Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

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

Abstract—State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet [1] and Fast R-CNN [2] have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck. In this work, we introduce a *Region Proposal Network* (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. An RPN is a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. We further merge RPN and Fast R-CNN into a single network by sharing their convolutional features—using the recently popular terminology of neural networks with "attention" mechanisms, the RPN component tells the unified network where to look. For the very deep VGG-16 model [3], our detection system has a frame rate of 5fps (*including all steps*) on a GPU, while achieving state-of-the-art object detection accuracy on PASCAL VOC 2007, 2012, and MS COCO datasets with only 300 proposals per image. In ILSVRC and COCO 2015 competitions, Faster R-CNN and RPN are the foundations of the 1st-place winning entries in several tracks. Code has been made publicly available.

Index Terms—Object Detection, Region Proposal, Convolutional Neural Network.

1 Introduction

Recent advances in object detection are driven by the success of region proposal methods (*e.g.*, [4]) and region-based convolutional neural networks (R-CNNs) [5]. Although region-based CNNs were computationally expensive as originally developed in [5], their cost has been drastically reduced thanks to sharing convolutions across proposals [1], [2]. The latest incarnation, Fast R-CNN [2], achieves near real-time rates using very deep networks [3], *when ignoring the time spent on region proposals*. Now, proposals are the test-time computational bottleneck in state-of-the-art detection systems.

Region proposal methods typically rely on inexpensive features and economical inference schemes. Selective Search [4], one of the most popular methods, greedily merges superpixels based on engineered low-level features. Yet when compared to efficient detection networks [2], Selective Search is an order of magnitude slower, at 2 seconds per image in a CPU implementation. EdgeBoxes [6] currently provides the best tradeoff between proposal quality and speed, at 0.2 seconds per image. Nevertheless, the region proposal step still consumes as much running time as the detection network.

- *S. Ren is with University of Science and Technology of China, Hefei, China. This work was done when S. Ren was an intern at Microsoft Research. Email: sqren@mail.ustc.edu.cn*
- *K. He and J. Sun are with Visual Computing Group, Microsoft Research. E-mail: {kahe,jiansun}@microsoft.com*
- *R. Girshick is with Facebook AI Research. The majority of this work was done when R. Girshick was with Microsoft Research. E-mail: rbg@fb.com*

One may note that fast region-based CNNs take advantage of GPUs, while the region proposal methods used in research are implemented on the CPU, making such runtime comparisons inequitable. An obvious way to accelerate proposal computation is to re-implement it for the GPU. This may be an effective engineering solution, but re-implementation ignores the down-stream detection network and therefore misses important opportunities for sharing computation.

In this paper, we show that an algorithmic change—computing proposals with a deep convolutional neural network—leads to an elegant and effective solution where proposal computation is nearly cost-free given the detection network's computation. To this end, we introduce novel *Region Proposal Networks* (RPNs) that share convolutional layers with state-of-the-art object detection networks [1], [2]. By sharing convolutions at test-time, the marginal cost for computing proposals is small (*e.g.*, 10ms per image).

Our observation is that the convolutional feature maps used by region-based detectors, like Fast R-CNN, can also be used for generating region proposals. On top of these convolutional features, we construct an RPN by adding a few additional convolutional layers that simultaneously regress region bounds and objectness scores at each location on a regular grid. The RPN is thus a kind of fully convolutional network (FCN) [7] and can be trained end-to-end specifically for the task for generating detection proposals.

RPNs are designed to efficiently predict region proposals with a wide range of scales and aspect ratios. In contrast to prevalent methods [8], [9], [1], [2] that use

Faster R-CNN：利用区域提议网络实现实时目标检测

任少卿，何恺明，Ross Girshick 和孙剑

摘要——最先进的物体检测网络依赖于区域提议算法来假设物体位置。SPPnet [1] 和 Fast R-CNN [2] 等进展已减少了这些检测网络的运行时间，使区域提议计算成为瓶颈。在本工作中，我们引入了一个*Region Proposal Network* (区域提议网络（RPN）)，它与检测网络共享全图像卷积特征，从而实现几乎无成本的区域提议。RPN 是一个全卷积网络，可同时预测每个位置上的物体边界和物体性分数。RPN 经过端到端训练以生成高质量的区域提议，供 Fast R-CNN 用于检测。我们通过共享卷积特征进一步将 RPN 和 Fast R-CNN 合并为一个单一网络——使用最近流行的“注意力”机制神经网络术语，RPN 组件告诉统一网络需要关注的位置。对于非常深的 VGG-16 模型 [3]，我们的检测系统在 GPU 上达到每秒 5 帧（*including all steps*）的处理速度，同时在 PASCAL VOC 2007、2012 和 MS COCO 数据集上实现了最先进的物体检测精度，且每幅图像仅使用 300 个提议。在 ILSVRC 和 COCO 2015 竞赛中，Faster R-CNN 和 RPN 是多个赛道第一名获奖方案的基础。代码已公开提供。

索引术语—目标检测，区域提议，卷积神经网络。

1 引言

R物体检测的最新进展得益于
th区域提议方法的成功（*e.g.*, [4]）
a基于区域的卷积神经网络（R-
C神经网络）[5]。尽管基于区域的卷积神经网络曾是
p正如最初在[5]中开发的那样，计算成本高昂，
th其成本因共享而大幅降低。
ing 跨提案的卷积 [1], [2]。最新的
in康乃馨，Fast R-CNN [2]，实现了接近实时的
ra使用非常深的网络进行测试 [3], *when ignoring the time spent on region proposals*现在，提案是
te最先进技术中的st-time计算瓶颈
d检测系统。

区域提议方法通常依赖于不精确的
p广泛的特征和经济高效的推理方案。选择性搜索[4]作为最流行的方法之一，基于设计的低级特征贪婪地合并超像素。然而，与高效的检测网络[2]相比，选择性搜索在CPU实现中每张图像耗时2秒，速度慢了一个数量级。EdgeBoxes[6]目前在建议质量和速度之间提供了最佳平衡，每张图像仅需0.2秒。尽管如此，区域建议步骤仍然消耗与检测网络相当的运行时间。

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，这使得运行时间的比较显得不公平。加速提议计算的一个明显方法是将其重新实现在GPU上。这可能是一个有效的工程解决方案，但重新实现忽略了下游的检测网络，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上的改进——使用深度卷积神经网络计算候选区域——带来了一个优雅而高效的解决方案，使得在给定检测网络计算量的情况下，候选区域的计算几乎无需额外成本。为此，我们引入了新颖的*Region Proposal Networks* (RPNs)，它们与当前最先进的目标检测网络[1]、[2]共享卷积层。通过在测试时共享卷积计算，计算候选区域的边际成本非常小（*e.g*例如，每张图像仅需10毫秒）。

我们的观察是，基于区域的检测器（如Fast R-CNN）所使用的卷积特征图同样可用于生成区域建议。在这些卷积特征之上，我们通过添加少量额外的卷积层构建了一个区域建议网络（RPN），这些卷积层能够在规则网格的每个位置上同时回归区域边界和目标性得分。因此，RPN是一种全卷积网络（FCN）[7]，可以端到端地专门训练以生成检测建议。

RPN旨在高效预测具有广泛尺度和长宽比的区域提议。与当前主流方法[8]、[9]、[1]、[2]使用{v*}的方式不同，

- *S. Ren is with University of Science and Technology of China, Hefei, China. This work was done when S. Ren was an intern at Microsoft Research. Email: sqren@mail.ustc.edu.cn*
- *K. He and J. Sun are with Visual Computing Group, Microsoft Research. E-mail: {kahe,jiansun}@microsoft.com*
- *R. Girshick is with Facebook AI Research. The majority of this work was done when R. Girshick was with Microsoft Research. E-mail: rbg@fb.com*

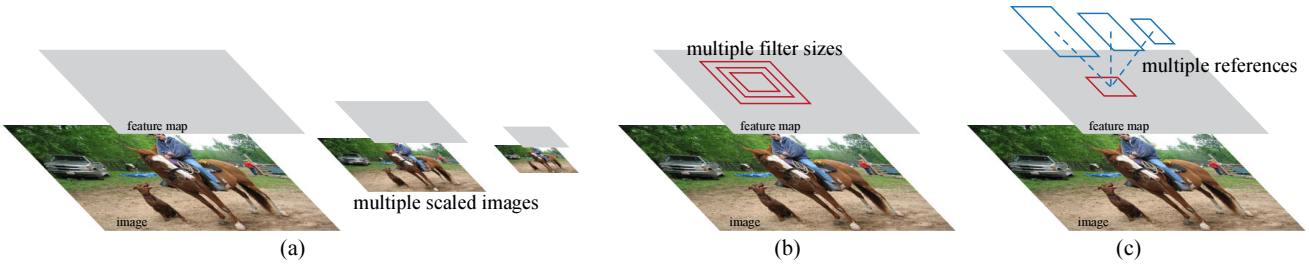

Figure 1: Different schemes for addressing multiple scales and sizes. (a) Pyramids of images and feature maps are built, and the classifier is run at all scales. (b) Pyramids of filters with multiple scales/sizes are run on the feature map. (c) We use pyramids of reference boxes in the regression functions.

pyramids of images (Figure 1, a) or pyramids of filters (Figure 1, b), we introduce novel "anchor" boxes that serve as references at multiple scales and aspect ratios. Our scheme can be thought of as a pyramid of regression references (Figure 1, c), which avoids enumerating images or filters of multiple scales or aspect ratios. This model performs well when trained and tested using single-scale images and thus benefits running speed.

To unify RPNs with Fast R-CNN [2] object detection networks, we propose a training scheme that alternates between fine-tuning for the region proposal task and then fine-tuning for object detection, while keeping the proposals fixed. This scheme converges quickly and produces a unified network with convolutional features that are shared between both tasks.[1]

We comprehensively evaluate our method on the PASCAL VOC detection benchmarks [11] where RPNs with Fast R-CNNs produce detection accuracy better than the strong baseline of Selective Search with Fast R-CNNs. Meanwhile, our method waives nearly all computational burdens of Selective Search at test-time—the effective running time for proposals is just 10 milliseconds. Using the expensive very deep models of [3], our detection method still has a frame rate of 5fps (*including all steps*) on a GPU, and thus is a practical object detection system in terms of both speed and accuracy. We also report results on the MS COCO dataset [12] and investigate the improvements on PASCAL VOC using the COCO data. Code has been made publicly available at `https://github.com/shaoqingren/faster_rcnn` (in MATLAB) and `https://github.com/rbgirshick/py-faster-rcnn` (in Python).

A preliminary version of this manuscript was published previously [10]. Since then, the frameworks of RPN and Faster R-CNN have been adopted and generalized to other methods, such as 3D object detection [13], part-based detection [14], instance segmentation [15], and image captioning [16]. Our fast and effective object detection system has also been built in commercial systems such as at Pinterests [17], with user engagement improvements reported.

In ILSVRC and COCO 2015 competitions, Faster R-CNN and RPN are the basis of several 1st-place entries [18] in the tracks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. RPNs completely learn to propose regions from data, and thus can easily benefit from deeper and more expressive features (such as the 101-layer residual nets adopted in [18]). Faster R-CNN and RPN are also used by several other leading entries in these competitions[2]. These results suggest that our method is not only a cost-efficient solution for practical usage, but also an effective way of improving object detection accuracy.

2 Related Work

Object Proposals. There is a large literature on object proposal methods. Comprehensive surveys and comparisons of object proposal methods can be found in [19], [20], [21]. Widely used object proposal methods include those based on grouping super-pixels (*e.g.*, Selective Search [4], CPMC [22], MCG [23]) and those based on sliding windows (*e.g.*, objectness in windows [24], EdgeBoxes [6]). Object proposal methods were adopted as external modules independent of the detectors (*e.g.*, Selective Search [4] object detectors, R-CNN [5], and Fast R-CNN [2]).

Deep Networks for Object Detection. The R-CNN method [5] trains CNNs end-to-end to classify the proposal regions into object categories or background. R-CNN mainly plays as a classifier, and it does not predict object bounds (except for refining by bounding box regression). Its accuracy depends on the performance of the region proposal module (see comparisons in [20]). Several papers have proposed ways of using deep networks for predicting object bounding boxes [25], [9], [26], [27]. In the OverFeat method [9], a fully-connected layer is trained to predict the box coordinates for the localization task that assumes a single object. The fully-connected layer is then turned

1. Since the publication of the conference version of this paper [10], we have also found that RPNs can be trained jointly with Fast R-CNN networks leading to less training time.

2. http://image-net.org/challenges/LSVRC/2015/results

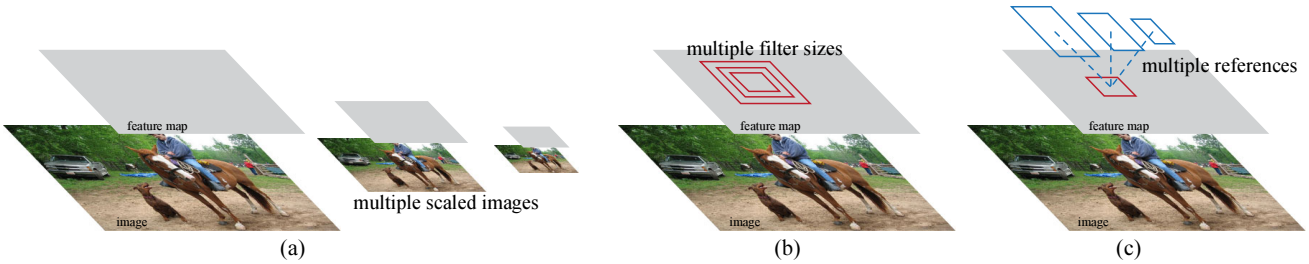

图1：处理多尺度和多尺寸的不同方案。(a) 构建图像和特征图的金字塔，并在所有尺度上运行分类器。(b) 在特征图上运行具有多尺度/多尺寸的滤波器金字塔。(c) 我们在回归函数中使用参考框金字塔。

图像金字塔（图1，a）或滤波器金字塔（图1，b）的基础上，我们引入了新颖的“锚点”框，这些框在多个尺度和宽高比下作为参考。我们的方案可被视为回归参考金字塔（图1，c），它避免了枚举多个尺度或宽高比的图像或滤波器。该模型在使用单尺度图像进行训练和测试时表现良好，从而有利于提升运行速度。

为了将RPN与Fast R-CNN[2]目标检测网络统一，我们提出了一种交替训练方案：先在区域提议任务上进行微调，然后在目标检测任务上进行微调，同时保持提议框不变。该方案能快速收敛，并生成一个具有卷积特征共享的双任务统一网络。[1]

我们在PASCAL VOC检测基准[11]上全面评估了我们的方法，其中采用Fast R-CNN的RPN在检测准确率上超越了基于选择性搜索结合Fast R-CNN的强基线。同时，我们的方法在测试阶段几乎完全免除了选择性搜索的计算负担——候选区域生成的有效运行时间仅为10毫秒。即使使用[3]中计算成本高昂的极深度模型，我们的检测方法在GPU上仍能达到5fps（*including all steps*）的帧率，因此在速度与精度方面均构成了实用的目标检测系统。我们还在MS COCO数据集[12]上报告了结果，并探究了利用COCO数据对PASCAL VOC指标的提升效果。相关代码已公开于https://github.com/shaoqingren/faster_rcnn（MATLAB版本）与https://github.com/rbgirshick/py-faster-rcnn（Python版本）。

本文的初版曾于先前发表[10]。自那时起，RPN与Faster R-CNN的框架已被采纳并推广至其他方法，例如3D目标检测[13]、基于部件的检测[14]、实例分割[15]以及图像描述生成[16]。我们快速高效的目标检测系统也已在商业应用中构建——例如在Pinterest等商业系统中[17]，据报告用户参与度有所提升。

在ILSVRC和COCO 2015竞赛中，Faster R-CNN和RPN是多个赛道（包括ImageNet检测、ImageNet定位、COCO检测和COCO分割）中若干第一名作品的基础[18]。RPN完全从数据中学习如何生成候选区域，因此能够轻松受益于更深层、更具表现力的特征（例如[18]中采用的101层残差网络）。在这些竞赛中，其他多个领先作品也采用了Faster R-CNN和RPN[2]。这些结果表明，我们的方法不仅是实际应用中具有成本效益的解决方案，也是提升目标检测精度的有效途径。

2 相关工作

物体提议。关于物体提议方法的文献非常丰富。全面的物体提议方法综述和比较可以在[19]、[20]、[21]中找到。广泛使用的物体提议方法包括基于超像素分组的方法（*e.g*，如选择性搜索[4]、CPMC[22]、MCG[23]）和基于滑动窗口的方法（*e.g*，如窗口物体性[24]、EdgeBoxes[6]）。物体提议方法曾作为独立于检测器的外部模块被采用（*e.g*，如选择性搜索[4]物体检测器、R-CNN[5]和Fast R-CNN[2]）。

用于目标检测的深度网络。R-CNN方法[5]通过端到端训练CNN，将候选区域分类为目标类别或背景。R-CNN主要充当分类器，不预测目标边界（除通过边界框回归进行优化外）。其准确度取决于区域建议模块的性能（参见[20]中的比较）。多篇论文提出了使用深度网络预测目标边界框的方法[25]、[9]、[26]、[27]。在OverFeat方法[9]中，训练全连接层来预测定位任务（假设存在单个目标）的边界框坐标。随后该全连接层被转换为

1. Since the publication of the conference version of this paper [10], we have also found that RPNs can be trained jointly with Fast R-CNN networks leading to less training time.

2. http://image-net.org/challenges/LSVRC/2015/results

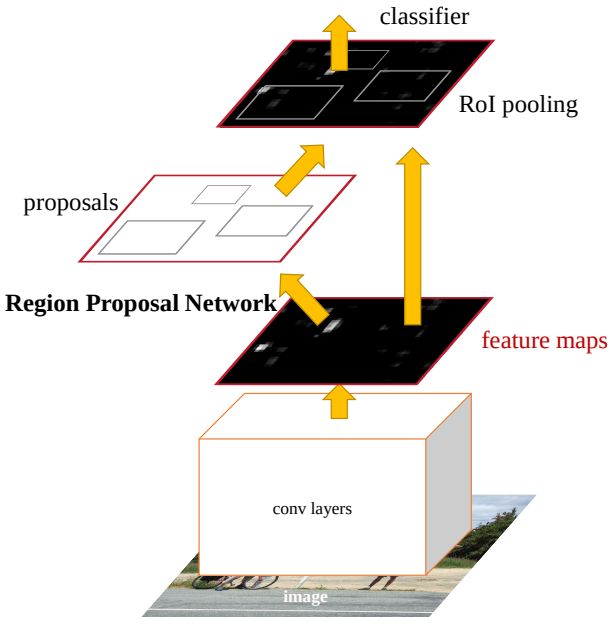

Figure 2: Faster R-CNN is a single, unified network for object detection. The RPN module serves as the 'attention' of this unified network.

into a convolutional layer for detecting multiple class-specific objects. The MultiBox methods [26], [27] generate region proposals from a network whose last fully-connected layer simultaneously predicts multiple class-agnostic boxes, generalizing the "single-box" fashion of OverFeat. These class-agnostic boxes are used as proposals for R-CNN [5]. The MultiBox proposal network is applied on a single image crop or multiple large image crops (*e.g.*, 224×224), in contrast to our fully convolutional scheme. MultiBox does not share features between the proposal and detection networks. We discuss OverFeat and MultiBox in more depth later in context with our method. Concurrent with our work, the DeepMask method [28] is developed for learning segmentation proposals.

Shared computation of convolutions [9], [1], [29], [7], [2] has been attracting increasing attention for efficient, yet accurate, visual recognition. The OverFeat paper [9] computes convolutional features from an image pyramid for classification, localization, and detection. Adaptively-sized pooling (SPP) [1] on shared convolutional feature maps is developed for efficient region-based object detection [1], [30] and semantic segmentation [29]. Fast R-CNN [2] enables end-to-end detector training on shared convolutional features and shows compelling accuracy and speed.

3 Faster R-CNN

Our object detection system, called Faster R-CNN, is composed of two modules. The first module is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN detector [2] that uses the proposed regions. The entire system is a single, unified network for object detection (Figure 2). Using the recently popular terminology of neural networks with 'attention' [31] mechanisms, the RPN module tells the Fast R-CNN module where to look. In Section 3.1 we introduce the designs and properties of the network for region proposal. In Section 3.2 we develop algorithms for training both modules with features shared.

3.1 Region Proposal Networks

A Region Proposal Network (RPN) takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score.[3] We model this process with a fully convolutional network [7], which we describe in this section. Because our ultimate goal is to share computation with a Fast R-CNN object detection network [2], we assume that both nets share a common set of convolutional layers. In our experiments, we investigate the Zeiler and Fergus model [32] (ZF), which has 5 shareable convolutional layers and the Simonyan and Zisserman model [3] (VGG-16), which has 13 shareable convolutional layers.

To generate region proposals, we slide a small network over the convolutional feature map output by the last shared convolutional layer. This small network takes as input an $n \times n$ spatial window of the input convolutional feature map. Each sliding window is mapped to a lower-dimensional feature (256-d for ZF and 512-d for VGG, with ReLU [33] following). This feature is fed into two sibling fully-connected layers—a box-regression layer (*reg*) and a box-classification layer (*cls*). We use $n = 3$ in this paper, noting that the effective receptive field on the input image is large (171 and 228 pixels for ZF and VGG, respectively). This mini-network is illustrated at a single position in Figure 3 (left). Note that because the mini-network operates in a sliding-window fashion, the fully-connected layers are shared across all spatial locations. This architecture is naturally implemented with an $n \times n$ convolutional layer followed by two sibling 1×1 convolutional layers (for *reg* and *cls*, respectively).

3.1.1 Anchors

At each sliding-window location, we simultaneously predict multiple region proposals, where the number of maximum possible proposals for each location is denoted as k. So the *reg* layer has $4k$ outputs encoding the coordinates of k boxes, and the *cls* layer outputs $2k$ scores that estimate probability of object or not object for each proposal[4]. The k proposals are parameterized *relative* to k reference boxes, which we call

3. "Region" is a generic term and in this paper we only consider *rectangular* regions, as is common for many methods (*e.g.*, [27], [4], [6]). "Objectness" measures membership to a set of object classes *vs.* background.

4. For simplicity we implement the *cls* layer as a two-class softmax layer. Alternatively, one may use logistic regression to produce k scores.

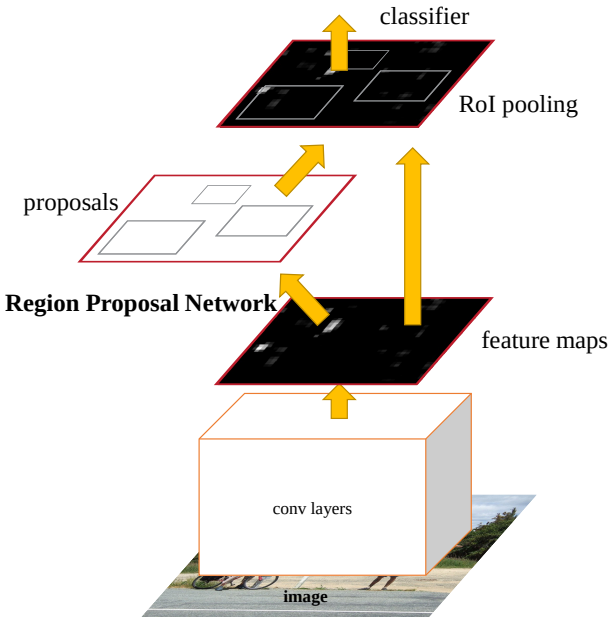

图2：Faster R-CNN是一个用于目标检测的单一统一网络。RPN模块充当该统一网络的“注意力”机制。

转换为用于检测多个类别特定对象的卷积层。MultiBox方法[26]、[27]通过一个网络生成区域提议，该网络的最后一个全连接层同时预测多个类别无关的边界框，从而推广了OverFeat的“单框”模式。这些类别无关的边界框被用作R-CNN[5]的提议。MultiBox提议网络应用于单个图像裁剪或多个大型图像裁剪（*e.g*，224×224），这与我们的全卷积方案形成对比。MultiBox在提议网络和检测网络之间不共享特征。我们将在后续结合我们的方法更深入地讨论OverFeat和MultiBox。与我们的工作同时，DeepMask方法[28]被开发用于学习分割提议。

卷积的共享计算[9]、[1]、[29]、[7]、[2]因其高效且准确的视觉识别能力而日益受到关注。OverFeat论文[9]通过图像金字塔计算卷积特征，用于分类、定位和检测。在共享卷积特征图上采用的自适应尺寸池化（SPP）[1]被开发用于高效的基于区域的目标检测[1]、[30]和语义分割[29]。Fast R-CNN[2]实现了在共享卷积特征上的端到端检测器训练，并展现出卓越的精度与速度。

3 更快的 R-CNN

我们的目标检测系统名为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，第二个模块是使用所提区域的Fast 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）。使用最近流行的带有“注意力”机制[31]的神经网络术语，RPN模块告诉Fast R-CNN模块应该关注哪些区域。在3.1节中，我们介绍了用于区域提议的网络设计和特性。在3.2节中，我们开发了用于训练两个共享特征模块的算法。

3.1 区域提议网络

区域提议网络（RPN）以任意尺寸的图像作为输入，输出一组矩形物体提议框，每个框都带有一个物体性评分。[3] 我们使用全卷积网络[7]对这一过程进行建模，并将在本节中详细描述。由于我们的最终目标是与Fast R-CNN物体检测网络[2]共享计算，我们假设两个网络共享一组共同的卷积层。在我们的实验中，我们研究了Zeiler和Fergus模型[32]（ZF，具有5个可共享卷积层）以及Simonyan和Zisserman模型[3]（VGG-16，具有13个可共享卷积层）。

为了生成区域提议，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。该小型网络以输入卷积特征图的$n \times n$空间窗口作为输入。每个滑动窗口被映射到一个低维特征（ZF网络为256维，VGG网络为512维，其后使用ReLU激活函数[33]）。该特征被送入两个并列的全连接层——边界框回归层(*reg*)和边界框分类层(*cls*)。本文采用n =3，需注意其在输入图像上的有效感受野较大（ZF和VGG网络分别为171和228像素）。图3（左）展示了该微型网络在单个位置的工作原理。值得注意的是，由于该微型网络以滑动窗口方式运行，全连接层在所有空间位置共享。该架构通过一个$n \times n$卷积层和两个并列的1×1卷积层（分别对应*reg*和*cls*）自然实现。

3.1.1 Anchors

在每个滑动窗口位置，我们同时预测多个区域建议，其中每个位置的最大可能建议数记为k。因此*reg*层具有4k个输出编码k个边界框的坐标，而*cls*层输出2k个分数，用于评估每个建议[4]属于目标或非目标的概率。这些k个建议被参数化为*relative*至k个参考框，我们称之为

3. “区域”是一个通用术语，在本文中我们仅考虑*rectangular*区域，这与许多方法（*e.g*，如[27]、[4]、[6]）的常见做法一致。“物体性”衡量的是对一组物体类别*vs*的隶属程度，而非背景。

4. 为简化起见，我们将 *cls* 层实现为一个二类 softmax 层。或者，也可以使用逻辑回归来生成 k 分数。

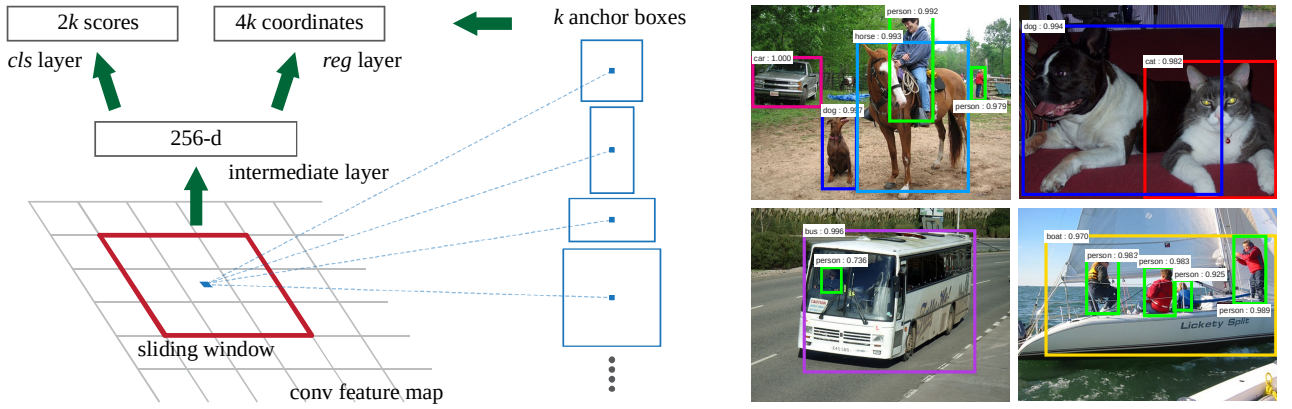

Figure 3: **Left**: Region Proposal Network (RPN). **Right**: Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

anchors. An anchor is centered at the sliding window in question, and is associated with a scale and aspect ratio (Figure 3, left). By default we use 3 scales and 3 aspect ratios, yielding $k = 9$ anchors at each sliding position. For a convolutional feature map of a size $W \times H$ (typically ~2,400), there are WHk anchors in total.

Translation-Invariant Anchors

An important property of our approach is that it is *translation invariant*, both in terms of the anchors and the functions that compute proposals relative to the anchors. If one translates an object in an image, the proposal should translate and the same function should be able to predict the proposal in either location. This translation-invariant property is guaranteed by our method[5]. As a comparison, the MultiBox method [27] uses k-means to generate 800 anchors, which are *not* translation invariant. So MultiBox does not guarantee that the same proposal is generated if an object is translated.

The translation-invariant property also reduces the model size. MultiBox has a $(4+1) \times 800$-dimensional fully-connected output layer, whereas our method has a $(4+2) \times 9$-dimensional convolutional output layer in the case of $k = 9$ anchors. As a result, our output layer has 2.8×10^4 parameters ($512 \times (4+2) \times 9$ for VGG-16), two orders of magnitude fewer than MultiBox's output layer that has 6.1×10^6 parameters ($1536 \times (4+1) \times 800$ for GoogleNet [34] in MultiBox [27]). If considering the feature projection layers, our proposal layers still have an order of magnitude fewer parameters than MultiBox[6]. We expect our method to have less risk of overfitting on small datasets, like PASCAL VOC.

5. As is the case of FCNs [7], our network is translation invariant up to the network's total stride.

6. Considering the feature projection layers, our proposal layers' parameter count is $3 \times 3 \times 512 \times 512 + 512 \times 6 \times 9 = 2.4 \times 10^6$; MultiBox's proposal layers' parameter count is $7 \times 7 \times (64+96+64+64) \times 1536 + 1536 \times 5 \times 800 = 27 \times 10^6$.

Multi-Scale Anchors as Regression References

Our design of anchors presents a novel scheme for addressing multiple scales (and aspect ratios). As shown in Figure 1, there have been two popular ways for multi-scale predictions. The first way is based on image/feature pyramids, *e.g.*, in DPM [8] and CNN-based methods [9], [1], [2]. The images are resized at multiple scales, and feature maps (HOG [8] or deep convolutional features [9], [1], [2]) are computed for each scale (Figure 1(a)). This way is often useful but is time-consuming. The second way is to use sliding windows of multiple scales (and/or aspect ratios) on the feature maps. For example, in DPM [8], models of different aspect ratios are trained separately using different filter sizes (such as 5×7 and 7×5). If this way is used to address multiple scales, it can be thought of as a "pyramid of filters" (Figure 1(b)). The second way is usually adopted jointly with the first way [8].

As a comparison, our anchor-based method is built on *a pyramid of anchors*, which is more cost-efficient. Our method classifies and regresses bounding boxes with reference to anchor boxes of multiple scales and aspect ratios. It only relies on images and feature maps of a single scale, and uses filters (sliding windows on the feature map) of a single size. We show by experiments the effects of this scheme for addressing multiple scales and sizes (Table 8).

Because of this multi-scale design based on anchors, we can simply use the convolutional features computed on a single-scale image, as is also done by the Fast R-CNN detector [2]. The design of multi-scale anchors is a key component for sharing features without extra cost for addressing scales.

3.1.2 Loss Function

For training RPNs, we assign a binary class label (of being an object or not) to each anchor. We assign a positive label to two kinds of anchors: (i) the anchor/anchors with the highest Intersection-over-Union (IoU) overlap with a ground-truth box, *or* (ii) an anchor that has an IoU overlap higher than 0.7 with

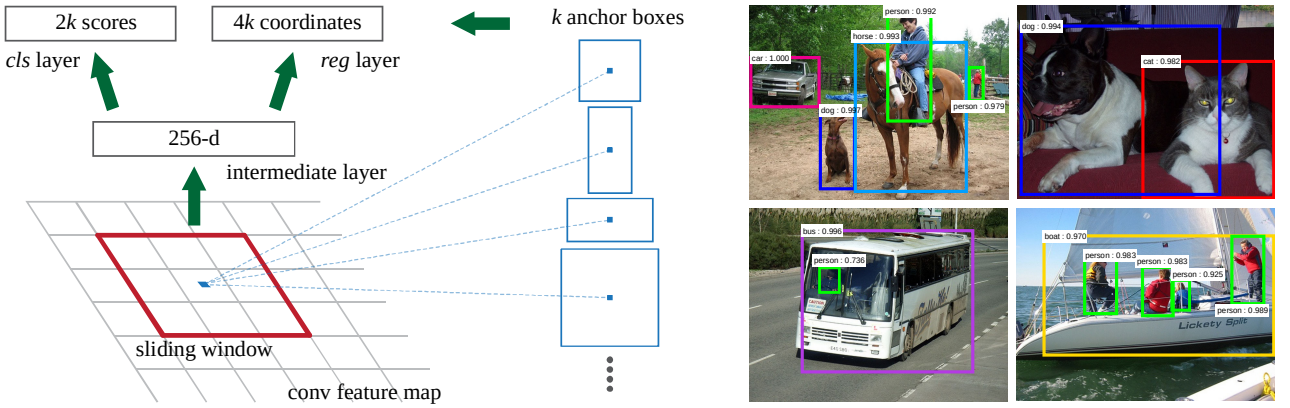

图3：左：区域提议网络（RPN）。右：在PASCA上使用RPN提议的检测示例 L
VOC 2007 测试集。我们的方法能够检测多种尺度和宽高比的目标物体。

*anchors*一个锚点以所讨论的滑动窗口为中心，并与一个尺度和长宽比相关联（图3，左）。默认情况下，我们使用3个尺度和3个长宽比，在每个滑动位置产生$k=9$个锚点。对于一个大小为$W \times H$ (通常为~2,400)的卷积特征图，总共有WHk个锚点。

平移不变锚点

我们方法的一个重要特性是，它在锚点和计算相对于锚点的提议函数方面都具有*translation invariant*。如果图像中的物体发生平移，提议框也应相应平移，且同一函数应能预测任意位置的提议框。这一平移不变性由我们的方法[5]保证。作为对比，MultiBox方法[27]使用k-means生成800个锚点，这些锚点*not*平移不变性。因此，当物体平移时，MultiBox无法保证生成相同的提议框。

平移不变性特性还减小了模型尺寸。MultiBox拥有一个(4+1)×800维全连接输出层，而我们的方法在采用$k=9$个锚点的情况下，使用了一个(4+2)×9维卷积输出层。因此，我们的输出层仅含2.8×10^4个参数（VGG-16架构下为512×（4+2）×9），比MultiBox输出层的6.1×10^6个参数（MultiBox[27]中GoogleNet[34]为1536×（4+1）×800）少两个数量级。若计入特征投影层，我们提出的检测层参数仍比MultiBox[6]少一个数量级。我们预期该方法在PASCAL VOC这类小数据集上具有更低的过拟合风险。

多尺度锚点作为回归参考

我们设计的锚点提出了一种处理多尺度（及宽高比）的新颖方案。如图1所示，多尺度预测目前存在两种主流方法。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]中采用的方案。图像被缩放到多个尺度，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)）。这种方法通常有效但耗时较多。第二种方法是在特征图上使用多尺度（及/或宽高比）的滑动窗口。例如在DPM[8]中，通过不同滤波器尺寸（如5×7和7×5）分别训练不同宽高比的模型。若将此方法用于处理多尺度问题，可将其视为“滤波器金字塔”（图1(b)）。第二种方法通常与第一种方法结合使用[8]。

作为对比，我们的基于锚点的方法建立在*a pyramid of anchors*之上，这种方法更具成本效益。我们的方法参考多尺度和宽高比的锚框对边界框进行分类和回归。它仅依赖于单尺度的图像和特征图，并使用单一尺寸的滤波器（特征图上的滑动窗口）。我们通过实验展示了该方案在处理多尺度和尺寸方面的效果（表8）。

由于这种基于锚点的多尺度设计，我们可以简单地使用在单尺度图像上计算出的卷积特征，正如Fast R-CNN检测器[2]所做的那样。多尺度锚点的设计是实现特征共享而无需额外处理尺度问题的关键组成部分。

3.1.2 Loss Function

在训练RPN时，我们为每个锚框分配一个二元类别标签（即是否属于物体）。我们将正标签赋予两类锚框：(i) 与真实边界框具有最高交并比（IoU）的锚框，*or* (ii) 以及 (ii) 与任意真实边界框的IoU重叠度高于0.7的锚框。

5. As is the case of FCNs [7], our network is translation invariant up to the network's total stride.

6. Considering the feature projection layers, our proposal layers' parameter count is $3\times3\times512\times512+512\times6\times9=2.4\times10^6$; MultiBox's proposal layers' parameter count is $7\times7\times(64+96+64+64)\times1536+1536\times5\times800=27\times10^6$.

any ground-truth box. Note that a single ground-truth box may assign positive labels to multiple anchors. Usually the second condition is sufficient to determine the positive samples; but we still adopt the first condition for the reason that in some rare cases the second condition may find no positive sample. We assign a negative label to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes. Anchors that are neither positive nor negative do not contribute to the training objective.

With these definitions, we minimize an objective function following the multi-task loss in Fast R-CNN [2]. Our loss function for an image is defined as:

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

Here, i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i being an object. The ground-truth label p_i^* is 1 if the anchor is positive, and is 0 if the anchor is negative. t_i is a vector representing the 4 parameterized coordinates of the predicted bounding box, and t_i^* is that of the ground-truth box associated with a positive anchor. The classification loss L_{cls} is log loss over two classes (object *vs*. not object). For the regression loss, we use $L_{reg}(t_i, t_i^*) = R(t_i - t_i^*)$ where R is the robust loss function (smooth L_1) defined in [2]. The term $p_i^* L_{reg}$ means the regression loss is activated only for positive anchors ($p_i^* = 1$) and is disabled otherwise ($p_i^* = 0$). The outputs of the *cls* and *reg* layers consist of $\{p_i\}$ and $\{t_i\}$ respectively.

The two terms are normalized by N_{cls} and N_{reg} and weighted by a balancing parameter λ. In our current implementation (as in the released code), the cls term in Eqn.(1) is normalized by the mini-batch size (*i.e.*, $N_{cls} = 256$) and the reg term is normalized by the number of anchor locations (*i.e.*, $N_{reg} \sim 2,400$). By default we set $\lambda = 10$, and thus both *cls* and *reg* terms are roughly equally weighted. We show by experiments that the results are insensitive to the values of λ in a wide range (Table 9). We also note that the normalization as above is not required and could be simplified.

For bounding box regression, we adopt the parameterizations of the 4 coordinates following [5]:

$$\begin{aligned} t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}), \\ t_{\mathrm{x}}^* &= (x^* - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}}^* = (y^* - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}}^* &= \log(w^*/w_{\mathrm{a}}), \quad t_{\mathrm{h}}^* = \log(h^*/h_{\mathrm{a}}), \end{aligned} \quad (2)$$

where x, y, w, and h denote the box's center coordinates and its width and height. Variables x, x_{a}, and x^* are for the predicted box, anchor box, and ground-truth box respectively (likewise for y, w, h). This can be thought of as bounding-box regression from an anchor box to a nearby ground-truth box.

Nevertheless, our method achieves bounding-box regression by a different manner from previous RoI-based (Region of Interest) methods [1], [2]. In [1], [2], bounding-box regression is performed on features pooled from *arbitrarily* sized RoIs, and the regression weights are *shared* by all region sizes. In our formulation, the features used for regression are of the *same* spatial size (3×3) on the feature maps. To account for varying sizes, a set of k bounding-box regressors are learned. Each regressor is responsible for one scale and one aspect ratio, and the k regressors do *not* share weights. As such, it is still possible to predict boxes of various sizes even though the features are of a fixed size/scale, thanks to the design of anchors.

3.1.3 Training RPNs

The RPN can be trained end-to-end by back-propagation and stochastic gradient descent (SGD) [35]. We follow the "image-centric" sampling strategy from [2] to train this network. Each mini-batch arises from a single image that contains many positive and negative example anchors. It is possible to optimize for the loss functions of all anchors, but this will bias towards negative samples as they are dominate. Instead, we randomly sample 256 anchors in an image to compute the loss function of a mini-batch, where the sampled positive and negative anchors have a ratio of *up to* 1:1. If there are fewer than 128 positive samples in an image, we pad the mini-batch with negative ones.

We randomly initialize all new layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. All other layers (*i.e.*, the shared convolutional layers) are initialized by pre-training a model for ImageNet classification [36], as is standard practice [5]. We tune all layers of the ZF net, and conv3_1 and up for the VGG net to conserve memory [2]. We use a learning rate of 0.001 for 60k mini-batches, and 0.0001 for the next 20k mini-batches on the PASCAL VOC dataset. We use a momentum of 0.9 and a weight decay of 0.0005 [37]. Our implementation uses Caffe [38].

3.2 Sharing Features for RPN and Fast R-CNN

Thus far we have described how to train a network for region proposal generation, without considering the region-based object detection CNN that will utilize these proposals. For the detection network, we adopt Fast R-CNN [2]. Next we describe algorithms that learn a unified network composed of RPN and Fast R-CNN with shared convolutional layers (Figure 2).

Both RPN and Fast R-CNN, trained independently, will modify their convolutional layers in different ways. We therefore need to develop a technique that allows for sharing convolutional layers between the

任何真实框。注意，单个真实框可能会为多个锚框分配正标签。通常第二个条件足以确定正样本；但我们仍采用第一个条件，因为在某些罕见情况下，第二个条件可能找不到正样本。对于所有真实框的交并比均低于0.3的非正锚框，我们为其分配负标签。既非正也非负的锚框不会对训练目标产生贡献。

根据这些定义，我们按照Fast R-CNN[2]中的多任务损失最小化目标函数。针对单张图像的损失函数定义为：

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

这里，i 是 mini-batch 中一个锚点的索引，p_i 是锚点 i 被预测为物体的概率。如果锚点为正样本，则其真实标签 p_i^* 为 1；如果锚点为负样本，则 p_i^* 为 0。t_i 是一个表示预测边界框 4 个参数化坐标的向量，t_i^* 则是与正样本锚点关联的真实边界框的对应坐标向量。分类损失 L_{cls} 是二分类（物体 *vs* 与非物体）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] 中定义的鲁棒损失函数（平滑 L_1）。项 $p_i^* L_{reg}$ 表示回归损失仅对正样本锚点激活（$p_i^* =$ 为 1），否则不激活（$p_i^* =$ 为 0）。*cls* 层和 *reg* 层的输出分别由 $\{p_i\}$ 和 $\{t_i\}$ 组成。

这两项分别通过N_{cls}和N_{reg}进行归一化，并通过平衡参数λ加权。在我们当前的实现（如已发布的代码所示）中，公式(1)中的cls项通过小批量大小（*i.e*，即 $N_{cls} = 256$）归一化，而reg项则通过锚点位置数量（*i.e*，即$N_{reg} \sim 2,400$）归一化。默认情况下，我们设定 $\lambda =$为10，因此*cls*和*reg*两项的权重大致相等。实验表明，结果在λ的广泛取值范围内均不敏感（见表9）。我们还注意到，上述归一化并非必需，可以进行简化。

对于边界框回归，我们采用[5]中4个坐标的参数化方法：

$$\begin{aligned} t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}), \\ t_{\mathrm{x}}^* &= (x^* - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}}^* = (y^* - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}}^* &= \log(w^*/w_{\mathrm{a}}), \quad t_{\mathrm{h}}^* = \log(h^*/h_{\mathrm{a}}), \end{aligned} \tag{2}$$

其中x、y、w和h表示边界框的中心坐标及其宽度和高度。变量x、x_{a}和x^*分别对应预测框、锚框和真实框（y, w, h同理）。

可以被视为从锚框到附近真实框的边界框回归。

尽管如此，我们的方法通过不同于以往基于感兴趣区域（RoI）的方法[1]、[2]的方式实现边界框回归。在[1]、[2]中，边界框回归是在从*arbitrarily*大小的RoI中池化得到的特征上进行的，且回归权重对所有区域尺寸都是*shared*。在我们的公式中，用于回归的特征在特征图上具有*same*的空间尺寸（3×3）。为了适应不同尺寸，我们学习了一组k个边界框回归器。每个回归器负责一个尺度和一个宽高比，且这些k个回归器*not*共享权重。因此，即使特征具有固定尺寸/尺度，得益于锚点的设计，我们仍能预测各种尺寸的边界框。

3.1.3 Training RPNs

RPN可以通过反向传播和随机梯度下降（SGD）进行端到端训练[35]。我们采用文献[2]中的“以图像为中心”采样策略来训练该网络。每个小批量数据来自单张图像，其中包含许多正负样本锚点。虽然可以对所有锚点的损失函数进行优化，但由于负样本占主导地位，这将导致训练偏向负样本。因此，我们改为在每张图像中随机采样256个锚点来计算小批量的损失函数，其中正负锚点的采样比例保持为*up to* 1:1。若图像中正样本少于128个，则使用负样本填充该小批量。

我们通过从标准差为0.01的零均值高斯分布中抽取权重，随机初始化所有新层。所有其他层（*i.e*，即共享卷积层）则按照标准做法[5]，通过为ImageNet分类任务预训练模型进行初始化[36]。为节省内存[2]，我们对ZF网络的所有层进行调优，对VGG网络则调整conv3_1及以上层。在PASCAL VOC数据集上，我们使用0.001的学习率训练60k个小批量，随后使用0.0001的学习率再训练20k个小批量。我们采用0.9的动量和0.0005的权重衰减[37]。我们的实现基于Caffe[38]。

3.2 RPN与Fast R-CNN的特征共享

到目前为止，我们描述了如何训练用于生成区域建议的网络，而没有考虑将利用这些建议的基于区域的目标检测CNN。对于检测网络，我们采用Fast R-CNN [2]。接下来，我们将描述学习由RPN和Fast R-CNN组成的统一网络的算法，该网络具有共享的卷积层（图2）。

RPN和Fast R-CNN在独立训练时，会以不同的方式修改各自的卷积层。因此我们需要开发一种技术，使得{v*}能够在两者间共享卷积层。

Table 1: the learned average proposal size for each anchor using the ZF net (numbers for $s = 600$).

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

two networks, rather than learning two separate networks. We discuss three ways for training networks with features shared:

(i) *Alternating training*. In this solution, we first train RPN, and use the proposals to train Fast R-CNN. The network tuned by Fast R-CNN is then used to initialize RPN, and this process is iterated. This is the solution that is used in all experiments in this paper.

(ii) *Approximate joint training*. In this solution, the RPN and Fast R-CNN networks are merged into one network during training as in Figure 2. In each SGD iteration, the forward pass generates region proposals which are treated just like fixed, pre-computed proposals when training a Fast R-CNN detector. The backward propagation takes place as usual, where for the shared layers the backward propagated signals from both the RPN loss and the Fast R-CNN loss are combined. This solution is easy to implement. But this solution ignores the derivative w.r.t. the proposal boxes' coordinates that are also network responses, so is approximate. In our experiments, we have empirically found this solver produces close results, yet reduces the training time by about 25-50% comparing with alternating training. This solver is included in our released Python code.

(iii) *Non-approximate joint training*. As discussed above, the bounding boxes predicted by RPN are also functions of the input. The RoI pooling layer [2] in Fast R-CNN accepts the convolutional features and also the predicted bounding boxes as input, so a theoretically valid backpropagation solver should also involve gradients w.r.t. the box coordinates. These gradients are ignored in the above approximate joint training. In a non-approximate joint training solution, we need an RoI pooling layer that is differentiable w.r.t. the box coordinates. This is a nontrivial problem and a solution can be given by an "RoI warping" layer as developed in [15], which is beyond the scope of this paper.

4-Step Alternating Training. In this paper, we adopt a pragmatic 4-step training algorithm to learn shared features via alternating optimization. In the first step, we train the RPN as described in Section 3.1.3. This network is initialized with an ImageNet-pre-trained model and fine-tuned end-to-end for the region proposal task. In the second step, we train a separate detection network by Fast R-CNN using the proposals generated by the step-1 RPN. This detection network is also initialized by the ImageNet-pre-trained model. At this point the two networks do not share convolutional layers. In the third step, we use the detector network to initialize RPN training, but we fix the shared convolutional layers and only fine-tune the layers unique to RPN. Now the two networks share convolutional layers. Finally, keeping the shared convolutional layers fixed, we fine-tune the unique layers of Fast R-CNN. As such, both networks share the same convolutional layers and form a unified network. A similar alternating training can be run for more iterations, but we have observed negligible improvements.

3.3 Implementation Details

We train and test both region proposal and object detection networks on images of a single scale [1], [2]. We re-scale the images such that their shorter side is $s = 600$ pixels [2]. Multi-scale feature extraction (using an image pyramid) may improve accuracy but does not exhibit a good speed-accuracy trade-off [2]. On the re-scaled images, the total stride for both ZF and VGG nets on the last convolutional layer is 16 pixels, and thus is ∼10 pixels on a typical PASCAL image before resizing (∼500×375). Even such a large stride provides good results, though accuracy may be further improved with a smaller stride.

For anchors, we use 3 scales with box areas of 128^2, 256^2, and 512^2 pixels, and 3 aspect ratios of 1:1, 1:2, and 2:1. These hyper-parameters are *not* carefully chosen for a particular dataset, and we provide ablation experiments on their effects in the next section. As discussed, our solution does not need an image pyramid or filter pyramid to predict regions of multiple scales, saving considerable running time. Figure 3 (right) shows the capability of our method for a wide range of scales and aspect ratios. Table 1 shows the learned average proposal size for each anchor using the ZF net. We note that our algorithm allows predictions that are larger than the underlying receptive field. Such predictions are not impossible—one may still roughly infer the extent of an object if only the middle of the object is visible.

The anchor boxes that cross image boundaries need to be handled with care. During training, we ignore all cross-boundary anchors so they do not contribute to the loss. For a typical 1000×600 image, there will be roughly 20000 ($\approx 60 \times 40 \times 9$) anchors in total. With the cross-boundary anchors ignored, there are about 6000 anchors per image for training. If the boundary-crossing outliers are not ignored in training, they introduce large, difficult to correct error terms in the objective, and training does not converge. During testing, however, we still apply the fully convolutional RPN to the entire image. This may generate cross-boundary proposal boxes, which we clip to the image boundary.

表1：使用ZF网络学习的每个锚点的平均提议大小（$s = 600$的数值）。锚点 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 提议 188×111 113×114 70×92 416×229 261×284 174×332 768×437 499×501 355×715

两个网络，而不是学习两个独立的网络。我们讨论了三种训练具有共享特征网络的方法：

(i) *Alternating training*。在此解决方案中，我们首先训练RPN，并使用其生成的候选区域来训练Fast R-CNN。随后，用经过Fast R-CNN调优的网络初始化RPN，并迭代这一过程。本文所有实验均采用此方案。

(ii) *Approximate joint training*。在此解决方案中，RPN和Fast R-CNN网络在训练期间合并为一个网络，如图2所示。在每次SGD迭代中，前向传播会生成区域建议，这些建议在训练Fast R-CNN检测器时被视为固定的、预计算的建议。反向传播照常进行，对于共享层，来自RPN损失和Fast R-CNN损失的反向传播信号会被合并。该方案易于实现，但忽略了针对同样作为网络响应的建议框坐标的导数，因此是近似解。在我们的实验中，我们经验性地发现此求解器能产生相近的结果，且与交替训练相比减少了约25-50%的训练时间。该求解器已包含在我们发布的Python代码中。

(iii) *Non-approximate joint training*。如上所述，RPN预测的边界框也是输入的函数。Fast R-CNN中的RoI池化层[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包含关于边界框坐标的梯度。在上述近似联合训练中，这些梯度被忽略了。在非近似的联合训练解决方案中，我们需要一个关于边界框坐标可微的RoI池化层。这是一个重要问题，可以通过[15]中提出的“RoI变形”层来解决，但这已超出本文的讨论范围。

四步交替训练。本文采用一种实用的四步训练算法，通过交替优化学习共享特征。第一步，我们按照第3.1.3节所述训练区域提议网络（RPN）。该网络以ImageNet预训练模型初始化，并针对区域提议任务进行端到端微调。第二步，我们使用第一步RPN生成的提议框，通过Fast R-CNN训练独立的检测网络。该检测网络同样由ImageNet预训练模型初始化。此时两个网络尚未共享卷积层。第三步，我们使用检测器网络初始化RPN训练，但保持{v*}不变。固定共享的卷积层，仅对RPN特有的层进行微调。现在两个网络共享卷积层。最后，保持共享卷积层固定，我们对Fast R-CNN特有的层进行微调。这样，两个网络共享相同的卷积层，形成一个统一的网络。可以进行更多轮次的类似交替训练，但我们观察到改进微乎其微。

3.3 实现细节

我们在单一尺度的图像上训练和测试区域提议网络和物体检测网络[1], [2]。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）可能会提高精度，但未能展现出良好的速度-精度权衡[2]。在重新缩放的图像上，ZF和VGG网络在最后一个卷积层的总步长为16像素，因此在调整大小前的典型PASCAL图像上为∼10像素（∼500×375）。即使这样大的步长也能提供良好的结果，尽管使用更小的步长可能进一步提高精度。

对于锚点，我们使用3种尺度，其边界框面积分别为128^2、256^2和512^2像素，以及3种长宽比：1:1、1:2和2:1。这些超参数是针对特定数据集*not*精心选择的，我们将在下一节中提供关于它们影响的消融实验。如前所述，我们的解决方案不需要图像金字塔或滤波器金字塔来预测多尺度区域，从而节省了大量运行时间。图3（右）展示了我们的方法在多种尺度和长宽比上的能力。表1展示了使用ZF网络学习到的每个锚点的平均建议框尺寸。我们注意到，我们的算法允许预测大于基础感受野的区域。这种预测并非不可能——即使只能看到物体的中间部分，人们仍然可以大致推断出物体的范围。

跨越图像边界的锚框需要谨慎处理。在训练过程中，我们忽略所有跨界锚框，使其不参与损失计算。对于典型的1000 × 600图像，总共约有20000个（≈ 60 × 40 × 9）锚框。忽略跨界锚框后，每张图像约有6000个锚框用于训练。若在训练中不忽略这些边界异常值，它们会在目标函数中引入难以纠正的大误差项，导致训练无法收敛。然而在测试阶段，我们仍将全卷积RPN应用于整张图像。这可能会产生跨界提议框，我们会将其裁剪至图像边界。

Table 2: Detection results on **PASCAL VOC 2007 test set** (trained on VOC 2007 trainval). The detectors are Fast R-CNN with ZF, but using various proposal methods for training and testing.

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

Some RPN proposals highly overlap with each other. To reduce redundancy, we adopt non-maximum suppression (NMS) on the proposal regions based on their *cls* scores. We fix the IoU threshold for NMS at 0.7, which leaves us about 2000 proposal regions per image. As we will show, NMS does not harm the ultimate detection accuracy, but substantially reduces the number of proposals. After NMS, we use the top-N ranked proposal regions for detection. In the following, we train Fast R-CNN using 2000 RPN proposals, but evaluate different numbers of proposals at test-time.

4 Experiments

4.1 Experiments on PASCAL VOC

We comprehensively evaluate our method on the PASCAL VOC 2007 detection benchmark [11]. This dataset consists of about 5k trainval images and 5k test images over 20 object categories. We also provide results on the PASCAL VOC 2012 benchmark for a few models. For the ImageNet pre-trained network, we use the "fast" version of ZF net [32] that has 5 convolutional layers and 3 fully-connected layers, and the public VGG-16 model[7] [3] that has 13 convolutional layers and 3 fully-connected layers. We primarily evaluate detection mean Average Precision (mAP), because this is the actual metric for object detection (rather than focusing on object proposal proxy metrics).

Table 2 (top) shows Fast R-CNN results when trained and tested using various region proposal methods. These results use the ZF net. For Selective Search (SS) [4], we generate about 2000 proposals by the "fast" mode. For EdgeBoxes (EB) [6], we generate the proposals by the default EB setting tuned for 0.7 IoU. SS has an mAP of 58.7% and EB has an mAP of 58.6% under the Fast R-CNN framework. RPN with Fast R-CNN achieves competitive results, with an mAP of 59.9% while using *up to* 300 proposals[8]. Using RPN yields a much faster detection system than using either SS or EB because of shared convolutional computations; the fewer proposals also reduce the region-wise fully-connected layers' cost (Table 5).

Ablation Experiments on RPN. To investigate the behavior of RPNs as a proposal method, we conducted several ablation studies. First, we show the effect of sharing convolutional layers between the RPN and Fast R-CNN detection network. To do this, we stop after the second step in the 4-step training process. Using separate networks reduces the result slightly to 58.7% (RPN+ZF, unshared, Table 2). We observe that this is because in the third step when the detector-tuned features are used to fine-tune the RPN, the proposal quality is improved.

Next, we disentangle the RPN's influence on training the Fast R-CNN detection network. For this purpose, we train a Fast R-CNN model by using the 2000 SS proposals and ZF net. We fix this detector and evaluate the detection mAP by changing the proposal regions used at test-time. In these ablation experiments, the RPN does not share features with the detector.

Replacing SS with 300 RPN proposals at test-time leads to an mAP of 56.8%. The loss in mAP is because of the inconsistency between the training/testing proposals. This result serves as the baseline for the following comparisons.

Somewhat surprisingly, the RPN still leads to a competitive result (55.1%) when using the top-ranked

7. www.robots.ox.ac.uk/~vgg/research/very_deep/

8. For RPN, the number of proposals (*e.g.*, 300) is the maximum number for an image. RPN may produce fewer proposals after NMS, and thus the average number of proposals is smaller.

表2：在PASCAL VOC 2007测试集上的检测结果（使用VOC 2007 trainval训练）。检测器a 重新使用ZF的Fast R-CNN，但在训练和测试中采用多种候选区域生成方法。

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

一些RPN提议彼此高度重叠。为了减少冗余，我们基于提议区域的*cls*分数采用非极大值抑制（NMS）。我们将NMS的IoU阈值固定为0.7，这样每张图像大约留下2000个提议区域。正如我们将展示的，NMS不会损害最终的检测精度，但显著减少了提议数量。在NMS之后，我们使用排名前*N*的提议区域进行检测。接下来，我们使用2000个RPN提议训练Fast R-CNN，但在测试时评估不同数量的提议。

4 实验

4.1 PASCAL VOC 实验

我们在PASCAL VOC 2007检测基准[11]上全面评估了我们的方法。该数据集包含约5千张训练验证图像和5千张测试图像，涵盖20个物体类别。我们还为部分模型提供了在PASCAL VOC 2012基准上的结果。对于ImageNet预训练网络，我们使用具有5个卷积层和3个全连接层的ZF网络“快速”版本[32]，以及具有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均值（mAP），因为这是物体检测的实际衡量标准（而非关注物体候选区域代理指标）。

表2（上）展示了使用不同区域提议方法进行训练和测试时Fast R-CNN的结果。这些结果使用了ZF网络。对于选择性搜索（SS）[4]，我们通过“快速”模式生成约2000个提议。对于EdgeBoxes（EB）[6]，我们采用针对0.7 IoU阈值调整的默认EB设置生成提议。

7. www.robots.ox.ac.uk/~vgg/research/very_deep/

IoU。在Fast R-CNN框架下，SS的mAP为58.7%，EB的mAP为58.6%。结合Fast R-CNN的RPN取得了具有竞争力的结果，在使用*up to*300个提议[8]的情况下mAP达到59.9%。由于共享卷积计算，使用RPN比使用SS或EB构建的检测系统快得多；更少的提议也降低了区域全连接层的成本（表5）。

RPN消融实验。为了研究RPN作为候选区域生成方法的表现，我们进行了多项消融实验。首先，我们展示了RPN与Fast R-CNN检测网络共享卷积层的效果。为此，我们在四步训练过程的第二步后停止训练。使用独立网络会使结果略微下降至58.7%（RPN+ZF，非共享，表2）。我们发现这是因为在第三步中，当使用检测器调优后的特征微调RPN时，候选区域质量得到了提升。

接下来，我们解析RPN对训练Fast R-CNN检测网络的影响。为此，我们使用2000个SS建议框和ZF网络训练了一个Fast R-CNN模型。我们固定该检测器，并通过在测试时改变使用的建议区域来评估检测mAP。在这些消融实验中，RPN与检测器不共享特征。

在测试时用300个RPN建议替换SS，得到的mAP为56.8%。mAP的下降是由于训练和测试建议之间的不一致性造成的。这一结果将作为后续比较的基线。

有些令人惊讶的是，当使用排名最高的

8. For RPN, the number of proposals (*e.g.*, 300) is the maximum number for an image. RPN may produce fewer proposals after NMS, and thus the average number of proposals is smaller.

Table 3: Detection results on **PASCAL VOC 2007 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07+12": union set of VOC 2007 trainval and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2000. †: this number was reported in [2]; using the repository provided by this paper, this result is higher (68.1).

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**

Table 4: Detection results on **PASCAL VOC 2012 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07++12": union set of VOC 2007 trainval+test and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 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.

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**

Table 5: **Timing** (ms) on a K40 GPU, except SS proposal is evaluated in a CPU. "Region-wise" includes NMS, pooling, fully-connected, and softmax layers. See our released code for the profiling of running time.

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**

100 proposals at test-time, indicating that the top-ranked RPN proposals are accurate. On the other extreme, using the top-ranked 6000 RPN proposals (without NMS) has a comparable mAP (55.2%), suggesting NMS does not harm the detection mAP and may reduce false alarms.

Next, we separately investigate the roles of RPN's *cls* and *reg* outputs by turning off either of them at test-time. When the *cls* layer is removed at test-time (thus no NMS/ranking is used), we randomly sample N proposals from the unscored regions. The mAP is nearly unchanged with $N = 1000$ (55.8%), but degrades considerably to 44.6% when $N = 100$. This shows that the *cls* scores account for the accuracy of the highest ranked proposals.

On the other hand, when the *reg* layer is removed at test-time (so the proposals become anchor boxes), the mAP drops to 52.1%. This suggests that the high-quality proposals are mainly due to the regressed box bounds. The anchor boxes, though having multiple scales and aspect ratios, are not sufficient for accurate detection.

We also evaluate the effects of more powerful networks on the proposal quality of RPN alone. We use VGG-16 to train the RPN, and still use the above detector of SS+ZF. The mAP improves from 56.8% (using RPN+ZF) to 59.2% (using RPN+VGG). This is a promising result, because it suggests that the proposal quality of RPN+VGG is better than that of RPN+ZF. Because proposals of RPN+ZF are competitive with SS (both are 58.7% when consistently used for training and testing), we may expect RPN+VGG to be better than SS. The following experiments justify this hypothesis.

Performance of VGG-16. Table 3 shows the results of VGG-16 for both proposal and detection. Using RPN+VGG, the result is 68.5% for *unshared* features, slightly higher than the SS baseline. As shown above, this is because the proposals generated by RPN+VGG are more accurate than SS. Unlike SS that is pre-defined, the RPN is actively trained and benefits from better networks. For the feature-*shared* variant, the result is 69.9%—better than the strong SS baseline, yet with nearly cost-free proposals. We further train the RPN and detection network on the union set of PASCAL VOC 2007 trainval and 2012 trainval. The mAP is **73.2%**. Figure 5 shows some results on the PASCAL VOC 2007 test set. On the PASCAL VOC 2012 test set (Table 4), our method has an mAP of **70.4%** trained on the union set of VOC 2007 trainval+test and VOC 2012 trainval. Table 6 and Table 7 show the detailed numbers.

表3：在PASCAL VOC 2007测试集上的检测结果。检测器为Fast R-CNN和VGG-16。训练数据："07"：VOC 2007训练验证集，"07+12"：VOC 2007训练验证集与VOC 2012训练验证集的并集。对于RPN，Fast R-CNN的训练阶段建议框数量为2000个。†：该数值引自文献[2]；使用本文提供的代码库时，此结果更高（68.1）。

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4：PASCAL VOC 2012测试集上的检测结果。检测器为Fast R-CNN与VGG-16。训练数据："07"：VOC 2007训练验证集，"07++12"：VOC 2007训练验证集+测试集与VOC 2012训练验证集的并集。对于RPN，Fast R-CNN训练阶段使用的候选框数量为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。

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5：在K40 GPU上的计时（毫秒），其中SS候选框生成在CPU上评估。"区域级"包括NMS、池化、全连接和softmax层。运行时间分析请参见我们发布的代码。

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**

在测试阶段使用100个提议，表明排名靠前的RPN提议是准确的。在另一个极端，使用排名前6000的RPN提议（不使用NMS）获得了可比的mAP（55.2%），这表明NMS不会损害检测mAP，并可能减少误报。

接下来，我们通过测试时关闭RPN的*cls*或*reg*输出，分别研究它们的作用。当测试时移除*cls*层（即不使用NMS/排序）时，我们从无评分区域随机采样N个候选框。当N =为1000时，mAP几乎保持不变（55.8%），但当N =为100时，mAP显著下降至44.6%。这表明*cls*分数决定了最高排名候选框的准确性。

另一方面，当在测试时移除*reg*层（此时建议框变为锚框），mAP下降至52.1%。这表明高质量的建议框主要归功于回归得到的边界框。尽管锚框具有多尺度和宽高比，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训练RPN，并继续采用上述SS+ZF检测器。mAP从56.8%得到提升（使用RPN+ZF）提升至59.2%（使用RPN+VGG）。这是一个令人鼓舞的结果，因为它表明RPN+VGG的候选区域质量优于RPN+ZF。由于RPN+ZF的候选区域与SS方法表现相当（在训练和测试中一致使用时均为58.7%），我们可以预期RPN+VGG将优于SS。后续实验验证了这一假设。

VGG-16的性能。表3展示了VGG-16在候选区域生成与目标检测两方面的结果。使用RPN+VGG时，基于*unshared*特征的结果为68.5%，略高于SS基线。如上所述，这是因为RPN+VGG生成的候选区域比SS更精确。与预定义的SS不同，RPN经过主动训练并能从更优的网络中受益。对于特征*shared*变体，结果达到69.9%——优于强大的SS基线，且候选区域生成几乎无需额外计算成本。我们进一步在PASCAL VOC 2007 trainval与2012 trainval的联合数据集上训练RPN和检测网络，获得了73.2%的mAP。图5展示了在PASCAL VOC 2007测试集上的部分结果。在PASCAL VOC 2012测试集上（表4），我们的方法在VOC 2007 trainval+test与VOC 2012 trainval联合训练后达到70.4%的mAP。详细数据见表6与表7。

Table 6: Results on PASCAL VOC 2007 test set with Fast R-CNN detectors and VGG-16. For RPN, the train-time proposals for Fast R-CNN are 2000. RPN* denotes the unsharing feature version.

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**

Table 7: Results on PASCAL VOC 2012 test set with Fast R-CNN detectors and VGG-16. For RPN, the train-time proposals for Fast R-CNN are 2000.

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**

Table 8: Detection results of Faster R-CNN on PASCAL VOC 2007 test set using **different settings of anchors**. The network is VGG-16. The training data is VOC 2007 trainval. The default setting of using 3 scales and 3 aspect ratios (69.9%) is the same as that in Table 3.

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**

Table 9: Detection results of Faster R-CNN on PASCAL VOC 2007 test set using **different values of** λ in Equation (1). The network is VGG-16. The training data is VOC 2007 trainval. The default setting of using $\lambda = 10$ (69.9%) is the same as that in Table 3.

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

In Table 5 we summarize the running time of the entire object detection system. SS takes 1-2 seconds depending on content (on average about 1.5s), and Fast R-CNN with VGG-16 takes 320ms on 2000 SS proposals (or 223ms if using SVD on fully-connected layers [2]). Our system with VGG-16 takes in total **198ms** for both proposal and detection. With the convolutional features shared, the RPN alone only takes 10ms computing the additional layers. Our region-wise computation is also lower, thanks to fewer proposals (300 per image). Our system has a frame-rate of 17 fps with the ZF net.

Sensitivities to Hyper-parameters. In Table 8 we investigate the settings of anchors. By default we use 3 scales and 3 aspect ratios (69.9% mAP in Table 8). If using just one anchor at each position, the mAP drops by a considerable margin of 3-4%. The mAP is higher if using 3 scales (with 1 aspect ratio) or 3 aspect ratios (with 1 scale), demonstrating that using anchors of multiple sizes as the regression references is an effective solution. Using just 3 scales with 1 aspect ratio (69.8%) is as good as using 3 scales with 3 aspect ratios on this dataset, suggesting that scales and aspect ratios are not disentangled dimensions for the detection accuracy. But we still adopt these two dimensions in our designs to keep our system flexible.

In Table 9 we compare different values of λ in Equation (1). By default we use $\lambda = 10$ which makes the two terms in Equation (1) roughly equally weighted after normalization. Table 9 shows that our result is impacted just marginally (by $\sim 1\%$) when λ is within a scale of about two orders of magnitude (1 to 100). This demonstrates that the result is insensitive to λ in a wide range.

Analysis of Recall-to-IoU. Next we compute the recall of proposals at different IoU ratios with ground-truth boxes. It is noteworthy that the Recall-to-IoU metric is just *loosely* [19], [20], [21] related to the ultimate detection accuracy. It is more appropriate to use this metric to *diagnose* the proposal method than to evaluate it.

In Figure 4, we show the results of using 300, 1000, and 2000 proposals. We compare with SS and EB, and the N proposals are the top-N ranked ones based on the confidence generated by these methods. The plots show that the RPN method behaves gracefully when the number of proposals drops from 2000 to 300. This explains why the RPN has a good ultimate detection mAP when using as few as 300 proposals. As we analyzed before, this property is mainly attributed to the *cls* term of the RPN. The recall of SS and EB drops more quickly than RPN when the proposals are fewer.

表6：使用Fast R-CNN检测器和VGG-16在PASCAL VOC 2007测试集上的结果。对于RPN，训练时 e Fast R-CNN的候选框数量为2000。RPN*表示未共享特征的版本。

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	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7：使用Fast R-CNN检测器和VGG-16在PASCAL VOC 2012测试集上的结果。对于RPN，Fast R-CNN训练时使用的候选框数量为2000。

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8：使用不同锚点设置的Faster R-CNN在PASCAL VOC 2007测试集上的检测结果。网络结构为VGG-16。训练数据为VOC 2007 trainval。采用3种尺度和3种长宽比的默认设置（69.9%）与表3中的设置相同。

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9：使用方程(1)中不同λ值时，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$\lambda = 10$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

在表5中，我们总结了整个物体检测系统的运行时间。选择性搜索（SS）根据内容需要1-2秒（平均约1.5秒），而基于VGG-16的Fast R-CNN处理2000个SS建议区域需要320毫秒（若在全连接层使用SVD[2]则需223毫秒）。我们采用VGG-16的系统在建议区域生成和检测环节总计耗时198毫秒。由于共享卷积特征，单独计算RPN额外层仅需10毫秒。得益于更少的建议区域数量（每图300个），我们的逐区域计算耗时也更低。使用ZF网络时，我们的系统帧率达到每秒17帧。

超参数敏感性。在表8中，我们研究了锚点的设置。默认情况下，我们使用3个尺度和3个宽高比（表8中为69.9% mAP）。若每个位置仅使用一个锚点，mAP会显著下降3-4%。使用3个尺度（搭配1种宽高比）或3种宽高比（搭配1个尺度）时mAP更高，这表明采用多尺寸锚点作为回归参考是有效的解决方案。在本数据集中，仅使用3个尺度搭配1种宽高比（69.8%）的效果与使用3个尺度搭配3种宽高比相当，说明尺度和宽高比并非影响检测精度的独立维度。但我们仍在设计中保留这两个维度，以维持系统的灵活性。

在表9中，我们比较了方程(1)中不同λ的取值。默认情况下我们使用$\lambda = 10$，这使得归一化后方程(1)中的两项具有大致相等的权重。表9显示，当λ在大约两个数量级（1到100）的范围内变化时，我们的结果仅受到轻微影响（约$\sim$ 1%）。这表明结果在较大范围内对λ不敏感。

召回率与IoU关系分析。接下来，我们计算候选框在不同IoU阈值下与真实框的召回率。值得注意的是，Recall-to-IoU指标仅与最终检测精度*loosely*相关[19], [20], [21]。相比直接评估候选框生成方法，使用该指标进行*diagnose*更为合适。

在图4中，我们展示了使用300、1000和2000个候选区域的结果。我们与SS和EB方法进行比较，其中N个候选区域是基于这些方法生成的置信度排名前N的提案。图表显示，当候选区域数量从2000降至300时，RPN方法表现依然稳定。这解释了为何RPN仅使用300个候选区域仍能获得良好的最终检测mAP。正如我们先前所分析的，这一特性主要归因于RPN的*cls*项。当候选区域数量减少时，SS和EB方法的召回率下降速度比RPN更快。

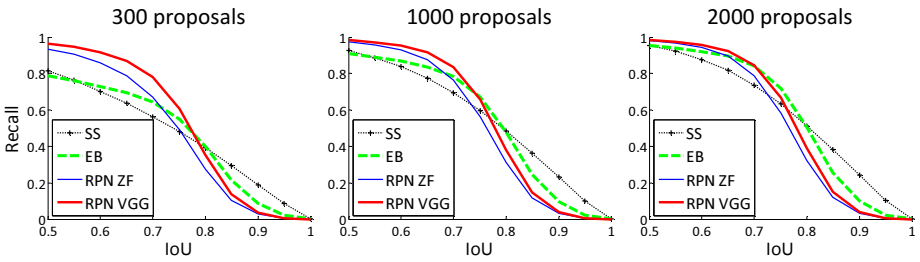

Figure 4: Recall *vs*. IoU overlap ratio on the PASCAL VOC 2007 test set.

Table 10: **One-Stage Detection *vs*. Two-Stage Proposal + Detection**. Detection results are on the PASCAL VOC 2007 test set using the ZF model and Fast R-CNN. RPN uses unshared features.

	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

One-Stage Detection *vs*. Two-Stage Proposal + Detection. The OverFeat paper [9] proposes a detection method that uses regressors and classifiers on sliding windows over convolutional feature maps. OverFeat is a *one-stage*, *class-specific* detection pipeline, and ours is a *two-stage cascade* consisting of class-agnostic proposals and class-specific detections. In OverFeat, the region-wise features come from a sliding window of one aspect ratio over a scale pyramid. These features are used to simultaneously determine the location and category of objects. In RPN, the features are from square (3×3) sliding windows and predict proposals relative to anchors with different scales and aspect ratios. Though both methods use sliding windows, the region proposal task is only the first stage of Faster R-CNN—the downstream Fast R-CNN detector *attends* to the proposals to refine them. In the second stage of our cascade, the region-wise features are adaptively pooled [1], [2] from proposal boxes that more faithfully cover the features of the regions. We believe these features lead to more accurate detections.

To compare the one-stage and two-stage systems, we *emulate* the OverFeat system (and thus also circumvent other differences of implementation details) by *one-stage* Fast R-CNN. In this system, the "proposals" are dense sliding windows of 3 scales (128, 256, 512) and 3 aspect ratios (1:1, 1:2, 2:1). Fast R-CNN is trained to predict class-specific scores and regress box locations from these sliding windows. Because the OverFeat system adopts an image pyramid, we also evaluate using convolutional features extracted from 5 scales. We use those 5 scales as in [1], [2].

Table 10 compares the two-stage system and two variants of the one-stage system. Using the ZF model, the one-stage system has an mAP of 53.9%. This is lower than the two-stage system (58.7%) by 4.8%. This experiment justifies the effectiveness of cascaded region proposals and object detection. Similar observations are reported in [2], [39], where replacing SS region proposals with sliding windows leads to ∼6% degradation in both papers. We also note that the one-stage system is slower as it has considerably more proposals to process.

4.2 Experiments on MS COCO

We present more results on the Microsoft COCO object detection dataset [12]. This dataset involves 80 object categories. We experiment with the 80k images on the training set, 40k images on the validation set, and 20k images on the test-dev set. We evaluate the mAP averaged for IoU $\in [0.5 : 0.05 : 0.95]$ (COCO's standard metric, simply denoted as mAP@[.5, .95]) and mAP@0.5 (PASCAL VOC's metric).

There are a few minor changes of our system made for this dataset. We train our models on an 8-GPU implementation, and the effective mini-batch size becomes 8 for RPN (1 per GPU) and 16 for Fast R-CNN (2 per GPU). The RPN step and Fast R-CNN step are both trained for 240k iterations with a learning rate of 0.003 and then for 80k iterations with 0.0003. We modify the learning rates (starting with 0.003 instead of 0.001) because the mini-batch size is changed. For the anchors, we use 3 aspect ratios and 4 scales (adding 64^2), mainly motivated by handling small objects on this dataset. In addition, in our Fast R-CNN step, the negative samples are defined as those with a maximum IoU with ground truth in the interval of $[0, 0.5)$, instead of $[0.1, 0.5)$ used in [1], [2]. We note that in the SPPnet system [1], the negative samples in $[0.1, 0.5)$ are used for network fine-tuning, but the negative samples in $[0, 0.5)$ are still visited in the SVM step with hard-negative mining. But the Fast R-CNN system [2] abandons the SVM step, so the negative samples in $[0, 0.1)$ are never visited. Including these $[0, 0.1)$ samples improves mAP@0.5 on the COCO dataset for both Fast R-CNN and Faster R-CNN systems (but the impact is negligible on PASCAL VOC).

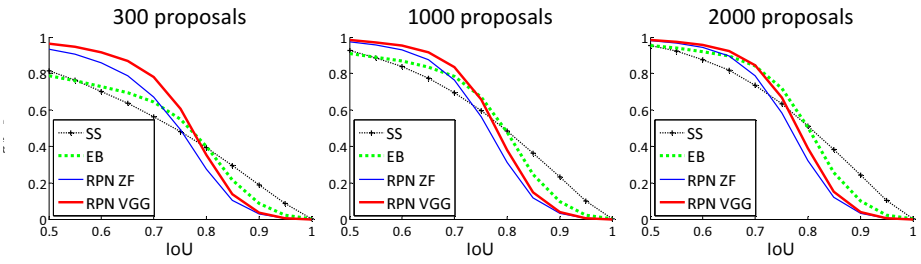

图 4: R　　召回 *vs*。在PASCAL VOC 2007上的IoU重叠率　　测试集。

表10：单阶段检测 ***vs***。两阶段提议 + 检测。检测结果基于PASCA。　　L
使用ZF模型和Fast R-CNN在VOC 2007测试集上的结果。RPN采用非共享特征。

	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

单阶段检测 ***vs***。两阶段提议 + 检测。OverFeat 论文 [9] 提出了一种检测方法，在卷积特征图上使用滑动窗口进行回归器和分类器的操作。OverFeat 是一个 *one-stage*、*class-specific* 检测流程，而我们的方法是一个 *two-stage cascade*，由类别无关的提议和类别特定的检测组成。在 OverFeat 中，区域特征来自尺度金字塔上单一长宽比的滑动窗口。这些特征用于同时确定对象的位置和类别。在 RPN 中，特征来自正方形（3×3）滑动窗口，并预测相对于不同尺度和长宽比的锚框的提议。尽管两种方法都使用滑动窗口，但区域提议任务只是 Faster R-CNN 的第一阶段——下游的 Fast R-CNN 检测器 *attends* 会对提议进行细化。在我们级联的第二阶段，区域特征是从更忠实覆盖区域特征的提议框中进行自适应池化 [1]、[2] 得到的。我们相信这些特征能带来更精确的检测。

为了比较单阶段和两阶段系统，我们通过*one-stage* Fast R-CNN 来*emulate* OverFeat 系统（从而也规避了其他实现细节的差异）。在该系统中，“候选框”是密集滑动窗口，包含3种尺度（128、256、512）和3种宽高比（1:1、1:2、2:1）。Fast R-CNN 被训练用于从这些滑动窗口中预测类别特定分数并回归边界框位置。由于 OverFeat 系统采用图像金字塔，我们也使用从5种尺度提取的卷积特征进行评估。我们按照文献[1]、[2]的方法使用这5种尺度。

表10比较了两阶段系统和一阶段系统的两个变体。使用ZF模型时，一阶段系统的mAP为53.9%，比两阶段系统（58.7%）低4.8%。该实验证明了级联区域建议与目标检测方案的有效性。[2]和[39]的研究也报告了类似现象，其中将SS滑动窗口的区域建议导致两篇论文中的性能都下降了～6%。我们还注意到，单阶段系统速度较慢，因为它需要处理更多的建议。

4.2 MS COCO 上的实验

我们在微软COCO目标检测数据集[12]上呈现更多结果。该数据集包含80个目标类别。我们使用训练集中的80k图像、验证集中的40k图像以及测试开发集中的20k图像进行实验。我们评估了IoU阈值在{0.5:0.05:0.95}范围内（即COCO标准指标，简记为mAP@[.5, .95]）的平均mAP，以及mAP@0.5（PASCAL VOC指标）。

针对该数据集，我们对系统进行了一些微调。我们在8-GPU配置上训练模型，RPN（每GPU 1个）的有效小批量大小为8，Fast R-CNN（每GPU 2个）为16。RPN阶段和Fast R-CNN阶段均训练240k次迭代，学习率为0.003，随后以0.0003学习率训练80k次迭代。由于小批量大小发生变化，我们调整了学习率（从0.003开始而非0.001）。在锚点设置上，我们采用3种宽高比和4种尺度（新增64^2），主要是为了应对该数据集中的小目标。此外，在Fast R-CNN阶段，我们将负样本定义为与真实标注框最大IoU处于[0, 0.5)区间的样本，而非[1]、[2]中使用的[0.1, 0.5)。需要说明的是，SPPnet系统[1]在微调网络时使用了[0.1, 0.5)区间的负样本，但[0, 0.5)区间的负样本仍会通过难例挖掘在SVM阶段参与训练。而Fast R-CNN系统[2]取消了SVM步骤，导致[0, 0.1)区间的负样本完全未被利用。实验表明，纳入这些[0, 0.1)区间的样本能提升Fast R-CNN和Faster R-CNN系统在COCO数据集上的mAP@0.5指标（但对PASCAL VOC数据集的影响可忽略不计）。

Table 11: Object detection results (%) on the **MS COCO** dataset. The model is VGG-16.

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**

The rest of the implementation details are the same as on PASCAL VOC. In particular, we keep using 300 proposals and single-scale ($s = 600$) testing. The testing time is still about 200ms per image on the COCO dataset.

In Table 11 we first report the results of the Fast R-CNN system [2] using the implementation in this paper. Our Fast R-CNN baseline has 39.3% mAP@0.5 on the test-dev set, higher than that reported in [2]. We conjecture that the reason for this gap is mainly due to the definition of the negative samples and also the changes of the mini-batch sizes. We also note that the mAP@[.5, .95] is just comparable.

Next we evaluate our Faster R-CNN system. Using the COCO training set to train, Faster R-CNN has 42.1% mAP@0.5 and 21.5% mAP@[.5, .95] on the COCO test-dev set. This is 2.8% higher for mAP@0.5 and **2.2% higher for mAP@[.5, .95]** than the Fast R-CNN counterpart under the same protocol (Table 11). This indicates that RPN performs excellent for improving the localization accuracy at higher IoU thresholds. Using the COCO trainval set to train, Faster R-CNN has 42.7% mAP@0.5 and 21.9% mAP@[.5, .95] on the COCO test-dev set. Figure 6 shows some results on the MS COCO test-dev set.

Faster R-CNN in ILSVRC & COCO 2015 competitions We have demonstrated that Faster R-CNN benefits more from better features, thanks to the fact that the RPN completely learns to propose regions by neural networks. This observation is still valid even when one increases the depth substantially to over 100 layers [18]. Only by replacing VGG-16 with a 101-layer residual net (ResNet-101) [18], the Faster R-CNN system increases the mAP from 41.5%/21.2% (VGG-16) to 48.4%/27.2% (ResNet-101) on the COCO val set. With other improvements orthogonal to Faster R-CNN, He *et al*. [18] obtained a single-model result of 55.7%/34.9% and an ensemble result of 59.0%/37.4% on the COCO test-dev set, which won the 1st place in the COCO 2015 object detection competition. The same system [18] also won the 1st place in the ILSVRC 2015 object detection competition, surpassing the second place by absolute 8.5%. RPN is also a building block of the 1st-place winning entries in ILSVRC 2015 localization and COCO 2015 segmentation competitions, for which the details are available in [18] and [15] respectively.

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**

4.3 From MS COCO to PASCAL VOC

Large-scale data is of crucial importance for improving deep neural networks. Next, we investigate how the MS COCO dataset can help with the detection performance on PASCAL VOC.

As a simple baseline, we directly evaluate the COCO detection model on the PASCAL VOC dataset, *without fine-tuning on any PASCAL VOC data*. This evaluation is possible because the categories on COCO are a superset of those on PASCAL VOC. The categories that are exclusive on COCO are ignored in this experiment, and the softmax layer is performed only on the 20 categories plus background. The mAP under this setting is 76.1% on the PASCAL VOC 2007 test set (Table 12). This result is better than that trained on VOC07+12 (73.2%) by a good margin, even though the PASCAL VOC data are not exploited.

Then we fine-tune the COCO detection model on the VOC dataset. In this experiment, the COCO model is in place of the ImageNet-pre-trained model (that is used to initialize the network weights), and the Faster R-CNN system is fine-tuned as described in Section 3.2. Doing so leads to 78.8% mAP on the PASCAL VOC 2007 test set. The extra data from the COCO set increases the mAP by 5.6%. Table 6 shows that the model trained on COCO+VOC has the best AP for every individual category on PASCAL VOC 2007. Similar improvements are observed on the PASCAL VOC 2012 test set (Table 12 and Table 7). We note that the test-time speed of obtaining these strong results is still about **200ms per image**.

5 Conclusion

We have presented RPNs for efficient and accurate region proposal generation. By sharing convolutional

表11：在MS COCO数据集上的目标检测结果（%）。模型为VGG-16。

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**

其余实现细节与PASCAL VOC相同。具体而言，我们继续使用300个候选框和单尺度（$s = 600$）测试。在COCO数据集上，每张图像的测试时间仍约为200毫秒。

在表11中，我们首先报告了采用本文实现的Fast R-CNN系统[2]的结果。我们的Fast R-CNN基线在test-dev数据集上取得了39.3%的mAP@0.5，高于文献[2]中报告的结果。我们推测这种差异主要源于负样本的定义方式以及小批量数据规模的调整。同时我们注意到，其mAP@[.5, .95]指标仅处于相当水平。

接下来我们评估我们的Faster R-CNN系统。使用COCO训练集进行训练时，Faster R-CNN在COCO test-dev数据集上取得了42.1%的mAP@0.5和21.5%的mAP@[.5, .95]指标。在相同实验协议下，该结果比Fast R-CNN对应指标分别高出2.8%（mAP@0.5）和2.2%（mAP@[.5, .95]）（见表11）。这表明RPN在更高IoU阈值下能显著提升定位精度。使用COCO trainval集训练时，Faster R-CNN在COCO test-dev数据集上达到42.7%的mAP@0.5和21.9%的mAP@[.5, .95]。图6展示了在MS COCO test-dev数据集上的部分检测结果。

在ILSVRC和COCO 2015竞赛中的Faster R-CNN　我们证明了Faster R-CNN能够从更优的特征中获益更多，这得益于区域提议网络（RPN）完全通过神经网络学习生成候选区域。即使将网络深度大幅增加至100层以上[18]，这一结论依然成立。仅将VGG-16替换为101层残差网络（ResNet-101）[18]，Faster R-CNN系统在COCO验证集上的mAP就从41.5%/21.2%（VGG-16）提升至48.4%/27.2%（ResNet-101）。通过结合其他与Faster R-CNN正交的改进方法，何*et al*等人[18]在COCO测试开发集上取得了单模型55.7%/34.9%和集成模型59.0%/37.4%的结果，赢得了COCO 2015目标检测竞赛冠军。同一系统[18]还在ILSVRC 2015目标检测竞赛中获得第一名，以绝对优势领先第二名8.5%。RPN同样是ILSVRC 2015定位竞赛和COCO 2015分割竞赛冠军方案的核心组件，具体细节可分别参考[18]和[15]。

表12：使用不同训练数据时，Faster R-CNN在PASCAL VOC 2007测试集和2012测试集上的检测mAP（%）。模型为VGG-16。“COCO”表示使用COCO trainval集进行训练。另见表6和表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**

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，我们将探讨MS COCO数据集如何帮助提升PASCAL VOC的检测性能。

作为一个简单的基线，我们直接在PASCAL VOC数据集上评估COCO检测模型，*without fine-tuning on any PASCAL VOC data*。这种评估是可行的，因为COCO的类别是PASCAL VOC类别的超集。本实验中忽略COCO独有的类别，softmax层仅针对20个类别加背景进行计算。在此设置下，PASCAL VOC 2007测试集上的mAP达到76.1%（表12）。这一结果明显优于在VOC07+12上训练的结果（73.2%），尽管我们并未利用PASCAL VOC数据。

然后我们在VOC数据集上对COCO检测模型进行微调。在此实验中，COCO模型替代了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系统进行微调。这样做使得PASCAL VOC 2007测试集上的mAP达到78.8%。来自COCO集的额外数据使mAP提升了5.6%。表6显示，在COCO+VOC上训练的模型在PASCAL VOC 2007的每个单独类别上都取得了最佳AP。在PASCAL VOC 2012测试集上也观察到了类似的改进（表12和表7）。我们注意到，获得这些强劲结果的测试速度仍约为每张图像200毫秒。

5 结论

我们提出了RPNs以实现高效且准确的区域建议生成。通过共享卷积

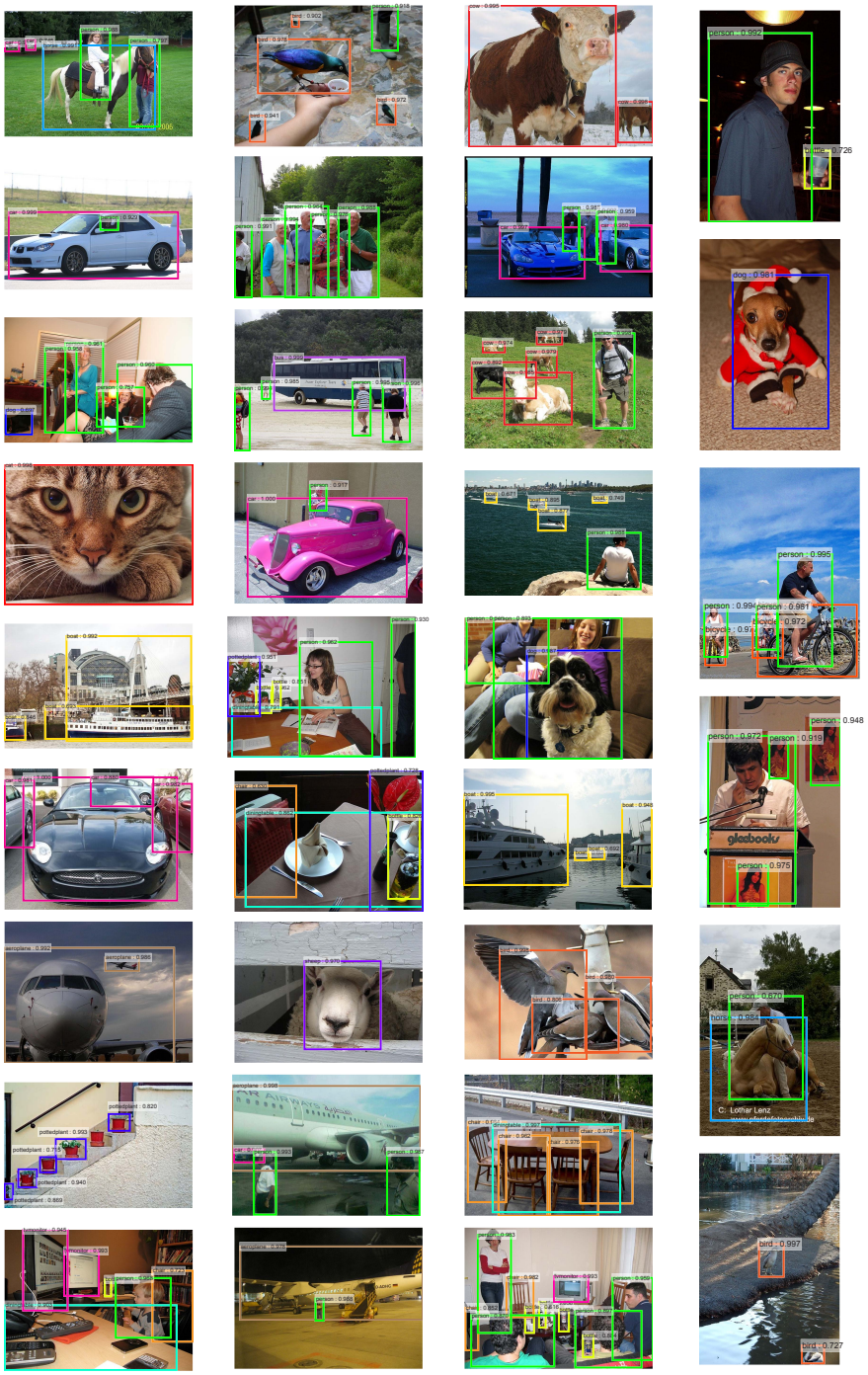

Figure 5: Selected examples of object detection results on the PASCAL VOC 2007 test set using the Faster R-CNN system. The model is VGG-16 and the training data is 07+12 trainval (73.2% mAP on the 2007 test set). Our method detects objects of a wide range of scales and aspect ratios. Each output box is associated with a category label and a softmax score in $[0, 1]$. A score threshold of 0.6 is used to display these images. The running time for obtaining these results is **198ms** per image, *including all steps*.

features with the down-stream detection network, the region proposal step is nearly cost-free. Our method enables a unified, deep-learning-based object detection system to run at near real-time frame rates. The learned RPN also improves region proposal quality and thus the overall object detection accuracy.

REFERENCES

[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

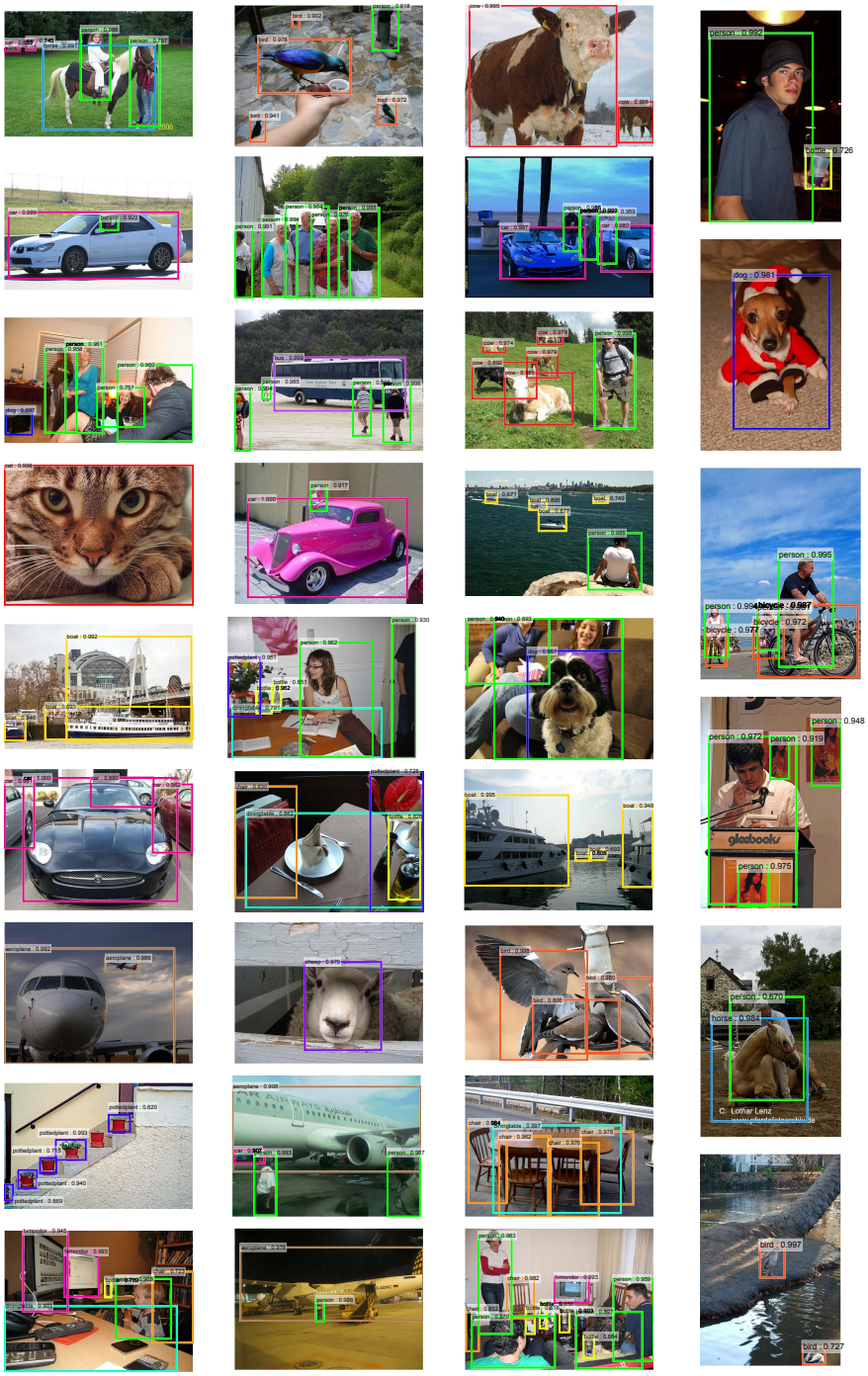

图5：使用Faster R-CNN系统在PASCAL VOC 2007测试集上选取的目标检测结果示例。模型采用VGG-16架构，训练数据为07+12 trainval（在2007测试集上达到73.2% mAP）。我们的方法能检测多种尺度和宽高比的目标物体。每个输出框都关联着类别标签和[0, 1]范围内的softmax置信度。图中展示的结果采用0.6的置信度阈值进行筛选，生成这些结果的单张图像处理时间为198毫秒*including all steps*。

通过与下游检测网络结合的特征，区域提议步骤几乎是零成本的。我们的方法使得一个统一的、基于深度学习的目标检测系统能够以接近实时的帧率运行。学习得到的RPN还提高了区域提议的质量，从而提升了整体目标检测的准确性。

参考文献

[1] K. He, X. Zhang, S. Ren, and J. Sun, “用于视觉识别的深度卷积网络中的空间金字塔池化，” 发表于 *European Conference on Computer Vision (ECCV)*, 2014.[2] R. Girshick, “Fast R-CNN，” 发表于 *IEEE International Conference on Computer Vision (ICCV)*, 2015.[3] K. Simonyan and A. Zisserman, “非常深的卷积

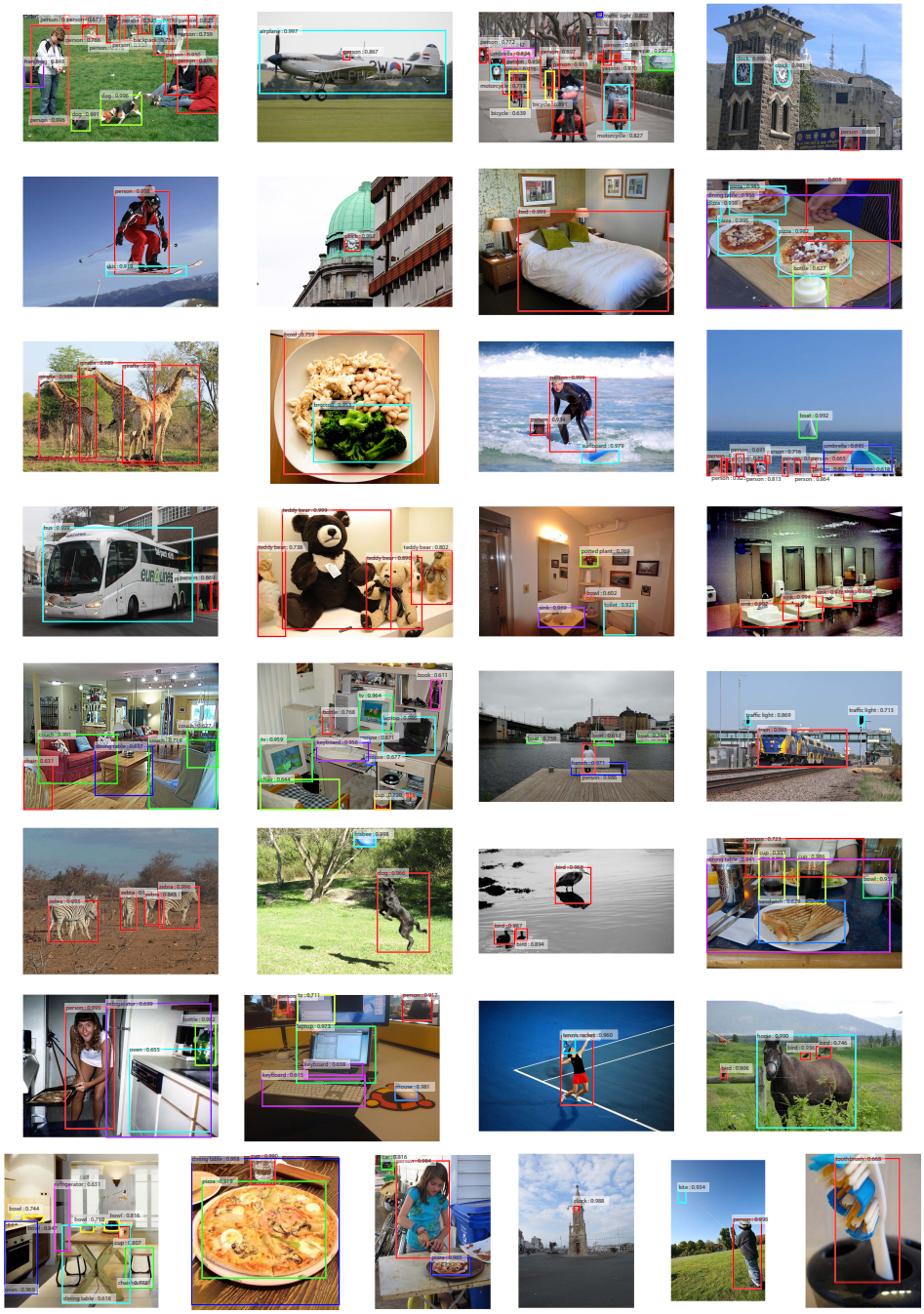

Figure 6: Selected examples of object detection results on the MS COCO test-dev set using the Faster R-CNN system. The model is VGG-16 and the training data is COCO trainval (42.7% mAP@0.5 on the test-dev set). Each output box is associated with a category label and a softmax score in $[0, 1]$. A score threshold of 0.6 is used to display these images. For each image, one color represents one object category in that image.

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. Dollár, "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 part-based 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

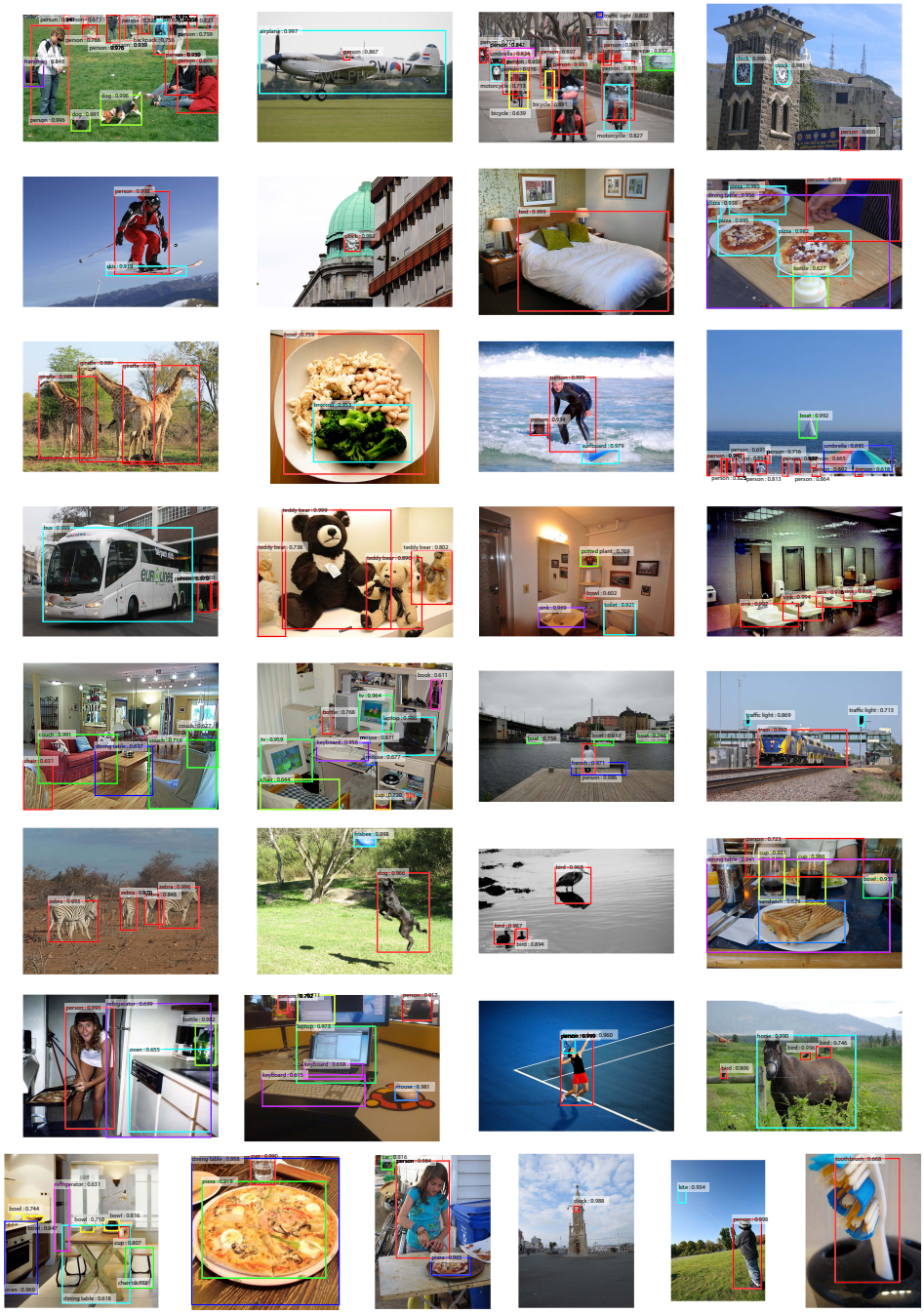

图6：使用Faster R-CNN系统在MS COCO测试开发集上选取的目标检测结果示例。模型采用VGG-16架构，训练数据为COCO训练验证集（在测试开发集上达到42.7%的mAP@0.5）。每个输出框均关联一个类别标签及[0, 1]范围内的softmax置信度分数。本展示采用0.6的分数阈值筛选图像，每张图像中同色边框代表同一目标类别。

用于大规模图像识别的网络，”于*International Conference on Learning Representations (ICLR)*，2015年。[4] J. R. Uijlings, K. E. van de Sande, T. Gevers, 和 A. W. Smeulders，“选择性搜索用于物体识别，”*InternationalJournal of Computer Vision (IJCV)*，2013年。[5] R. Girshick, J. Donahue, T. Darrell, 和 J. Malik，“用于精确物体检测和语义分割的丰富特征层次结构，”于*IEEE Conference on Computer Vision and PatternRecognition (CVPR)*，2014年。[6] C. L. Zitnick 和 P. Dollár，“边缘框：从边缘定位物体提议，”于*European Conference on ComputerVision (ECCV)*，2014年。

[7] J. Long, E. Shelhamer, 与 T. Darrell，“用于语义分割的全卷积网络”，发表于 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015年。[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, 与 D. Ramanan，“使用基于判别性训练部件模型的物体检测”，*IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010年。[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, 与 Y. LeCun，“Overfeat：使用卷积网络进行集成识别、定位与检测”，发表于 *International Conference on Learning Representations (ICLR)*, 2014年。

[10] S. Ren, K. He, R. Girshick, 与 J. Sun，“Faster R-CNN: 迈向

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. Dollár, and C. L. Zitnick, "Microsoft COCO: Common 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. Dollár, 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. Arbeláez, 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.

基于区域建议网络的实时目标检测》，发表于
Neural Information Processing Systems (NIPS)，2015年。[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, 与 A. Zisserman，《PASCAL视觉目标分类挑战赛2007（VOC2007）结果》，2007年。[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, 与 C. L. Zitnick，《Microsoft COCO：上下文中的常见目标》，发表于*European Conference on ComputerVision (ECCV)*，2014年。[13] S. Song 与 J. Xiao，《用于RGB-D图像中非模态3D目标检测的深度滑动形状》，*arXiv:1511.02300*，2015年。[14] J. Zhu, X. Chen, 与 A. L. Yuille，《DeePM：一种用于目标检测与语义部件定位的深度部件模型》，*arXiv:1511.07131*，2015年。[15] J. Dai, K. He, 与 J. Sun，《通过多任务网络级联实现实例感知的语义分割》，*arXiv:1512.04412*，2015年。[16] J. Johnson, A. Karpathy, 与 L. Fei-Fei，《Densecap：用于密集描述的全卷积定位网络》，*arXiv:1511.07571*，2015年。[17] D. Kislyuk, Y. Liu, D. Liu, E. Tzeng, 与 Y. Jing，《人工筛选与卷积网络：驱动Pinterest上的物品到物品推荐》，*arXiv:1511.04003*，2015年。[18] K. He, X. Zhang, S. Ren, 与 J. Sun，《用于图像识别的深度残差学习》，*arXiv:1512.03385*，2015年。[19] J. Hosang, R. Benenson, 与 B. Schiele，《检测建议到底有多好？》，发表于
British Machine Vision Conference(BMVC)，2014年。[20] J. Hosang, R. Benenson, P. Dollár, 与 B. Schiele，《什么造就了有效的检测建议？》，*IEEE Transactions on Pattern
Analysis and Machine Intelligence (TPAMI)*，2015年。[21] N. Chavali, H. Agrawal, A. Mahendru, 与 D. Batra，《目标建议评估协议是“可操纵的”》，*arXiv:1505.05836*，2015年。[22] J. Carreira 与 C. Sminchisescu，《CPMC：使用约束参数最小割的自动目标分割》，
IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)，2012年。[23] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, 与 J. Malik，《多尺度组合分组》，发表于*IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*，2014年。[24] B. Alexe, T. Deselaers, 与 V. Ferrari，《测量图像窗口的目标性》，
*IEEE Transactions on Pattern Analysis
and Machine Intelligence (TPAMI)*，2012年。[25] C. Szegedy, A. Toshev, 与 D. Erhan，《用于目标检测的深度神经网络》，发表于
Neural Information Processing Systems(NIPS)，2013年。[26] D. Erhan, C. Szegedy, A. Toshev, 与 D. Anguelov，《使用深度神经网络的可扩展目标检测》，发表于*IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*，2014年。[27] C. Szegedy, S. Reed, D. Erhan, 与 D. Anguelov，《可扩展的高质量目标检测》，*arXiv:1412.1441 (v1)*，2015年。[28] P. O. Pinheiro, R. Collobert, 与 P. Dollar，《学习分割候选目标》，发表于
Neural Information ProcessingSystems (NIPS)，2015年。[29] J. Dai, K. He, 与 J. Sun，《用于联合目标与背景分割的卷积特征掩码》，发表于*IEEE Conference on
Computer Vision and Pattern Recognition (CVPR)*，2015年。[30] S. Ren, K. He, R. Girshick, X. Zhang, 与 J. Sun，《卷积特征图上的目标检测网络》，*arXiv:1504.06066*，2015年。[31] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, 与 Y. Bengio，《基于注意力的语音识别模型》，发表于*Neural Information Processing Systems (NIPS)*，2015年。[32] M. D. Zeiler 与 R. Fergus，《可视化与理解卷积神经网络》，发表于*European Conference onComputer Vision (ECCV)*，2014年。[33] V. Nair 与 G. E. Hinton，《修正线性单元改进受限玻尔兹曼机》，发表于*International Conference onMachine Learning (ICML)*，2010年。[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, 与 A. Rabinovich，《更深入的卷积网络》，发表于
*IEEE Conference on Computer Vision and Pattern
Recognition (CVPR)*，2015年。[35] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, 与 L. D. Jackel，《反向传播应用于手写邮政编码识别》，*Neural computation*，1989年。

[36] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, 与 L. Fei-Fei，“ImageNet 大规模视觉识别挑战赛，”发表于
International Journal of Computer Vision (IJCV), 2015年。[37] A. Krizhevsky, I. Sutskever, 与 G. Hinton，“使用深度卷积神经网络进行 ImageNet 分类，”发表于 *Neural Information Processing Systems (NIPS)*, 2012年。[38] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, 与 T. Darrell，“Caffe：用于快速特征嵌入的卷积架构，”*arXiv:1408.5093*, 2014年。[39] K. Lenc 与 A. Vedaldi，“R-CNN 减去 R，”发表于 *British Machine Vision Conference (BMVC)*, 2015年。