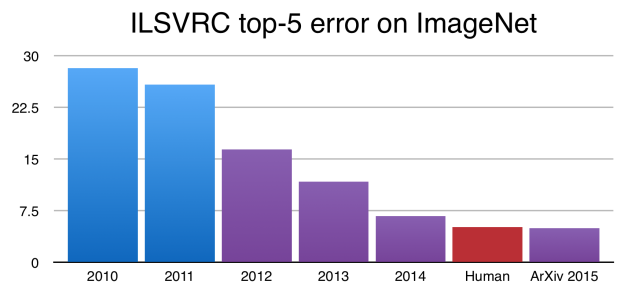
**图像分割发展史 mask rcnn**

- by 小韩

（来源： <https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4> ）

* [图像分割发展史 mask rcnn](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E5%9B%BE%E5%83%8F%E5%88%86%E5%89%B2%E5%8F%91%E5%B1%95%E5%8F%B2-mask-rcnn)
  + [本文目标](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E6%9C%AC%E6%96%87%E7%9B%AE%E6%A0%87)
    - [2014: R-CNN CNNs 在目标检测中的早期应用](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#2014-r-cnn-cnns-%E5%9C%A8%E7%9B%AE%E6%A0%87%E6%A3%80%E6%B5%8B%E4%B8%AD%E7%9A%84%E6%97%A9%E6%9C%9F%E5%BA%94%E7%94%A8)
    - [理解 R-CNN](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E7%90%86%E8%A7%A3-r-cnn)
    - [改进边框](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E6%94%B9%E8%BF%9B%E8%BE%B9%E6%A1%86)
  + [2015: Fast R-CNN - 更快更简单的 R-CNN](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#2015-fast-r-cnn-%E6%9B%B4%E5%BF%AB%E6%9B%B4%E7%AE%80%E5%8D%95%E7%9A%84-r-cnn)
    - [Fast R-CNN 第一个思想：RoI (Region of Interest) Pooling](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#fast-r-cnn-%E7%AC%AC%E4%B8%80%E4%B8%AA%E6%80%9D%E6%83%B3roi-region-of-interest-pooling)
    - [Fast R-CNN 第二个思想：结合所有的模型到一个网络中](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#fast-r-cnn-%E7%AC%AC%E4%BA%8C%E4%B8%AA%E6%80%9D%E6%83%B3%E7%BB%93%E5%90%88%E6%89%80%E6%9C%89%E7%9A%84%E6%A8%A1%E5%9E%8B%E5%88%B0%E4%B8%80%E4%B8%AA%E7%BD%91%E7%BB%9C%E4%B8%AD)
  + [2016: Faster R-CNN - 加速候选区域](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#2016-faster-r-cnn-%E5%8A%A0%E9%80%9F%E5%80%99%E9%80%89%E5%8C%BA%E5%9F%9F)
    - [候选区域是怎样生成的](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E5%80%99%E9%80%89%E5%8C%BA%E5%9F%9F%E6%98%AF%E6%80%8E%E6%A0%B7%E7%94%9F%E6%88%90%E7%9A%84)
  + [2017: Mask R-CNN - Faster R-CNN 扩展到像素级分割](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#2017-mask-r-cnn-faster-r-cnn-%E6%89%A9%E5%B1%95%E5%88%B0%E5%83%8F%E7%B4%A0%E7%BA%A7%E5%88%86%E5%89%B2)
    - [RoiAlign - 调整 RoIPool 使更精确](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#roialign-%E8%B0%83%E6%95%B4-roipool-%E4%BD%BF%E6%9B%B4%E7%B2%BE%E7%A1%AE)
  + [代码](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E4%BB%A3%E7%A0%81)
    - [Faster R-CNN](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#faster-r-cnn)
    - [Mask R-CNN](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#mask-r-cnn)
  + [展望](file:///C:\Program%20Files\Microsoft%20VS%20Code\resources\app\out\vs\workbench\parts\webview\electron-browser\webview.html#%E5%B1%95%E6%9C%9B)

自从Alex Krizhevsky、Geoff Hinton 和 Ilya Sutskever 在2012年赢得 ImageNet 挑战赛以来，卷积神经网络（CNN）已经成为图像分类的黄金标准。 事实上从那时起，CNN已经改善到现在在ImageNet挑战中胜过人类的程度！



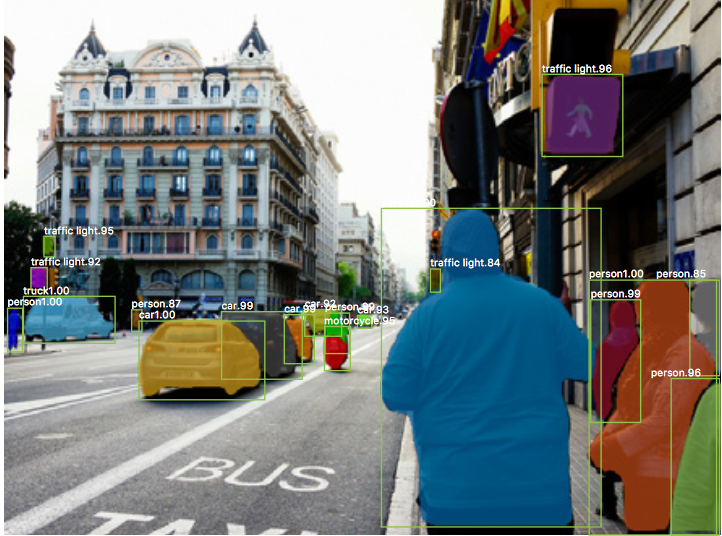
虽然这些结果令人印象深刻，但图像分类远比人类视觉理解的复杂性和多样性简单得多。



在分类中，通常有一个图像，其中有单个目标作为焦点，任务是说明该目标是什么。但是当我们观察周围的世界时我们会执行更复杂的任务。



我们看到的景象有许多重叠的目标和不同的背景，我们不仅要对这些不同的目标分类，还要确定它们之间的界限，差异和关系。



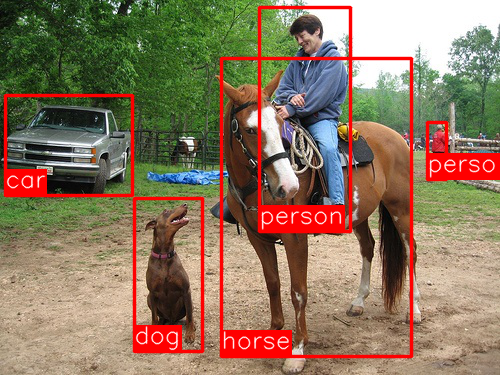
CNN 可以帮助我们完成这么复杂的任务吗？ 也就是说，给定一个更复杂的图像，我们可以使用 CNN 来识别图像中的不同目标及其边界吗？ 正如 Ross Girshick 和他的同事在过去几年所表明的那样，答案是肯定的。

**本文目标**

通过这篇文章，我们将介绍在目标检测和分割中使用的一些主要技术背后的原理，并了解它们是如何从一个实现发展到下一个的。 特别的，我们将介绍 R-CNN（Regional CNN），CNNs 的原始应用，以及它的后代 Fast R-CNN 和 Faster R-CNN。 最后，我们将介绍Facebook Research 发布的一篇文章 Mask R-CNN，该文章对这种目标检测技术进行了扩展以提供像素级的分割。 下面是本文中引用的论文：

1. R-CNN: <https://arxiv.org/abs/1311.2524>
2. Fast R-CNN: <https://arxiv.org/abs/1504.08083>
3. Faster R-CNN: <https://arxiv.org/abs/1506.01497>
4. Mask R-CNN: <https://arxiv.org/abs/1703.06870>

**2014: R-CNN CNNs 在目标检测中的早期应用**



受多伦多大学 Hinton 实验室研究的启发，由加州大学伯克利分校的 Jitendra Malik 教授领导的小团队开始探索一个在今天看来是一个不可避免的问题：

[Krizhevsky 等人的结果]可以在多大程度上推广到目标检测？

目标检测的任务是在图像中查找不同的目标并对其进行分类（如上图所示）， 由 Ross Girshick ，Jeff Donahue 和 Trevor Darrel 组成的团队发现，Krizhevsky 的结果可以解决这个问题, 并通过 PASCAL VOC Challenge 的测试，这是一种类似于 ImageNet 的目标检测挑战。 他们写道：

本文首次表明，与基于简单 HOG 类功能的系统相比，CNN 可以在 PASCAL VOC 上实现更高的目标检测性能。

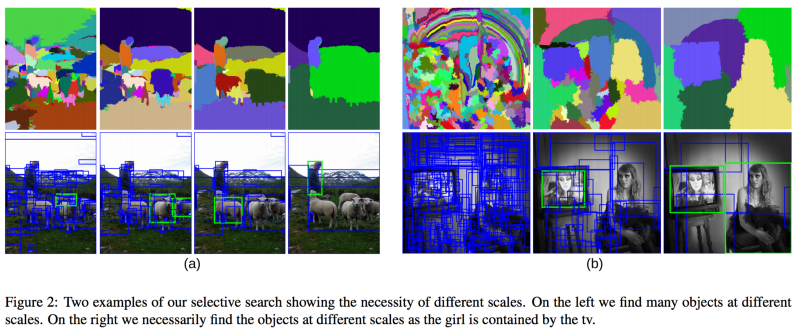
现在来了解他们的架构，Regions With CNNs（R-CNN）是怎样工作的。

**理解 R-CNN**

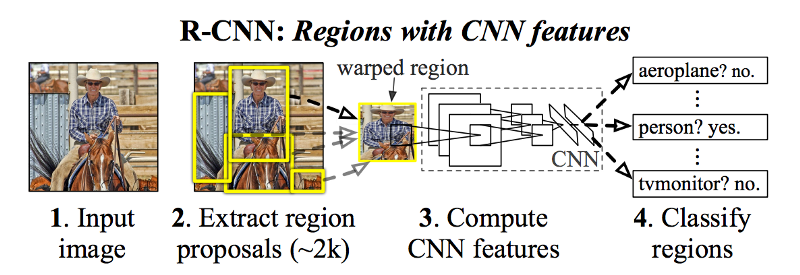
R-CNN 的目标是获取图像，并正确识别图像中主要目标（用边框（bounding box）表示）的位置。

* **输入：** 图像
* **输出：** 图像中每个目标的边界框（bounding box）和标签（label）。

但是我们如何找出这些边界框的位置？ R-CNN 按照我们的直觉来做做 -- 首先在图像中标出许多边界框，然后判断每个边界框中是否实际对应一个目标。



R-CNN 使用称为选择性搜索（Selective Search）的方法创建这些边界框或候选区域。 在较高的层次上，选择性搜索（如上图所示）通过不同大小的窗口查看图像，并且对于每个尺寸，尝试通过纹理，颜色或强度将相邻像素组合在一起以识别目标。



一旦创建了一些候选区域，R-CNN 就会该区域变为标准的方形大小，并将其传递给修改过的 AlexNet（2012 年 ImageNet 的获奖提交），如上图所示。

在 CNN 的最后一层，R-CNN 增加了一个支持向量机（SVM），它简单地判断这是否是一个目标，如果是的话，是什么目标。 见上图中的第 4 步。

**改进边框**

现在，在边框内找到了这个目标，我们可以缩小边界框到目标的实际大小吗？ 答案是可以，这就是 R-CNN 的最后一步。R-CNN 对候选区域进行简单的线性回归，生成更紧密的边界框坐标获得最终结果。 以下是这个回归模型的输入和输出：

* **输入：** 图像相应目标的子区域
* **输出：** 子区域中新的目标边界框

总结一下，R-CNN 有以下几个步骤：

1. 生成一系列的候选边框。
2. 将边框中的图像输入预先训练的 AlexNet，最后通过 SVM 确认边界框中是什么目标。
3. 如果图像中有目标，就将边框中图像输入线性回归模型，输出更紧密的边界框坐标。

**2015: Fast R-CNN - 更快更简单的 R-CNN**



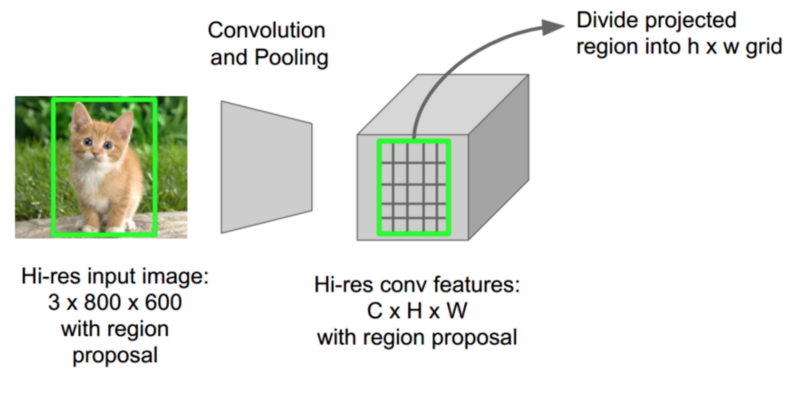
R-CNN 运行的挺好，但是比较慢，有下面的原因：

1. 每张图像的每个候选区域都要输入到 CNN(AlexNet)中（每个图像大约有2000个！）。
2. 需要单独训练三个不同的模型: 生成图像特征的 CNN ，预测目标类别的分类器，生成更紧密边界框的回归模型。这使得模型非常难以训练。

在2015年，R-CNN 的第一作者 Ross Girshick 解决了这两个问题，提出了只有短暂历史的第二个算法 - Fast R-CNN。 回顾一下它的主要思想。

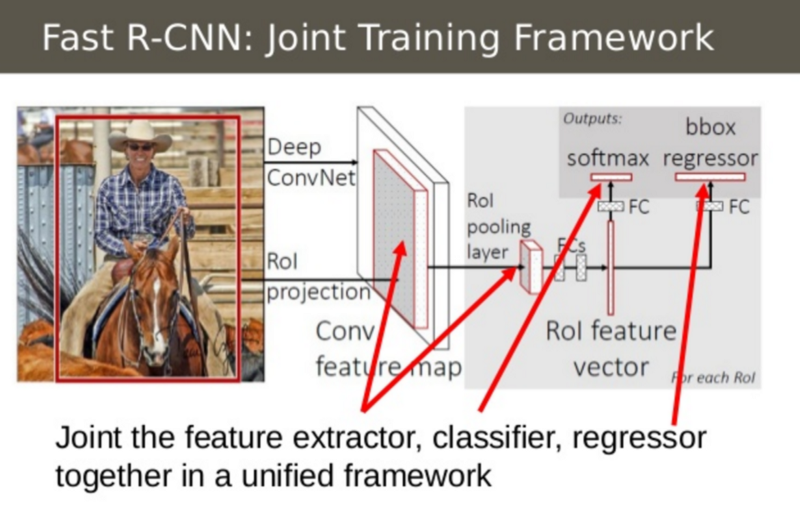
**Fast R-CNN 第一个思想：RoI (Region of Interest) Pooling**

Girshick 意识到 CNN 中每一张图片有许多重复的候选区域，因此有很多重复的 CNN 计算（约2000次）。他的想法很简单——为什么不让每张图片只做一次 CNN 计算然后找一个方法使这约 2000 个候选区域共享计算结果？



这正是 Fast R-CNN 使用的称为 RoIPool（Region of Interest Pooling）所做的事情。 RoIPool 的核心就是让候选区域分享 CNN 的结果。 在上图中，每个区域的 CNN 特征都是通过从 CNN 特征图选择相应的区域来获得的。 然后每个区域再经过池化（通常是最大池化）。 所以我们只需要计算一次原始图像而不是之前的2000次！

**Fast R-CNN 第二个思想：结合所有的模型到一个网络中**



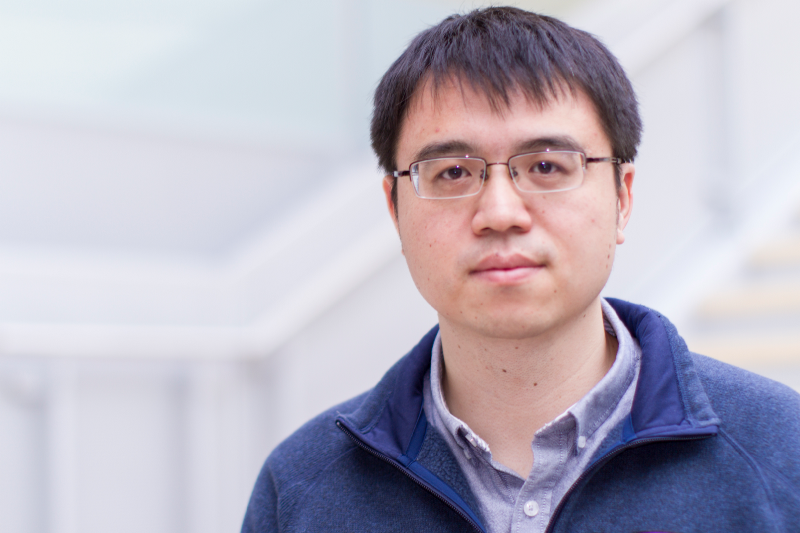
Fast R-CNN 第二个思想就是将 CNN，分类器和边界框回归器放在一个模型中。相比之前的三个不同的模型，图像特征（CNN）、分类器（SVM）、边界框（回归），Fast R-CNN 只使用了一个网络计算。

可以在上图中看到是怎样完成的。 Fast R-CNN 用 SVM 分类器替换原本 CNN 顶部的 softmax 层。 它还添加了一个与 softmax 层平行的线性回归层用来输出边界框坐标。 这样，所需的所有输出都来自于一个网络！ 下面是这个整体模型的输入和输出：

* **输入：** 有候选区域的图像
* **输出：** 每个区域的目标分类和更紧密的边界框。

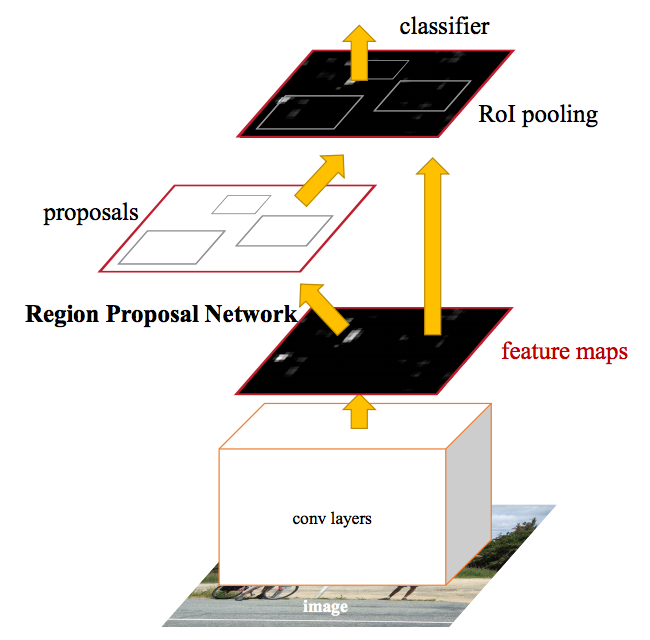
**2016: Faster R-CNN - 加速候选区域**

即使有了这些进步，Fast R-CNN 的过程仍然存在一个瓶颈 — 候选区域。 正如之前看到的，检测目标位置的第一步是生成许多潜在的边界框或感兴趣区域进行测试。 在 Fast R-CNN 中，这些区域是使用选择性搜索（Selective Search）创建的，这是一个相当缓慢的过程，是整个过程的瓶颈。



在2015年中期，由 Shaoqing Ren，Kaiming He，Ross Girshick 和 Jian Sun 组成的微软研究团队找到了一种方法，他们称为 Faster R-CNN 的架构，使生成候选区域几乎不花费额外时间。

Faster R-CNN 的思想是，候选区域取决于已经通过 CNN 计算的图像的特征（分类的第一步）。 生成候选区域时为什么不重用 CNN 计算结果而要单独运行选择性搜索算法呢？



实际上，这正是 Faster R-CNN 团队所取得的成就。 在上图中，可以看到单次 CNN 计算是怎样得到候选区域和分类。 这样，只需计算一次 CNN 就可以获得候选区域！ 作者写道：

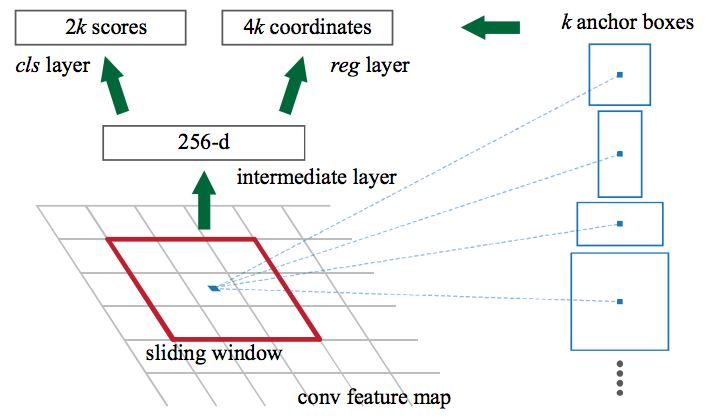
我们的观察结果是，基于区域的探测器（如Fast R-CNN）使用的卷积特征图也可用于生成候选区域（几乎无成本）。

模型的输入和输出：

* **输入：** 图像（不需要候选区域）。
* **输出：** 图像中目标的分类和边界框坐标。

**候选区域是怎样生成的**

让我们看看 Faster R-CNN 是怎样生成候选区域的。 Faster R-CNN 在 CNN 的特征上面增加了一个全卷积网络，也就是候选区域网络（Region Proposal Network）。



候选区域网络在 CNN 特征图上滑动一个窗口，每个窗口输出 k 个可能的边界框并预测每个边界框的好坏程度，打一个分数。 这 k 个边界框代表什么？



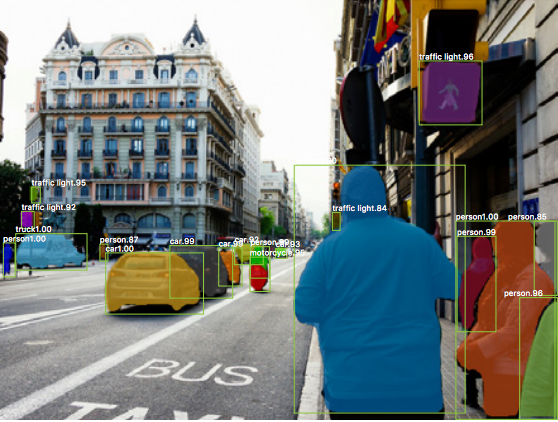
直观上，图像中的目标应该适合某些常见的长宽比和大小。 例如，如果想要一些类似于人类形状的矩形盒子就不会看到很多非常薄的框。 用这种方式创建 k 个这样常见的长宽比的框，称之为 anchor boxes。 对于每个 anchor boxes，输出一个边界框并对图像中的每个位置打分。

候选区域网络的输入和输出：

* **输入：** CNN 特征图。
* **输出：** 每个 anchor 一个边界框。 一个该边界框中有目标的可能性的分数。

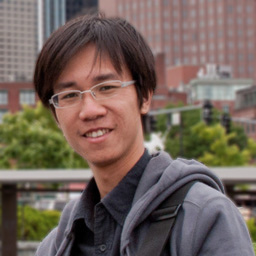
然后将每个可能有目标的边界框传递到 Fast R-CNN，生成分类和更小的边界框。

**2017: Mask R-CNN - Faster R-CNN 扩展到像素级分割**

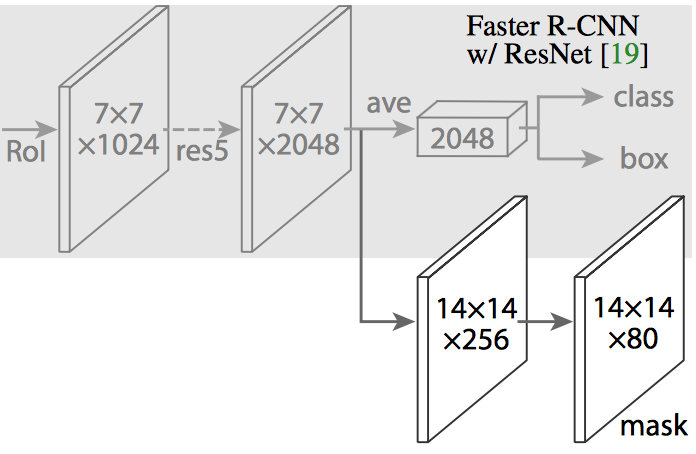


到目前为止，我们看到了以多种方式使用 CNN 来有效地定位图像中带有边界框的的不同目标。

我们能否进一步扩展这样的技术来定位每一个目标的像素而不仅仅只是一个边界框？这个问题就是图像分割，是 Kaiming He 和包括 Girshick，Facebook AI 的研究团队使用的 Mask R-CNN 结构。



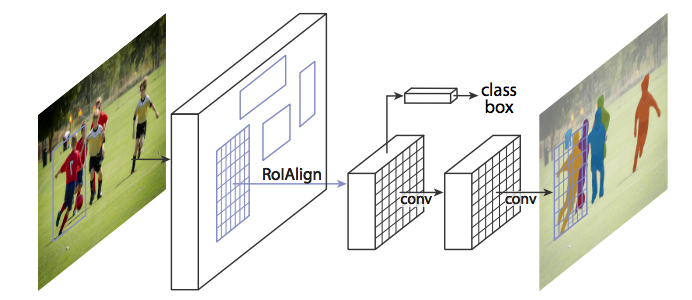
像 Fast R-CNN 和 Faster R-CNN 一样，Mask R-CNN 直观上很直接。鉴于 Faster R-CNN 在目标检测方面的效果非常好，我们是否可以将它扩展到像素级分割？



Mask R-CNN 通过向 Faster R-CNN 添加分支来完成此操作，该分支输出二进制掩码（binary mask），该掩码表示这个像素是否是该目标的一部分。 如前所述，分支（上图中的白色）只是基于 CNN 的特征映射之上的全卷积网络。 以下是其输入和输出：

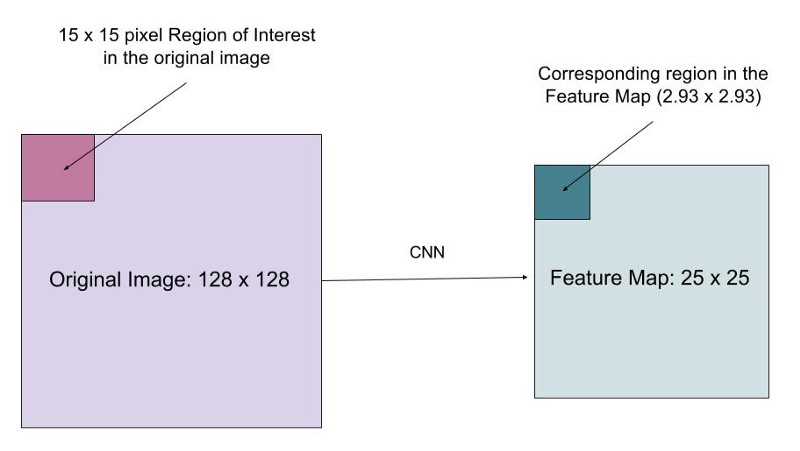
* **输入：** CNN 特征图
* **输出：** 像素在属于目标的所有位置上是 1 并且在其他位置是 0（称为二进制掩码）的矩阵。

**RoiAlign - 调整 RoIPool 使更精确**



当运行在原始的没有修改的 Faster R-CNN 上时，Mask R-CNN 的作者意识到由 RoIPool 选择的特征图的区域与原始图像的区域对应略微不准确。 与边界框不同，图像分割需要像素级的特性，这自然会导致不准确。

作者巧妙地通过调整 RoIPool 来解决这个问题，使用称为 RoIAlign 的方法使对齐更精确。



想象一下，我们有一个大小为 128 \* 128 的图像和一个大小为 25 \* 25 的特征图。 我们想要的特征区域对应于原始图像中左上角的 15 \* 15 的像素（见上图）。 我们怎样从要素图中选择这些像素？

原始图像中的每个像素对应于特征图中的 ~25/128 像素。 要从原始图像中选择15个像素，我们只从特征图中选择 15 \* 25 / 128 ~= 2.93 个像素。

在 RoIPool 中，我们向下舍入只选择2个像素，导致轻微的错位。 但是，在 RoIAlign 中，我们不使用舍入。 相反，我们用双线性插值法来准确还原 2.93 个像素对应原图像的内容。 这在很大程度上避免了 RoIPool 引起的错位。

一旦生成了这些掩码，Mask R-CNN 将掩码和 Faster R-CNN 生成的分类和边界框组合在一起，生成更加精确的分割：



**代码**

如果您有兴趣了解这些算法，这里有相关的代码：

**Faster R-CNN**

* Caffe: <https://github.com/rbgirshick/py-faster-rcnn>
* PyTorch: <https://github.com/longcw/faster_rcnn_pytorch>
* MatLab: <https://github.com/ShaoqingRen/faster_rcnn>

**Mask R-CNN**

* PyTorch: <https://github.com/felixgwu/mask_rcnn_pytorch>
* TensorFlow: <https://github.com/CharlesShang/FastMaskRCNN>

**展望**

在短短3年时间里，我们已经看到研究界如何从Krizhevsky 等人的原始成果到 R-CNN，最后一直到 Mask R-CNN 这样强大的成果。 孤立地看，像 Mask R-CNN 这样的成果看起来像天才般难以置信的飞跃是无法达到的。 然而，通过这篇文章，我希望你看到这些进步是通过多年的努力和协作缓慢实现的。 R-CNN，Fast R-CNN，Faster R-CNN 以及最终的 Mask R-CNN 提出的每个想法都不一定是质的跳跃，但它们的结合已经产生了非常显著的结果，更接近人类视力的水平。

让我特别兴奋的是，R-CNN 和 Mask R-CNN 之间的时间只有三年！ 通过不断增加的关注和支持，未来三年计算机视觉是否能够进一步提升？