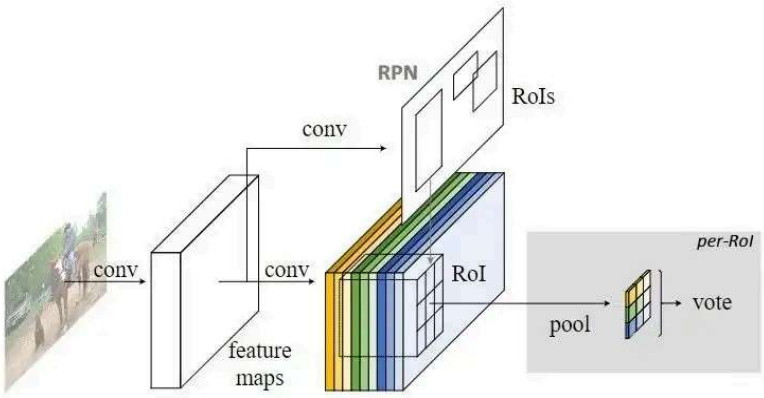


R-FCN

巴纳纳尔岛主很忙
好好学习哇

收录于 • CVpapers ›

> R-FCN：Object Detection via Region-base Fully Convolutional Network
>
> arxiv.org/pdf/1605.0640...

摘要

我们提出了基于区域的全卷积网络，用于准确高效的目标检测。有别于先前基于区域的检测器，例如Fast/Faster R-CNN，每个区域提议框都将分别计算，我们的基于区域的检测器几乎是共享所有计算的全卷积网络。为了实现这个目标，我们提出了一个位置敏感得分图来摆脱图像分类的平移不变性和目标检测的平移同变性的之间的困境。因此，我们的方法可以自然地使用全卷积的图像分类骨干网络用于目标检测。使用ResNet-101的R-FCN在VOC 2007上获得了83.6%的mAP，同时，每张图像测试时间为170ms，速度是Faster R-CNN的2.5至20倍。代码可用如下：

> arxiv.org/pdf/1605.0640...

1、引言

广为流行的用于目标检测的深度网络SPPNet、Fast/Faster R-CNN可以通过兴趣区域池化层（RoI pooling）划分为两个子网络：（1）一个共享的独立于RoI的"全卷积"子网络，（2）一个不共享计算的RoI-wise子网络。这种分解最初出现在开创性分类架构中，例如AlexNet和VGG，其设计为两个子网络——一个以空间池化层结尾的卷积子网络，另一个是由若干全连接层组成子网络。因此在图像分类网络中的池化层自然变成目标检测网络中的RoI池化层。

但是最近图像分类最佳网络如ResNet和Inception均设计为全卷积网络（仅仅最后一层是全连接层，其在目标检测微调训练时被移除和替换）。以此类推，目标检测架构中的共享的卷积子网络全部使用卷积层和RoI-wise子网络去掉全连接的隐藏层是件自然而然的事情。然而，若以此进行实验研究，这个幼稚的想法所构造的模型却有着与优越的分类精度不匹配的检测精度。为了解决这个问题，在ResNet论文中，Faster R-CNN的RoI池化层反常地插入两组卷积层之间（conv4_x和conv5_x），此操作加深了RoI-wise子网络以提升精度，但由于并不共享的RoI计算而付出了速度代价。

我们认为这种反常的设计是由于图像分类的平移不变性和目标检测的平移同变性之间的困境所引起的。一方面，图像分类需要平移不变性——图像内目标移动不加区别。因此，在ImageNet分类上领先结果ResNet和Inception证明尽可能保持平移不变性的深度卷积架构是可取的。另一方面，目标检测需要在一定程度上具有平移同变性的定位表示。例如，提议框中的目标的平移应该产生有意义的响应来描述提议框和目标的重叠程度。我们假设在一个图像分类网络中更深的卷积层对平移更为不敏感。为了摆脱这种困境，ResNet论文中的目标检测流程将RoI池化层插入卷积层之间——这个特定区域的操作打破了平移不变性，并且RoI后的卷积层在评估不同区域时不再具有平移不变性。然而，这种设计牺牲了训练和测试速度，因为其在RoI-wise子网络中引入了相当数量的卷积层，如下表1所示。

Table 1: Methodologies of *region-based* detectors using **ResNet-101** [9].

	R-CNN [7]	Faster R-CNN [19, 9]	R-FCN [ours]
depth of shared convolutional subnetwork	0	91	101
depth of RoI-wise subnetwork	101	10	**0**

> **translation invariant**：平移不变性，即目标平移之后系统响应相同，主要用于形容分类卷积网络，因为分类卷积网络包含**卷积层**和**池化层**，总会生成1×1×n特征。表达式✦为：represent(x) = represent(transform(x))。
> **translation equivarint**：平移同变性，即目标平移之后系统响应相应平移，主要用于形容**卷积层**，表达式为：transform(represent(x)) = represent(transform(x))。
> **translation variant**：平移变性，意同平移同变性。

本文我们为目标检测开发了一个名为基于区域的全卷积网络(R-FCN)。如同FCN网络，我们的网络由共享全卷积架构组成。为了将平移同变性合并到全卷积网络✦中，我们通过使用一组特殊的卷积层构造一组位置敏感的得分图作为全卷积网络的输出。这些得分图中的每一个都编码了关于相对空间✦位置的位置信息（例如，一个目标的左侧）。在这个全卷积网络之后，我们添加了一个位置敏感的RoI池化层以处理得分图信息，而之后便无任何权重层跟随。整个架构是端到端学习的。所有可学习的层都是在整个图像上共享的卷积层，但编码了目标检测所需要的空间信息。表1比较了基于区域的检测器之间区别，如下图1表明了这种核心思想。

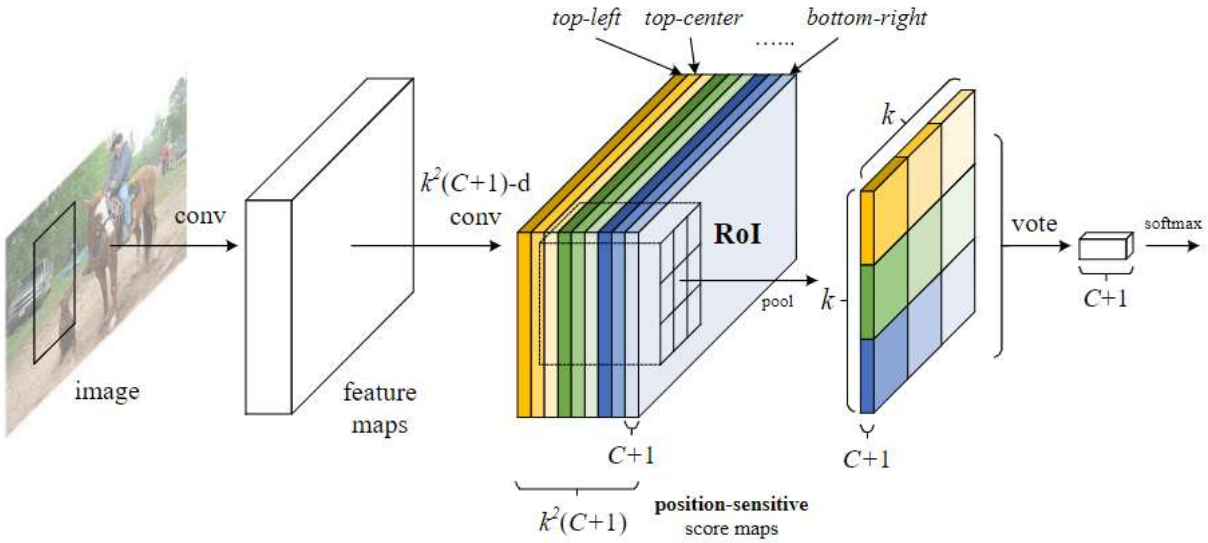


Figure 1: Key idea of **R-FCN** for object detection. In this illustration, there are $k \times k = 3 \times 3$ position-sensitive score maps generated by a fully convolutional network. For each of the $k \times k$ bins in an RoI, pooling is only performed on one of the k^2 maps (marked by different colors).

使用ResNet-101作为骨干网络，我们的R-FCN在VOC 2007数据集上得到83.6% mAP，在VOC 2012数据集上得到82.0% mAP。同时，使用ResNet-101，我们模型的测试速度是每张图像170ms，其是Faster R-CNN+ ResNet-101方法的2.5倍至20倍。这些实验表明我们的方法摆脱了平移不变性和平移同变性之间的困境，并且全卷积图像分类器如ResNet可以高效转换为全卷积目标检测器。代码公开可用如下：

> github.com/daijifeng001...

2、我们的方法

概述。遵循R-CNN，我们采用流行的两阶段目标检测策略✦，组成部分有二：（i）区域提议，（ii）区域分类。尽管存在不依赖于区域提议的方法，如YOLO V1和SSD，但是基于区域的系统在几个基准上依旧拥有领先的精度。我们通过全卷积RPN网络提取提议区域。遵循Faster R-CNN，我们在RPN和R-FCN之间共享特征。下图2展示了系统概貌。

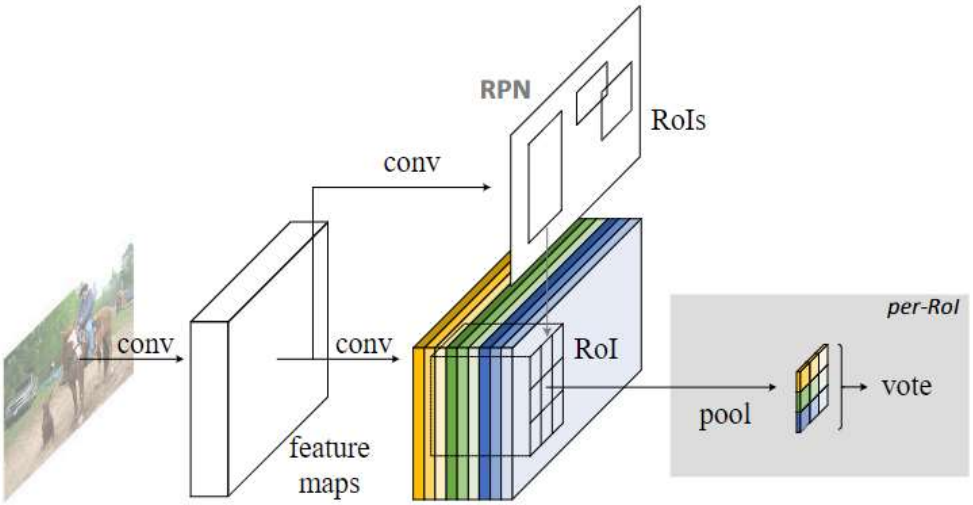


Figure 2: Overall architecture of R-FCN. A Region Proposal Network (RPN) [18] proposes candidate RoIs, which are then applied on the score maps. All learnable weight layers are convolutional and are computed on the entire image; the per-RoI computational cost is negligible.

给定提议区域（RoI），R-FCN架构被设计为将RoI分类为目标类别和背景。在R-FCN中，所有可学习的权重层都是卷积层，并且是在整个图像上计算的。最后一个卷积层为每一个类别生成k×k个位置敏感得分图，因此其通道数为k×k×(C+1),其中1代表背景类。k×k个得分图对应描述相对位置的k×k个空间网格。例如，若k×k=3×3，则有9个得分图对每一个目标类别的九个部分{左上，中上，右上，……，右下}进行编码。

R-FCN以位置敏感RoI池化层结束。位置敏感RoI池化层聚合最后一层卷积层的输出，并生成每一个RoI的分数。不同于SPPNet和Fast R-CNN，我们的位置敏感池化层进行选择性池化，k×k 个bins中的每一个都只聚合来自k×k 得分图之一的响应。伴随着端到端训练，RoI层引导着最后的卷积层去学习特定的位置敏感得分图。图1表明了这种核心思想。图3和图4展示了一个示例。

骨干架构。尽管有AlexNet和VGG可用，但是本文我们使用ResNet-101作为R-FCN的骨干网络。ResNet-101有100层卷积，1层平均池化，1层全连接。我们移除平均池化层和全连接层，并且仅使用卷积层进行计算特征图。我们使用ResNet原作者发布的在ImageNet预训练过的ResNet-101。ResNet-101里最后一个卷积层是2048-d的，我们接上一个随机初始化的1024-d的1×1卷积层以减少维度。然后我们应用一个k×k×(C+1)通道的卷积层以生成得分图。

位置敏感得分图和位置敏感RoI池化。为了显式编码位置信息进入每一个RoI，我们使用网格将RoI划分为k×k个bins。对于一个w×h的RoI，每个bin的大小是w/k×h/k。在我们的方法里，最后一个卷积层将会为每一个类别生成k×k个得分图。在(i,j)-th bin中(0≤i,j≤k-1)，我们定义一个位置敏感RoI池化操作，其仅仅在(i,j)-th得分图上进行池化：

$$r_c(i,j \mid \Theta) = \sum_{(x,y)\in \text{bin}(i,j)} z_{i,j,c}(x + x_0, y + y_0 \mid \Theta)/n. \quad (1)$$

其中rc(i,j)是c-th类(i,j)-th bin中池化后的响应，zi,j,c是k×k×(C+1)个得分图之一，(x0,y0)表示RoI的左上角在得分图上的坐标，n是一个bin中的像素的个数，Θ表示网络中可学习的参数。(i,j)-th bin坐标范围：

$$\lfloor i\tfrac{w}{k} \rfloor \le x < \lceil (i+1)\tfrac{w}{k} \rceil \text{ and } \lfloor j\tfrac{h}{k} \rfloor \le y < \lceil (j+1)\tfrac{h}{k} \rceil.$$

表达式1的操作如图1所示，其中一种颜色代表一个bin。表达式1是执行平均池化，但是也可以进行最大池化。

然后根据k×k位置敏感得分对RoI进行投票。在本文中，我们简单地通过平均得分实现投票，为每一个RoI产生一个C+1维向量：

$$r_c(\Theta) = \sum_{i,j} r_c(i,j \mid \Theta).$$

然后我们计算类别间的softmax响应：

$$s_c(\Theta) = e^{r_c(\Theta)} / \sum_{c'=0}^{C} e^{r_{c'}(\Theta)}.$$

他们被用于在训练期间评估交叉熵损失以及在推理期间进行类别排名。

我们进一步以和R-CNN和Fast R-CNN相似的方法解决了边界框回归。除了上述的k×k×(C+1)-d的卷积层，为了边界框回归我们添加了一个4×k^2-d卷积层。位置敏感RoI池化在4×k^2 maps上执行，为每一个RoI生成一个4×k^2-d特征。然后通过平均得分生成一个4-d向量。遵循Fast R-CNN，这个4-d向量将边界框参数化为t=(tx,ty,tw,th)。但是我们注意到，为了简单起见，我们使用了类别无关的边界框回归，但是类别特定的边界框回归也是适用的。

位置敏感得分图的概念部分受到instance-sensitive FCN的启发，后者开发了用于实例级语义分割的FCN。我们进一步介绍了位置敏感RoI池化层，它引导了用于目标检测的得分图的学习。RoI层之后无可学习的层，可以实现几乎0成本region-wise计算，并加速训练和推理。

训练。通过预先计算的区域提议框，很容易端到端地训练R-FCN架构。遵循Fast R-CNN，我们定义在每个RoI上的损失函数是交叉熵损失和框回归损失之和：

$$L_{cls}(s_{c^*}) = -\log(s_{c^*})$$

$$L(s, t_{x,y,w,h}) = L_{cls}(s_{c^*}) + \lambda[c^* > 0] L_{reg}(t, t^*).$$

其中c*是RoI的真实类别（c*=0表示背景），Lcls表示分类的交叉熵损失，Lreg表示边界框回归，t*表示真实框，[c*>0]表示若c*>0则为1，否则为0。如同Fast R-CNN，我们设置权重系数λ=1。我们定义与真实框交并比至少为0.5的RoI为正样本，否则为负样本。

在训练期间我们的方法易于采用在线难例挖掘（OHEM）。我们微不足道的per-RoI计算确保几乎0成本难例挖掘。假定每个图像有N个提议框，在前向传播中，我们评估了所有N个提议框的loss。然后我们以loss排序RoI，并且选择B个具有最高loss的RoI。基于选择的这些难例进行反向传播。由于我们的per-RoI计算是微不足道的，所以前向传播的时间几乎不受提议框数量N的影响，相反OHEM的Fast R-CNN可能需要双倍的训练时间。在下一节中表3，我们将提供全面的时间统计。

我们使用的权重衰减因子为0.0005和一个动量因子0.9。默认我们使用单尺度训练：将图像短边缩放为600。每个GPU处理一个图像，并且选择B=128个RoIs进行反向传播。我们使用8个GPU训练模型，所以高效的mini-batch大小为8x。我们在VOC数据集上使用20k批次的0.001学习率，使用10k批次的0.0001学习率。为了让R-FCN和RPN共享特征，我们采用Faster R-CNN中的4-step交替训练，以在训练RPN和训练R-FCN之间交替。

推理。如图2所示，经过一些列卷积计算而来的特征图共享于RPN和R-FCN之间。然后RPN部分提议RoI，R-FCN在其上评估类别得分和边界框。在推理期间，为了公平比较，我们像Faster R-CNN一样评估了300个RoIs。作为标准做法，使用IoU阈值为0.3的NMS对结果进行后处理。

空洞卷积⁺和步长。我们的全卷积网络架构受益于广泛用于语义分割的各种FCN网络变体。特别是，我们将ResNet-101的有效步长从32减至16，进而增加了得分图的分辨率。conv4阶段之前和之上的所有层都不变；conv5阶段的第一个卷积块上的stride=2修改为stride=1，conv5阶段上的所有卷积层都使用hole算法进行修改以弥补步长的减少。为了公平比较，如同ResNet论文中的Faster R-CNN一样，RPN紧接在conv4阶段之后，所以RPN不受空洞卷积的影响。下表展示了R-FCN的消融结果（k=7，未使用难例挖掘）。空洞卷积提升了2.6% mAP。

R-FCN with ResNet-101 on:	conv4, stride=16	conv5, stride=32	conv5, *à trous*, stride=16
mAP (%) on VOC 07 test	72.5	74.0	76.6

可视化⁺。在图3和图4中，我们可视化了当k=7时，R-FCN学习到的位置敏感得分图。这些特定的得分图⁺预计会在目标的特定的相对位置上被强烈激活。例如，top-center-sensitive得分图在接近一个目标的top-center位置处展示了高分。如果一个提议框与真实目标精确重叠如图3所示，那么RoI里k^2 bins大部分将被强烈激活，进而它们的投票导致更高的分。相反，如果一个提议框未能与真实目标正确重叠如图4所示，那么RoI里k^2 bins一些将不能被激活，进而它们的投票得分是低的。

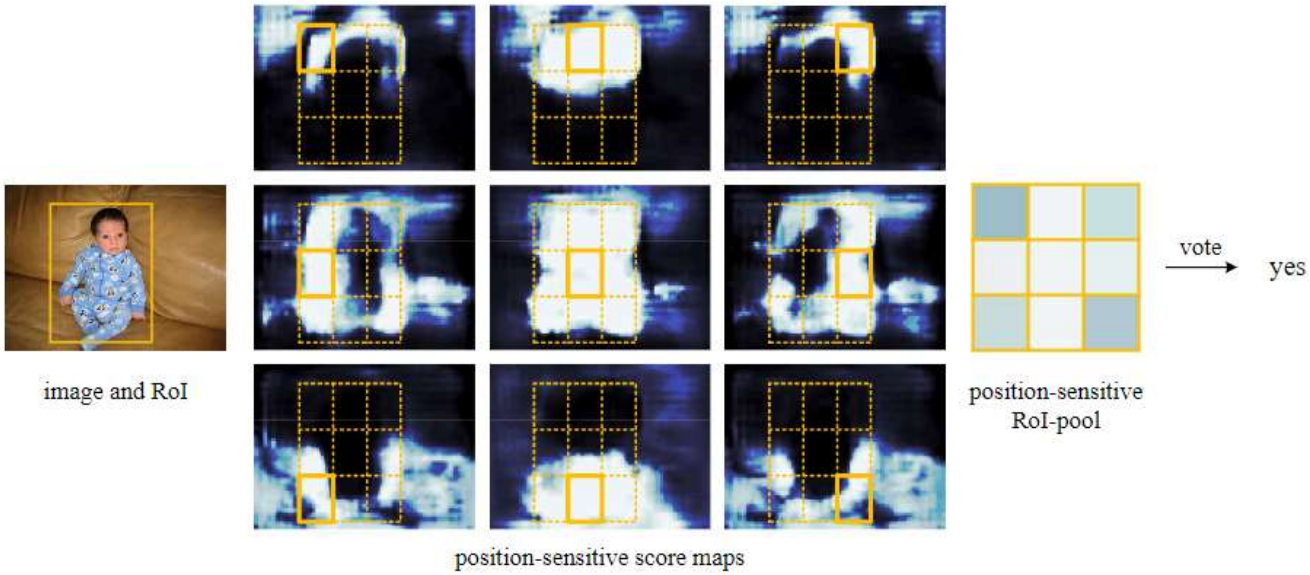


Figure 3: Visualization of R-FCN ($k \times k = 3 \times 3$) for the *person* category.

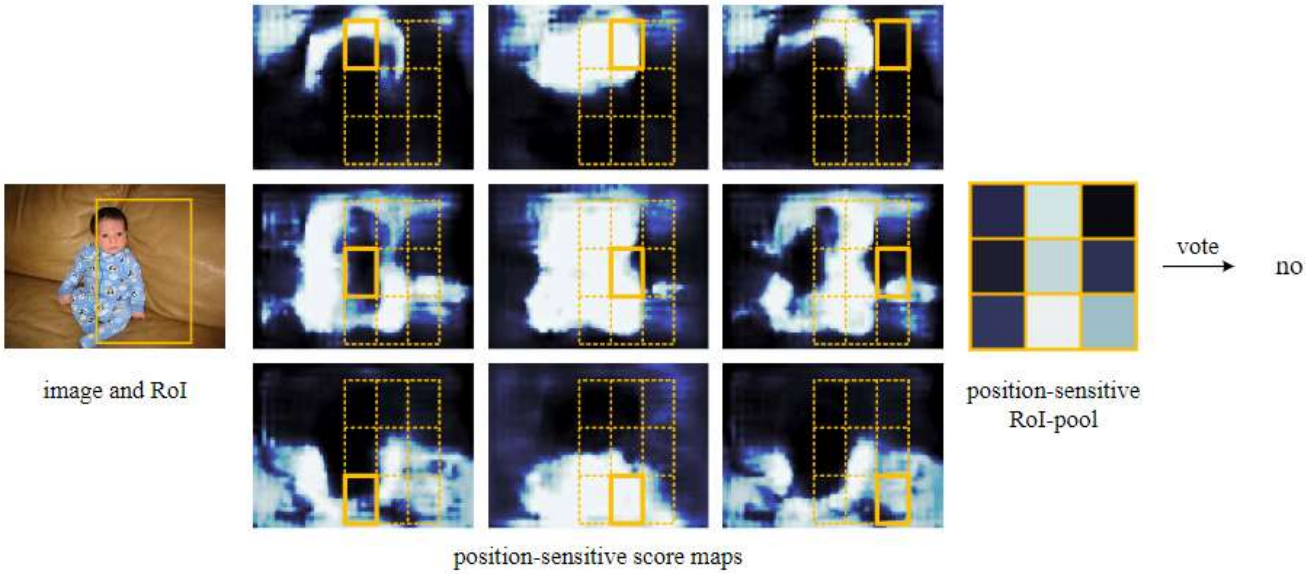


Figure 4: Visualization when an RoI does not correctly overlap the object.

3、相关工作

R-CNN已经展示了使用区域提议+selective search和深度网络的有效性。R-CNN在RoI上评估卷积网络，而RoI之间并未共享计算。SPPNet、Fast R-CNN、Faster R-CNN都都是半卷积网络，其中一个子网络对整个图像执行共享计算，另一个子网络评估每一个区域。

已经存在被认为是全卷积目标检测模型了。OverFeat在共享的卷积特征图上通过多尺度滑窗检测目标；相似地，在Fast R-CNN中，取代区域提议的滑窗已经在研究了。在这些情况，可以单尺度的滑窗构造为单个卷积层。在Faster R-CNN中RPN组件是一个基于多尺度参考框anchors预测边界框的全卷积检测器。在Faster R-CNN中的原始RPN是类别无关的，但是正如我们下面所评估的那样，类别特定的RPN也是可用的，如SSD。

另一类检测器采用全连接层在整个图像上生成所有目标检测的结果，如selective search+。

4、实验

4.1、在VOC数据集上的实验

我们在有20目标类的VOC上进行实验。遵循Fast R-CNN，我们在VOC 2007 trainval和VOC 2012 trainval联合数据集上进行训练，在VOC 2007测试集上进行评估。目标检测评价指标使用mAP。

与其他全卷积策略进行比较

尽管可以使用全卷积检测器，但是实验表明，它们要获得良好的精度并非易事。我们使用ResNet-101研究以下全卷积策略（或对于每个RoI仅有一个全卷积层+的近乎全卷积策略）：

幼稚的Faster R-CNN。如引言中所述，可以使用ResNet-101中的所有卷积层用于计算共享特征图，并在最后一个卷积层即conv5之后采用RoI池化。最后，在每一个RoI上进行一个简单的21分类全连接层评估，因此这种变体是近乎全卷积的。同时为了公平比较，我们在ResNet-101的conv5层使用了空洞卷积。

类别特定的RPN。除了2分类（object or not）卷积分类层替换为21分类的卷积分类层，类别特定的RPN按照Faster R-CNN中的RPN进行训练。为了公平比较，对于这个类别特定的RPN，我们在ResNet-101的conv5层使用了空洞卷积。

不具有位置敏感的R-FCN。通过设置k=1，我们移除了R-FCN中的位置敏感。这时位置敏感池化等同于每个RoI上的全局池化。

分析。表2展示了对比结果。在ResNet论文中的标准Faster R-CNN使用ResNet-101达到了76.4% mAP，其将RoI池化层插入在conv4和conv5之间。作为比较，幼稚的Faster R-CNN具有一个大幅降低的68.9% mAP。这个比较从实验上证明了通过在Faster R-CNN的卷积层中插入RoI池化以慎重处理空间信息的重要性。类似的观察报告在论文《Object detection networks on convolutional feature maps》中。

Table 2: Comparisons among fully convolutional (or "almost" fully convolutional) strategies using **ResNet-101**. *All competitors in this table use the à trous trick.* Hard example mining is not conducted.

method	RoI output size ($k \times k$)	mAP on VOC 07 (%)
naïve Faster R-CNN	1×1	61.7
	7×7	68.9
class-specific RPN	-	67.6
R-FCN (w/o position-sensitivity)	1×1	*fail*
R-FCN	3×3	75.5
	7×7	**76.6**

类别特定的RPN有一个67.6% mAP，其低于标准Faster R-CNN的76.4% 9个点。这种比较符合Fast R-CNN和R-CNN minus R中的观察结果——实际上，类别特定的RPN相似于使用密集滑窗作为提议框的一种特殊形式的Fast R-CNN。

另一方面，我们的R-FCN有着更高的精度，其76.6% mAP与标准Faser R-CNN的76.4% mAP相当。这些结果表明了，我们的位置敏感策略为定位目标编码了有用的空间信息，而不在RoI池化之后使用任何学习层。

为了进一步展示位置敏感的重要性，我们设置k=1，但R-FCN无法收敛。在这种退化情况下，一个RoI中的空间信息无法显式捕获。同时，如果RoI池化$^+$输出分辨率为1×1，幼稚的Faster R-CNN能够收敛，但是mAP大幅减少至61.7%。

与使用ResNet-101$^+$的Faster R-CNN比较。

接下来，我们与VOC和COCO以及ImageNet基准上的最强竞争者和顶级表现者的标准Faster R-CNN（Faster R-CNN + ResNet-101）。下面我们使用k=7的R-FCN。表3展示了对比结果。Faster R-CNN使用一个10层的RoI-wise的子网络以获得更高的精度，但是R-FCN却有着微不足道的计算成本。在采用300 RoIs的测试阶段，在一个K40 GPU上，我们的R-FCN每张图像的推理时间是0.17s（在一个Titan X GPU是0.11s），而Faster R-CNN每张图像的推理时间是0.42s。在训练阶段，R-FCN也比Faster R-CNN快。此外，对于R-FCN$^+$训练而言，难例挖掘无计算成本增加。当从2000 RoIs上进行难例挖掘时，训练R-FCN是可行的，在这种情况下，Faster R-CNN$^+$的一张图像的训练时间是R-FCN的6倍。但是实验表明更多的候选框并无益处。因此为了性训练和推理，我们在本文的其它部分使用300 RoIs。

Table 3: Comparisons between Faster R-CNN and R-FCN using ResNet-101. Timing is evaluated on a single Nvidia K40 GPU. With OHEM, N RoIs per image are computed in the forward pass, and 128 samples are selected for backpropagation. 300 RoIs are used for testing following [18].

	depth of per-RoI subnetwork	training w/ OHEM?	train time (sec/img)	test time (sec/img)	mAP (%) on VOC07
Faster R-CNN	10		1.2	0.42	76.4
R-FCN	0		0.45	0.17	76.6
Faster R-CNN	10	✓ (300 RoIs)	1.5	0.42	79.3
R-FCN	0	✓ (300 RoIs)	0.45	0.17	**79.5**
Faster R-CNN	10	✓ (2000 RoIs)	2.9	0.42	*N/A*
R-FCN	0	✓ (2000 RoIs)	0.46	0.17	79.3

表4展示了更多对比结果。遵循SPPNet中的多尺度训练，我们在每一个训练迭代时重新调整图像大小为从{400,500,600,700,800}中随机采样的尺度。我们依旧以单一尺度为600的图像进行测试，所以测试时间成本并未增加，而mAP为80.5%。此外我们先在COCO trainval set进行训练，然后在VOC set上进行微调训练。R-FCN实现了83.6% mAP，其接近于同样使用ResNet-101的Faster R-CNN +++。我们注意到R-FCN的推理时间是Faster R-CNN的20倍，因为Faster R-CNN进一步结合了box regression、context和multi-scale testing。这些比较也可在表5中VOC 2012测试集上观察到。

Table 4: Comparisons on PASCAL VOC 2007 *test* set using **ResNet-101**. "Faster R-CNN +++" [9] uses iterative box regression, context, and multi-scale testing.

	training data	mAP (%)	test time (sec/img)
Faster R-CNN [9]	07+12	76.4	0.42
Faster R-CNN +++ [9]	07+12+COCO	**85.6**	3.36
R-FCN	07+12	79.5	0.17
R-FCN multi-sc train	07+12	80.5	0.17
R-FCN multi-sc train	07+12+COCO	**83.6**	0.17

Table 5: Comparisons on PASCAL VOC 2012 *test* set using **ResNet-101**. "07++12" [6] denotes the union set of 07 *trainval+test* and 12 *trainval*. †: http://host.robots.ox.ac.uk:8080/anonymous/44L5HI.html ‡: http://host.robots.ox.ac.uk:8080/anonymous/MVCM2L.html

	training data	mAP (%)	test time (sec/img)
Faster R-CNN [9]	07++12	73.8	0.42
Faster R-CNN +++ [9]	07++12+COCO	**83.8**	3.36
R-FCN multi-sc train	07++12	77.6†	0.17
R-FCN multi-sc train	07++12+COCO	**82.0**‡	0.17

> 本文中所指的标准Faster R-CNN为ResNet论文中所构建的Faster R-CNN，而并非Faster R-CNN原论文中的Faster R-CNN。Faster R-CNN原论文模型检测精度如下：

Table 12: Detection mAP (%) of Faster R-CNN on PASCAL VOC 2007 test set and 2012 test set using different training data. The model is VGG-16. "COCO" denotes that the COCO trainval set is used for training. See also Table 6 and Table 7.

training data	2007 test	2012 test
VOC07	69.9	67.0
VOC07+12	73.2	-
VOC07++12	-	70.4
COCO (no VOC)	76.1	73.0
COCO+VOC07+12	**78.8**	-
COCO+VOC07++12	-	**75.9**

关于深度的影响。

下表展示了使用不同深度的ResNet的R-FCN结果。当深度从50增加到101时，我们的检测精度增加了，但深度为152时变得饱和。

	training data	test data	ResNet-50	ResNet-101	ResNet-152
R-FCN	07+12	07	77.0	79.5	79.6
R-FCN multi-sc train	07+12	07	78.7	80.5	80.4

关于区域提议的影响。

R-FCN能够很容易应用其他区域提议方法，例如Selective Search（SS）和Edge Boxes。下表展示了使用不同区域提议方法的结果。R-FCN使用SS或EB依旧具有很好精度，展示了R-FCN的通用性。

	training data	test data	RPN [18]	SS [27]	EB [28]
R-FCN	07+12	07	**79.5**	77.2	77.8

4.2、在COCO数据集上的实验

接下来，我们在具有80目标类的COCO数据集上进行评估。我们的实验涉及80k训练集、40k验证集和20k测试集。我们使用有效的mini-batch大小为8，同时将90k次迭代的学习率设为0.001，接下来30k次迭代设为0.0001。当共享特征时，我们将交替训练从4-step扩展到5-step（例如，再执行一个RPN训练step），这在这个数据集上会稍微提升精度；当不共享特征时，我们使用2-step训练也能达到相当好的精度。

> 4-step 交替训练：采用一个实用的4-step训练算法通过交替优化去学习共享特征。step-1，训练一个RPN：此网络使用ImageNet-pre-trained初始化，然后端到端进行区域提议微调训练。
> step-2，使用step-1训练的RPN进行区域提议，通过Fast R-CNN训练一个独立的检测网络。此测器也使用ImageNet-pre-trained初始化。但此时RPN和检测器并不共享卷积层。
> step-3，使用检测网络初始化RPN，固定共享卷积层，微调训练RPN独有的层。此时RPN和检测器共享卷积层。
> step-4，固定共享卷积层，微调训练Fast R-CNN独有的层。

结果如表6所示。我们的单尺度训练的R-FCN基线在val set上mAP有着48.9%/27.6%的结果。这与Faster R-CNN基线的48.4%/27.2%相当，但是R-FCN的测试时间是Faster R-CNN的2.5倍。值得注意的是，R-FCN在小目标上表现得更好。我们多尺度训练单尺度测试的R-FCN在val set上mAP有着49.1%/27.8%的结果，在test-dev上有着51.5%/29.2%的结果。考虑到COCO数据集的目标尺度的宽泛，我们进一步参考ResNet论文进行一个多尺度测试，其测试尺度为{200,400,600,800,1000}，其mAP为53.2%/31.5%。这个结果接近于COCO 2015竞赛的第一名（使用ResNet-101的Faster R-CNN +++，55.7%/34.9%）。尽管如此，我们的方法是更简单和没有任何花里胡哨的东西，如Faster R-CNN +++采用的上下文或迭代框回归，并且训练和测试都更快。

Table 6: Comparisons on MS COCO dataset using **ResNet-101**. The COCO-style AP is evaluated @ IoU $\in [0.5, 0.95]$. AP@0.5 is the PASCAL-style AP evaluated @ IoU $= 0.5$.

	training data	test data	AP@0.5	AP	AP small	AP medium	AP large	test time (sec/img)
Faster R-CNN [9]	train	val	48.4	27.2	6.6	28.6	45.0	0.42
R-FCN	train	val	48.9	27.6	8.9	30.5	42.0	0.17
R-FCN multi-sc train	train	val	49.1	27.8	8.8	30.8	42.2	0.17
Faster R-CNN +++ [9]	trainval	test-dev	**55.7**	**34.9**	15.6	38.7	50.9	3.36
R-FCN	trainval	test-dev	51.5	29.2	10.3	32.4	43.3	0.17
R-FCN multi-sc train	trainval	test-dev	51.9	29.9	10.8	32.8	45.0	0.17
R-FCN multi-sc train, test	trainval	test-dev	**53.2**	**31.5**	14.3	35.5	44.2	1.00

5、结论和未来工作

我们提出的基于区域的全卷积网络是一个简单但准确高效的目标检测框架。我们的网络自然采用当前最佳图像分类网络，例如全卷积的ResNet。我们的方法实现的精度与Faster R-CNN相当，但是训练和测试都更快。

我们有意使R-FCN保持简单。已经有一系列FCN的扩展被开发用于语义分割，以及用于基于区域的目标检测。我们希望我们的网络能够轻松享受该领域的进步所带来的好处。

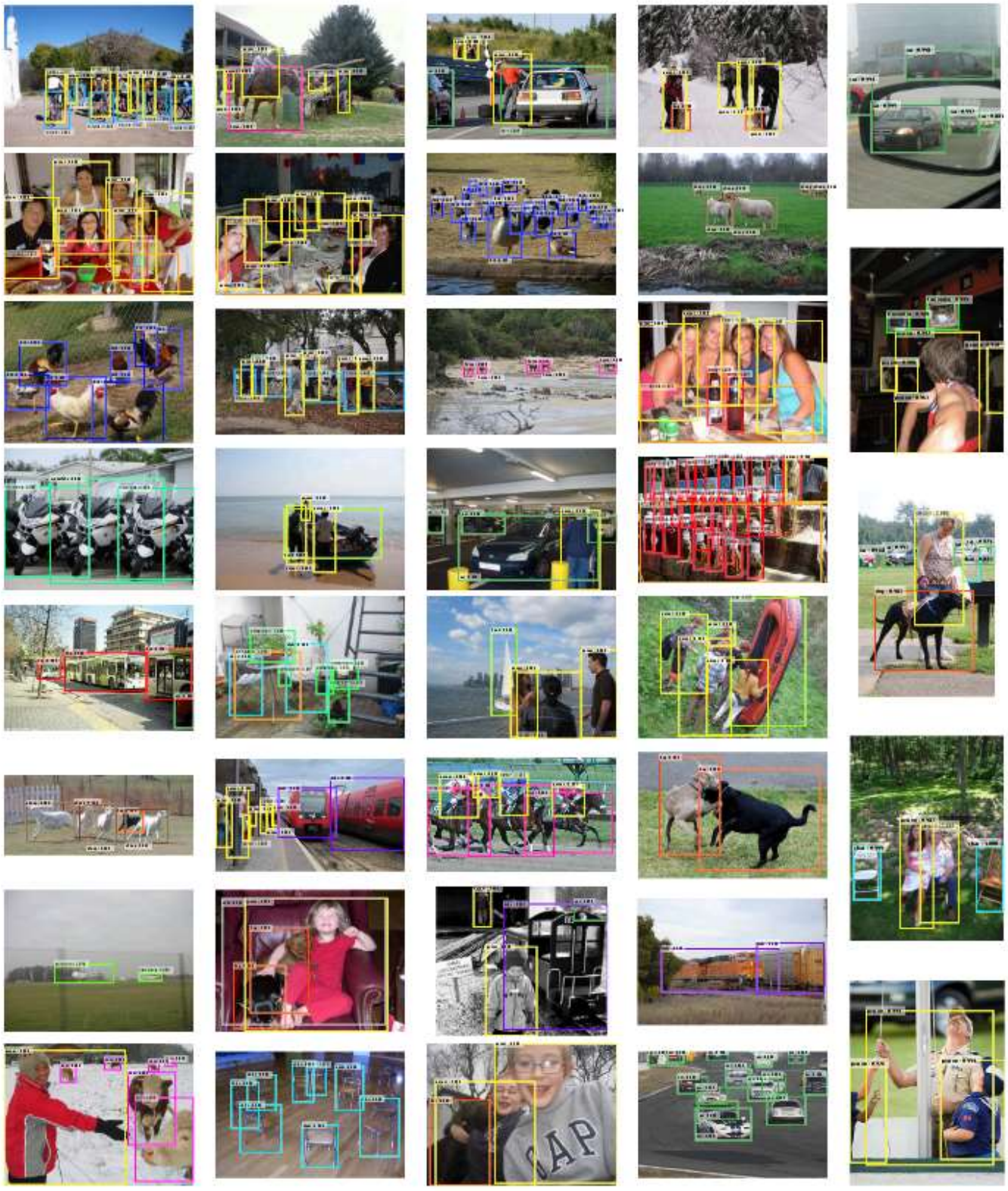


Figure 5: Curated examples of R-FCN results on the PASCAL VOC 2007 test set (83.6% mAP). The network is ResNet-101, and the training data is 07+12+COCO. A score threshold of 0.6 is used for displaying. The running time per image is 170ms on one Nvidia K40 GPU.

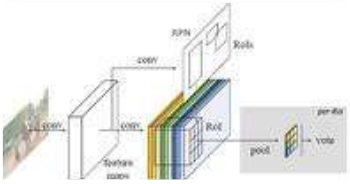
编辑于 2021-09-13 02:18

目标检测 卷积 ResNet

未登录用户

还没有评论，发表第一个评论吧

推荐阅读



详解R-FCN

麦田守望者

目标检测：R-FCN（NIPS 2016）

TeddyZhang

【目标检测】R-FCN

R-FCN这篇文章发表于2016年的NIPS上，它在Faster R-CNN的基础上进行改进，使得目标检测准确且更快。 1. 动机(Motivation)虽然Faster R-CNN把整个的检测过程集成到了一个可以end-to-end训练...

Jacqueline

基于R-FCN的物体检测

基于R-FCN的物体检测作者 Jifeng Dai Yi Li Kaiming He Jian Sun Microsoft Research Tsinghua University Microsoft Research Microsoft Research 译者 ~~~~~~~~~~~~~~~~~~~~~...

哈紫卡西