原 R-CNN 最直观的理解

2018年04月27日 21:37:30 WeisongZhao 阅读数：2095 标签： CNN 人工智能 目标识别 更多

从2014年CNN就成为用于目标检测的极热门的工具，至于起因还要从RCNN---Fast RCNN----Faster RCNN说起，[illegible]讨论一些RCNN的方法：

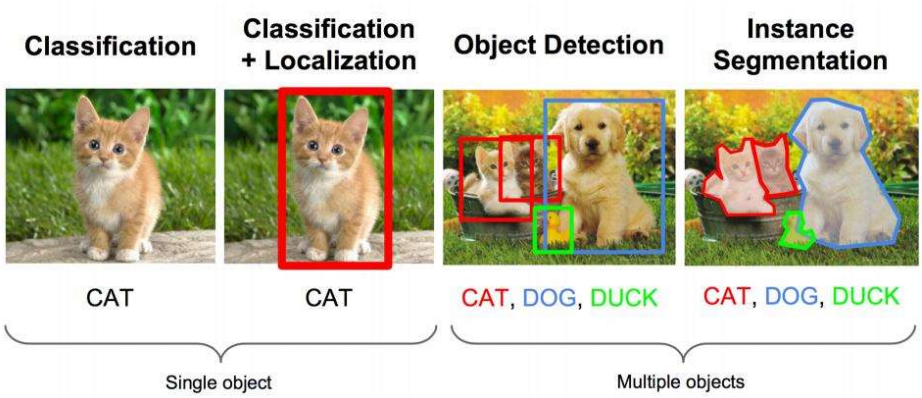
object detection技术的演进：

RCNN->SppNET->Fast-RCNN->Faster-RCNN

从图像识别的任务说起

这里有一个图像任务：

既要把图中的物体识别出来，又要用方框框出它的位置。



上面的任务用专业的说法就是：图像识别+定位

图像识别（classification）：

输入：图片

输出：物体的类别

评估方法：准确率

定位（localization）：

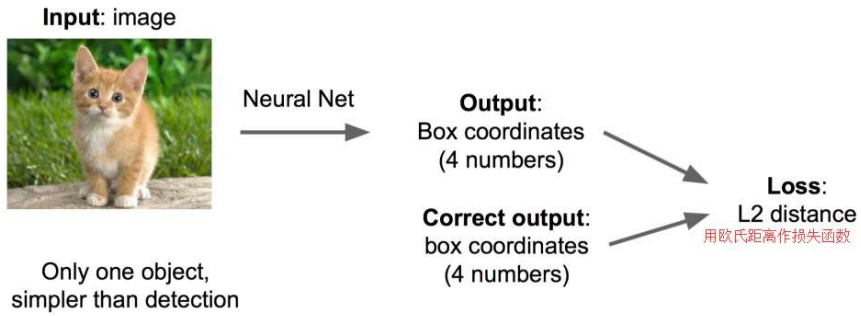
输入：图片

输出：方框在图片中的位置（x,y,w,h）

评估方法：检测评价函数 intersection-over-union (IOU)

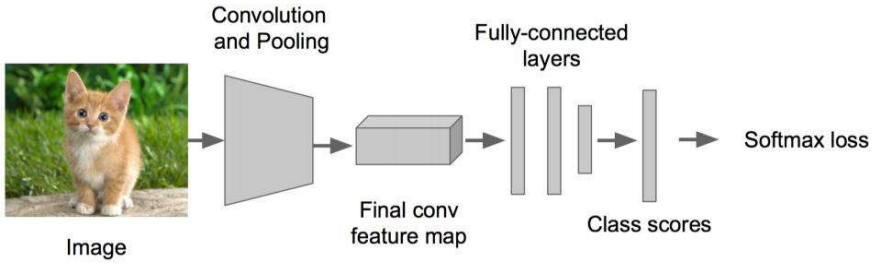
卷积神经网络CNN已经帮我们完成了图像识别（判定是猫还是狗）的任务了，我们只需要添加一些额外的功能来完成定位任务即可。

看做回归问题，我们需要预测出（x,y,w,h）四个参数的值，从而得出方框的位置。



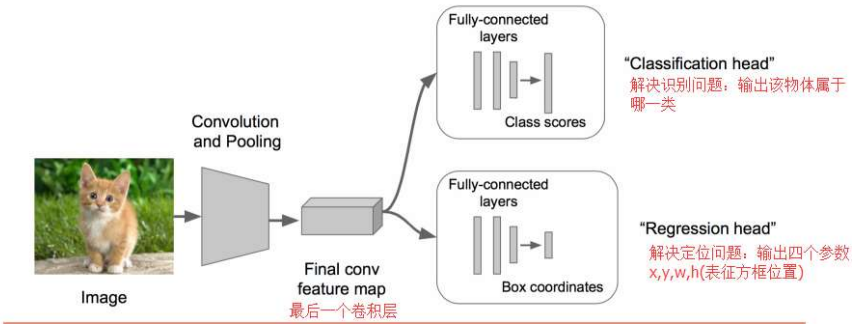
步骤1:

- 先解决简单问题， 搭一个识别图像的神经网络
- 在AlexNet VGG GoogleLenet上fine-tuning一下



步骤2:

- 在上述神经网络的尾部展开（也就说CNN前面保持不变，我们对CNN的结尾处作出改进：加了两个头："分类头"和"回归头"）
- 成为classification + regression模式



步骤3:

- Regression那个部分用欧氏距离损失
- 使用SGD训练

步骤4:

- 预测阶段把2个头部拼上
- 完成不同的功能

这里需要进行两次fine-tuning

第一次在ALexNet上做，第二次将头部改成regression head，前面不变，做一次fine-tuning

Regression的部分加在哪?

有两种处理方法:

- 加在最后一个卷积层后面（如VGG）

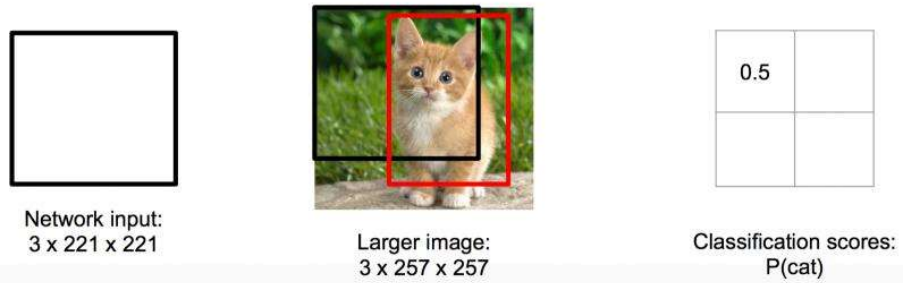
- 加在最后一个全连接层后面（如R-CNN）

regression太难做了，应想方设法转换为classification问题。
regression的训练参数收敛的时间要长得多，所以上面的网络采取了用classification的网络来计算出网络共同部分的连接权值。

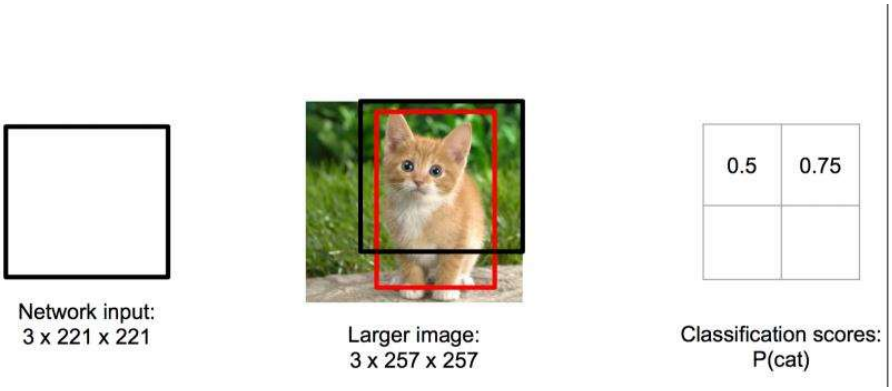
思路二：取图像窗口

- 还是刚才的classification + regression思路
- 咱们取不同的大小的“框”
- 让框出现在不同的位置，得出这个框的判定得分
- 取得分最高的那个框

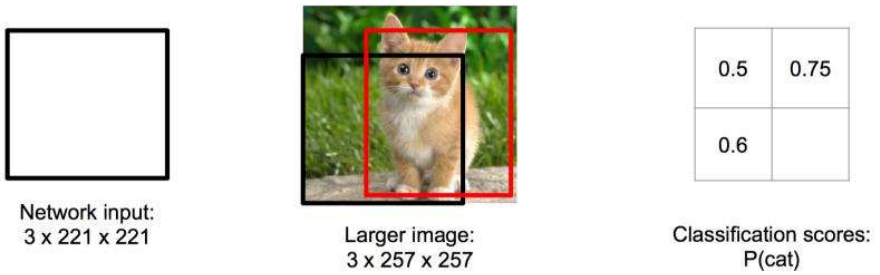
左上角的黑框：得分0.5



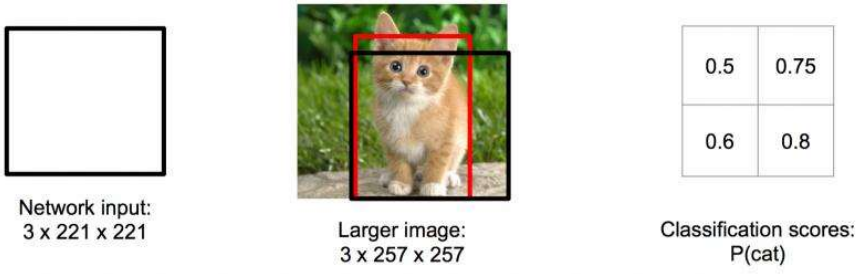
右上角的黑框：得分0.75



左下角的黑框：得分0.6



右下角的黑框：得分0.8



根据得分的高低，我们选择了右下角的黑框作为目标位置的预测。

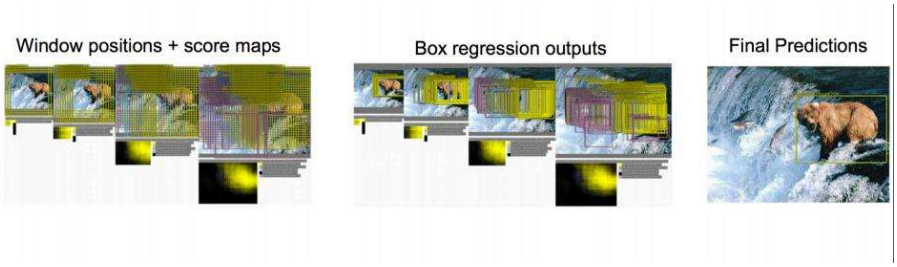
注：有的时候也会选择得分最高的两个框，然后取两框的交集作为最终的位置预测。

疑惑：框要取多大?

取不同的框，依次从左上角扫到右下角。非常粗暴啊。

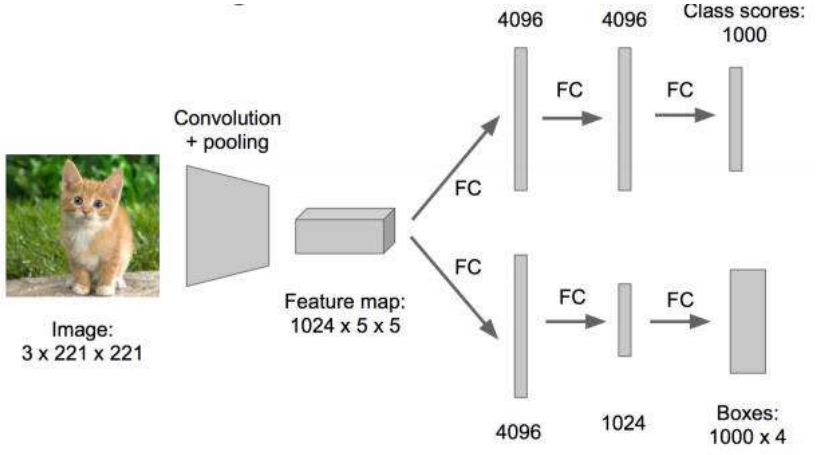
总结一下思路：

对一张图片，用各种大小的框（遍历整张图片）将图片截取出来，输入到CNN，然后CNN会输出这个框的得分（classification）以及这个框图片对应的x,y,h,w（regression）

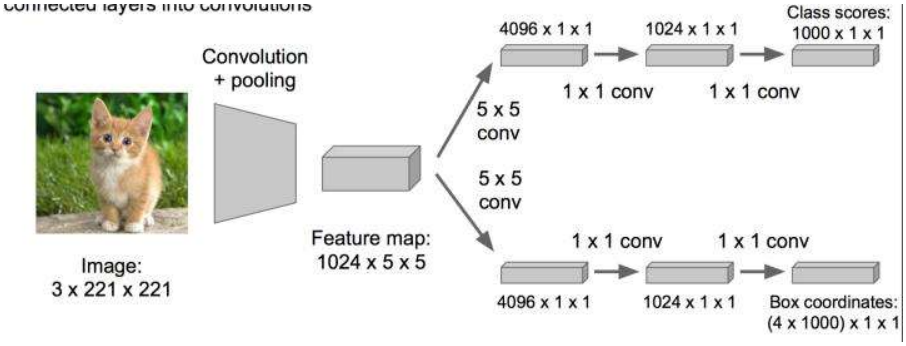


这方法实在太耗时间了，做个优化。

原来网络是这样的：



优化成这样：把全连接层改为卷积层，这样可以提提速。

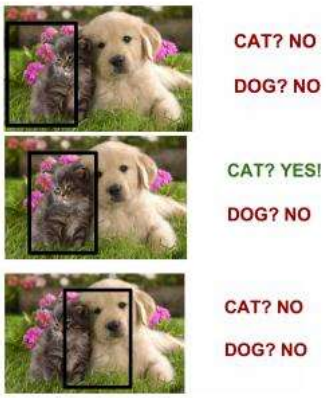


物体检测（Object Detection）

当图像有很多物体怎么办的？难度可是一下暴增啊。

那任务就变成了：多物体识别+定位多个物体

那把这个任务看做分类问题？



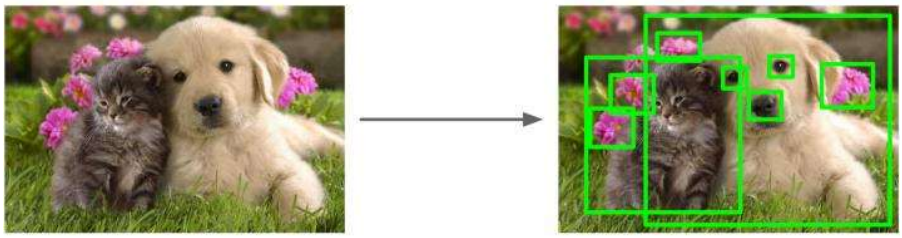
看成分类问题有何不妥？

- 你需要找很多位置， 给很多个不同大小的框
- 你还需要对框内的图像分类
- 当然， 如果你的GPU很强大， 恩， 那加油做吧…

看做classification， 有没有办法优化下？我可不想试那么多框那么多位置啊！

有人想到一个好方法：

找出可能含有物体的框（也就是候选框，比如选1000个候选框），这些框之间是可以互相重叠互相包含的，这样我们就可以避免暴力枚举的所有框了。



大牛们发明好多选定候选框的方法，比如EdgeBoxes和Selective Search。

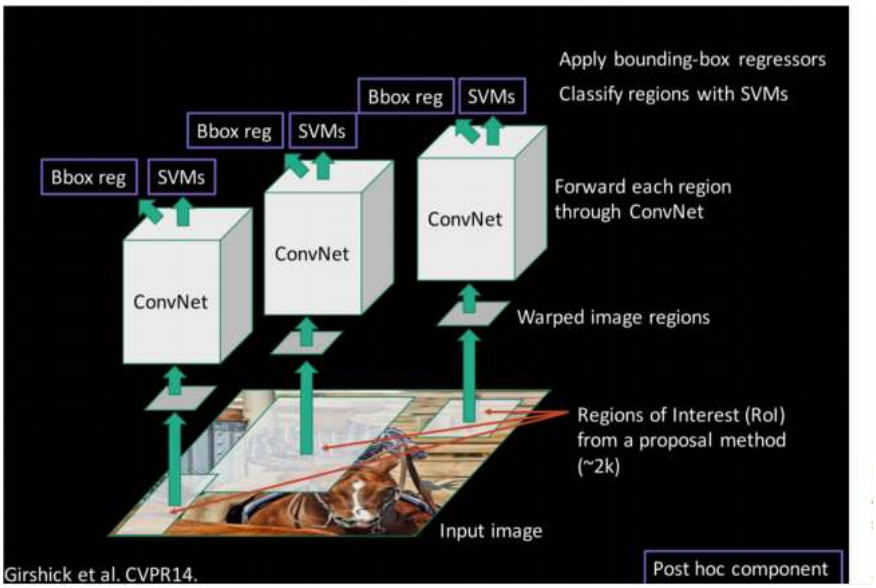
以下是各种选定候选框的方法的性能对比。

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	⋆⋆⋆	⋆	·
CPMC [19]	Grouping	✓	✓	✓	250	-	⋆⋆	⋆
EdgeBoxes [20]	Window scoring		✓	✓	0.3	⋆⋆	⋆⋆⋆	⋆⋆⋆
Endres [21]	Grouping	✓	✓	✓	100	-	⋆⋆⋆	⋆⋆
Geodesic [22]	Grouping	✓		✓	1	⋆	⋆⋆⋆	⋆⋆
MCG [23]	Grouping	✓	✓	✓	30	⋆	⋆⋆⋆	⋆⋆⋆
Objectness [24]	Window scoring		✓	✓	3	·	⋆	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	⋆
RandomizedPrim's [26]	Grouping	✓		✓	1	⋆	⋆	⋆⋆
Rantalankila [27]	Grouping	✓		✓	10	⋆⋆	·	⋆⋆
Rigor [28]	Grouping	✓		✓	10	⋆	⋆⋆	⋆⋆
SelectiveSearch [29]	Grouping	✓	✓	✓	10	⋆⋆	⋆⋆⋆	⋆⋆⋆
Gaussian				✓	0	·	·	⋆
SlidingWindow				✓	0	⋆⋆⋆	·	·
Superpixels		✓			1	⋆	·	·
Uniform				✓	0	·	·	·

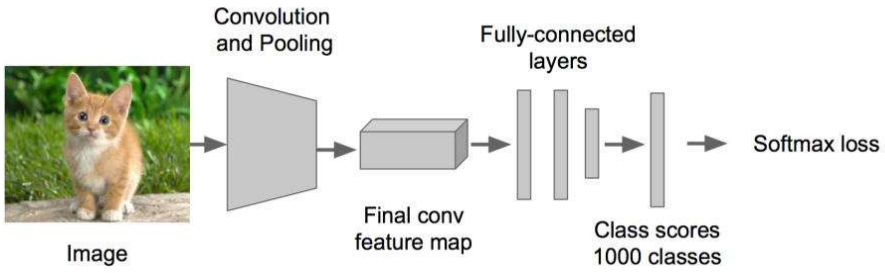
有一个很大的疑惑，提取候选框用到的算法“选择性搜索”到底怎么选出这些候选框的呢？那个就得好好看看它的论文了，这里就不介绍了。

R-CNN横空出世

基于以上的思路，RCNN的出现了。

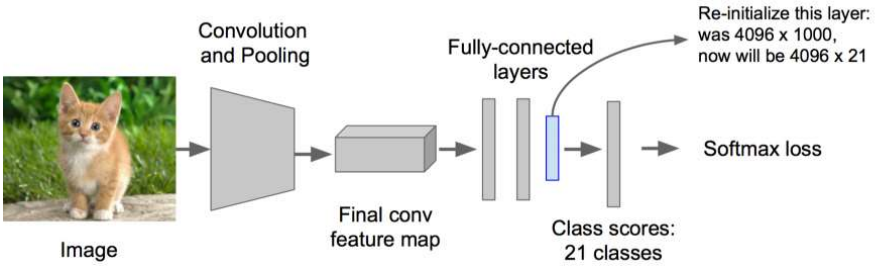


步骤一：训练（或者下载）一个分类模型（比如AlexNet）



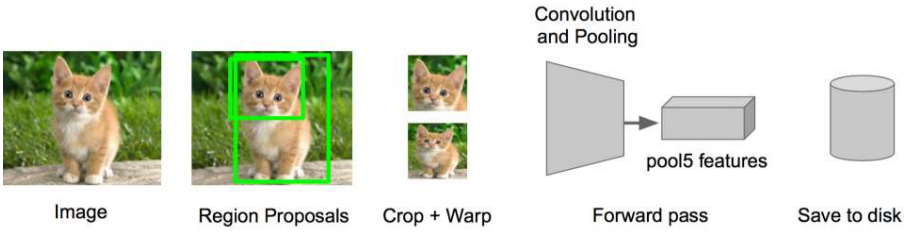
步骤二：对该模型做fine-tuning

- 将分类数从1000改为20
- 去掉最后一个全连接层



步骤三：特征提取

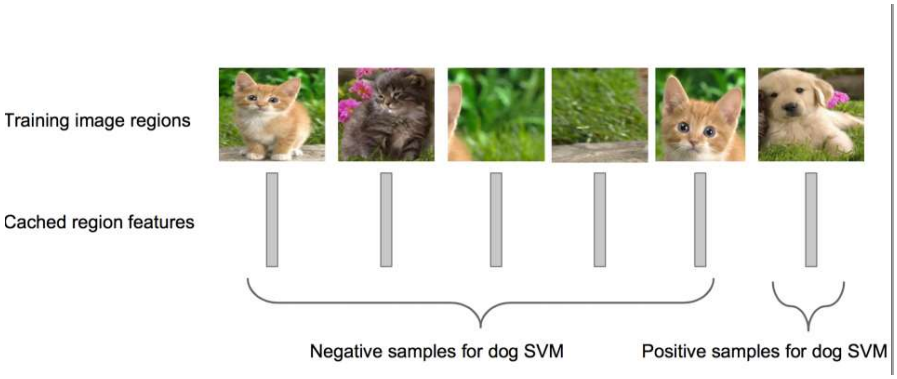
- 提取图像的所有候选框（选择性搜索）
- 对于每一个区域：修正区域大小以适合CNN的输入，做一次前向运算，将第五个池化层的输出（就是对候选框提取到的特征）存到硬盘



步骤四：训练一个SVM分类器（二分类）来判断这个候选框里物体的类别

每个类别对应一个SVM，判断是不是属于这个类别，是就是positive，反之nagative

比如下图，就是狗分类的SVM



步骤五：使用回归器精细修正候选框位置：对于每一个类，训练一个线性回归模型去判定这个框是否框得完美。

RCNN的进化中SPP Net的思想对其贡献很大，这里也简单介绍一下SPP Net。

SPP Net

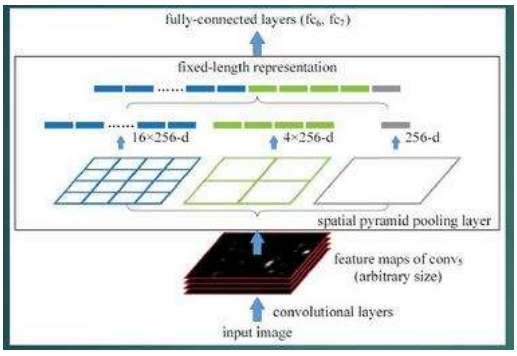
SPP：Spatial Pyramid Pooling（空间金字塔池化）

它的特点有两个:

1.结合空间金字塔方法实现CNNs的对尺度输入。

一般CNN后接全连接层或者分类器，他们都需要固定的输入尺寸，因此不得不对输入数据进行crop或者warp，这些预处理会造成数据的丢失或几何的失真。SPP Net的第一个字塔思想加入到CNN，实现了数据的多尺度输入。

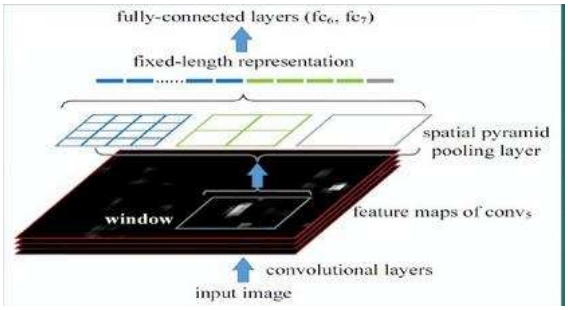
如下图所示，在卷积层和全连接层之间加入了SPP layer。此时网络的输入可以是任意尺度的，在SPP layer中每一个pooling的filter会根据输入调整大小，而SPP的输出尺度



2.只对原图提取一次卷积特征

在R-CNN中，每个候选框先resize到统一大小，然后分别作为CNN的输入，这样是很低效的。

所以SPP Net根据这个缺点做了优化：只对原图进行一次卷积得到整张图的feature map，然后找到每个候选框zaifeature map上的映射patch，将此patch作为每个候选框的卷
SPP layer和之后的层。节省了大量的计算时间，比R-CNN有一百倍左右的提速。

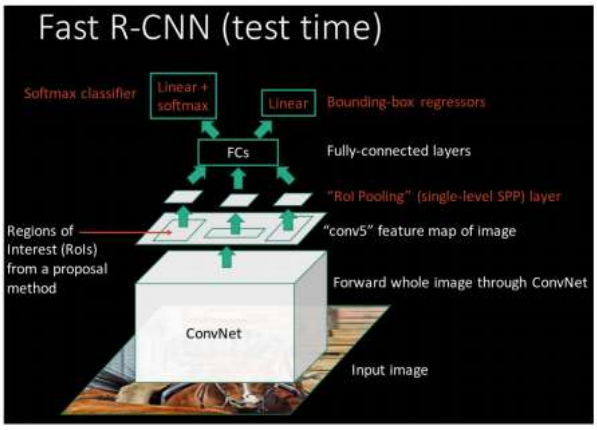


Fast R-CNN

SPP Net真是个好方法，R-CNN的进阶版Fast R-CNN就是在RCNN的基础上采纳了SPP Net方法，对RCNN作了改进，使得性能进一步提高。

R-CNN与Fast RCNN的区别有哪些呢？

先说RCNN的缺点：即使使用了selective search等预处理步骤来提取潜在的bounding box作为输入，但是RCNN仍会有严重的速度瓶颈，原因也很明显，就是计算机对所有r提取时会有重复计算，Fast-RCNN正是为了解决这个问题诞生的。



大牛提出了一个可以看做单层sppnet的网络层，叫做ROI Pooling，这个网络层可以把不同大小的输入映射到一个固定尺度的特征向量，而我们知道，conv、pooling、relu等固定size的输入，因此，在原始图片上执行这些操作后，虽然输入图片size不同导致得到的feature map尺寸也不同，不能直接接到一个全连接层进行分类，但是可以加入这个Pooling层，对每个region都提取一个固定维度的特征表示，再通过正常的softmax进行类型识别。另外，之前RCNN的处理流程是先提proposal，然后CNN提取特征，之后用最后再做bbox regression，而在Fast-RCNN中，作者巧妙的把bbox regression放进了神经网络内部，与region分类和并成为了一个multi-task模型，实际实验也证明，这两个卷积特征，并相互促进。Fast-RCNN很重要的一个贡献是成功的让人们看到了Region Proposal+CNN这一框架实时检测的希望，原来多类检测真的可以在保证准确率的同时度，也为后来的Faster-RCNN做下了铺垫。

画一画重点：

R-CNN有一些相当大的缺点（把这些缺点都改掉了，就成了Fast R-CNN）。

大缺点：由于每一个候选框都要独自经过CNN，这使得花费的时间非常多。

解决：共享卷积层，现在不是每一个候选框都当做输入进入CNN了，而是输入一张完整的图片，在第五个卷积层再得到每个候选框的特征

原来的方法：许多候选框（比如两千个）-->CNN-->得到每个候选框的特征-->分类+回归

现在的方法：一张完整图片-->CNN-->得到每张候选框的特征-->分类+回归

所以容易看见，Fast RCNN相对于RCNN的提速原因就在于：不过不像RCNN把每个候选区域给深度网络提特征，而是整张图提一次特征，再把候选框映射到conv5上，而S一次特征，剩下的只需要在conv5层上操作就可以了。

在性能上提升也是相当明显的：

	R-CNN	Fast R-CNN
Training Time:	84 hours	**9.5 hours**
(Speedup)	1x	**8.8x**
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**

Faster!

FASTER!

Faster R-CNN

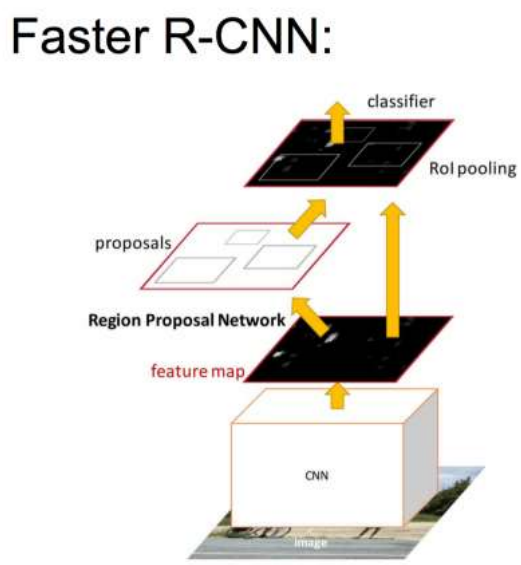
Fast R-CNN存在的问题：存在瓶颈：选择性搜索，找出所有的候选框，这个也非常耗时。那我们能不能找出一个更加高效的方法来求出这些候选框呢？

解决：加入一个提取边缘的神经网络，也就说找到候选框的工作也交给神经网络来做了。

做这样的任务的神经网络叫做Region Proposal Network(RPN)。

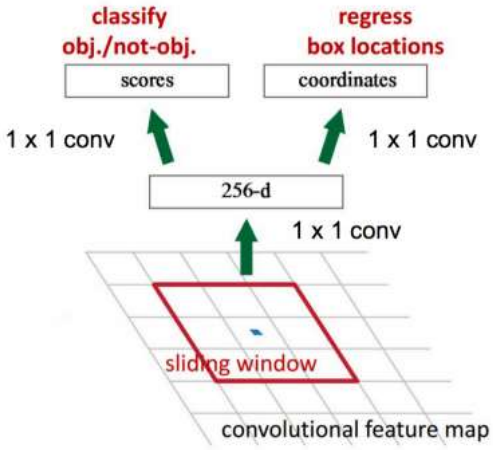
具体做法：

- 将RPN放在最后一个卷积层的后面
- RPN直接训练得到候选区域



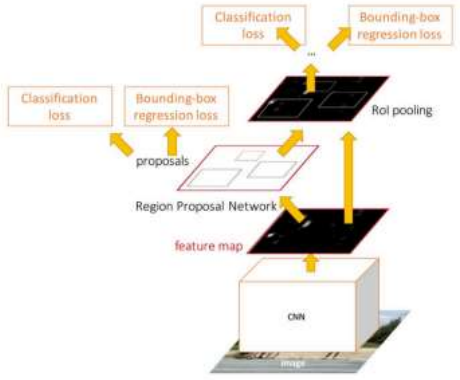
RPN简介：

- 在feature map上滑动窗口
- 建一个神经网络用于物体分类+框位置的回归
- 滑动窗口的位置提供了物体的大体位置信息
- 框的回归提供了框更精确的位置



一种网络，四个损失函数;

- RPN calssification(anchor good.bad)
- RPN regression(anchor->propoasal)
- Fast R-CNN classification(over classes)
- Fast R-CNN regression(proposal ->box)



速度对比

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

Faster R-CNN的主要贡献是设计了提取候选区域的网络RPN，代替了费时的选择性搜索，使得检测速度大幅提高。

最后总结一下各大算法的步骤：

RCNN

1. 在图像中确定约1000-2000个候选框 (使用选择性搜索)
2. 每个候选框内图像块缩放至相同大小，并输入到CNN内进行特征提取
3. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
4. 对于属于某一特征的候选框，用回归器进一步调整其位置

Fast RCNN

1. 在图像中确定约1000-2000个候选框 (使用选择性搜索)
2. 对整张图片输进CNN，得到feature map
3. 找到每个候选框在feature map上的映射patch，将此patch作为每个候选框的卷积特征输入到SPP layer和之后的层
4. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
5. 对于属于某一特征的候选框，用回归器进一步调整其位置

Faster RCNN

1. 对整张图片输进CNN，得到feature map
2. 卷积特征输入到RPN，得到候选框的特征信息
3. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
4. 对于属于某一特征的候选框，用回归器进一步调整其位置

总的来说，从R-CNN, SPP-NET, Fast R-CNN, Faster R-CNN一路走来，基于深度学习目标检测的流程变得越来越精简，精度越来越高，速度也越来越快。可以说基于regio CNN系列目标检测方法是当前目标检测技术领域最主要的一个分支。

参考文献：https://www.cnblogs.com/skyfsm/p/6806246.html