每次请回顾：

Faster RCNN 学习笔记

faster rcnn源码解析

Faster RCNN：RPN，anchor，sliding windows

1、这里以faster rcnn举例。在faster rcnn里面，anchor（或者说RPN网络）的作用是代替以往rcnn使用的selective search的方法寻找图片里面可能存在物体的区域。当一张图片输入resnet或者vgg，在最后一层的feature map上面，寻找可能出现物体的位置，这时候分别以这张feature map的每一个点为中心，在原图上画出9个尺寸不一anchor。然后计算anchor与GT（ground truth） box的iou（重叠率），满足一定iou条件的anchor，便认为是这个anchor包含了某个物体。

目标检测的思想是，首先在图片中寻找**“可能存在物体的位置（regions）”**，然后再判断**“这个位置里面的物体是什么东西”**，所以region proposal就参与了判断物体可能存在位置的过程。

region proposal是让模型学会去看哪里有物体，GT box就是给它进行参考，告诉它是不是看错了，该往哪些地方看才对。

2、Region Proposal有什么作用？

（1）、COCO数据集上总共只有80类物体，如果不进行Region Proposal，即网络最后的classification是对所有anchor框定的Region进行识别分类，**会严重拖累网络的分类性能，难以收敛**。原因在于，存在过多的不包含任何有用的类别（80类之外的，例如各种各样的天空、草地、水泥墙、玻璃反射等等）的Region输入分类网络，而这些无用的Region占了所有Region的很大比例。换句话说，这

些Region数量庞大，却并不能为softmax分类器带来有用的性能提升（因为无论怎么预测，其类别都是背景，对于主体的80类没有贡献）。

（2）、大量无用的Region都需要单独进入分类网络，而分类网络由几层卷积层和最后一层全连接层组成，**参数众多，十分耗费计算时间**，Faster R-CNN本来就不能做到实时，这下更慢了。

链接：https://www.zhihu.com/question/265345106/answer/294410307

链接：https://www.zhihu.com/question/265345106/answer/292998341

关于感受野和映射的一些问题：感受野的计算

https://blog.csdn.net/u010725283/article/details/78593410

https://zhuanlan.zhihu.com/p/24780433

【先说说感受野的计算】

卷积神经网络(CNN)简介　里就曾经画图推导过的一个公式

隐藏层边长（输出的边长） = （W - K + 2P）/S + 1

（其中 W是输入特征的大小，K是卷积核大小，P是填充大小，S是步长（stride））

为了理解方便我就把公式先用英文写一下：

output field size = (input field size - kernel size + 2*padding) / stride + 1

(output field size 是卷积层的输出，input field size 是卷积层的输入)

反过来问你： 卷积层的输入（也即前一层的感受野） = ？

> 答案必然是： input field size = （output field size - 1）* stride - 2*padding + kernel size

再重申一下：卷积神经网络CNN中，某一层输出结果中一个元素所对应的输入层的区域大小，被称作感受野receptive field。感受野的大小是由kernel size，stride，padding ， outputsize 一起决定的。

从Concepts and Tricks In CNN(长期更新) 里截张图你感受一下：

公式化一下：

- 对于Convolution/Pooling layer）：
- 对于 Neuronlayer(ReLU/Sigmoid/..) : (显然如此)

上面只是给出了 前一层在后一层的感受野，如何计算最后一层在原始图片上的感受野呢？ 从后向前级联一下就可以了（先计算最后一层到倒数第二层的感受野，再计算倒数第二层到倒数第三层的感受野，依次从后往前推导就可以了）

【再谈谈感受野上面的坐标映射 （Coordinate Mapping）】

> 通常，我们需要知道网络里面任意两个feature map之间的坐标映射关系（一般是中心点之间的映射），如下图，我们想得到map 3上的点p3映射回map 2所在的位置p2（橙色框的中心点）

计算公式：

- 对于 Convolution/Pooling layer:
- 对于Neuronlayer(ReLU/Sigmoid/..) :

上面是计算任意一个layer输入输出的坐标映射关系，如果是计算任意feature map之间的关系，只需要用简单的组合就可以得到，下图是一个简单的例子：

言归正传！！！

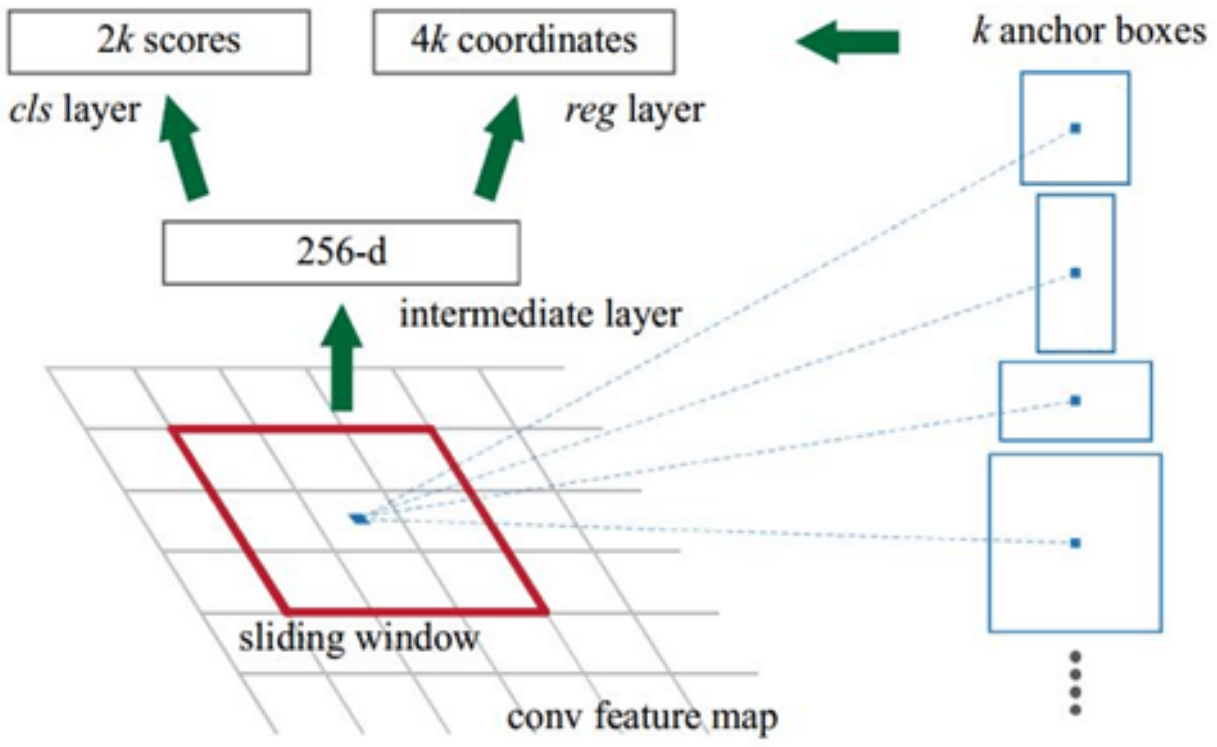
RPN的实现方式：在conv5-3的卷积feature map上用一个n*n的滑窗（论文中作者选用了n=3，即3*3的滑窗）生成一个长度为256（对应于ZF网络）或512（对应于VGG网络）维长度的全连接特征.然后在这个256维或512维的特征后产生两个分支的全连接层：

(1)reg-layer（回归层）：用于预测proposal的中心锚点对应的proposal的坐标x，y和宽高w，h；

(2)cls-layer（分类层）：用于判定该proposal是前景还是背景。sliding window的处理方式保证reg-layer和cls-layer关联了conv5-3的全部特征空间。事实上，作者用全连接层实现方式介绍RPN层实现容易帮助我们理解这一过程，但在实现时作者选用了卷积层实现全连接层的功能。

(3)个人理解：全连接层本来就是特殊的卷积层，如果产生256或512维的fc特征，事实上可以用Num_out=256或512, kernel_size=3*3, stride=1的卷积层实现conv5-3到第一个全连接特征的映射。然后再用两个Num_out分别为2*9=18和4*9=36，kernel_size=1*1，stride=1的卷积层实现上一层特征到两个分支cls层和reg层的特征映射。

(4)注意：这里2*9中的2指cls层的分类结果包括前后背景两类，4*9的4表示一个Proposal的中心点坐标x，y和宽高w，h四个参数。采用卷积的方式实现全连接处理并不会减少参数的数量，但是使得输入图像的尺寸可以更加灵活。在RPN网络中，我们需要重点理解其中的anchors概念，Loss fucntions计算方式和RPN层训练数据生成的具体细节。



Anchors:

字面上可以理解为锚点，位于之前提到的n*n的sliding window的中心处。对于一个sliding window，我们可以同时预测多个proposal，假定有k个。k个proposal即k个reference boxes，每一个reference box又可以用一个scale，一个aspect_ratio和sliding window中的锚点唯一确定。所以，我们在后面说一个anchor，你就理解成一个anchor box 或一个reference box。作者在论文中定义k=9，即3种scales和3种aspect_ratio确定出当前sliding window位置处对应的9个reference boxes， 4*k个reg-layer的输出和2*k个cls-layer的score输出。对于一幅W*H的feature map，对应W*H*k个锚点。所有的锚点都具有尺度不变性。

Loss functions:

在计算Loss值之前，作者设置了anchors的标定方法。正样本标定规则：

1)　　如果Anchor对应的reference box与ground truth的IoU值最大，标记为正样本；

2) 如果Anchor对应的reference box与ground truth的IoU>0.7，标记为正样本。事实上，采用第2个规则基本上可以找到足够的正样本，但是对于一些极端情况，例如所有的Anchor对应的reference box与groud truth的IoU不大于0.7，可以采用第一种规则生成。

3) 负样本标定规则：如果Anchor对应的reference box与ground truth的IoU<0.3，标记为负样本。

4) 剩下的既不是正样本也不是负样本，不用于最终训练.

5) 训练RPN的Loss是有classification loss （即softmax loss）和regression loss （即L1 loss）按一定比重组成的.

计算softmax loss需要的是anchors对应的groundtruth标定结果和预测结果，计算regression loss需要三组信息：

i. 预测框，即RPN网络预测出的proposal的中心位置坐标x,y和宽高w,h;

ii. 锚点reference box:

之前的9个锚点对应9个不同scale和aspect_ratio的reference boxes，每一个reference boxes都有一个中心点位置坐标x_a，y_a和宽高w_a，h_a;

iii. ground truth：标定的框也对应一个中心点位置坐标x*，y*和宽高w*，h*。因此计算regression loss和总Loss方式如下：

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a,$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a),$$
$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a,$$
$$t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a),$$
$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*).$$

RPN训练设置：

(1)在训练RPN时，一个Mini-batch是由一幅图像中任意选取的256个proposal组成的（？），其中正负样本的比例为1：1。

(2)如果正样本不足128，则多用一些负样本以满足有256个Proposal可以用于训练，反之亦然。

(3)训练RPN时，与VGG共有的层参数可以直接拷贝经ImageNet训练得到的模型中的参数；剩下没有的层参数用标准差=0.01的高斯分布初始化。

另外一篇

https://blog.csdn.net/docrazy5351/article/details/78993413

Faster R-CNN是目标检测界的大神Ross Girshick 2015年提出的一个很经典的检测结构，它将传统的Selective Search提取目标的方法替换成网络训练来实现，使得全流程的检测、分类速度大幅提升。

图1是Faster R-CNN的基本结构，由以下4个部分构成：

1、特征提取部分：用一串卷积+pooling从原图中提取出feature map；

2、RPN部分：这部分是Faster R-CNN全新提出的结构，作用是通过网络训练的方式从feature map中获取目标的大致位置；

3、Proposal Layer部分：利用RPN获得的大致位置，继续训练，获得更精确的位置；

4、ROI Pooling部分：利用前面获取到的精确位置，从feature map中抠出要用于分类的目标，并pooling成固定长度的数据；

一、特征提取部分

特征提取部分是输入图片和feature map间的那一串卷积+pooling，这部分和普通的CNN网络中特征提取结构没有区别，可以用VGG、ResNet、Inception等各种常见的结构实现(只使用全连接层之前的部分)，这部分不再详述。

二、RPN部分

目标识别有两个过程：首先你要知道目标在哪里，要从图片中找出要识别的前景，然后才是拿前景去分类。在Faster R-CNN提出之前常用的提取前景(本文称为提取proposal)的方法是Selective Search，简称SS法，通过比较相邻区域的相似度来把相似的区域合并到一起，反复这个过程，最终就得到目标区域，这种方法相当耗时以至于提取proposal的过程比分类的过程还要慢，完全达不到实时的目的；到了Faster R-CNN时，作者就想出把提取proposal的过程也通过网络训练来完成，部分网络还可以和分类过程共用，新的方法称为Reginal Proposal Network(RPN)，速度大大提升。

RPN做两件事：

1、把feature map分割成多个小区域，识别出哪些小区域是前景，哪些是背景，简称RPN Classification；

2、获取前景区域的大致坐标，简称RPN bounding box regression。

1、RPN Classification

RPN Classification的过程就是个二分类的过程。先要在feature map上均匀的划分出KxHxW个区域（称为anchor，K=9，H是feature map的高度，W是宽度），通过比较这些anchor和ground truth间的重叠情况来决定哪些anchor是前景，哪些是背景，也就是给每一个anchor都打上前景或背景的label。有了labels，你就可以对RPN进行训练使它对任意输入都具备识别前景、背景的能力。

rpn_cls_score_reshape模块输出的结构是[1,9*H,W,2]，就是9xHxW个anchor二分类为前景、背景的概率；anchor_target_layer模块输出的是每一个anchor标注的label，拿它和二分类概率一比较就能得出分类的loss。

一个feature map有9xHxW个anchor，就是说每个点对应有9个anchor，这9个anchor有1:1、1:2、2:1三种长宽比，每种长宽比都有三种尺寸。一般来说原始输入图片都要缩放到固定的尺寸才能作为网络的输入，这个尺寸在作者源码里限制成800x600，9种anchor还原到原始图片上基本能覆盖800x600图片上各种尺寸的坐标。

feature map每个点对应个不同尺寸的anchor

要注意的是在实际应用时并不是把全部HxWx9个anchor都拿来做label标注，这里面有些规则来去除效果不好的anchor，具体的规则如下：

1、覆盖到feature map边界线上的anchor不参与训练；

2、前景和背景交界地带的anchor不参与训练。这些交界地带即不作为前景也不作为背景，以防出现错误的分类。在作者原文里把IOU>0.7作为标注成前景的门限，把IOU<0.3作为标注成背景的门限，之间的值就不参与训练，IOU是anchor与ground truth的重叠区域占两者总覆盖区域的比例，见示意图4；

3、训练时一个batch的样本数是256，对应同一张图片的256个anchor，前景的个数不能超过一半，如果超出，就随机取128个做为前景，背景也有类似的筛选规则；

2、RPN bounding box regression

RPN bounding box regression用于得出前景的大致位置，要注意这个位置并不精确，准确位置的提取在后面的Proposal Layer bounding box regression章节会介绍。提取的过程也是个训练的过程，前面的RPN classification给所有的anchor打上label后，我们需用一个表达式来建立anchor与ground truth的关系，假设anchor中心位置坐标是[Ax, Ay]，长高为Aw和Ah，对应ground truth的4个值为[Gx,Gy,Gw,Gh]，他们间的关系可以用公式 1 来表示。[dx(A), dy(A), dw(A), dh(A)]就是anchor与ground truth之间的偏移量。

有了这 4 个偏移量，你就可以拿他们去训练图2 RPN中下面一个分支的输出。完成训练后RPN就具备识别每一个anchor到与之对应的最优proposal偏移量的能力（[d'x(A), d'y(A), d'w(A), d'h(A)]），换个角度看就是得到了所有proposal的位置和尺寸。要注意的是如果一个feature map中有多个ground truth，每个anchor只会选择和它重叠度最高的ground truth来计算偏移量。

3、RPN的loss计算（论文粘图！）

RPN训练时要把RPN classification和RPN bounding box regression的loss加到一起来实现联合训练。公式3中Ncls是一个batch的大小256，Lcls(pi, pi*)是前景和背景的对数损失，pi是anchor预测为目标的概率，就是前面rpn_cls_score_reshape输出的前景部分score值，pi*是前景的label值，就是1，将一个batch所有loss求平均就是RPN classification的损失；公式3中Nreg是anchor的总数，λ是两种 loss的平衡比例，ti是图2中rpn_bbox_pred模块输出的[d'x(A), d'y(A), d'w(A), d'h(A)]，t*i是训练时每一个anchor与ground truth间的偏移量，t*i与ti用smooth L1方法来计算loss就是RPN bounding box regression的损失：

三、Proposal Layer部分

得到proposal大致位置后下一步就是要做精确位置的回归了。在RPN的训练收敛后我们能得到anchor相对于proposal的偏移量[d'x(A), d'y(A), d'w(A), d'h(A)](要注意这里是想对于proposal的，而不是相对于ground truth的)，有了偏移量再根据公式1就能算出proposal的大致位置。在这个过程中HxWx9个anchor能算出HxWx9个proposal，大多数都是聚集在ground truth周围的候选框，这么多相近的proposal完全没必要反而增加了计算量，这时就要用一些方法来精选出最接近ground truth的proposal，Ross Girshick给了三个步骤：

1、先选出前景概率最高的N个proposal;

2、做非极大值抑制(NMS)

3、NMS后再次选择前景概率最高的M个proposal;

经历这三个步骤后能够得到proposal的大致位置，但这还不够，为了得到更精确的坐标，你还要利用公式2再反推出这个大致的proposal和真实的ground truth间还有多少偏移量，对这个新的偏移量再来一次回归才是完成了精确的定位。

上面的过程比较绕，反复在偏移量、anchor、ground truth间切换。

proposal精确位置回归时计算loss的公式和RPN bounding box regression的loss计算方法完全相同，也用smooth L1方法。

四、ROI Pooling部分

ROI Pooling做了两件事：

1、从feature maps中“抠出”proposals（大小、位置由RPN生成）区域;

2、把“抠出”的区域pooling成固定长度的输出。

图6是pooling过程的示意图，feature map中有两个不同尺寸的proposals，但pooling后都是7x7=49个输出，这样就能为后面的全连接层提供固定长度的输入。这种pooling方式有别于传统的pooling，没有任何tensorflow自带的函数能

实现这种功能，你可以自己用python写个ROI Pooling的过程，但这样就调用不了GPU的并行计算能力，所以作者的源码里用C++来实现整个ROI Pooling。

为什么要pooling成固定长度的输出呢？这个其实来自于更早提出的SPP Net，RPN网络提取出的proposal大小是会变化的，而分类用的全连接层输入必须固定长度，所以必须有个从可变尺寸变换成固定尺寸输入的过程。在较早的R-CNN和Fast R-CNN结构中都通过对proposal进行拉升（warp）或裁减（crop）到固定尺寸来实现，拉升、裁减的副作用就是原始的输入发生变形或信息量丢失，以致分类不准确。而ROI Pooling就完全规避掉了这个问题，proposal能完整的pooling成全连接的输入，而且没有变形，长度也固定。

五、训练过程

前面介绍了Faster R-CNN的结构，最后看下训练方法，为了便于说明，我们把RPN中的rpn classification和rpn bounding box regression统称为RPN训练；把proposal layer中对proposal精确位置的训练和最终的准确分类训练统称为R-CNN训练。Ross Girshick在论文中介绍了3种训练方法：

- Alternating training：RPN训练和R-CNN训练交替进行，共交替两次。训练时先用ImageNet预训练的结果来初始化网络，训练RPN，用得到的proposal再训练R-CNN，之后用R-CNN训练出的参数来初始网络，再训练一次RPN，最后用RPN训练出的参数来初始化网络，最后训练次R-CNN，就完成了全部的训练过程。
- Approximate joint training：这里与前一种方法不同，不再是串行训练RPN和R-CNN，而是尝试把二者融入到一个网络内一起训练。这里Approximate 的意思是指把RPN bounding box regression部分反向计算得到的梯度完全舍弃，不用做更新网络参数的权重。Approximate joint training相对于Alternating traing减少了25-50%的训练时间。
- Non-approximate training：该方法和Approximate joint training基本一致，只是不再舍弃RPN bounding box regression部分得到的梯度。

本文开头提供的源码使用的是第三种方法，把 4 个部分的loss都加到了一起来训练，它的速度要更快。

- **关于ancher（回原图进行，不是在特征图上）**

https://zhuanlan.zhihu.com/p/24916624

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**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），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。
- 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 ()，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（），因此RPN的总体Loss函数可以定义为：
  - i表示第i个anchor，当anchor是正样本时 =1 ，是负样本则 =0 。 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， 分别表示 predicted box, anchor box, and ground truth box （y,w,h同理） 表示 predict box相对于anchor box的偏移， 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。
  - 其中 是：
  - 表示这些regressor的loss指针对正样本而言，因为负样本时 =0该项被消去。
  - 是关于两种类别 (object vs. not object) 的log loss

训练：

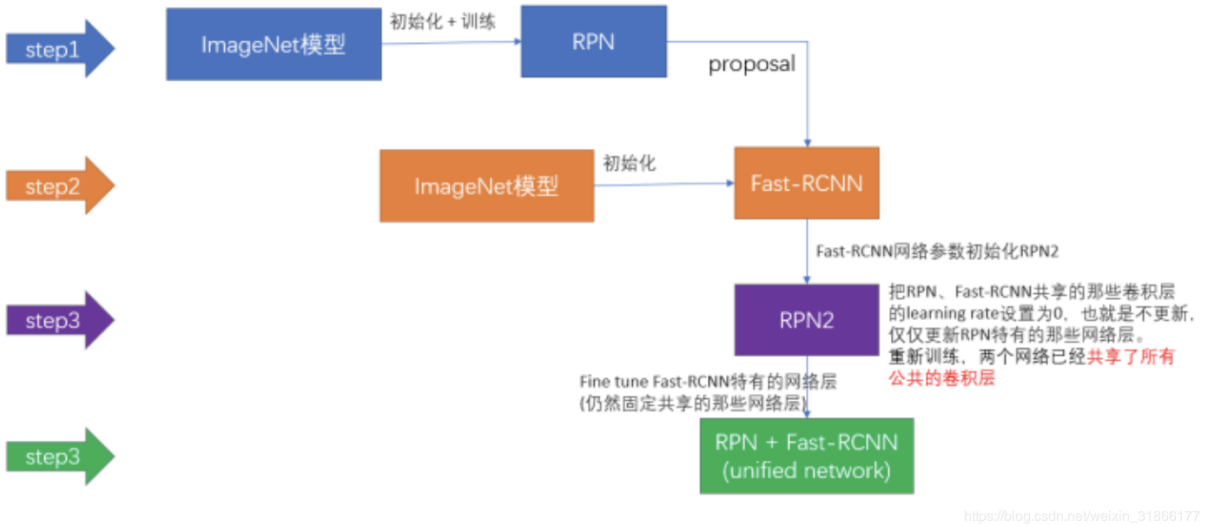
正负样本的选择

- 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)

网络模型整体训练过程：

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。



1. Faster RCNN 学习笔记
2. faster rcnn源码解析
3. R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD, R-FCN系列深度学习检测方法梳理

未看：https://zhuanlan.zhihu.com/p/30720870

https://zhuanlan.zhihu.com/p/32404424

region proposal的作用：
https://www.zhihu.com/question/265345106/answer/294410307

https://www.cnblogs.com/dudumiaomiao/p/6560841.html

数据标注工具：机器学习——数据标注工具使用

数据集使用：ICFHR 2014