更快的 R-CNN：利用区域建议网络实现实时物体检测

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun

摘要--最先进的物体检测网络依赖于区域建议算法来假设物体的位置。SPPnet [1] 和 Fast R-CNN [2] 等技术的进步缩短了这些检测网络的运行时间，但却暴露出区域建议计算的瓶颈。在这项工作中，我们引入了区域建议网络（RPN），它与检测网络共享全图像卷积特征，从而实现了几乎无成本的区域建议。RPN 是一个全卷积网络，可同时预测每个位置的对象边界和对象性得分。RPN 经过端到端训练，可生成高质量的区域建议，并被快速 R-CNN 用于检测。通过共享卷积特征，我们进一步将 RPN 和快速 R-CNN 合并为一个网络--使用最近流行的具有 "注意力 "机制的神经网络术语，RPN 部分会告诉统一网络去哪里寻找。对于深度 VGG-16 模型[3]，我们的检测系统在 GPU 上的帧频为 5fps（包括所有步骤），同时在 PASCAL VOC 2007、2012 和 MS COCO 数据集上实现了一流的物体检测精度，每幅图像只需 300 个建议。在 ILSVRC 和 COCO 2015 比赛中，Faster R-CNN 和 RPN 是多个赛道第一名获奖作品的基础。代码已公开发布。

索引词条-物体检测、区域建议、卷积神经网络。

1. 引言

区域提议法（例如 [4]）和基于区域的卷积神经网络（RCNN）[5]的成功推动了物体检测领域的最新进展。虽然基于区域的卷积神经网络最初在 [5] 中开发时计算成本很高，但由于在提议中共享卷积，其成本已大大降低 [1], [2]。最新的快速 R-CNN [2]，在忽略区域提议所花费时间的情况下，利用非常深的网络[3]实现了接近实时的速度。现在，在最先进的检测系统中，提议是测试时间计算的瓶颈。

区域建议方法通常依赖于廉价的特征和经济的推理方案。选择性搜索[4]是最流行的方法之一，它根据设计好的低级特征贪婪地合并超像素。然而，与高效检测网络[2]相比，选择性搜索的速度慢了一个数量级，在 CPU 实现中，每幅图像需要 2 秒钟。目前，EdgeBoxes [6] 在建议质量和速度之间做出了最佳权衡，每幅图像只需 0.2 秒。尽管如此，区域建议步骤消耗的运行时间仍然与检测网络一样多。

人们可能会注意到，基于区域的快速 CNN 利用了 GPU 的优势，而研究中使用的区域提议方法是在 CPU 上实现的，因此这种运行时间比较并不公平。加速建议计算的一个显而易见的方法是在 GPU 上重新实施。这可能是一种有效的工程解决方案，但重新实施忽略了下游检测网络，因此错过了共享计算的重要机会。

在本文中，我们展示了一种算法变化--用深度卷积神经网络计算建议，从而获得一种优雅而有效的解决方案，在这种解决方案中，建议计算几乎不需要检测网络的计算成本。为此，我们推出了新型区域建议网络（RPN），它与最先进的物体检测网络共享卷积层[1]、[2]。通过在测试时间共享卷积，计算建议的边际成本很小（例如，每幅图像 10 毫秒）。

我们发现，基于区域的检测器（如快速 RCNN）所使用的卷积特征图也可用于生成区域建议。在这些卷积特征图的基础上，我们通过添加一些额外的卷积层来构建一个 RPN，这些卷积层可以在一个规则网格上的每个位置同时回归区域边界和对象性得分。因此，RPN 是一种全卷积网络（FCN）[7]，可以专门针对生成检测建议的任务进行端到端训练。

RPN 设计用于有效预测各种比例和长宽比的区域提案。与使用图像金字塔（图 1，a）

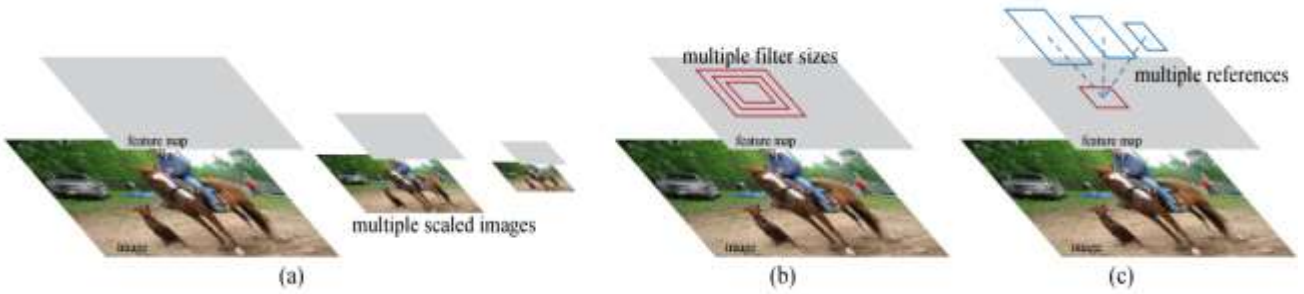

图 1：处理多种尺度和尺寸的不同方案。(a) 建立图像和特征图的金字塔，并在所有尺度上运行分类器。(b) 在特征图上运行多尺度/尺寸的金字塔过滤器。(c) 我们在回归函数中使用参考盒金字塔。

或滤波器金字塔（图 1，b）的主流方法 [8]、[9]、[1]、[2]相比，我们引入了新颖的 "锚 "框，作为多种比例和长宽比的参考。我们的方案可以看作是一个回归参考的金字塔（图 1，c），它避免了枚举多个尺度或纵横比的图像或滤镜。该模型在使用单比例图像进行训练和测试时表现良好，因此有利于提高运行速度。

为了将 RPN 与快速 R-CNN [2] 物体检测网络统一起来，我们提出了一种训练方案，即在区域建议任务中交替进行微调，然后在对象检测中进行微调，同时保持建议固定不变。1 我们在 PASCAL VOC 检测基准[11]上对我们的方法进行了全面评估，在这些基准中，使用快速 R-CNN 的 RPN 的检测精度优于使用快速 R-CNN 的选择性搜索的强基准。同时，我们的方法在测试时几乎免除了选择性搜索的所有计算负担--建议的有效运行时间仅为 10 毫秒。使用 [3] 中昂贵的深度模型，我们的检测方法在 GPU 上的帧频仍能达到 5fps（包括所有步骤），因此在速度和准确性方面都是一个实用的物体检测系统。我们还报告了 MS COCO 数据集[12]的结果，并利用 COCO 数据研究了 PASCAL VOC 的改进。代码已在 https://github.com/shaoqingren/faster_rcnn（MATLAB 中）和 https://github.com/ rbgirshick/py-faster-rcnn（Python 中）公开。

本手稿的初步版本已于之前发表[10]。从那时起，RPN 和 Faster R-CNN 框架已被采用并推广到其他方法中，如 3D 物体检测 [13]、基于部件的检测 [14]、实例分割 [15] 和图像标题 [16]。我们快速有效的物体检测系统还被用于商业系统，如 Pinterests [17]，据报道，用户参与度得到了提高。

在 ILSVRC 和 COCO 2015 比赛中，Faster R-CNN 和 RPN 是 ImageNet 检测、ImageNet 定位、COCO 检测和 COCO 分割赛道中多个第一名作品 [18] 的基础。RPN 完全学会了从数据中提出区域，因此很容易从更深层次和更具表现力的特征（如 [18] 中采用的 101 层残差网）中获益。更快的 R-CNN 和 RPN 也被这些竞赛中的其他几个领先参赛项目所采用 2。这些结果表明，我们的方法不仅是一种经济实用的解决方案，也是提高物体检测精度的有效方法。

2 相关工作

对象提议：关于对象提议方法的文献很多。有关对象提议方法的全面调查和比较可参见 [19]、[20]、[21]。广泛使用的对象提议方法包括基于超像素分组的方法（如选择性搜索 [4]、CPMC [22]、MCG [23]）和基于滑动窗口的方法（如窗口中的对象性 [24]、EdgeBoxes [6]）。对象建议方法作为独立于检测器的外部模块采用（例如，选择性搜索 [4] 对象检测器、RCNN [5] 和快速 R-CNN [2]）。

用于物体检测的深度网络：R-CNN 方法 [5] 对 CNN 进行端到端训练，将提议区域划分为物体类别或背景。R-CNN 主要起分类器的作用，并不预测物体边界（除了通过边界框回归进行细化）。其准确性取决于区域建议模块的性能（见 [20] 中的比较）。有几篇论文提出了使用深度网络预测物体边界框的方法 [25]、[9]、[26]、[27]。在 OverFeat 方法[9]中，对一个全连接层进行训练，以预测假定为单个物体的定位任务的方框坐标。然后将全连接层转化为卷积层，用于检测多个特定类别的物体。MultiBox 方法[26]、[27] 通过一个网络生成区域建议，该网络的最后一个全连接层可

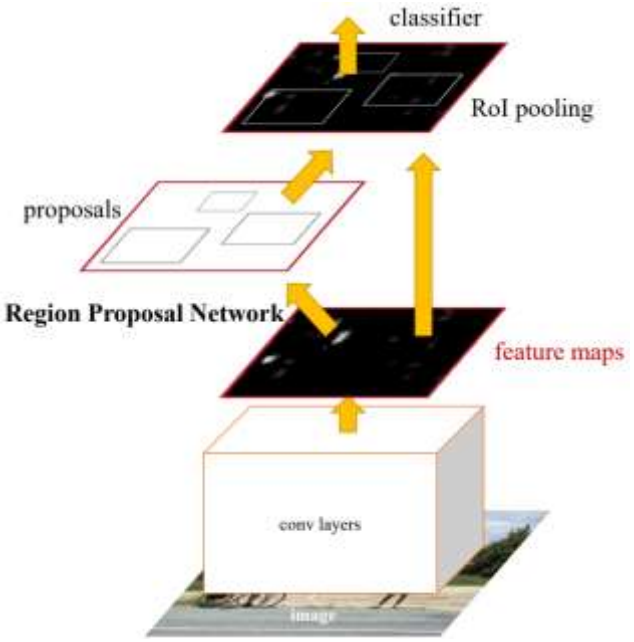

图 2：Faster R-CNN 是一个用于物体检测的单一统一网络。RPN 模块是这个统一网络的 "关注点"。

同时预测多个与类别无关的方框，从而推广了 OverFeat 的 "单方框 "方式。这些与类别无关的盒子被用作 R-CNN [5] 的建议。与我们的全卷积方案不同，MultiBox 提议网络适用于单一图像裁剪或多个大图像裁剪（如 224×224）。MultiBox 不会在提议网络和检测网络之间共享特征。稍后，我们将结合我们的方法更深入地讨论 OverFeat 和 MultiBox。与我们的工作同时进行的还有 DeepMask 方法[28]，用于学习分割建议。

卷积的共享计算 [9]、[1]、[29]、[7]、[2] 在高效、准确的视觉识别方面受到越来越多的关注。OverFeat 论文 [9] 从图像金字塔中计算卷积特征，用于分类、定位和检测。在共享卷积特征图上开发的自适应大小池（SPP）[1]可用于高效的基于区域的物体检测[1]、[30]和语义分割[29]。快速 R-CNN [2] 可在共享卷积特征上进行端到端检测器训练，并显示出令人信服的准确性和速度。

3 FASTER R-CNN

我们的物体检测系统名为 Faster R-CNN，由两个模块组成。第一个模块是提出区域的深度全卷积网络，第二个模块是使用提出的区域的快速 R-CNN 检测器 [2]。整个系统是一个单一、统一的物体检测网络（图 2）。使用最近流行的具有 "注意力 "机制的神经网络术语[31]，RPN 模块会告诉快速 R-CNN 模块去哪里寻找。在第 3.1 节中，我们将介绍区域建议网络的设计和特性。在第 3.2 节中，我们将为这两个模块开发共享特征的训练算法。

3.1 区域建议网络

区域建议网络（RPN）以图像（任意大小）为输入，输出一组矩形物体建议，每个建议都有一个物体度得分。由于我们的最终目标是与快速 R-CNN 物体检测网络共享计算[2]，因此我们假设这两个网络共享一组卷积层。在实验中，我们研究了拥有 5 个可共享卷积层的 Zeiler 和 Fergus 模型[32]（ZF），以及拥有 13 个可共享卷积层的 Simonyan 和 Zisserman 模型[3]（VGG-16）。

为了生成区域建议，我们在最后一个共享卷积层输出的卷积特征图上滑动一个小型网络。这个小型网络将输入卷积特征图的 n × n 空间窗口作为输入。每个滑动窗口都被映射为一个低维特征（ZF 为 256-d，VGG 为 512-d，ReLU[33]紧随其后）。该特征被送入两个完全连接的同胞层--盒式回归层（reg）和盒式分类层（cls）。我们在本文中使用 n = 3，因为输入图像的有效感受野很大（ZF 和 VGG 分别为 171 和 228 像素）。图 3（左）展示了这一微型网络的单个位置。请注意，由于迷你网络以滑动窗口的方式运行，因此所有空间位置的全连接层都是共享的。这种结构自然是通过一个 n×n 卷积层和两个同级的 1×1 卷积层（分别用于 reg 和 cls）来实现的。

3.1.1 Anchors

在每个滑动窗口位置，我们同时预测多个区域提案，每个位置的最大可能提案数表示为 k。因此，reg 层有 4k 个输出，编码 k 个方框的坐标，而 cls 层则输出 2k 个分数，估计每个提案是否有对象的概率 4。k 个提议是相对于 k 个参考方框的参数，我们称之为锚点。锚点以相关滑动窗口为中心，并与比例和长宽比相关联（图 3，左）。默认情况下，我们使用 3 种比例和 3 种长宽比，在每个滑动位置产生 k = 9 个锚点。对于大小为 W × H（通常

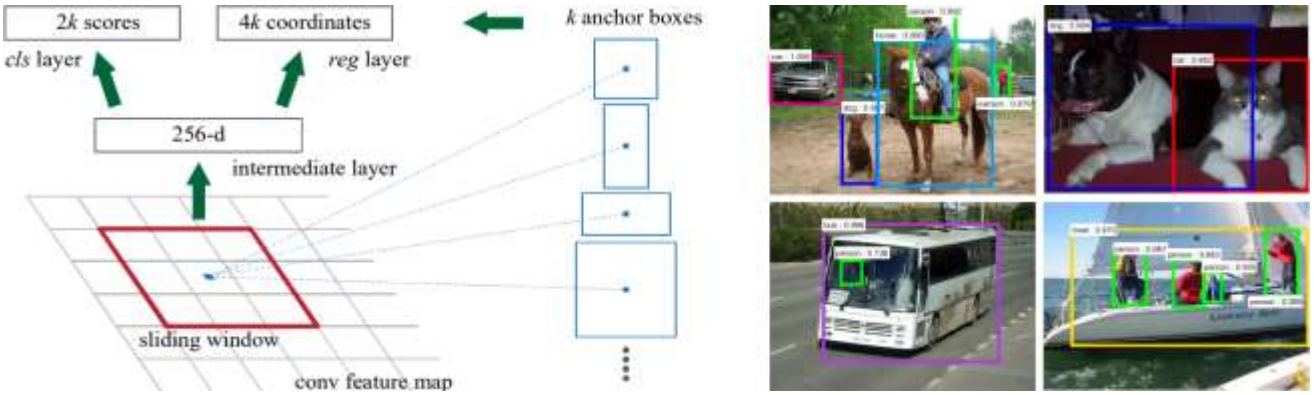

图 3：左：区域建议网络 (RPN)。右图在 PASCAL VOC 2007 测试中使用 RPN 建议进行检测的示例。我们的方法可以检测到各种比例和长宽比的物体。

为 ~2,400）的卷积特征图，总共有 WHk 个锚点。

翻译不变性的锚点

我们的方法的一个重要特性是它具有翻译不变性，这既体现在锚点方面，也体现在计算相对于锚点的建议的函数方面。如果将图像中的一个物体平移，提议也应随之平移，而相同的函数应能预测任一位置的提议。我们的方法保证了这一平移不变的特性 5。作为比较，MultiBox 方法[27] 使用 k-means 生成 800 个锚点，而这些锚点并不具有翻译不变性。因此，MultiBox 无法保证在对象被翻译的情况下生成相同的建议。

平移不变的特性还缩小了模型的尺寸。MultiBox 有一个 (4 + 1) × 800 维的全连接输出层，而我们的方法在 k = 9 锚点的情况下只有一个 (4 + 2) × 9 维的卷积输出层。因此，我们的输出层有 2.8 × 104 个参数（VGG-16 为 512 × (4 + 2) × 9），比 MultiBox 输出层的 6.1×106 个参数（MultiBox [27] 中的 GoogleNet [34] 为 1536 × (4 + 1) × 800）少两个数量级。如果考虑到特征投影层，我们的建议层的参数仍比 MultiBox6 少一个数量级。我们希望我们的方法在小型数据集（如 PASCAL VOC）上的过拟合风险更小。

作为回归参考的多尺度锚点

我们的锚点设计提出了一种解决多尺度（和纵横比）问题的新方案。如图 1 所示，目前有两种流行的多尺度预测方法。第一种方法基于图像/特征金字塔，例如 DPM [8] 和基于 CNN 的方法 [9]、[1]、[2]。在多个尺度上调整图像大小，并为每个尺度计算特征图（HOG [8] 或深度卷积特征 [9]、[1]、[2]）（图 1(a)）。这种方法通常很有用，但比较耗时。第二种方法是在特征图上使用多个尺度（和/或纵横比）的滑动窗口。例如，在 DPM [8]中，使用不同尺寸的滤波器（如 5×7 和 7×5）分别训练不同纵横比的模型。如果使用这种方法来处理多个尺度，则可以将其视为 "滤波器金字塔"（图 1(b)）。第二种方法通常与第一种方法联合使用 [8]。相比之下，我们基于锚点的方法建立在锚点金字塔的基础上，更具成本效益。我们的方法参照多种比例和长宽比的锚点框对边界框进行分类和回归。它只依赖于单一比例的图像和特征图，并使用单一尺寸的过滤器（特征图上的滑动窗口）。我们通过实验展示了这种方法在处理多种尺度和尺寸时的效果（表 8）。

由于这种基于锚点的多尺度设计，我们可以像快速 R-CNN 检测器[2]那样，简单地使用在单尺度图像上计算的卷积特征。多尺度锚点的设计是共享特征的关键部分，无需为处理尺度付出额外成本。

3.1.2Loss Function

在训练 RPN 时，我们为每个锚点分配一个二元类标签（是或不是对象）。我们为两种锚点分配正标签：(i) 与地面实况箱重叠度（Intersection-over-Union，IoU）最高的锚点，或 (ii) 与任何地面实况箱重叠度大于 0.7 的锚点。请注意，一个地面实况箱可能会给多个锚点分配正标签。通常情况下，第二个条件足以确定正样本；但我们仍然采用第一个条件，因为在某些罕见情况下，第二个条件可

能找不到正样本。如果一个非正向锚点的 IoU 比率在所有地面实况箱中都低于 0.3，我们就会给该锚点分配一个负向标签。既不是正面也不是负面的锚点不会对训练目标做出贡献。

根据这些定义，我们按照快速 R-CNN [2] 中的多任务损失最小化目标函数。我们对图像的损失函数定义如下：

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*). \quad (1)$$

这里，i 是迷你批次中锚点的索引，pi 是锚点 i 是对象的预测概率。ti 是表示预测边界框 4 个参数化坐标的向量，t∗i 是与正锚相关联的地面实况框的坐标。分类损失 Lcls 是两个类别（对象与非对象）的对数损失。对于回归损失，我们使用 $L_{reg}(t_i,t^*_i) = R(t_i - t^*_i)$ 其中 R 是 [2] 中定义的稳健损失函数（平滑 L1）。p∗iLreg 表示回归损耗仅在正锚（p∗i = 1）时激活，否则无效（p∗i = 0）。cls 层和 reg 层的输出分别由 {pi} 和 {ti} 组成。

在我们目前的实现中（与已发布的代码一样），式（1）中的 cls 项按迷你批量大小（即 Ncls = 256）归一化，reg 项按锚点数量（即 Nreg ∼ 2 400）归一化。我们默认设置 λ = 10，因此 cls 和 reg 项的权重大致相同。实验表明，在很大范围内，结果对 λ 值并不敏感（表 9）。我们还注意到，上述归一化并不是必需的，可以简化。

在边界框回归中，我们采用了 [5] 的 4 个坐标参数：

$$\begin{aligned} t_x &= (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \\ t_w &= \log(w/w_a), \quad t_h = \log(h/h_a), \\ t^*_x &= (x^* - x_a)/w_a, \quad t^*_y = (y^* - y_a)/h_a, \\ t^*_w &= \log(w^*/w_a), \quad t^*_h = \log(h^*/h_a), \end{aligned} \quad (2)$$

其中 x、y、w 和 h 表示方框的中心坐标及其宽度和高度。变量 x、xa 和 x∗ 分别表示预测方框、锚方框和地面实况方框（y、w、h 也是如此）。这可以看作是从锚点框到附近的地面实况框的边界框回归。

不过，我们的方法实现边界框回归的方式不同于之前基于兴趣区域（RoI）的方法 [1]、[2]。在 [1]、[2] 中，边界框回归是在任意大小的 RoI 中汇集的特征上进行的，回归权重由所有大小的区域共享。在我们的方案中，用于回归的特征在特征图上具有相同的空间大小（3 × 3）。为了考虑不同的大小，我们学习了一组 k 边框回归因子。每个回归因子负责一个比例和一个纵横比，并且 k 个回归因子不共享权重。因此，由于锚点的设计，即使特征的尺寸/比例是固定的，也能预测出各种尺寸的方框。

3.1.3Training RPNs

RPN 可以通过反向传播和随机梯度下降（SGD）[35] 进行端到端训练。我们采用 [2] 中的 "以图像为中心 "的采样策略来训练该网络。每个迷你批次由包含许多正反示例锚点的单个图像产生。对所有锚点的损失函数进行优化是可行的，但这会偏向于负样本，因为负样本占主导地位。相反，我们在图像中随机抽样 256 个锚点，计算迷你批次的损失函数，其中正负锚点的抽样比例最高为 1:1。如果图像中的正样本少于 128 个，我们就用负样本填充迷你批次。

我们从标准差为 0.01 的零均值高斯分布中提取权重，随机初始化所有新层。所有其他层（即共享卷积层）都是通过预训练 ImageNet 分类模型[36]来初始化的，这也是标准做法[5]。我们调整了 ZF 网络的所有层，以及 VGG 网络的 conv3 1 及以上层，以节省内存[2]。在 PASCAL VOC 数据集上，我们对 60k 个迷你批次使用 0.001 的学习率，对接下来的 20k 个迷你批次使用 0.0001 的学习率。我们使用 0.9 的动量和 0.0005 的权重衰减[37]。我们采用 Caffe [38]。

3.2 共享 RPN 和 Fast R-CNN 的特征

到目前为止，我们已经介绍了如何训练一个用于生成区域建议的网络，但并未考虑将利用这些建议的基于区域的物体检测 CNN。对于检测网络，我们采用快速 R-CNN [2]。接下来，我们将介绍由 RPN 和共享卷积层的快速 R-CNN 组成的统一网络的学习算法（图 2）。

anchor	128^2, 2:1	128^2, 1:1	128^2, 1:2	256^2, 2:1	256^2, 1:1	256^2, 1:2	512^2, 2:1	512^2, 1:1	512^2, 1:2
proposal	188×111	113×114	70×92	416×229	261×284	174×332	768×437	499×501	355×715

表 1：使用 ZF 网络学习到的每个锚点的平均提案大小（s = 600 时的数字）。

独立训练的 RPN 和快速 R-CNN 将以不同的方式修改其卷积层。因此，我们需要开发一种技术，允许两个网络共享卷积层，而不是学习两个独立的网络。我们将讨论训练共享特征网络的三种方法：

i. *交替训练。在这个方案中，我们首先训练 RPN，然后利用建议训练快速 R-CNN。快速 R-CNN 调整后的网络再用于初始化 RPN，这一过程反复进行。这就是本文所有实验中使用的解决方案。*

ii. 近似联合训练。在这种方案中，RPN 网络和快速 R-CNN 网络在训练过程中合并为一个网络，如图 2 所示。在每次 SGD 迭代中，前向传递都会生成区域建议，在训练快速 R-CNN 检测器时，这些建议就像固定的、预先计算好的建议一样。后向传播照常进行，在共享层中，来自 RPN 损失和快速 R-CNN 损失的后向传播信号被合并在一起。这种解决方案很容易实现。但是，该方案忽略了对提案框坐标的导数，而提案框坐标也是网络响应，因此是近似的。在我们的实验中，我们根据经验发现这种求解器产生的结果很接近，但与交替训练相比，训练时间减少了约 25-50%。这个求解器已包含在我们发布的 Python 代码中。

iii. 非近似联合训练。如上所述，RPN 预测的边界框也是输入的函数。快速 R-CNN 中的 RoI 池层[2]接受卷积特征和预测的边界框作为输入，因此理论上有效的反向传播求解器也应涉及与边界框坐标相关的梯度。在上述近似联合训练中，这些梯度被忽略。在非近似联合训练解决方案中，我们需要一个在方框坐标上可微分的 RoI 池层。这是一个非难解决的问题，可以通过[15]中开发的 "RoI 扭曲 "层来解决，这超出了本文的讨论范围。

第四步 交替训练. 在本文中，我们采用了一种实用的四步训练算法，通过交替优化来学习共享特征。第一步，我们按照第 3.1.3 节所述的方法训练 RPN。该网络使用 ImageNet 预先训练的模型进行初始化，并针对区域建议任务进行端到端的微调。在第二步中，我们使用由第一步 RPN 生成的建议，通过快速 R-CNN 训练一个单独的检测网络。该检测网络也由 ImageNet 预训练模型初始化。此时，两个网络不共享卷积层。在第三步中，我们使用检测网络来初始化 RPN 训练，但我们会固定共享的卷积层，只微调 RPN 独有的卷积层。现在，两个网络共享卷积层。最后，在固定共享卷积层的基础上，我们对快速 R-CNN 的独特层进行微调。这样，两个网络就共享了相同的卷积层，形成了一个统一的网络。类似的交替训练可以进行更多的迭代，但我们观察到的改进微乎其微。

3.3 实施细节

我们在单一比例的图像[1]、[2]上训练和测试区域建议和物体检测网络。我们重新缩放图像，使其短边为 s = 600 像素[2]。多尺度特征提取（使用图像金字塔）可能会提高准确度，但在速度和准确度之间并没有很好的权衡[2]。在重新缩放的图像上，最后一个卷积层上的 ZF 网和 VGG 网的总跨距为 16 像素，因此在重新缩放前（~500×375）的典型 PASCAL 图像上的跨距为 10 像素。即使是这么大的跨距也能获得良好的效果，不过如果跨距更小，精度可能会进一步提高。

对于锚点，我们使用了 3 种尺度，方框面积分别为 128^2, 256^2, and 512^2 像素，长宽比分别为 1:1、1:2 和 2:1。这些超参数并不是针对特定数据集精心选择的，我们将在下一节中对其效果进行消融实验。如前所述，我们的解决方案不需要图像金字塔或滤波金字塔来预测多个尺度的区域，从而节省了大量的运行时间。图 3（右）显示了我们的方法在各种尺度和长宽比下的能力。表 1 显示了使用

train-time region proposals		test-time region proposals		
method	# boxes	method	# proposals	mAP (%)
SS	2000	SS	2000	58.7
EB	2000	EB	2000	58.6
RPN+ZF, shared	2000	RPN+ZF, shared	300	**59.9**
ablation experiments follow below				
RPN+ZF, unshared	2000	RPN+ZF, unshared	300	58.7
SS	2000	RPN+ZF	100	55.1
SS	2000	RPN+ZF	300	56.8
SS	2000	RPN+ZF	1000	56.3
SS	2000	RPN+ZF (no NMS)	6000	55.2
SS	2000	RPN+ZF (no *cls*)	100	44.6
SS	2000	RPN+ZF (no *cls*)	300	51.4
SS	2000	RPN+ZF (no *cls*)	1000	55.8
SS	2000	RPN+ZF (no *reg*)	300	52.1
SS	2000	RPN+ZF (no *reg*)	1000	51.3
SS	2000	RPN+VGG	300	59.2

表 2：PASCAL VOC 2007 测试集的检测结果（根据 VOC 2007 trainval 训练）。这些检测器都是带有 ZF 的快速 R-CNN 检测器，但在训练和测试时使用了不同的建议方法。

ZF 网预测每个锚点的平均建议尺寸。我们注意到，我们的算法允许预测结果大于底层感受野。这种预测并不是不可能的--如果只有物体的中间部分可见，我们仍然可以大致推断出物体的范围。

需要小心处理跨越图像边界的锚点框。在训练过程中，我们会忽略所有跨边界的锚点，这样它们就不会造成损失。对于一幅典型的 1000 × 600 图像，总共会有大约 20000 个（≈ 60 × 40 × 9）锚点。如果忽略跨边界锚点，每幅图像大约有 6000 个锚点用于训练。如果在训练中不忽略越界异常值，就会在目标中引入大量难以纠正的误差项，训练无法收敛。不过，在测试过程中，我们仍然会对整个图像应用完全卷积 RPN。这可能会产生交叉边界建议框，我们会将其剪切到图像边界。

一些 RPN 建议彼此高度重叠。为了减少冗余，我们根据提案区域的 cls 分数对其采用非最大抑制（NMS）。我们将 NMS 的 IoU 阈值固定为 0.7，这样每幅图像就有大约 2000 个提议区域。正如我们将要展示的那样，NMS 不会损害最终的检测精度，但会大大减少提议区域的数量。在 NMS 之后，我们使用排名前 N 的建议区域进行检测。在下文中，我们使用 2000 个 RPN 建议来训练快速 R-CNN，但在测试时会评估不同数量的建议。

4 实验

4.1 PASCAL VOC 实验

我们在 PASCAL VOC 2007 检测基准[11]上对我们的方法进行了全面评估。该数据集包括约 5k 张训练图像和 5k 张测试图像，涉及 20 个物体类别。我们还提供了几个模型在 PASCAL VOC 2012 基准上的结果。在 ImageNet 预训练网络中，我们使用了拥有 5 个卷积层和 3 个全连接层的 "快速 "版 ZF net [32]，以及拥有 13 个卷积层和 3 个全连接层的公开 VGG-16 模型 7 [3]。我们主要评估检测平均精度 (mAP)，因为这是物体检测的实际指标（而不是侧重于物体建议的代理指标）。

表 2（顶部）显示了快速 R-CNN 使用各种区域建议方法进行训练和测试的结果。这些结果使用的是 ZF 网络。对于选择性搜索（SS）[4]，我们使用 "快速 "模式生成了约 2000 条建议。对于 EdgeBoxes (EB)[6]，我们使用调整为 0.7 IoU 的 EB 默认设置生成建议。在快速 R-CNN 框架下，SS 的 mAP 为 58.7%，

method	# proposals	data	mAP (%)
SS	2000	07	66.9†
SS	2000	07+12	70.0
RPN+VGG, unshared	300	07	68.5
RPN+VGG, shared	300	07	69.9
RPN+VGG, shared	300	07+12	**73.2**
RPN+VGG, shared	300	COCO+07+12	**78.8**

表 3：PASCAL VOC 2007 测试集的检测结果。检测器为快速 R-CNN 和 VGG-16。训练数据："07":VOC 2007 训练数据，"07+12"：VOC 2007 训练数据和 VOC 2012 训练数据的联合集。对于 RPN，快速 R-CNN 的训练时间建议为 2000。† ：[2]中报告了这一数字；使用本文提供的存储库，这一结果更高（68.1）。

method	# proposals	data	mAP (%)
SS	2000	12	65.7
SS	2000	07++12	68.4
RPN+VGG, shared†	300	12	67.0
RPN+VGG, shared‡	300	07++12	**70.4**
RPN+VGG, shared§	300	COCO+07++12	**75.9**

表 4：PASCAL VOC 2012 测试集的检测结果。检测器为快速 R-CNN 和 VGG-16。训练数据："07":VOC 2007 trainval，"07++12"：VOC 2007 trainval+test 和 VOC 2012 trainval 的联合集。对于 RPN，快速 R-CNN 的训练时间建议为 2000。†： http://host.robots.ox.ac.uk:8080/anonymous/HZJTQA.html。‡ : http://host.robots.ox.ac.uk:8080/anonymous/YNPLXB.html. § : http://host.robots.ox.ac.uk:8080/anonymous/XEDH10.html.

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	**10**	47	**198**	**5 fps**
ZF	RPN + Fast R-CNN	31	**3**	25	**59**	**17 fps**

表 5：K40 GPU 上的时序（毫秒），CPU 评估 SS 方案除外。"区域 "包括 NMS、池化、全连接和 softmax 层。运行时间剖析请参见我们发布的代码

EB 的 mAP 为 58.6%。使用快速 R-CNN 的 RPN 取得了具有竞争力的结果，在使用多达 300 个建议时，mAP 为 59.9%8。由于共享卷积计算，使用 RPN 产生的检测系统比使用 SS 或 EB 快得多；较少的提议也降低了区域全连接层的成本（表 5）。

RPN 的烧蚀实验：为了研究 RPN 作为一种提议方法的行为，我们进行了几项消融研究。首先，我们展示了 RPN 与快速 R-CNN 检测网络共享卷积层的效果。为此，我们在 4 步训练过程的第二步后停止。使用单独网络的结果略微降低到 58.7%（RPN+ZF，未共享，表 2）。我们发现，这是因为在第三步中，当使用检测到的特征对 RPN 进行微调时，提议的质量得到了提高。

接下来，我们分析了 RPN 对训练快速 R-CNN 检测网络的影响。为此，我们使用 2000 个 SS 提议和 ZF 网训练快速 R-CNN 模型。我们固定该检测器，并通过改变测试时使用的提议区域来评估检测 mAP。在这些消融实验中，RPN 不与检测器共享特征。

在测试时用 300 个 RPN 建议替换 SS，可使 mAP 达到 56.8%。mAP 的损失是由于训练/测试建议之间的不一致性造成的。这一结果可作为以下比较的基准。

令人稍感意外的是，在测试时使用排名最靠前的 100 个建议时，RPN 仍能产生有竞争力的结果（55.1%），这表明排名靠前的 RPN 建议是准确的。另一个极端是，使用排名最靠前的 6000 个 RPN 建议（不含 NMS），其 mAP 与之相当（55.2%），这表明 NMS 不会损害检测 mAP，并可能减少误报。

接下来，我们通过在测试时关闭 RPN 的 cls 和 reg 输出，分别研究它们的作用。当测试时移除 cls 层时（因此没有使用 NMS/ranking），我们从未获分区域随机抽取 N 个建议。当 N = 1000 时，mAP 几乎保持不变（55.8%），但当 N = 100 时，mAP 大幅下降至 44.6%。这表明，cls 分数影响了排名最高的建议的准确性。

另一方面，当在测试时移除 reg 层时（因此建议变成了锚框框），mAP 降至 52.1%。这

method	# box	data	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
SS	2000	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
SS	2000	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
RPN*	300	07	68.5	74.1	77.2	67.7	53.9	51.0	75.1	79.2	78.9	50.7	78.0	61.1	79.1	81.9	72.2	75.9	37.2	71.4	62.5	77.4	66.4
RPN	300	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
RPN	300	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
RPN	300	COCO+07+12	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9

表 6：使用快速 R-CNN 检测器和 VGG-16 对 PASCAL VOC 2007 测试集的结果。对于 RPN，快速 R-CNN 的训练时间建议为 2000。RPN* 表示无共享特征版本。

method	# box	data	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
SS	2000	12	65.7	80.3	74.7	66.9	46.9	37.7	73.9	68.6	87.7	41.7	71.1	51.1	86.0	77.8	79.8	69.8	32.1	65.5	63.8	76.4	61.7
SS	2000	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
RPN	300	12	67.0	82.3	76.4	71.0	48.4	45.2	72.1	72.3	87.3	42.2	73.7	50.0	86.8	78.7	78.4	77.4	34.5	70.1	57.1	77.1	58.9
RPN	300	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
RPN	300	COCO+07++12	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2

表 7：使用快速 R-CNN 检测器和 VGG-16 在 PASCAL VOC 2012 测试集上的结果。对于 RPN，快速 R-CNN 的训练时间建议为 2000。

settings	anchor scales	aspect ratios	mAP (%)
1 scale, 1 ratio	128^2	1:1	65.8
	256^2	1:1	66.7
1 scale, 3 ratios	128^2	{2:1, 1:1, 1:2}	68.8
	256^2	{2:1, 1:1, 1:2}	67.9
3 scales, 1 ratio	$\{128^2, 256^2, 512^2\}$	1:1	**69.8**
3 scales, 3 ratios	$\{128^2, 256^2, 512^2\}$	{2:1, 1:1, 1:2}	**69.9**

表 8：使用不同的锚点设置，Faster R-CNN 在 PASCAL VOC 2007 测试集上的检测结果。网络为 VGG-16。训练数据为 VOC 2007 trainval。使用 3 种比例和 3 种长宽比的默认设置（69.9%）与表 3 中的设置相同。

λ	0.1	1	10	100
mAP (%)	67.2	68.9	69.9	69.1

表 9：在 PASCAL VOC 2007 测试集上使用方程 (1) 中不同的 λ 值时，Faster R-CNN 的检测结果。网络为 VGG-16。训练数据为 VOC 2007 trainval。使用 λ = 10 (69.9%) 的默认设置与表 3 中的设置相同。

表明，高质量的建议主要归功于回归框边界。锚点框虽然具有多种比例和长宽比，但不足以实现精确检测。

我们还评估了更强大的网络对 RPN 单独提案质量的影响。我们使用 VGG-16 来训练 RPN，并仍然使用上述 SS+ZF 检测器。mAP 从 56.8%（使用 RPN+ZF）提高到 59.2%（使用 RPN+VGG）。这是一个很有希望的结果，因为它表明 RPN+VGG 的提议质量优于 RPN+ZF。由于 RPN+ZF 的建议与 SS 的建议具有竞争性（在一致使用 RPN+ZF 进行训练和测试时，两者的建议质量都是 58.7%），我们可以预期 RPN+VGG 的建议质量将优于 SS。下面的实验证明了这一假设。

VGG-16 的性能。 表 3 显示了 VGG-16 在提议和检测方面的结果。使用 RPN+VGG 时，非共享特征的检测结果为 68.5%，略高于 SS 基线。如上所述，这是因为 RPN+VGG 生成的提议比 SS 更准确。与预先定义的 SS 不同，RPN 是主动训练的，能从更好的网络中获益。对于特征共享变体，结果是 69.9%--优于强 SS 基线，而且几乎不需要成本。我们在 PASCAL VOC 2007 trainval 和 2012 trainval 的联合集上进一步训练 RPN 和检测网络。mAP 为 73.2%。图 5 显示了 PASCAL VOC 2007 测试集上的一些结果。在 PASCAL VOC 2012 测试集上（表 4），我们的方法在 VOC 2007 trainval+test 和 VOC 2012 trainval 的联合集上训练的 mAP 为 70.4%。表 6 和表 7 显示了详细数据。

表 5 总结了整个物体检测系统的运行时间。根据内容不同，SS 需要 1-2 秒（平均约 1.5 秒），而使用 VGG-16 的快速 R-CNN 在 2000 次 SS 提议中需要 320 毫秒（如果在全连接层上使用 SVD，则需要 223 毫秒[2]）。我们的系统使用 VGG-16 时，提议和检测总共需要 198 毫秒。在共享卷积特征的情况下，仅 RPN 计算附加层就只需 10 毫秒。由于提议次数较少（每幅图像 300 次），我们的区域计算时间也更短。使用 ZF 网络时，我们的系统帧频为 17 帧/秒。

对超参数的敏感性：在表 8 中，我们研究了锚点的设置。默认情况下，我们使用 3 种比例和 3 种长宽比（表 8 中的 mAP 为 69.9%）。

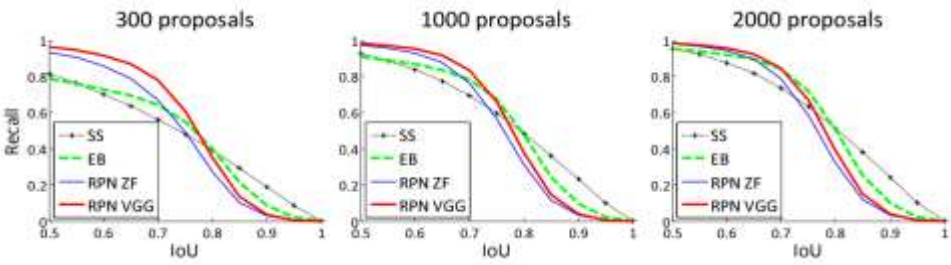

图 4：PASCAL VOC 2007 测试集上的召回率与 IoU 重叠率对比。

	proposals		detector	mAP (%)
Two-Stage	RPN + ZF, unshared	300	Fast R-CNN + ZF, 1 scale	58.7
One-Stage	dense, 3 scales, 3 aspect ratios	20000	Fast R-CNN + ZF, 1 scale	53.8
One-Stage	dense, 3 scales, 3 aspect ratios	20000	Fast R-CNN + ZF, 5 scales	53.9

表 10：单阶段检测与双阶段建议 + 检测对比。检测结果是使用 ZF 模型和快速 R-CNN 在 PASCAL VOC 2007 测试集上得出的。RPN 使用非共享特征。

如果在每个位置只使用一个锚点，mAP 会大幅下降 3-4%。如果使用 3 个比例（1 个纵横比）或 3 个纵横比（1 个比例），则 mAP 会更高，这表明使用多种尺寸的锚点作为回归参考是一种有效的解决方案。在该数据集上，仅使用 3 个尺度和 1 个纵横比（69.8%）与使用 3 个尺度和 3 个纵横比的效果一样好，这表明尺度和纵横比并不是检测精度的分离维度。但我们在设计中仍然采用了这两个维度，以保持系统的灵活性。在表 9 中，我们比较了方程 (1) 中不同的 λ 值。默认情况下，我们使用 λ = 10，这使得等式 (1) 中的两个项在归一化后权重大致相等。表 9 显示，当 λ 在大约两个数量级（1 到 100）的范围内时，我们的结果受到的影响很小（1%~）。这表明，在很大范围内，结果对 λ 并不敏感。 召回率-IoU 分析。接下来，我们计算了不同 IoU 比的提案与地面实况框的召回率。值得注意的是，Recall-to-IoU 指标与最终检测精度的关系并不紧密[19]、[20]、[21]。使用该指标诊断建议方法比评估建议方法更合适。

图 4 显示了使用 300、1000 和 2000 个建议的结果。我们与 SS 和 EB 进行了比较，根据这些方法产生的置信度，N 个建议是排名前 N 的建议。从图中可以看出，当建议数从 2000 降到 300 时，RPN 方法的表现非常优美。这就解释了为什么 RPN 在使用少至 300 个建议时也能获得良好的最终检测 mAP。正如我们之前所分析的，这一特性主要归功于 RPN 的 cls 项。当建议数较少时，SS 和 EB 的召回率比 RPN 下降得更快。

单阶段检测与两阶段建议+检测的比较：

OverFeat 论文[9]提出了一种在卷积特征图上的滑动窗口上使用回归器和分类器的检测方法。OverFeat 是一种单阶段、针对特定类别的检测管道，而我们的方法则是一种两阶段级联，包括与类别无关的建议和针对特定类别的检测。在 OverFeat 中，区域特征来自比例金字塔上一个长宽比的滑动窗口。这些特征用于同时确定物体的位置和类别。在 RPN 中，特征来自正方形（3×3）滑动窗口，并预测相对于不同比例和纵横比的锚点的建议。虽然这两种方法都使用滑动窗口，但区域建议任务只是 Faster RCNN 的第一阶段--下游的快速 R-CNN 检测器会对建议进行细化。在我们级联的第二阶段，区域特征从提案框中自适应地汇集[1]、[2]，以更忠实地覆盖区域特征。我们相信这些特征能带来更准确的检测。

为了比较单级系统和双级系统，我们用单级快速 R-CNN 模拟 OverFeat 系统（从而也规避了其他实施细节上的差异）。在该系统中，"提案 "是 3 种比例（128、256、512）和 3 种纵横比（1:1、1:2、2:1）的密集滑动窗口。对快速 R-CNN 进行训练，以预测特定类别的分数，并根据这些滑动窗口回归方框位置。由于 OverFeat 系统采用的是图像金字塔，我们还使用从 5 个尺度提取的卷积特征进行评估。我们使用的是 [1] 和 [2] 中的 5 个尺度。

表 10 比较了两级系统和一级系统的两个变体。使用 ZF 模型，单级系统的 mAP 为 53.9%。这比两阶段系统（58.7%）低 4.8%。这一实验证明了级联区域建议和目标检测的

method	proposals	training data	COCO val		COCO test-dev	
			mAP@.5	mAP@[.5, .95]	mAP@.5	mAP@[.5, .95]
Fast R-CNN [2]	SS, 2000	COCO train	-	-	35.9	19.7
Fast R-CNN [impl. in this paper]	SS, 2000	COCO train	38.6	18.9	39.3	19.3
Faster R-CNN	RPN, 300	COCO train	41.5	21.2	42.1	21.5
Faster R-CNN	RPN, 300	COCO trainval	-	-	**42.7**	**21.9**

表 11：MS COCO 数据集的物体检测结果（%）。模型为 VGG-16。

有效性。在 [2] 和 [39] 中也有类似的观察结果，在这两篇论文中，用滑动窗口代替 SS 区域提案会导致 ~6% 的性能下降。我们还注意到，单级系统的处理速度较慢，因为它需要处理的提案要多得多。

4.2 关于 MS COCO 的实验

我们展示了微软 COCO 物体检测数据集 [12]的更多结果。该数据集涉及 80 个对象类别。我们在训练集上使用了 8 万张图像，在验证集上使用了 4 万张图像，在测试开发集上使用了 2 万张图像。我们评估了 IoU ∈ [0.5 : 0.05 : 0.95]（COCO 标准指标，简称 mAP@[.5, .95]）和 mAP@0.5（PASCAL VOC 指标）的 mAP 平均值。

针对该数据集，我们的系统做了一些小改动。我们在 8 个 GPU 上训练我们的模型，RPN 的有效迷你批次规模变为 8（每个 GPU 1 个），Fast R-CNN 变为 16（每个 GPU 2 个）。RPN 步骤和快速 R-CNN 步骤均以 0.003 的学习率进行 240k 次迭代训练，然后以 0.0003 的学习率进行 80k 次迭代训练。我们修改了学习率（从 0.003 开始，而不是 0.001），因为迷你批次的大小发生了变化。对于锚点，我们使用了 3 种长宽比和 4 种比例（增加了 642），这主要是为了处理该数据集上的小型物体。此外，在我们的快速 R-CNN 步骤中，负样本被定义为与地面实况的最大 IoU 在 [0,0.5) 之间的样本，而不是 [1], [2] 中使用的 [0.1,0.5)。我们注意到，在 SPPnet 系统[1]中，[0.1,0.5] 中的负样本用于网络微调，但在 SVM 步骤中，[0,0.5] 中的负样本仍会被访问，并进行硬负挖掘。但快速 R-CNN 系统 [2] 放弃了 SVM 步骤，因此 [0,0.1) 中的负样本永远不会被访问。对于快速 R-CNN 和更快 R-CNN 系统而言，将这些 [0,0.1) 样本包括在内可改善 COCO 数据集上的 mAP@0.5（但对 PASCAL VOC 的影响可以忽略不计）。

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**

表 12：在 PASCAL VOC 2007 测试集和 2012 测试集上使用不同训练数据的更快 R-CNN 的检测率（%）。模型为 VGG-16。"COCO "表示使用 COCO 训练集进行训练。另请参见表 6 和表 7。

其余实施细节与 PASCAL VOC 相同。特别是，我们继续使用 300 个提案和单尺度（s = 600）测试。在 COCO 数据集上，每幅图像的测试时间仍然约为 200 毫秒。

在表 11 中，我们首先报告了使用本文实现的快速 R-CNN 系统 [2] 的结果。我们的快速 R-CNN 基线在测试-开发集上的 mAP@0.5 率为 39.3%，高于 [2] 中报告的结果。我们推测，造成这一差距的原因主要是负样本的定义以及迷你批量大小的变化。我们还注意到，mAP@[.5, .95]与之相当。

接下来，我们对 Faster R-CNN 系统进行评估。使用 COCO 训练集进行训练，Faster R-CNN 在 COCO 测试开发集上的 mAP@0.5 和 mAP@[.5, .95] 分别为 42.1% 和 21.5%。与相同协议下的快速 RCNN 相比，mAP@0.5 高 2.8%，mAP@[.5, .95] 高 2.2%（表 11）。这表明，在较高的 IoU 门限下，RPN 在提高定位精度方面表现出色。使用 COCO trainval 集进行训练时，Faster RCNN 在 COCO test-dev 集上的 mAP@0.5 和 mAP@[.5, .95] 分别为 42.7% 和 21.9%。图 6 显示了在 MS COCO 测试-开发集上的一些结果。

Faster R-CNN 在 ILSVRC 和 COCO 2015 比赛中脱颖而出: 我们已经证明，Faster R-CNN 能从更好的特征中获益更多，这要归功

于 RPN 完全学会了通过神经网络提出区域建议。即使深度大幅增加到 100 层以上，这一观察结果仍然有效[18]。只有用 101 层残差网络（ResNet-101）[18] 替换 VGG-16 后，Faster R-CNN 系统在 COCO val set 上的 mAP 才从 41.5%/21.2% (VGG16)提高到 48.4%/27.2% (ResNet-101)。通过与 Faster RCNN 正交的其他改进，He 等人[18] 在 COCO test-dev 集上获得了 55.7%/34.9% 的单模型结果和 59.0%/37.4% 的集合结果，并赢得了 COCO 2015 物体检测竞赛的第一名。同一系统[18]还在 ILSVRC 2015 物体检测竞赛中获得第一名，以绝对优势超过第二名 8.5%。RPN 也是 ILSVRC 2015 定位竞赛和 COCO 2015 分割竞赛第一名获奖作品的组成部分，详情分别见 [18] 和 [15]。

4.3 从 MS COCO 到 PASCAL VOC

大规模数据对于改进深度神经网络至关重要。接下来，我们将研究 MS COCO 数据集如何帮助提高 PASCAL VOC 的检测性能。

作为一个简单的基准，我们直接在 PASCAL VOC 数据集上评估 COCO 检测模型，而不对任何 PASCAL VOC 数据进行微调。之所以可以进行这种评估，是因为 COCO 上的类别是 PASCAL VOC 上类别的超集。在本实验中，COCO 上独有的类别将被忽略，而 softmax 层仅在 20 个类别加上背景上执行。在这种设置下，PASCAL VOC 2007 测试集的 mAP 为 76.1%（表 12）。尽管没有使用 PASCAL VOC 数据，但这一结果远远优于在 VOC07+12 上训练的结果（73.2%）。

然后，我们在 VOC 数据集上对 COCO 检测模型进行微调。在本实验中，COCO 模型取代了 ImageNet 预训练模型（用于初始化网络权重），并按照第 3.2 节所述对 Faster R-CNN 系统进行了微调。这样，在 PASCAL VOC 2007 测试集上的 mAP 率达到 78.8%。COCO 测试集的额外数据使 mAP 增加了 5.6%。表 6 显示，在 PASCAL VOC 2007 上，在 COCO+VOC 上训练的模型在每个类别上的 AP 都是最好的。在 PASCAL VOC 2012 测试集上也有类似的改进（表 12 和表 7）。我们注意到，获得这些优异结果的测试时间仍然是每幅图像约 200 毫秒。

5 结论

我们提出了用于高效、准确地生成区域建议的 RPN。通过与下游检测网络共享卷积特征，区域建议步骤几乎不需要成本。我们的方法使基于深度学习的统一物体检测系统能够以接近实时的帧速率运行。学习到的 RPN 还能提高区域建议的质量，从而提高整体物体检测的准确性。

参考文献

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in *European Conference on Computer Vision (ECCV)*, 2014.

[2] R. Girshick, “Fast R-CNN,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[3] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations (ICLR)*, 2015.

[4] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *International Journal of Computer Vision (IJCV)*, 2013.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[6] C. L. Zitnick and P. Dollar, “Edge boxes: Locating object´ proposals from edges,” in *European Conference on Computer Vision (ECCV)*, 2014.

[7] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained partbased models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” in *International Conference on Learning Representations (ICLR)*, 2014.

[10] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Neural Information Processing Systems (NIPS)*, 2015.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results,” 2007.

[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, “Microsoft COCO: Com-´ mon Objects in Context,” in *European Conference on Computer Vision (ECCV)*, 2014.

[13] S. Song and J. Xiao, “Deep sliding shapes for amodal 3d object detection in rgb-d images,” *arXiv:1511.02300*, 2015.

[14] J. Zhu, X. Chen, and A. L. Yuille, “DeePM: A deep part-based model for object detection and semantic part localization,” *arXiv:1511.07131*, 2015.

[15] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” *arXiv:1512.04412*, 2015.

[16] J. Johnson, A. Karpathy, and L. Fei-Fei, “Densecap: Fully convolutional localization networks for dense captioning,” *arXiv:1511.07571*, 2015.

[17] D. Kislyuk, Y. Liu, D. Liu, E. Tzeng, and Y. Jing, “Human curation and convnets: Powering item-to-item recommendations on pinterest,” *arXiv:1511.04003*, 2015.

[18] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv:1512.03385*, 2015.

[19] J. Hosang, R. Benenson, and B. Schiele, “How good are detection proposals, really?” in *British Machine Vision Conference (BMVC)*, 2014.

[20] J. Hosang, R. Benenson, P. Dollar, and B. Schiele, “What makes´ for effective detection proposals?” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.

[21] N. Chavali, H. Agrawal, A. Mahendru, and D. Batra, “Object-Proposal Evaluation Protocol is ’Gameable’,” *arXiv: 1505.05836*, 2015.

[22] J. Carreira and C. Sminchisescu, “CPMC: Automatic object segmentation using constrained parametric min-cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.

[23] P. Arbelaez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik,´ “Multiscale combinatorial grouping,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[24] B. Alexe, T. Deselaers, and V. Ferrari, “Measuring the objectness of image windows,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.

[25] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Neural Information Processing Systems (NIPS)*, 2013.

[26] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[27] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov, “Scalable, high-quality object detection,” *arXiv:1412.1441 (v1)*, 2015.

[28] P. O. Pinheiro, R. Collobert, and P. Dollar, “Learning to segment object candidates,” in *Neural Information Processing Systems (NIPS)*, 2015.

[29] J. Dai, K. He, and J. Sun, “Convolutional feature masking for joint object and stuff segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[30] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, “Object detection networks on convolutional feature maps,” *arXiv:1504.06066*, 2015.

[31] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in *Neural Information Processing Systems (NIPS)*, 2015.

[32] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional neural networks,” in *European Conference on Computer Vision (ECCV)*, 2014.

[33] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *International Conference on Machine Learning (ICML)*, 2010.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, and A. Rabinovich, “Going deeper with convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[35] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, 1989.

[36] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” in *International Journal of Computer Vision (IJCV)*, 2015.

[37] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Neural Information Processing Systems (NIPS)*, 2012.

[38] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv:1408.5093*, 2014.

[39] K. Lenc and A. Vedaldi, “R-CNN minus R,” in *British Machine Vision Conference (BMVC)*, 2015.