchor与ground true box 的IOU要么是所有anchor中最大，要么大于0.7）
  - x,y,w,h分别表示box的中心坐标和宽高，x, x_a, x^* 分别表示 predicted box, anchor box, and ground truth box （y,w,h同理）t_i 表示predict box相对于anchor box的偏移，t_i^* 表示ground true box相对于anchor box的偏移，学习目标自然就是让前者接近后者的值。

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



- 其中 L_{reg} 是:

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



- p_i^* 表示这些regressor的loss指针对正样本而言，因为负样本时 p_i^* =0该项被消去。
- L_{cls} 是关于两种类别 (object vs. not object) 的log loss

训练：正负样本的选择，文中提到如果对每幅图的所有anchor都去优化loss function，那么最终会因为负样本过多导致最终得到的模型对正样本预测准确率很低（It is possible to optimize for the loss functions of all anchors, but this will bias towards negative samples as they are dominate）。

- 训练RPN：文中提到如果每幅图的所有anchor都去参与优化loss function，那么最终会因为负样本过多导致最终得到的模型对正样本预测准确率很低。因此 在每幅图像中随机采样256个anchors去参与计算一次mini-batch的损失。正负比例1:1(如果正样本少于128则补充采样负样本)

> We randomly initialize all new layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. All other layers (*i.e.*, the shared convolutional layers) are initialized by pre-training a model for ImageNet classification [36], as is standard practice [5]. We tune all layers of the ZF net, and conv3_1 and up for the VGG net to conserve memory [2]. We use a learning rate of 0.001 for 60k mini-batches, and 0.0001 for the next 20k mini-batches on the PASCAL VOC dataset. We use a momentum of 0.9 and a weight decay of 0.0005 [37]. Our implementation uses Caffe [38].

注意点：

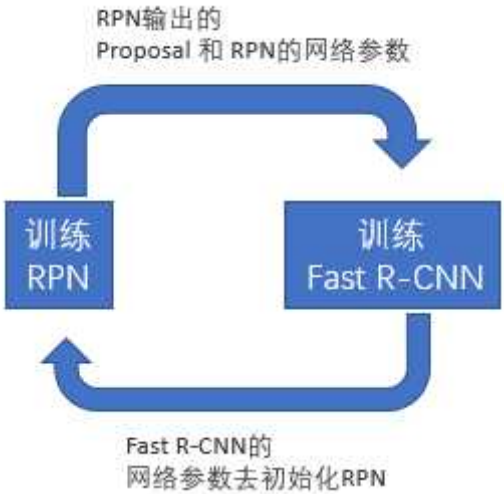
- 在到达全连接层之前，卷积层和Pooling层对图片输入大小其实没有size的限制，因此RCNN系列的网络模型其实是不需要实现把图片resize到固定大小的；
- n=3看起来很小，但是要考虑到这是非常高层的feature map，其size本身也没有多大，因此 3×33×3 9个矩形中，每个矩形窗框都是可以感知到很大范围的。

Sharing Features for RPN and Fast R-CNN

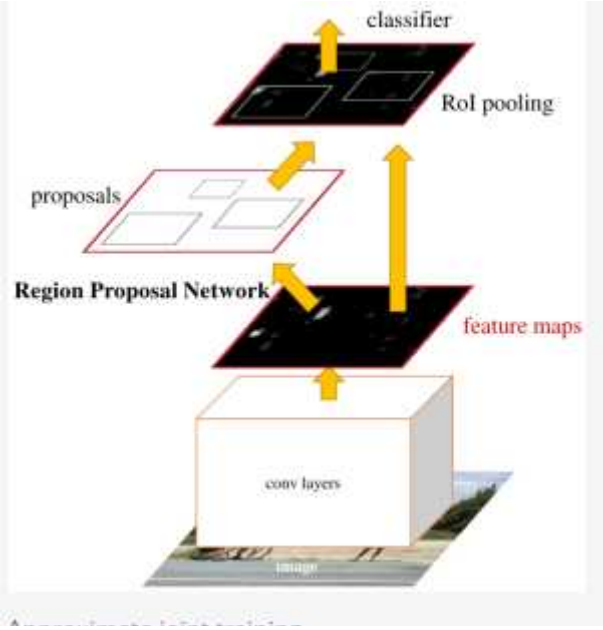
前面已经讨论如何训练提取proposal的RPN，分类采用Fast R-CNN。如何把这两者放在同一个网络结构中训练出一个共享卷积的Multi-task网络模型。

我们知道，如果是分别训练两种不同任务的网络模型，即使它们的结构、参数完全一致，但各自的卷积层内的卷积核也会向着不同的方向改变，导致无法共享网络权重，论文作者提出了三种可能的方式：

1. Alternating training：此方法其实就是一个不断迭代的训练过程，既然分别训练RPN和Fast-RCNN可能让网络朝不同的方向收敛，a)那么我们可以先独立训练RPN，然后用这个RPN的网络权重对Fast-RCNN网络进行初始化并且用之前RPN输出proposal作为此时Fast-RCNN的输入训练Fast R-CNN。b) 用Fast R-CNN的网络参数去初始化RPN。之后不断迭代这个过程，即循环训练RPN、Fast-RCNN。



2. Approximate joint training：这里与前一种方法不同，不再是串行训练RPN和Fast-RCNN，而是尝试把二者融入到一个网络内，具体融合的网络结构如下图所示，可以看到，proposals是由中间的RPN层输出的，而不是从网络外部得到。需要注意的一点，名字中的"approximate"是因为反向传播阶段RPN产生的cls score能够获得梯度用以更新参数，但是proposal的坐标预测则直接把梯度舍弃了，这个设置可以使backward时该网络层能得到一个解析解（closed results），并且相对于Alternating traing减少了25-50%的训练时间。(此处不太理解： 每次mini-batch的RPN输出的proposal box坐标信息固定，让Fast R-CNN的regressor去修正位置？）

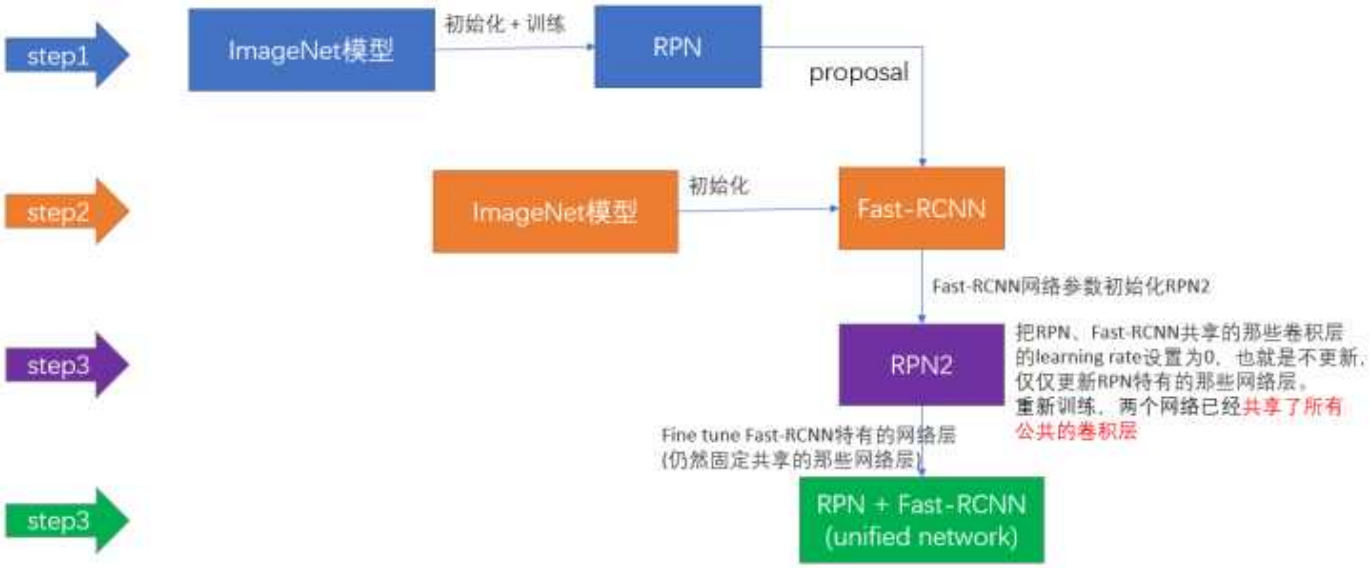


Approximate joint training

3. Non-approximate training：上面的Approximate joint training把proposal的坐标预测梯度直接舍弃，所以被称作approximate，那么理论上如果不舍弃是不是能更好的提升RPN部分网络的性能呢？作者把这种训练方式称为“ Non-approximate joint training”，但是此方法在paper中只是一笔带过，表示“This is a nontrivial problem and a solution can be given by an “RoI warping” layer as developed in [15], which is beyond the scope of this paper”，

上面说完了三种可能的训练方法，可非常神奇的是作者发布的源代码里却用了另外一种叫做4-Step Alternating Training的方法，思路和迭代的Alternating training有点类似，但是细节有点差别：

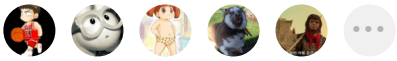
1. 第一步：用ImageNet模型初始化，独立训练一个RPN网络；
2. 第二步：仍然用ImageNet模型初始化，但是使用上一步RPN网络产生的proposal作为输入，训练一个Fast-RCNN网络，至此，两个网络每一层的参数完全不共享；
3. 第三步：使用第二步的Fast-RCNN网络参数初始化一个新的RPN网络，但是把RPN、Fast-RCNN共享的那些卷积层的learning rate设置为0，也就是不更新，仅仅更新RPN特有的那些网络层，重新训练，此时，两个网络已经共享了所有公共的卷积层；
4. 第四步：仍然固定共享的那些网络层，把Fast-RCNN特有的网络层也加入进来，形成一个unified network，继续训练，fine tune Fast-RCNN特有的网络层，此时，该网络已经实现我们设想的目标，即网络内部预测proposal并实现检测的功能。



参考：

- RCNN, Fast-RCNN, Faster-RCNN的一些事
- 目标检测--从RCNN到Faster RCNN 串烧
- Focusing on your own Mind :) - > 站在巨人的肩膀上

神经网络　深度学习（Deep Learning）　目标检测



陈西沐

你的学习顺序跟我一样！0v0

1 年前

陈仲

输入图片的尺寸不对吧?

1 年前

晓雷（作者） 回复 **陈仲** 查看对话

的确不对，图是在原作者画的一张图上修补一下，足够示意了。

1 年前

萌面女xia有点萌

很透彻的阐述啊！【手动点赞】

10 个月前

张洪星

不能再赞了

10 个月前

器鸢

楼主，想请教下： 求取的特征 51*39*256中的 256 的含义是什么?

10 个月前

晓雷（作者） 回复 **器鸢** 查看对话

256是深度(channel)

10 个月前

azxs4869 回复 **陈仲** 查看对话

请问你们是在说51*39不对吗，看了很久不知道这个51*39是怎么回事

10 个月前

器鸢 回复 **晓雷**（作者） 查看对话

好的，谢谢。我今天又找了一下，这篇博客——

faster-rcnn 之 RPN网络的结构解析

——也有解释上述问题。

10 个月前

Oh233

4 step training是为了赶nips deadline

10 个月前 1 赞

下一页

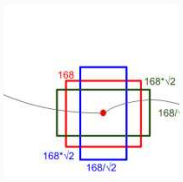
文章被以下专栏收录

晓雷机器学习笔记

记录自己如何从零基础到掌握机器学习关键算法

进入专栏

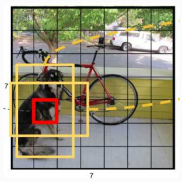
推荐阅读



SSD

参考： SSD: Single Shot MultiBox Detectordeepsystems.io背景介绍： 基于“Proposal + C... 查看全文 >

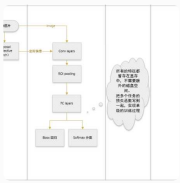
晓雷 · 1 年前 · 发表于 晓雷机器学习笔记



图解YOLO

YOLO核心思想：从R-CNN到Fast R-CNN一直采用的思路是proposal+分类 （proposal 提供位置信息... 查看全文 >

晓雷 · 1 年前 · 发表于 晓雷机器学习笔记



Fast R-CNN

首先声明：本文很多内容来自两个博客： RCNN, Fast-RCNN, Faster-RCNN的一些事目标检测--从... 查看全文 >

晓雷 · 1 年前 · 发表于 晓雷机器学习笔记

Faster R-CNN

Faster RCNN github Faster RCNN paper经过RCNN和Fast RCNN的积淀，Ross B. Girshick在2016年提出了新的Faster RCNN，在结构上，Faster RCN已经将特征抽取(feature extraction)，proposal提... 查看全文 >

张老板进工地了 · 13 天前