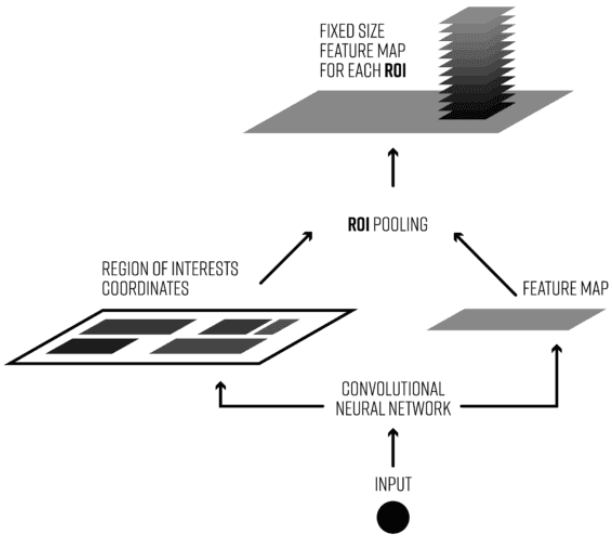
ROI Pooling 原理及实现

目标检测 architecture 通常可以分为两个阶段：
（1）region proposal：给定一张输入 image 找出 objects 可能存在的所有位置。这一阶段的输出应该是一系列 object 可能位置的 bounding box。这些通常称之为 region proposals 或者 regions of interest（ROI），在这一过程中用到的方法是基于滑窗的方式和 selective search。
（2）final classification：确定上一阶段的每个 region proposal 是否属于目标一类或者背景。



这个 architecture 存在的一些问题是：

产生大量的 region proposals 会导致 performance problems，很难达到实时目标检测。
在处理速度方面是 suboptimal。
无法做到 end-to-end training。
这就是 ROI pooling 提出的根本原因，ROI pooling 层能实现 training 和 testing 的显著加速，并提高检测 accuracy。该层有两个输入：

从具有多个卷积核池化的深度网络中获得的固定大小的 feature maps；
一个表示所有 ROI 的 N*5 的矩阵，其中 N 表示 ROI 的数目。第一列表示图像 index，其余四列表示其余的左上角和右下角坐标；

ROI pooling 具体操作如下：

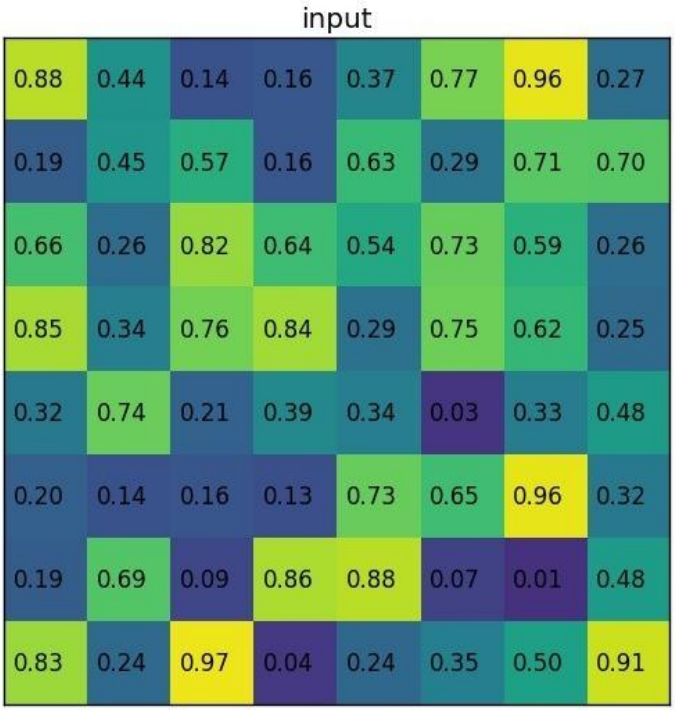
根据输入 image，将 ROI 映射到 feature map 对应位置；
将映射后的区域划分为相同大小的 sections（sections 数量与输出的维度相同）；
对每个 sections 进行 max pooling 操作；
这样我们就可以从不同大小的方框得到固定大小的相应 的 feature maps。值得一提的是，输出的 feature maps 的大小不取决于 ROI 和卷积 feature maps 大小。ROI pooling 最大的好处

就在于极大地提高了处理速度。

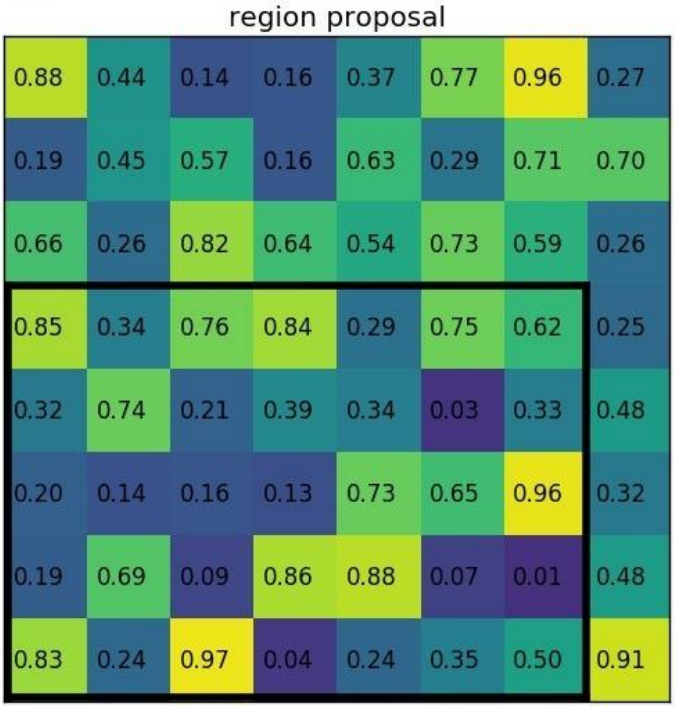
ROI pooling example

我们有一个 8*8 大小的 feature map，一个 ROI，以及输出大小为 2*2.

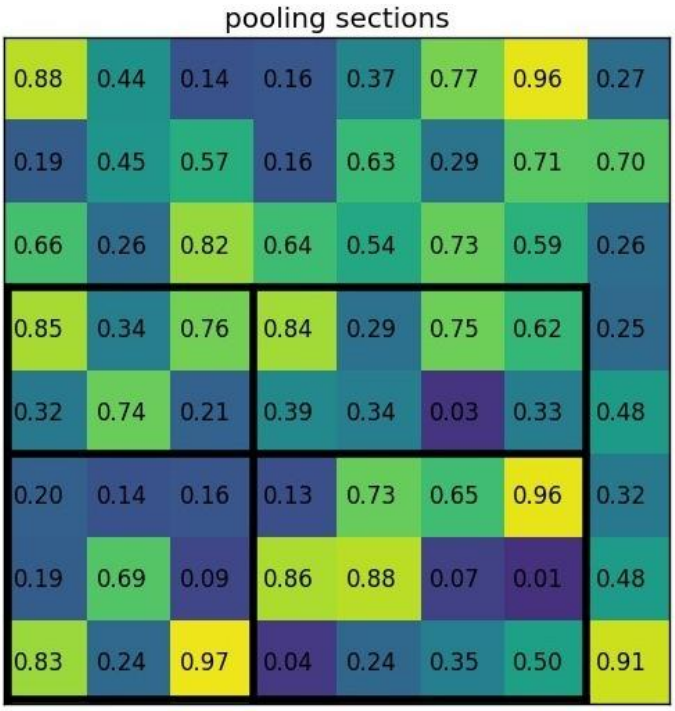
1.输入的固定大小的 feature map



2. region proposal 投影之后位置（左上角，右下角坐标）:（0，3），（7，8）



3. 将其划分为（2*2）个 sections（因为输出大小为 2*2），我们可以得到：



4. 对每个 section 做 max pooling，可以得到：

ROI Pooling 就是将大小不同的 feature map 池化成大小相同的 feature map，利于输出到下一层网络中。