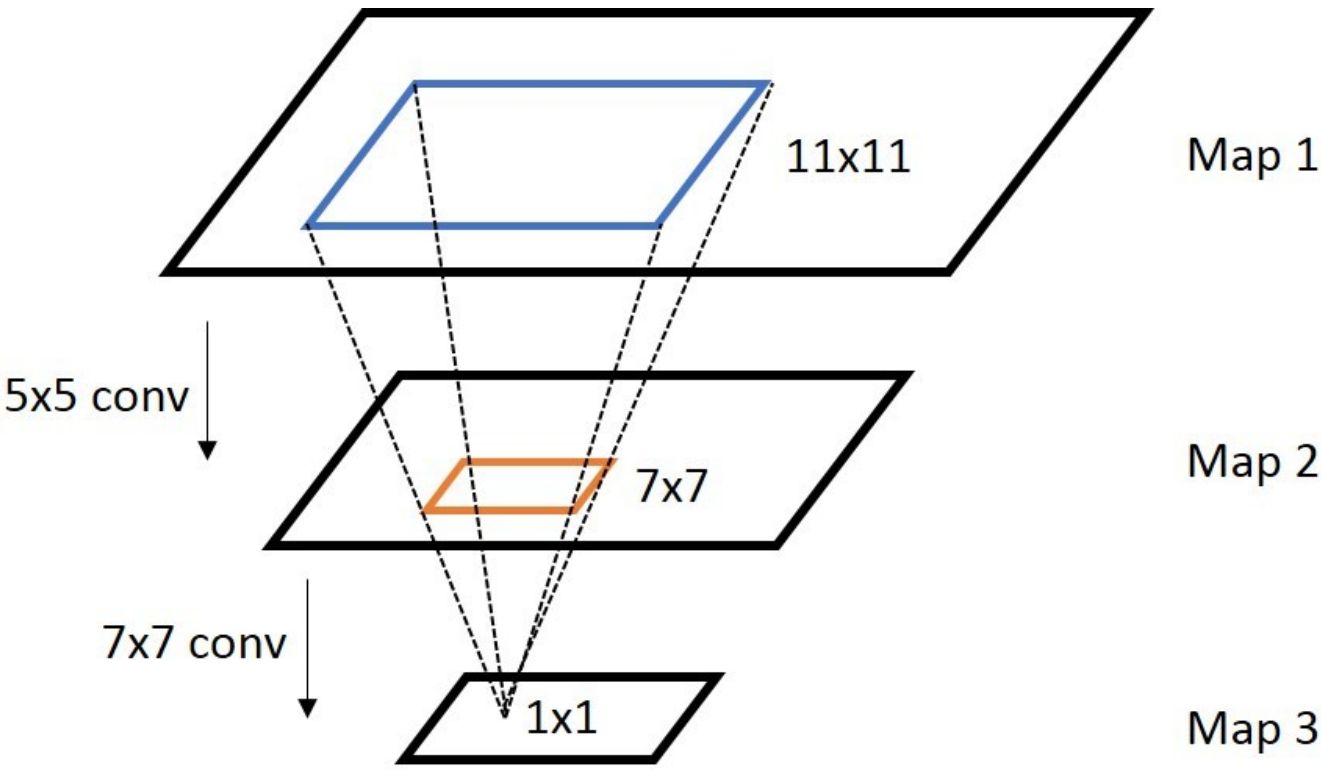


138



原始图片中的ROI如何映射到feature map?



晓雷

138 人赞了该文章

在SPP-net中的难点一曾提到：ROI如何对应到feature map?

这个地方遇到不少坑，看了很多资料都没有太明白，感觉太绕。

先数数遇到的坑：

- 《Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition》原文是这样写的，一脸懵逼。

if rounding is needed, we take the floor operation on the left/top boundary and ceiling on the right/bottom boundary.

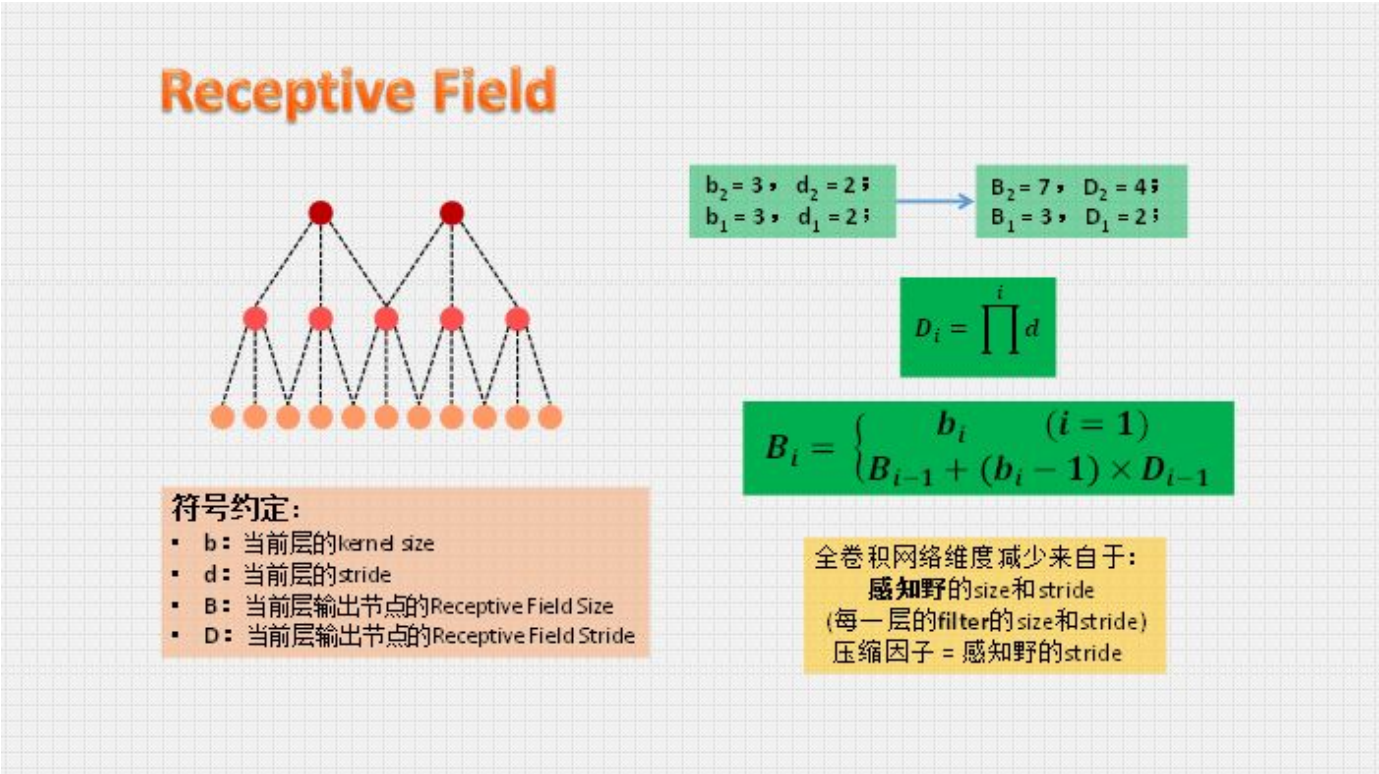
**Mapping a Window to Feature Maps.**

In the detection algorithm (and multi-view testing on feature maps), a window is given in the image domain, and we use it to crop the convolutional feature maps (e.g., conv<sub>5</sub>) which have been sub-sampled several times. So we need to align the window on the feature maps.

In our implementation, we project the corner point of a window onto a pixel in the feature maps, such that this corner point in the image domain is closest to the center of the receptive field of that feature map

pixel. The mapping is complicated by the padding of all convolutional and pooling layers. To simplify the implementation, during deployment we pad  $\lfloor p/2 \rfloor$  pixels for a layer with a filter size of  $p$ . As such, for a response centered at  $(x', y')$ , its effective receptive field in the image domain is centered at  $(x, y) = (Sx', Sy')$  where  $S$  is the product of all previous strides. In our models,  $S = 16$  for ZF-5 on conv<sub>5</sub>, and  $S = 12$  for Overfeat-5/7 on conv<sub>5/7</sub>. Given a window in the image domain, we project the left (top) boundary by:  $x' = \lfloor x/S \rfloor + 1$  and the right (bottom) boundary  $x' = \lceil x/S \rceil - 1$ . If the padding is not  $\lfloor p/2 \rfloor$ , we need to add a proper offset to  $x$ .

- 找了张图是这样画的：有那么点意思，好像是从前向后推出各个层的感受野，可是还是不懂为啥这样。



- 这两张图，看的有点摸不着头脑



如何将图像的ROI映射到feature map？

说实话，笔者还是没有完全弄懂这里的操作。先记录目前能够理解的部分。

总体的映射思路为：In our implementation, we project the corner point of a window onto a pixel in the feature maps, such that this corner point (in the image domain) is closest to the center of the receptive field of that pixel.

略绕，我的理解是：

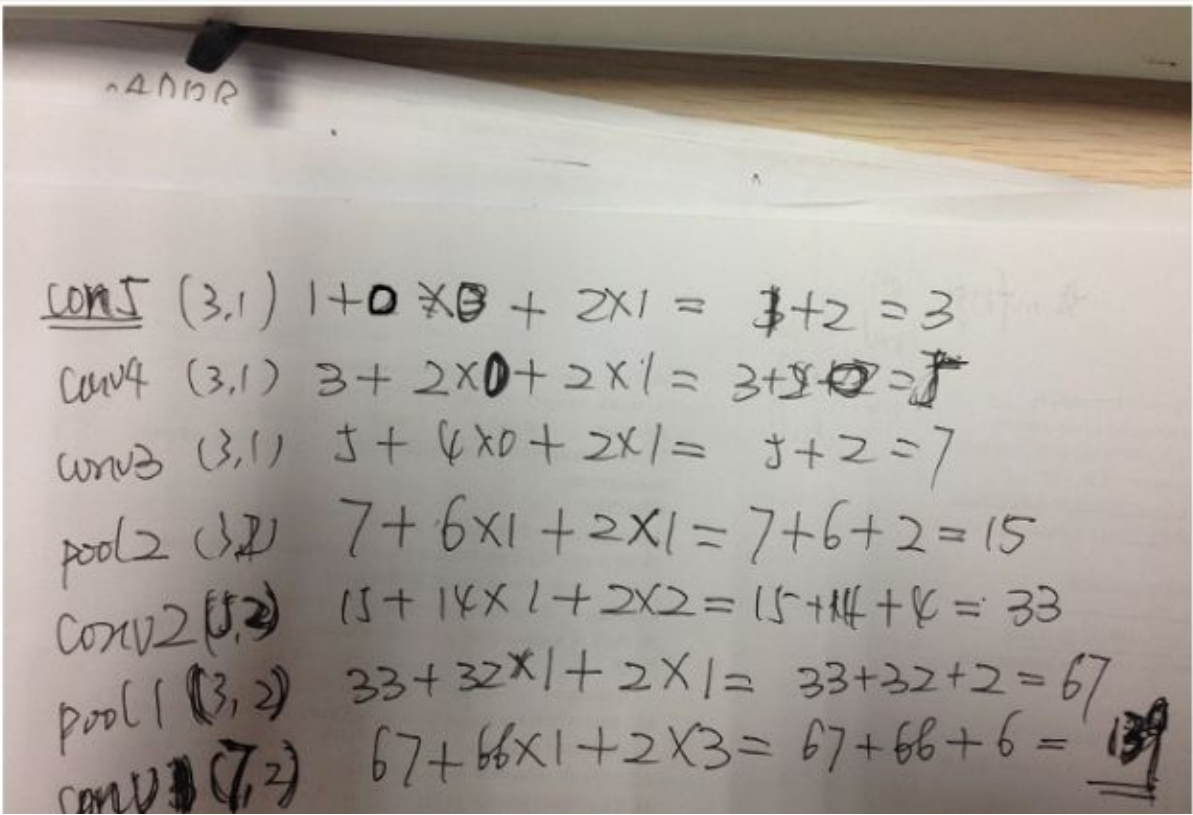
- 映射的是ROI的两个角点，左上角和右下角，这两个角点就可以唯一确定ROI的位置了。
- 将feature map的pixel映射回来图片空间
- 从映射回来的pixel中选择一个距离角点最近的pixel，作为映射。

如果以ZF-5为例子，具体的计算公式为：

image domain, and the effective stride of  $\text{conv}_5$  in the image domain is 16. Denote  $x$  and  $x_{\text{conv}5}$  as the coordinates in the image domain and  $\text{conv}_5$  (we use the MATLAB convention, i.e.,  $x$  starts from 1). We project the top-left corner by:  $x_{\text{conv}5} = \lfloor (x - 139/2 + 63)/16 \rfloor + 1$ . Here 139/2 is the radius of the receptive field, 16 is the effective stride, and 63 is an offset. The offset is computed by the top-left corner of the receptive field in the image domain. Similarly, we project the bottom-right corner by:  $x_{\text{conv}5} = \lceil (x + 139/2 - 75)/16 \rceil - 1$ . Here

这里有几个变量

- 139代表的是感受野的直径，计算这个也需要一点技巧了：如果一个filter的 kernelsize=x, stride=y，而输出的reponse map的长度是n，那么其对应的感受野的长度为：n+(n-1)\*(stride-1)+2\*((kernelsize-1)/2)



知乎



首发于  
晓雷机器学习笔记

写文章



分享

- 接着找了何凯明在ICCV2015上演讲的PPT：《iccv2015\_tutorial\_convolutional\_feature\_maps\_kaiminghe.》，算是有了点眉目。可是还是不造咋回事，只知道有那么个公式，却不知道怎么推出来的，满心的疑惑。

Receptive Field



How to compute the center of the receptive field

- A simple solution
  - For each layer, pad  $\lfloor F/2 \rfloor$  pixels for a filter size  $F$  (e.g., pad 1 pixel for a filter size of 3)
  - On each feature map, the response at (0, 0) has a receptive field centered at (0, 0) on the image
  - On each feature map, the response at (x, y) has a receptive field centered at (Sx, Sy) on the image (stride S)
- A general solution
$$\alpha_L = \alpha_L(i_L) = \alpha_L(i_L - 1) + \beta_L$$
$$\alpha_L = \prod_{j=1}^L s_j$$
$$\beta_L = 1 + \sum_{j=1}^L \left( \prod_{k=1}^{j-1} s_k \right) \left( \frac{F_j - 1}{2} - p_j \right)$$

See [Karel Lenc & Andrea Vedaldi] "R-CNN minus R", BMVC 2015.

ICCV15 Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition", ECCV 2014.

回归正题

最后找到一篇靠谱的文章 卷积神经网络物体检测之感受野大小计算 - machineLearning - 博客园，它给出了一个不错的启发，还附带了代码，最关键的是它给出了参考链接。于是我终于在参考链接找到了这篇 Concepts and Tricks In CNN(长期更新) 最佳博文，不仅清晰易懂，而且公式详细。（不过感觉略有不足，所以下面就详细介绍一下这个大坑）

【先说说感受野的计算】

回忆一下我之前在 卷积神经网络(CNN)简介 里就曾经画图推导过的一个公式（这个公式很常见，可是竟然很少有人去讲它怎么来的，当时我在写 CNN简介就顺便画图推导了一下，没细看的同学可以回头看看）

$$\text{output field size} = (\text{input field size} - \text{kernel size} + 2 \cdot \text{padding}) / \text{stride} + 1$$

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

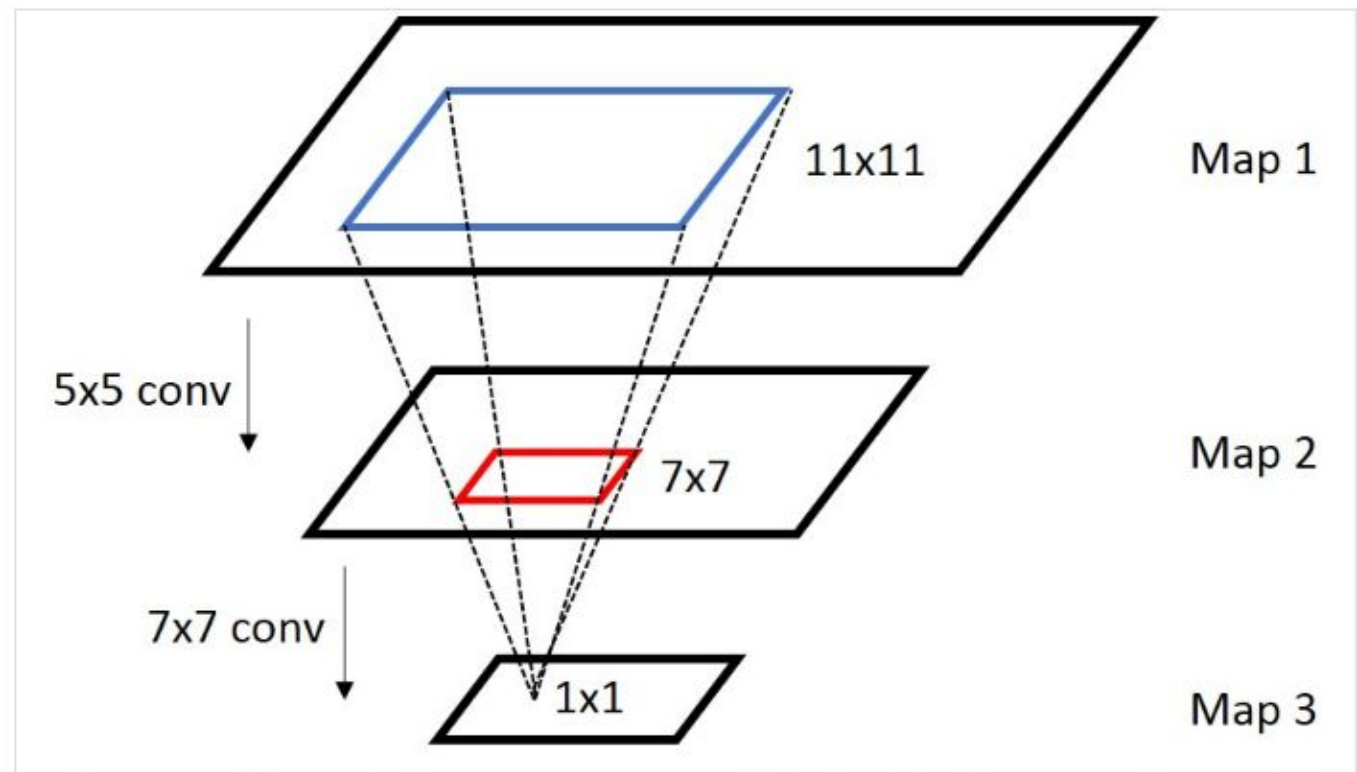
138

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：  $\text{input field size} = (\text{output field size} - 1) \cdot \text{stride} - 2 \cdot \text{padding} + \text{kernel size}$

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：



在上图里面，map 3里1x1的区域对应map 2的receptive field是那个红色的7x7的区域，而map 2里7x7的区域对应于map 1的receptive field是蓝色的11x11的区域，所以map 3里1x1的区域对应map 1的receptive field是蓝色的11x11的区域。

公式化一下：

- 对于Convolution/Pooling layer（博文作者忘了加上padding本文在这里补上）：  
$$r_i = s_i \cdot (r_{i+1} - 1) + k_i - 2 \cdot padding$$
- 对于 Neuronlayer(ReLU/Sigmoid/..):  $r_I = r_{i+1}$ （显然如此）

138 22 条评论 分享 收藏 ... [下一](#) **Fast R-CNN**

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）

卷积神经网络物体检测之感受野大小计算 - machineLearning - 博客园 中用如下核心代码计算了 Alexnet zf-5和VGG16网络每层输出feature map的感受野大小。

```
net_struct = {'alexnet': {'net': [[11,4,0],[3,2,0],[5,1,2],[3,2,0],[3,1,1],[3,1,1],[3,1,1],
    'name': ['conv1', 'pool1', 'conv2', 'pool2', 'conv3', 'conv4', 'conv5', 'pool5'],
    'vgg16': {'net': [[3,1,1],[3,1,1],[2,2,0],[3,1,1],[3,1,1],[2,2,0],[3,1,1],[3,1,1],|
    [2,2,0],[3,1,1],[3,1,1],[3,1,1],[2,2,0],[3,1,1],[3,1,1],[3,1,1],|
    'name': ['conv1_1', 'conv1_2', 'pool1', 'conv2_1', 'conv2_2', 'pool2', 'conv3_1',
    'conv3_2', 'conv3_3', 'pool3', 'conv4_1', 'conv4_2', 'conv4_3', 'pool4', 'conv5_1',
    'conv5_2', 'conv5_3', 'pool5],
    'zf-5': {'net': [[7,2,3],[3,2,1],[5,2,2],[3,2,1],[3,1,1],[3,1,1],[3,1,1]],
    'name': ['conv1', 'pool1', 'conv2', 'pool2', 'conv3', 'conv4', 'conv5']}]

imsize = 224

def outFromIn(isz, net, layernum):#从前向后算输出维度
    totstride = 1
    insize = isz
    for layer in range(layernum):
        fsize, stride, pad = net[layer]
        outsize = (insize - fsize + 2*pad) / stride + 1
        insize = outsize
        totstride = totstride * stride
    return outsize, totstride

def inFromOut(net, layernum):#从后向前算感受野 返回该层元素在原始图片中的感受野
    RF = 1
    for layer in reversed(range(layernum)):
        fsize, stride, pad = net[layer]
        RF = ((RF - 1)* stride) + fsize
    return RF
```



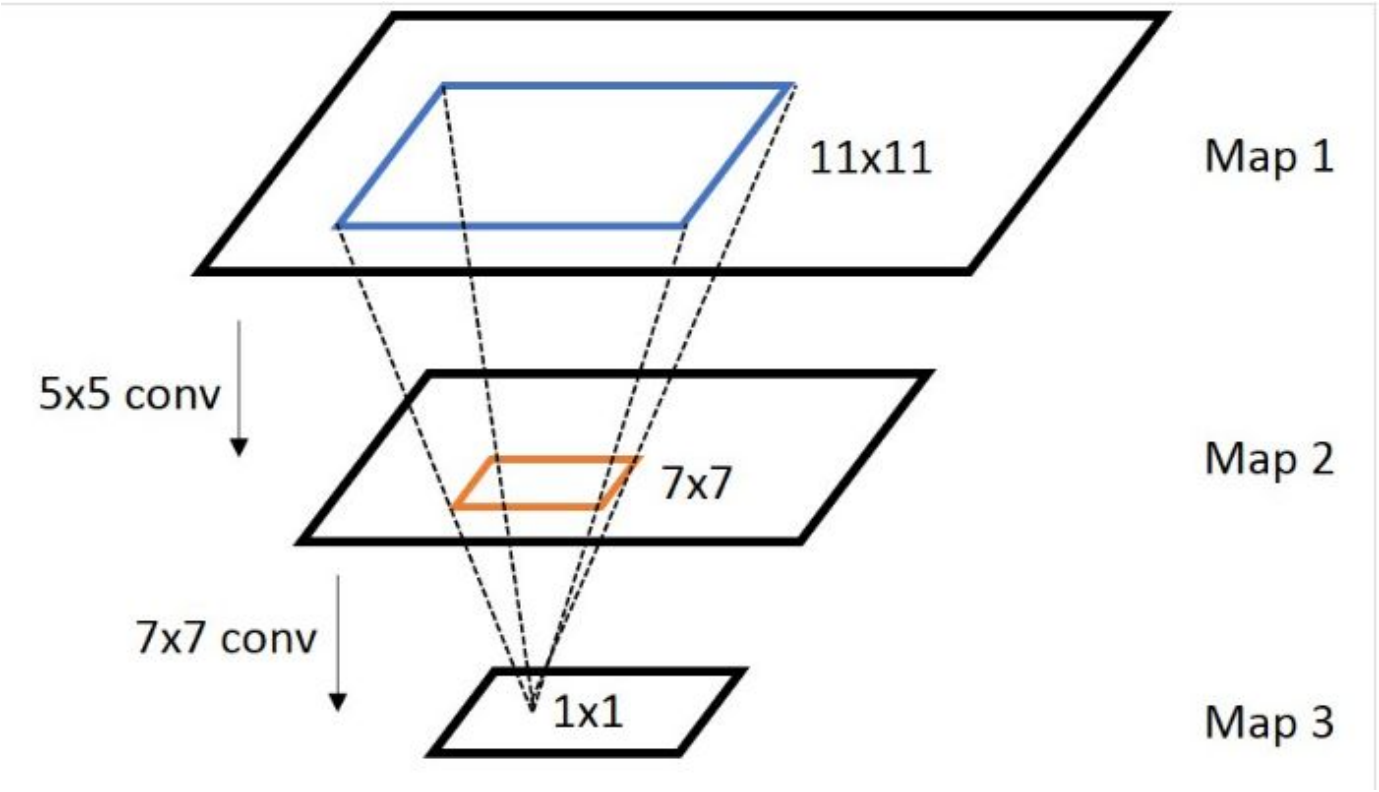
138

```
Layer Name = conv1_2, Output size = 224, Stride = 1, RF size = 5
Layer Name = pool1, Output size = 112, Stride = 2, RF size = 6
Layer Name = conv2_1, Output size = 112, Stride = 2, RF size = 16
Layer Name = conv2_2, Output size = 112, Stride = 2, RF size = 14
Layer Name = pool2, Output size = 56, Stride = 4, RF size = 16
Layer Name = conv3_1, Output size = 56, Stride = 4, RF size = 24
Layer Name = conv3_2, Output size = 56, Stride = 4, RF size = 32
Layer Name = conv3_3, Output size = 56, Stride = 4, RF size = 46
Layer Name = pool3, Output size = 28, Stride = 8, RF size = 44
Layer Name = conv4_1, Output size = 28, Stride = 8, RF size = 66
Layer Name = conv4_2, Output size = 28, Stride = 8, RF size = 76
Layer Name = conv4_3, Output size = 28, Stride = 8, RF size = 92
Layer Name = pool4, Output size = 14, Stride = 16, RF size = 100
Layer Name = conv5_1, Output size = 14, Stride = 16, RF size = 132
Layer Name = conv5_2, Output size = 14, Stride = 16, RF size = 164
Layer Name = conv5_3, Output size = 14, Stride = 16, RF size = 196
Layer Name = pool5, Output size = 7, Stride = 32, RF size = 212
*****net structrue name is zf-5*****
Layer Name = conv1, Output size = 112, Stride = 2, RF size = 7
Layer Name = pool1, Output size = 56, Stride = 4, RF size = 11
Layer Name = conv2, Output size = 28, Stride = 8, RF size = 27
Layer Name = pool2, Output size = 14, Stride = 16, RF size = 43
Layer Name = conv3, Output size = 14, Stride = 16, RF size = 75
Layer Name = conv4, Output size = 14, Stride = 16, RF size = 107
Layer Name = conv5, Output size = 14, Stride = 16, RF size = 139
*****net structrue name is alexnet*****
Layer Name = conv1, Output size = 54, Stride = 4, RF size = 11
Layer Name = pool1, Output size = 26, Stride = 8, RF size = 19
Layer Name = conv2, Output size = 26, Stride = 8, RF size = 51
Layer Name = pool2, Output size = 12, Stride = 16, RF size = 67
Layer Name = conv3, Output size = 12, Stride = 16, RF size = 99
Layer Name = conv4, Output size = 12, Stride = 16, RF size = 131
Layer Name = conv5, Output size = 12, Stride = 16, RF size = 163
Layer Name = pool5, Output size = 5, Stride = 32, RF size = 195
```

【再谈谈感受野上面的坐标映射（Coordinate Mapping）】

为了完整性直接摘录博客内容了：

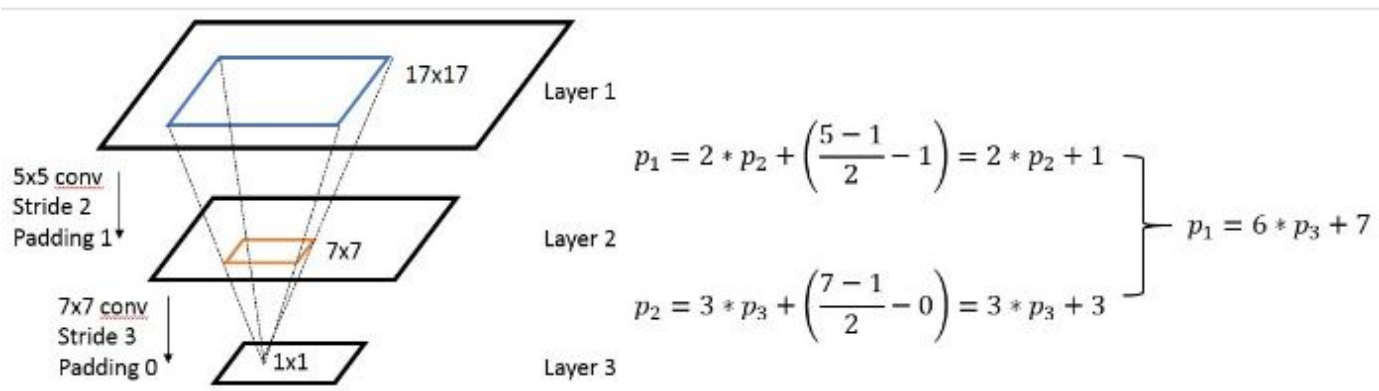
通常，我们需要知道网络里面任意两个feature map之间的坐标映射关系（一般是中心点之间的映射），如下图，我们想得到map 3上的点p3映射回map 2所在的位置p2（橙色框的中心点）



计算公式：

- 对于 Convolution/Pooling layer:  $p_i = s_i \cdot p_{i+1} + ((k_i - 1)/2 - padding)$
- 对于Neuronlayer(ReLU/Sigmoid/..):  $p_I = p_{i+1}$

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：



最后说一下 前面那张PPT里的公式。

Receptive Field



How to compute the center of the receptive field

- A simple solution
  - For each layer, pad  $\lfloor F/2 \rfloor$  pixels for a filter size  $F$  (e.g., pad 1 pixel for a filter size of 3)
  - On each feature map, the response at (0, 0) has a receptive field centered at (0, 0) on the image
  - On each feature map, the response at (x, y) has a receptive field centered at (Sx, Sy) on the image (stride S)
- A general solution
$$i_L = g_L(i_L) = \alpha_L(i_L - 1) + \beta_L$$
$$\alpha_L = \prod_{p=1}^L S_p$$
$$\beta_L = 1 + \sum_{p=1}^L \left( \prod_{q=1}^{p-1} S_q \right) \left( \frac{F_p - 1}{2} - P_p \right)$$

See [Karel Lenc & Andrea Vedaldi]  
"R-CNN minus R", BMVC 2015.



138

- 对于上面 A simple solution：其实也是何凯明在SPP-net中采用的方法。其实就是巧妙的化简一下公式  $p_i = s_i \cdot p_{i+1} + ((k_i - 1)/2 - padding)$  令每一层的padding都为  $padding = \lfloor k_i/2 \rfloor \Rightarrow p_i = s_i \cdot p_{i+1} + ((k_i - 1)/2 - \lfloor k_i/2 \rfloor)$ 
  - 当  $k_i$  为奇数  $((k_i - 1)/2 - \lfloor k_i/2 \rfloor) = 0$  所以  $p_i = s_i \cdot p_{i+1}$
  - 当  $k_i$  为偶数  $((k_i - 1)/2 - \lfloor k_i/2 \rfloor) = -0.5$  所以  $p_i = s_i \cdot p_{i+1} - 0.5$
  - 而  $p_i$  是坐标值，不可能取小数 所以基本上可以认为  $p_i = s_i \cdot p_{i+1}$  。公式得到了化简：感受野中心点的坐标  $p_i$  只跟前一层  $p_{i+1}$  有关。
- 对于上面的 A general solution: 其实就是把 公式  $p_i = s_i \cdot p_{i+1} + ((k_i - 1)/2 - padding)$  级联消去整合一下而已。

A general solution

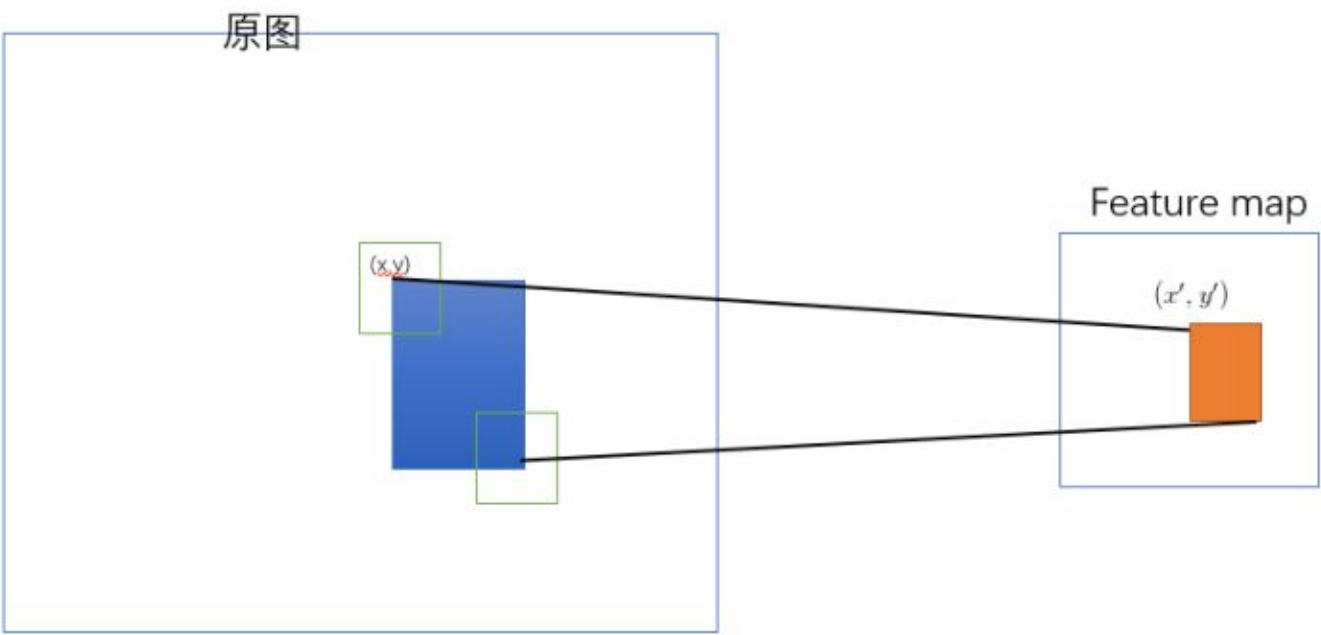
$$p_i = p_0(s_1 \cdot s_2 \cdot \dots \cdot s_i) + p_0((k_1 - 1)/2 - \lfloor k_1/2 \rfloor + \dots + (k_i - 1)/2 - \lfloor k_i/2 \rfloor)$$

$$p_0 = \prod_{j=1}^i s_j$$
$$p_0 = 1 + \sum_{j=1}^i \left( \prod_{k=1}^{j-1} s_k \right) \left( \frac{k_j - 1}{2} - \lfloor \frac{k_j}{2} \rfloor \right)$$

See [Karri Lenc & Andrea Vedaldi]  
"B-CHN minus R". BMVC 2015.

SPP-net的ROI映射做法详解

SPP-net 是把原始ROI的左上角和右下角 映射到 feature map上的两个对应点。 有了feature map上的两队角点就确定了 对应的 feature map 区域(下图中橙色)。



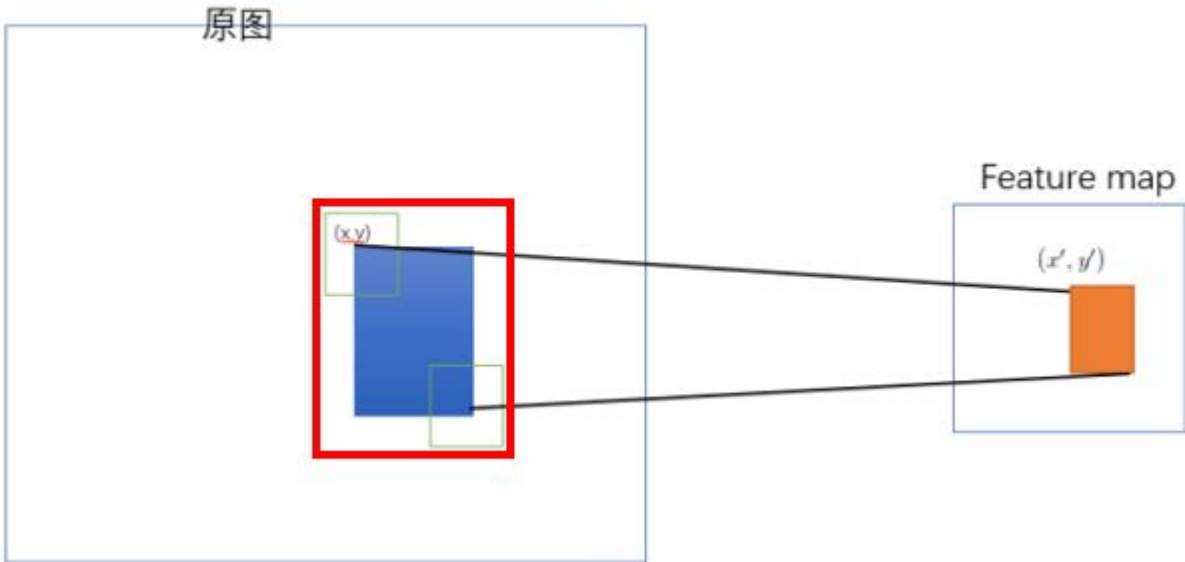
如何映射？

左上角的点 (x,y) 映射到 feature map上的  $(x', y')$ ：使得  $(x', y')$  在原始图上感受野（上图绿色框）的中心点 与 (x,y) 尽可能接近。

对应点之间的映射公式是啥？

- 就是前面每层都填充padding/2 得到的简化公式： $p_i = s_i \cdot p_{i+1}$
- 需要把上面公式进行级联得到  $p_0 = S \cdot p_{i+1}$  其中  $(S = \prod_0^i s_i)$
- 对于feature map 上的  $(x', y')$  它在原始图的对应点为  $(x, y) = (Sx', Sy')$
- 论文中的最后做法：把原始图片中的ROI映射为 feature map中的映射区域（上图橙色区域）其中 左上角取： $x' = \lfloor x/S \rfloor + 1, y' = \lfloor y/S \rfloor + 1$ ，；右下角的点取： 界取y'的x值： $x' = \lceil x/S \rceil - 1, y' = \lceil y/S \rceil - 1$ 。下图可见  $\lfloor x/S \rfloor + 1, \lceil x/S \rceil - 1$  的作用效果分别是增加和减少。也就是 左上角要向右下偏移，右下角要想要向左上偏移。个人理解采取这样的策略是因为论文中的映射方法（左上右下映射）会导致feature map上的区域反射回原始ROI时有多余的区域（下图左边红色框是比蓝色区域大的）

	X/S	$\lfloor x/S \rfloor + 1$	$\lceil x/S \rceil - 1$
有小数情况	1.2	1+1 = 2 （多 0.8）	2-1 = 1 （少 0.2）
无小数情况	2	2+1 = 3 （多 1）	2-1 = 1 （少 1）
		总是比X/S 多 0~1	总是比X/S 少 0~1





138

参考:

《Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition》

*Concepts and Tricks In CNN(长期更新)*

卷积神经网络物体检测之感受野大小计算 - machineLearning - 博客

iccv2015\_tutorial\_convolutional\_feature\_maps\_kaiminghe.

卷积神经网络(CNN)简介

编辑于 2017-01-20

深度学习 (Deep Learning) 卷积神经网络 (CNN) 机器学习

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

### 详解 roi-pooling 层的实现

y1n

### 李宏毅ML课程[1]Learning Map

忆臻 发表于机器学习算...

### Dict(hashtable, Map) implementation in python

简单粗暴，但是基本还原了python关于dict类的实现： open addressing, (random) probing (这里是quadratic), array resizing. yield 实现keys 和 values. \_\_author\_\_ = 'bozeng' ## this ...

曾博 发表于普通de世...

### map

第一幅很古典 2这幅以中国为中心2 中国人口与都市圈的“泡状图” 2古代地图都把中国画很大，相对于支离破碎的其他地区，一体的、一个大块的中国确实看似挺得天独厚，独得恩宠的。2可能是美国...

深具世界眼光



22 条评论

切换为时间排序

写下你的评论...

- 何志

1 年前

恩？ 还是没太明白， 意思是根据feature map反向寻找ROI？

1
- 晓雷 (作者) 回复 何志

1 年前

已经更新文章，在最后有详细解释。

1 查看对话
- 何志 回复 晓雷 (作者)

1 年前

我懂了， 其实就是有点像图像金字塔那样

1 查看对话
- 梁晓昱

1 年前

有个问题没看懂，请教：  
ri和pi  
这两个公式都是计算感受野的吧？  
第一个ri题主按照卷积反算我还比较明白；  
第二个pi的公式不明白，难道这两个公式不应该是一样（等价）的么？ 如果不是，为什么会不一样？  
谢谢

4
- Yiru Shen

1 年前

最近正在看这篇文章 看到feature map找对应位置也是看了好多遍都没懂 多谢分享！

赞
- 知乎用户

1 年前

答主，我有一个疑问。我觉得计算感受野时，与pad无关。所以公式里没有-2×padding

1
- 晓雷 (作者) 回复 知乎用户

1 年前

我自己理解是，计算感受野的公式是卷积公式，公式中的pad，在某些情况下，并非有padding，某些情况下，是padding，这取决于具体的实现。在计算感受野时，我们关心的是卷积核的覆盖范围，而不是padding的大小。所以公式里没有-2×padding。

138

 晓雷 (作者) 回复 梁晓昱

1 年前

pi所在公式是坐标点映射关系(映射感受野的中心点)，ri所在公式是感受野的尺度计算公式(映射感受野的长宽)。中心点和长宽就可以完全确定一个矩形的感受野。

pi代表坐标点。假设A特征图卷积到B特征图。那么此公式就可以计算 B上的某个特征点(x1,y1,w1,h1)到A上的某个感受野的中心点(x0,y0,w0,h0)。至于这个公式怎么来的。找了很久没找到推导过程。似乎可以从几何关系推导。

 1

 查看对话

 银发绅士

1 年前

博主你好，首先很感谢你这篇文章，给我带来了很大的收获。我这里有一个地方不是太明白，就是何凯明的那个PPT里边使用的通用公式(也就是他的PPT里边提及的文章“RCNN minus R”里的公式)。我用你的文章里的特殊值， $p_1=2*p_2+1$ ， $p_2=3*p_3+3$ ，以及 $p_1=6*p_3+7$ 这三个套那个公式，感觉不对。请问你有推导过吗，那个通用公式里的Beta\_L的计算式子感觉不大对劲

 赞

 教科书般的男人 回复 晓雷 (作者)

1 年前

个人理解：计算感受野的时候是不需要考虑padding的，因为计算感受野我们求得是后面一层1个神经元（或一个区域）与前一层的神经元或区域（）连接的范围，而补零是出现在边界的的，也就是说，单纯计算感受野是不需要考虑padding。在计算坐标映射的时候，这个时候就需要考虑padding了，因为有padding会导致坐标的变化，而计算感受野不管有没有padding，感受野只和上一层的卷积核和步长有关系，不管对输入的图片是否padding，其感受野的范围是不变的。

 10

 查看对话

 知乎用户 回复 教科书般的男人

1 年前

同意，我也是这样理解的，单纯计算感受野的大小时，不需要考虑padding。  
btw, 卷积神经网络物体检测之感受野大小计算 - machineLearning - 博客这个是我写的，哈哈

 赞

 查看对话

 乔豆麻袋

11 个月前

请问那个坐标映射的公式你怎么推到的？？我推倒的没有k-1而是k

 赞

 DaGgER14

10 个月前

还是想问一开始的那个公式里面offset是怎么计算的？


 赞

 水云天

10 个月前

1.第一个RF是3，第二个是5，第三个为什么是6  
2.如果是倒过去，第三个RF不应该通过pool得到？

 赞

 god boy

9 个月前

好像你引用的那个博客打不开了=\_=

 赞

 知乎用户 回复 晓雷 (作者)

6 个月前

不需要padding的。你的意思是直接对卷积过程逆推，而这样逆推出来的其实是输出map对应的输入map的大小，而不是输出map的感受野，相当于在感受野上做了减法。不管有没有padding，感受野都只是由kernel和stride决定。

你可以这么想，如果在外围加了padding，但是卷积时不使用padding，那么这时感受野是跟没加padding时是一样的。也就是说padding的作用在于调整输出map的大小，而感受野始终没有变化。

 赞

 查看对话

 王佳吉

6 个月前

我直觉地想象，在原图上某个区域，经过多次卷积后就被压缩成一个点。  
所以反过来，投射到最后一层feature map上的RoI，其上的角点，应该对应于原图片上的一片区域。

 赞

 知乎用户 回复 晓雷 (作者)

6 个月前

我个人感觉是这样的:  
 $s_i*p_{i+1}$ 项表示(i+1)层的第 $p_{i+1}$ (指在(i+1)层的坐标)的值所接受(i)层的感受野的坐标基数，  
( $k_i-1$ )/2项代表感受野中心  
padding决定(i)层的坐标偏移量

 赞

 查看对话

 吕归尘

5 个月前

假设S=16的话，原图上就不能有小于等于16\*16的ROI了吧？要不然就映射到feature map上就没了。这样是不是就没法检测小目标了？而且16\*16的目标对于imagenet的214\*214左右的图像来说感觉也不小了.....

 赞

 王思宇

5 个月前

感受野上面的坐标映射的问题

