# 我们从基于区域的目标检测器（更快的R-CNN，R-FCN，FPN）中学到了什么？

在本系列中，我们将全面介绍对象检测。在第一部分中，我们将介绍基于区域的目标检测器，包括快速R-CNN、快速R-CNN、R-FCN和FPN。在第二部分中，我们将研究单点探测器。在第3部分中，我们将讨论性能和一些实现问题。通过在一个环境中研究它们，我们研究什么是有效的，什么是重要的，以及哪里可以改进。希望通过研究我们是如何到达这里的，它能给我们更多关于我们前进方向的见解。

：我们从基于区域的对象检测器（更快的R-CNN、R-FCN、FPN）中学到了什么？

：我们从单点目标探测器（SSD、YOLO）、FPN和焦距损失中学到了什么？

：物体探测的设计选择、经验教训和趋势？

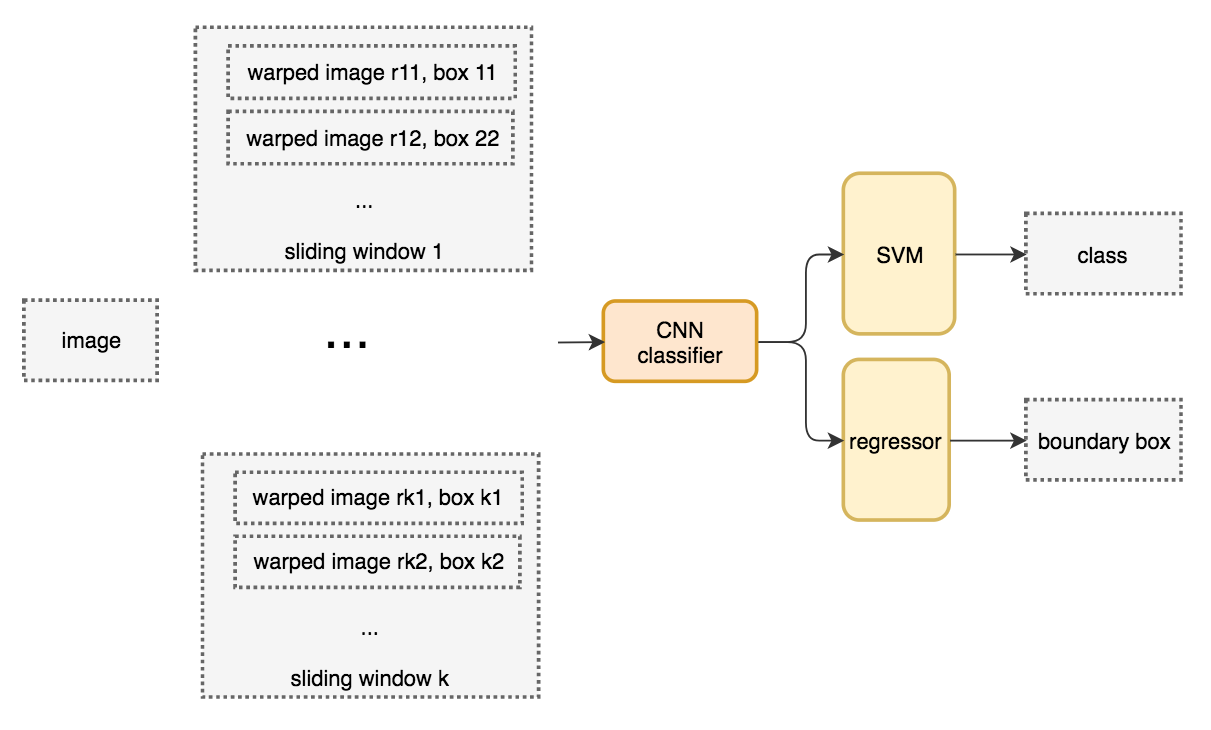
### 滑动窗口探测器

自从AlexNet赢得2012年ILSVRC挑战赛以来，CNN在分类领域的应用一直占据主导地位。一种强力的目标检测方法是从左到右滑动窗口，从上到下滑动窗口，使用分类来识别目标。为了在不同的视距下检测不同的对象类型，我们使用不同大小和长宽比的窗口。

我们根据滑动的窗户从图片上剪下补丁。由于许多分类器只获取固定大小的图像，因此这些面片是扭曲的。然而，这不应该影响分类精度，因为分类器被训练来处理扭曲的图像。



将变形后的图像块输入CNN分类器，提取4096个特征。然后应用支持向量机分类器对类进行识别，并对边界盒进行线性回归。



下面是伪代码。我们创建了许多窗口来检测不同位置的不同对象形状。为了提高性能，一个明显的解决方案是减少窗口的数量。

### 选择性搜索

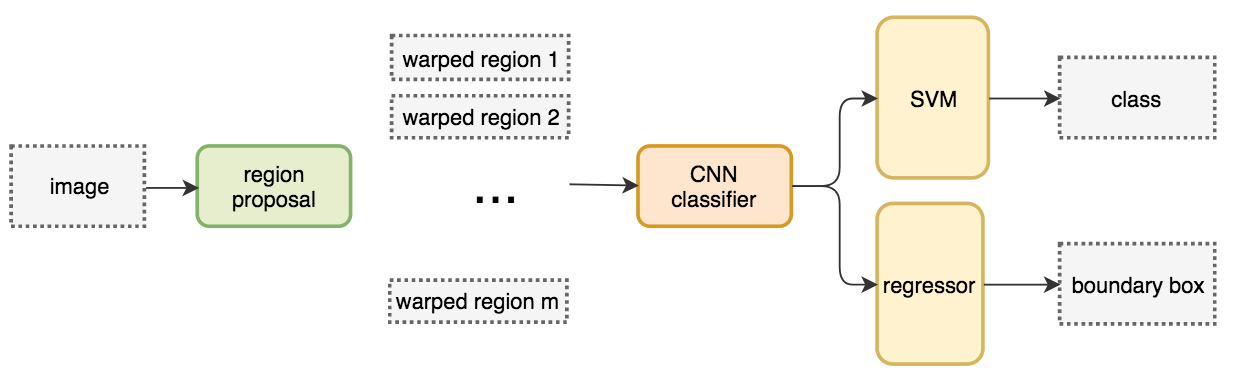
我们使用区域建议方法来建立感兴趣区域（roi）来侦测物体，而不是使用暴力方法。在选择性搜索（SS）中，我们从每个像素作为自己的组开始。接下来，我们计算每个组的纹理，并将最接近的两个组合并。但为了避免一个区域吞噬其他区域，我们更喜欢先将较小的组分组。我们继续合并区域，直到所有的东西都合并在一起。在下面的第一行中，我们展示了如何扩展区域，第二行中的蓝色矩形显示了合并过程中生成的所有可能的roi。



### 美国有线电视新闻网

R-CNN使用区域建议方法创建大约2000个ROI（感兴趣的区域）。这些区域被扭曲成固定大小的图像，并分别输入CNN网络。然后是完全连接的层，对对象进行分类并细化边界框。

这是系统流程。



与滑动窗口相比，R-CNN具有更少但更高质量的ROI，运行速度更快且更精确。

#### 边界盒回归器

区域建议方法计算量大。为了加快这个过程，我们经常选择一种成本较低的区域建议方法来创建roi，然后使用线性回归器（使用完全连接的层）来进一步细化边界框。

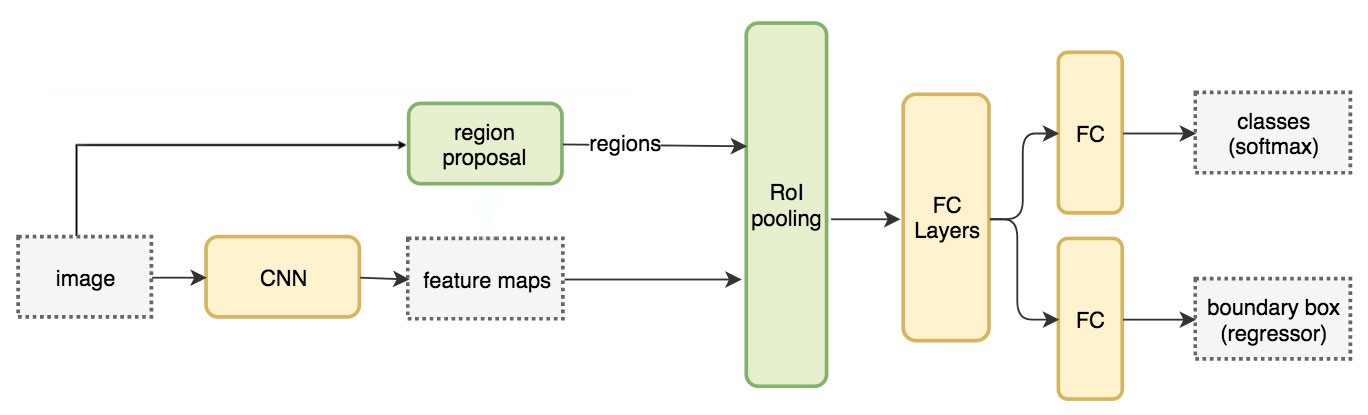


### 快速R-CNN

R-CNN需要许多提案才能准确，而且许多地区相互重叠。R-CNN训练和推理速度慢，如果我们有2000个方案，每个方案都由CNN单独处理，即对不同的roi重复2000次特征提取。

我们不是从头开始为每个图像补丁提取特征，而是先使用特征提取器（CNN）为整个图像提取特征。我们还使用外部区域建议方法，如选择性搜索，来创建roi，然后将其与相应的特征映射相结合，形成用于对象检测的面片。我们使用ROI池将补丁扭曲到一个固定的大小，并将它们提供给完全连接的层进行分类和定位（检测对象的位置）。通过不重复特征提取，快速R-CNN显著缩短了处理时间。

下面是网络流程：



在下面的伪代码中，昂贵的特征提取正在从for循环中移出，这是一个显著的速度改进，因为它是为所有2000个roi执行的。快速R-CNN的训练速度是R-CNN的10倍，推理速度是R-CNN的150倍。

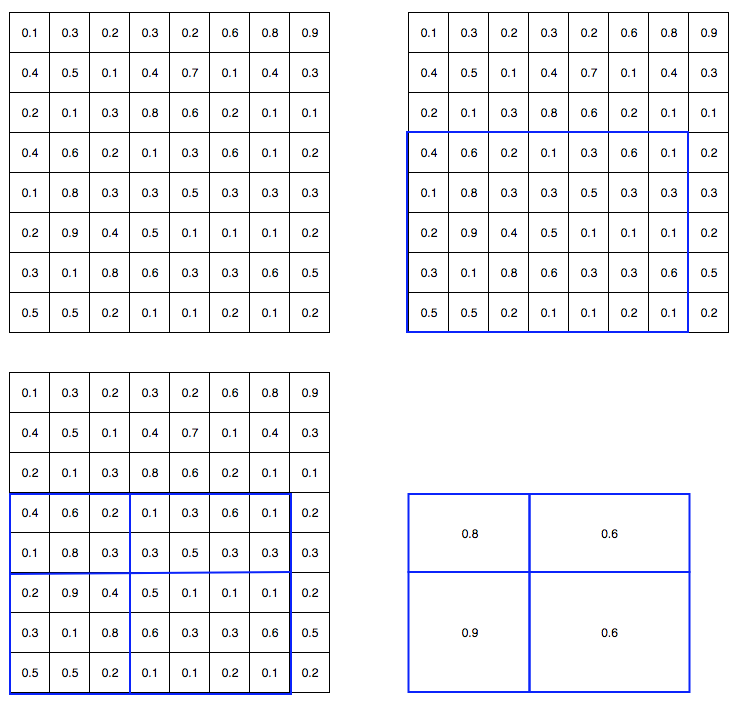
快速R-CNN的一个主要优点是，整个网络（特征抽取器、分类器和边界盒回归器）都是端到端训练的，具有多任务损失（分类损失和定位损失）。这样可以提高精确度。

投资回报池

因为Fast R-CNN使用完全连接的层，所以我们应用ROI池将可变大小的ROI扭曲成预定义的大小形状。

让我们通过将8×8特征映射转换为预定义的2×2形状来简化讨论。

* 左上角：我们的功能图。
* 右上：我们将ROI（蓝色）与功能图重叠。
* 左下角：我们将roi拆分为目标维度。例如，对于我们的2×2目标，我们将roi分成4个大小相似或相等的部分。
* 右下角：为每个部分找到最大值，结果是我们的扭曲特征图。

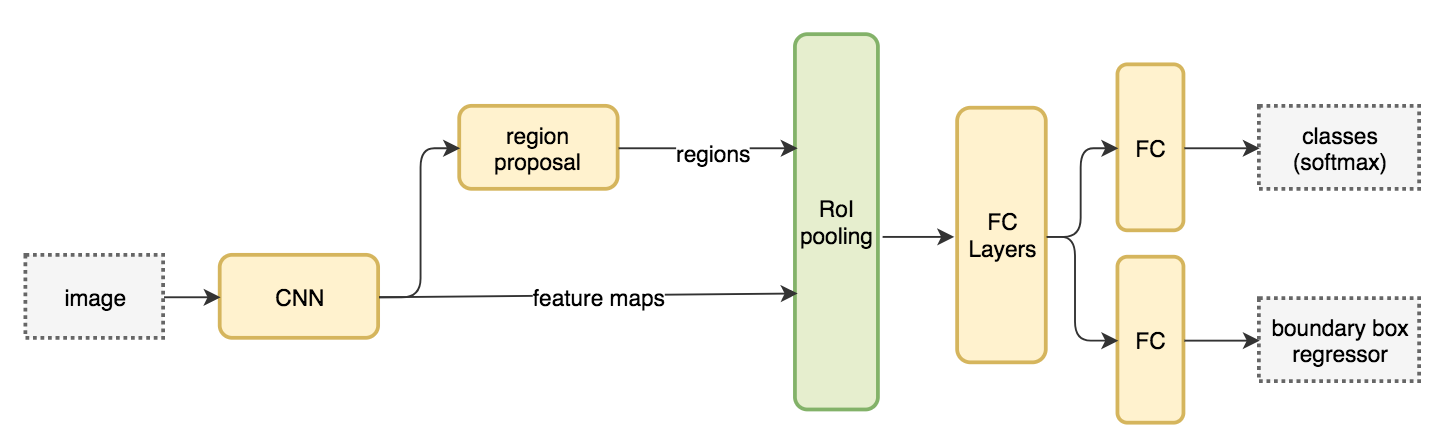


因此我们得到了一个2×2的特征块，我们可以将其输入到分类器和box回归器中。

### 更快的R-CNN

快速R-CNN依赖于一种外部区域提议方法，如选择性搜索。然而，这些算法运行在CPU上，而且速度很慢。在测试中，快速R-CNN需要2.3秒来做出预测，其中2秒用于生成2000个ROI。

快速R-CNN采用了与快速R-CNN相似的设计，只是它用一个内部的深层网络代替了区域建议方法，而roi是从特征映射中导出的新的区域建议网络（RPN）在生成roi时效率更高，每幅图像的运行时间为10ms。

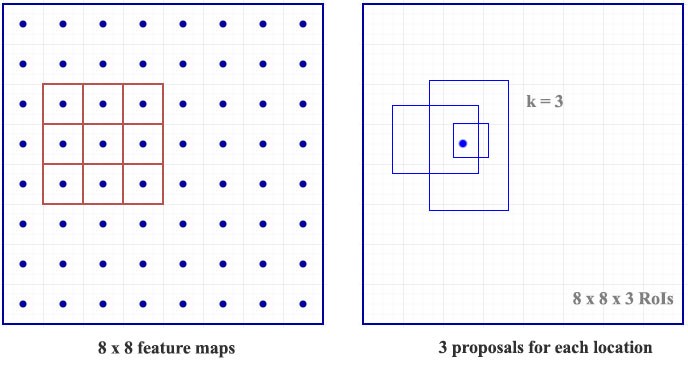


网络流是相似的，但是区域建议现在被卷积网络（RPN）取代。

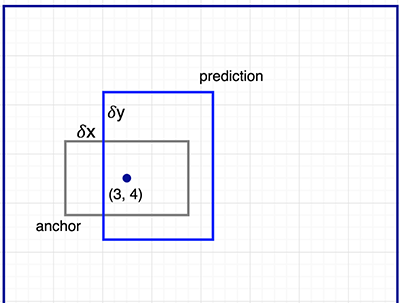
区域建议网络

区域建议网络（RPN）以第一卷积网络的输出特征映射为输入。它在特征映射上滑动3×3滤波器，使用类似于ZF网络的卷积网络（见下文）提出类不可知的区域建议。其他的深度网络如VGG或ResNet可以以牺牲速度为代价用于更全面的特征提取。ZF网络输出256个值，这些值被输入到2个独立的完全连接的层中，以预测一个边界框和2个对象得分。objectness度量框是否包含对象。我们可以使用一个回归器来计算一个单一的对象得分，但是为了简单起见，更快的R-CNN使用一个有两个可能类的分类器：一个用于“have an object”类别，另一个不用于（即背景类）。

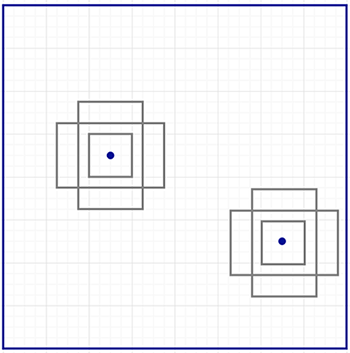
对于功能图中的每个位置，RPN进行k次猜测。因此，RPN为每个位置输出4×k坐标和2×k分数。下图显示了带有3×3滤波器的8×8特征映射，它总共输出8×8×3 roi（对于k=3）。右边的图表演示了由一个位置提出的3个建议。



在这里，我们得到3个猜测，稍后我们将完善我们的猜测。因为我们只需要一个是正确的，我们会更好，如果我们最初的猜测有不同的形状和大小。因此，更快的R-CNN不会提出随机的边界盒建议。相反，它预测了一些称为锚定的参考框左上角的偏移量，如𝛿x，𝛿y。我们限制这些偏移的值，因此我们的猜测仍然类似于锚。

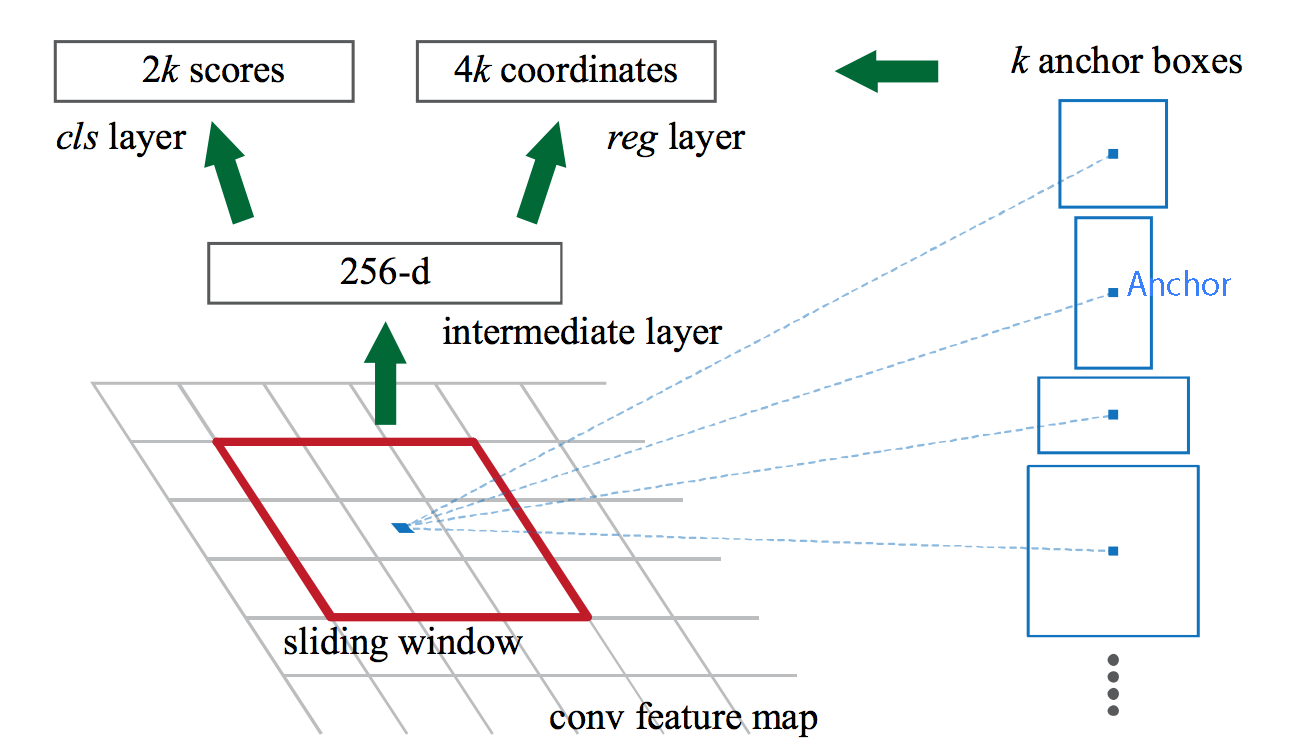


为了对每个位置进行k个预测，我们需要k个锚定在每个位置的中心。每个预测都与特定的锚关联，但不同的位置共享相同的锚形状。



这些锚是精心预选的，因此它们是多样的，并涵盖现实生活中的对象在不同的规模和长宽比合理。这将以更好的猜测来指导初始训练，并允许每个预测专门用于特定的形状。这种策略使早期训练更加稳定和容易。

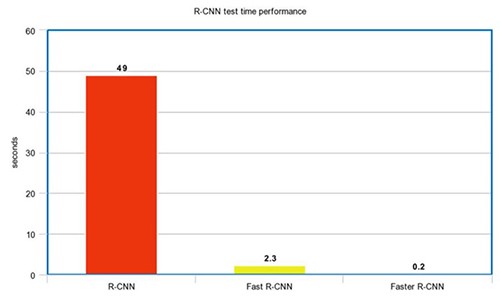
更快的R-CNN使用更多的锚。它部署了9个锚箱：3个不同的比例，3个不同的长宽比。每个位置使用9个锚定，生成2×9的对象分数和每个位置4×9的坐标。



在不同的论文中，锚也被称为prior或默认边界框。

### R-CNN方法的性能

如下图所示，更快的R-CNN甚至更快。



### 基于区域的全卷积网络（R-FCN）

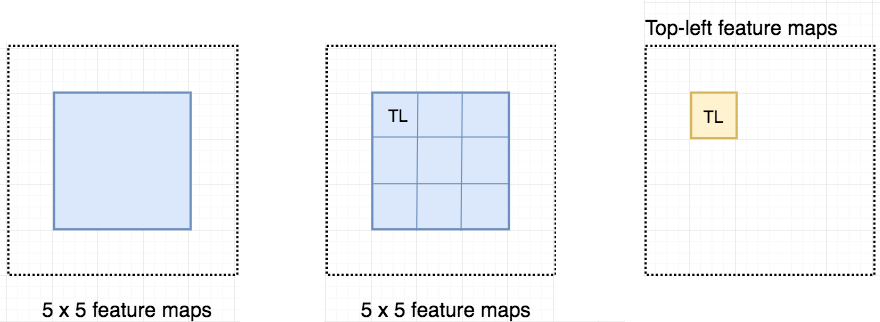
假设我们只有一个特征图来检测人脸的右眼。我们能用它定位一张脸吗？应该的。因为右眼应该在脸部照片的左上角，我们可以用它来定位脸部。

如果我们有专门用于检测左眼、鼻子或嘴巴的其他特征地图，我们可以将结果结合起来，更好地定位面部。

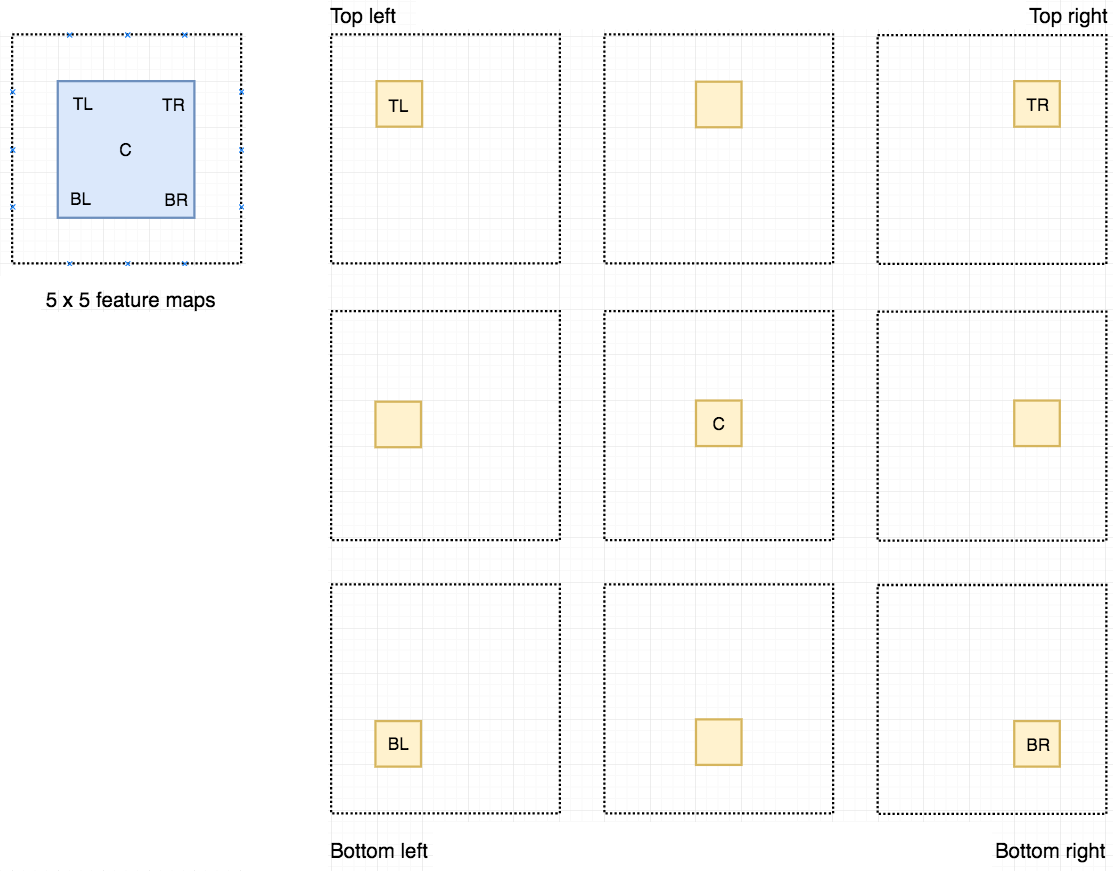
所以我们为什么要经历所有的麻烦。在更快的R-CNN中，探测器应用多个完全连接的层进行预测。有了2000个roi，它可能会很贵。

R-FCN通过减少每个ROI所需的工作量来提高速度。上述基于区域的特征图独立于ROI，可以在每个ROI之外计算。剩下的工作简单得多，因此R-FCN比R-CNN快。

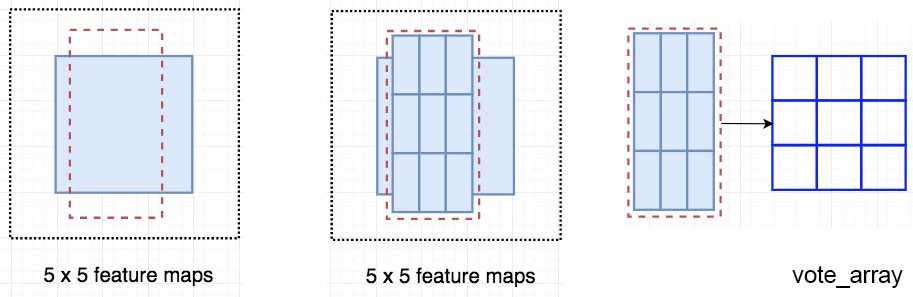
让我们考虑一个5×5特征图M，里面有一个蓝色的正方形对象。我们把正方形物体平均分成3×3个区域。现在，我们从M创建一个新的特征映射，只检测正方形的左上角（TL）。新的功能图看起来像下面右边的功能图。只有黄色网格单元[2，2]被激活。



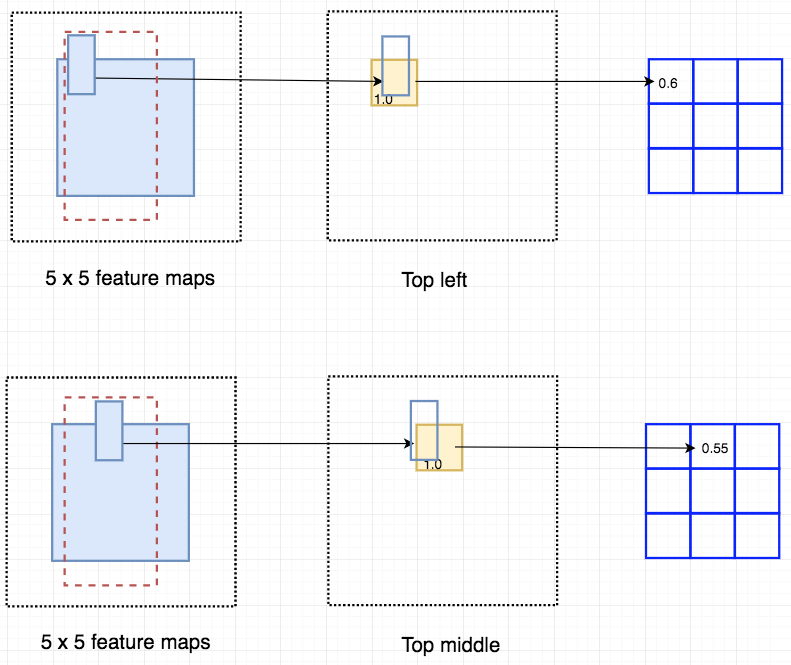
由于我们将正方形分为9部分，我们可以创建9个特征图，每个特征图检测对象的相应区域。这些特征映射称为位置敏感分数映射，因为每个映射检测（分数）对象的一个子区域。



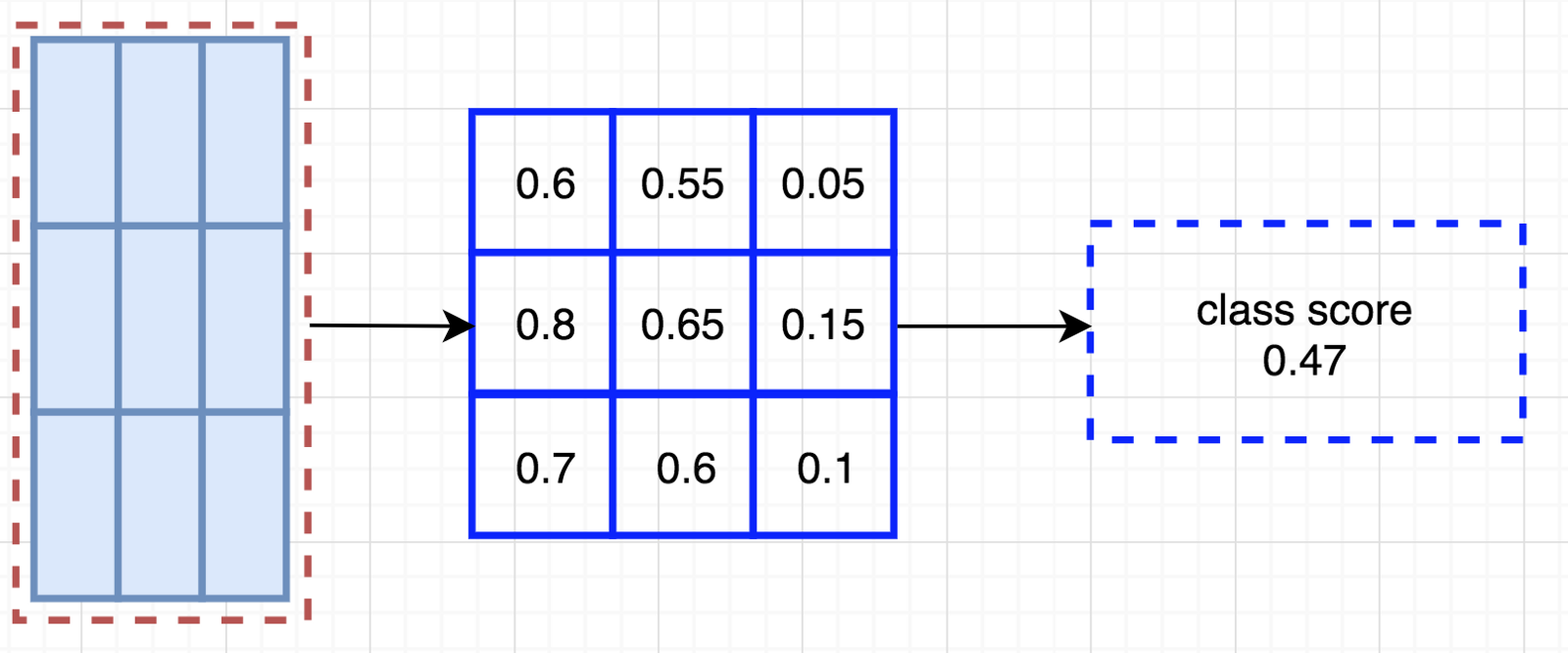
假设下面的红色虚线矩形是建议的ROI。我们将它分成3×3个区域，并询问每个区域包含对象相应部分的可能性。例如，左上方的ROI区域包含左眼的可能性有多大。我们将结果存储到下图中的3×3投票数组中。例如，vote\_数组[0][0]包含是否找到正方形对象左上区域的分数。



将分数图和ROI映射到投票数组的过程称为位置敏感ROI池。这个过程非常接近我们之前讨论过的ROI池。我们不会进一步讨论，但您可以参考未来阅读部分了解更多信息。



在计算位置敏感的ROI池的所有值之后，类得分是其所有元素的平均值。



假设我们要检测C类。我们将它扩展到C+1类，这样就包含了一个新的背景类（非对象）。每个班级都有自己的3×3分数图，因此总共有（C+1）×3×3分数图。使用它自己的一组分数图，我们为每个类预测一个类分数。然后对这些分数应用softmax来计算每个类的概率。

以下是数据流。在我们的例子中，下面有k=3。

### 我们到目前为止的旅程

我们从基本的滑动窗口算法开始。

然后我们尝试减少窗口的数量，并将尽可能多的工作移到for循环之外。

在中，我们将进一步完全移除for循环。单镜头探测器在单镜头中进行目标检测，不需要单独的区域建议步骤。

### 进一步阅读FPN，R-FCN和Mask R-CNN

FPN和R-FCN都比我们在这里描述的要复杂。如需进一步研究，请参阅：

* [Feature Pyramid Networks (FPN) for object detection.](https://medium.com/@jonathan_hui/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c)
* .

[**Image segmentation with Mask R-CNN** In a previous article, we discuss the use of region based object detector like Faster R-CNN to detect objects. Instead…medium.com](https://medium.com/@jonathan_hui/image-segmentation-with-mask-r-cnn-ebe6d793272)

### 资源

：Facebook Research使用Caffe2实现更快的R-CNN和屏蔽R-CNN。

在MATLAB中的正式实现。

在TensorFlow中实现。

在MXNet中实现。

在Caffe和MATLAB中的实现。

[R-FCN](https://github.com/xdever/RFCN-tensorflow) implementation in TensorFlow.