Faster R-CNN

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.

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

Faster R-CNN

摘要

最先进的目标检测网络依靠区域提出算法来假设目标的位置。SPPnet[1]和 Fast R-CNN[2]等研究已经减少了这些检测网络的运行时间，使得区域提出计算成为一个瓶颈。在这项工作中，我们引入了一个区域提出网络（RPN），该网络与检测网络共享全图像的卷积特征，从而使近乎零成本的区域提出成为可能。RPN 是一个全卷积网络，可以同时在每个位置预测目标边界和目标分数。RPN 经过端到端的训练，可以生成高质量的区域提出，由 Fast R-CNN 用于检测。我们将 RPN 和 Fast R-CNN 通过共享卷积特征进一步合并为一个单一的网络——使用最近流行的具有“注意力”机制的神经网络术语，RPN 组件告诉统一网络在哪里寻找。对于非常深的 VGG-16 模型[3]，我们的检测系统在 GPU 上的帧率为 5fps（包括所有步骤），同时在 PASCAL VOC 2007，2012 和 MS COCO 数据集上实现了最新的目标检测精度，每个图像只有 300 个提出。在 ILSVRC 和 COCO 2015 竞赛中，Faster R-CNN 和 RPN 是多个比赛中获得第一名输入的基础。代码可公开获得。

1.介绍

目标检测的最新进展是由区域提出方法（例如[4]）和基于区域的卷积神经网络（R-CNN）[5]的成功驱动的。尽管在[5]中最初开发的基于区域的 CNN 计算成本很高，但是由于在各种提议中共享卷积，所以其成本已经大大降低了[1]，[2]。忽略花费在区域提议上的时间，最新版本 Fast R-CNN[2]利用非常深的网络[3]实现了接近实时的速率。现在，提议是最新的检测系统中测

bottleneck in state-of-the-art detection systems.

试时间的计算瓶颈。

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.

区域提议方法通常依赖廉价的特征和简练的推断方案。选择性搜索[4]是最流行的方法之一，它贪婪地合并基于设计的低级特征的超级像素。然而，与有效的检测网络[2]相比，选择性搜索速度慢了一个数量级，在 CPU 实现中每张图像的时间为 2 秒。EdgeBoxes[6]目前提供了在提议质量和速度之间的最佳权衡，每张图像 0.2 秒。尽管如此，区域提议步骤仍然像检测网络那样消耗同样多的运行时间。

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.

有人可能会注意到，基于区域的快速 CNN 利用 GPU，而在研究中使用的区域提议方法在 CPU 上实现，使得运行时间比较不公平。加速区域提议计算的一个显而易见的方法是将其在 GPU 上重新实现。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检测网络，因此错过了共享计算的重要机会。

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).

在本文中，我们展示了算法的变化——用深度卷积神经网络计算区域提议——导致了一个优雅和有效的解决方案，其中在给定检测网络计算的情况下区域提议计算接近领成本。为此，我们引入了新的区域提议网络（RPN），它们共享最先进目标检测网络的卷积层[1]，[2]。通过在测试时共享卷积，计算区域提议的边际成本很小（例如，每张图像 10ms）。

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

我们的观察是，基于区域的检测器所使用的卷积特征映射，如 Fast R-CNN，也可以用于生成区域提议。在这些卷积特征之

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 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.

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

上，我们通过添加一些额外的卷积层来构建 RPN，这些卷积层同时在规则网格上的每个位置上回归区域边界和目标分数。因此 RPN 是一种全卷积网络（FCN）[7]，可以针对生成检测区域建议的任务进行端到端的训练。

RPN 旨在有效预测具有广泛尺度和长宽比的区域提议。与使用图像金字塔（图 1，a）或滤波器金字塔（图 1，b）的流行方法[8]，[9]，[1]相比，我们引入新的“锚”盒作为多种尺度和长宽比的参考。我们的方案可以被认为是回归参考金字塔（图 1，c），它避免了枚举多种比例或长宽比的图像或滤波器。这个模型在使用单尺度图像进行训练和测试时运行良好，从而有利于运行速度。

为了将 RPN 与 Fast R-CNN [2]目标检测网络相结合，我们提出了一种训练方案，在微调区域提议任务和微调目标检测之间进行交替，同时保持区域提议的固定。该方案快速收敛，并产生两个任务之间共享的具有卷积特征的统一网络。

我们在 PASCAL VOC 检测基准数据集上[11]综合评估了我们的方法，其中具有 Fast R-CNN 的 RPN 产生的检测精度优于使用选择性搜索的 Fast R-CNN 的强基准。同时，我们的方法在测试时几乎免除了选择性搜索的所有计算负担——区域提议的有效运行时间仅为 10 毫秒。使用[3]的昂贵的非常深的模型，我们的检测方法在 GPU 上仍然具有 5fps 的帧率（包括所有步骤），因此在速度和准确性方面是实用的目标检测系统。我们还报告了在 MS COCO 数据集上[12]的结果，并使用 COCO 数据研究了

made publicly available at https://github.com/shaoqingren/faster_rcnn (in MATLAB) and https://github.com/rbgirshick/py-faster-rcnn (in Python).

在 PASCAL VOC 上的改进。代码可公开获得 https://github.com/shaoqingren/faster_rcnn（在 MATLAB 中）和 https://github.com/rbgirshick/py-faster-rcnn（在 Python 中）。

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.

这个手稿的初步版本是以前发表的[10]。从那时起，RPN 和 Faster R-CNN 的框架已经被采用并推广到其他方法，如 3D 目标检测[13]，基于部件的检测[14]，实例分割[15]和图像标题[16]。我们快速和有效的目标检测系统也已经在 Pinterest[17]的商业系统中建立了，并报告了用户参与度的提高。

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. 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.

在 ILSVRC 和 COCO 2015 竞赛中，Faster R-CNN 和 RPN 是 ImageNet 检测，ImageNet 定位，COCO 检测和 COCO 分割中几个第一名参赛者[18]的基础。RPN 完全从数据中学习提议区域，因此可以从更深入和更具表达性的特征（例如[18]中采用的 101 层残差网络）中轻松获益。Faster R-CNN 和 RPN 也被这些比赛中的其他几个主要参赛者所使用。这些结果表明，我们的方法不仅是一个实用合算的解决方案，而且是一个提高目标检测精度的有效方法。

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

2.相关工作

目标提议。目标提议方法方面有大量的文献。目标提议方法的综合调查和比较可以在[19]，[20]，[21]中找到。广泛使用的目标提议方法包括基于超像素分组（例如，选择性搜索[4]，CPMC[22]，MCG[23]）和那些基于滑动窗口的方法（例如窗口中的目标[24]，EdgeBoxes[6]）。目标提议方法被采用为独立于检测

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 into a convolutional layer for detecting multiple classspecific 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

器（例如，选择性搜索[4]目标检测器，R-CNN[5]和 Fast R-CNN[2]）的外部模块。

用于目标检测的深度网络。R-CNN 方法[5]端到端地对 CNN 进行训练，将提议区域分类为目标类别或背景。R-CNN 主要作为分类器，并不能预测目标边界（除了通过边界框回归进行细化）。其准确度取决于区域提议模块的性能（参见[20]中的比较）。一些论文提出了使用深度网络来预测目标边界框的方法[25]，[9]，[26]，[27]。在 OverFeat 方法[9]中，训练一个全连接层来预测假定单个目标定位任务的边界框坐标。然后将全连接层变成卷积层，用于检测多个类别的目标。MultiBox 方法[26]，[27]从网络中生成区域提议，网络最后的全连接层同时预测多个类别不相关的边界框，并推广到 OverFeat 的“单边界框”方式。这些类别不可知的边界框框被用作 R-CNN 的提议区域[5]。与我们的全卷积方案相比，MultiBox 提议网络适用于单张裁剪图像或多张大型裁剪图像（例如 224×224）。MultiBox 在提议区域和检测网络之间不共享特征。稍后在我们的方法上下文中会讨论 OverFeat 和 MultiBox。与我们的工作同时进行的，DeepMask 方法[28]是为学习分割提议区域而开发的。

卷积的共享计算[9]，[1]，[29]，[7]，[2]已经越来越受到人们的关注，因为它可以有效而准确地进行视觉识别。OverFeat 论文[9]计算图像金字塔的卷积特征用于分类，定位和检测。共享卷积特征映射的自适应大小池化（SPP）[1]被开发用于有效的基于区域的目标检测[1]，[30]和语义分割[29]。Fast R-CNN[2]能够

[2] enables end-to-end detector training on shared convolutional features and shows compelling accuracy and speed.

对共享卷积特征进行端到端的检测器训练，并显示出令人信服的准确性和速度。

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.Faster R-CNN

我们的目标检测系统，称为 Faster R-CNN，由两个模块组成。第一个模块是提议区域的深度全卷积网络，第二个模块是使用提议区域的 Fast R-CNN 检测器[2]。整个系统是一个单个的，统一的目标检测网络（图 2）。使用最近流行的“注意力”[31]机制的神经网络术语，RPN 模块告诉 Fast R-CNN 模块在哪里寻找。在第 3.1 节中，我们介绍了区域提议网络的设计和属性。在第 3.2 节中，我们开发了用于训练具有共享特征模块的算法。

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, which has 5 shareable convolutional layers and the Simonyan and Zisserman model 3, which has 13 shareable convolutional layers.

3.1 区域提议网络

区域提议网络（RPN）以任意大小的图像作为输入，输出一组矩形的目标提议，每个提议都有一个目标得分。我们用全卷积网络[7]对这个过程进行建模，我们将在本节进行描述。因为我们的最终目标是与 Fast R-CNN 目标检测网络[2]共享计算，所以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，我们研究了具有 5 个共享卷积层的 Zeiler 和 Fergus 模型[32]（ZF）和具有 13 个共享卷积层的 Simonyan 和 Zisserman 模型[3]（VGG-16）。

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×n** spatial window of the input convolutional feature map. Each sliding window is mapped to a lower-

为了生成区域提议，我们在最后的共享卷积层输出的卷积特征映射上滑动一个小网络。这个小网络将输入卷积特征映射的 n×n 空间窗口作为输入。每个滑动窗口映射到一个低维特征（ZF

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 × n convolutional layer followed by two sibling 1x1 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. The k proposals are parameterized *relative* to k reference boxes, which we call 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*H (typically ~2,400), there are W*H*k 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. As a comparison, the MultiBox method [27] uses k-means to

为 256 维，VGG 为 512 维，后面是 ReLU[33]）。这个特征被输入到两个子全连接层——一个边界框回归层（reg）和一个边界框分类层（cls）。在本文中，我们使用 n=3，注意输入图像上的有效感受野是大的（ZF 和 VGG 分别为 171 和 228 个像素）。图 3（左）显示了这个小型网络的一个位置。请注意，因为小网络以滑动窗口方式运行，所有空间位置共享全连接层。这种架构通过一个 nxn 卷积层，后面是两个子 1x1 卷积层（分别用于 reg 和 cls）自然地实现。

3.1.1Anchors

在每个滑动窗口位置，我们同时预测多个区域提议，其中每个位置可能提议的最大数目表示为 k。因此，*reg* 层具有 4k 个输出，编码 k 个边界框的坐标，*cls* 层输出 2k 个分数，估计每个提议是目标或不是目标的概率。相对于我们称之为锚点的 k 个参考边界框，k 个提议是参数化的。锚点位于所讨论的滑动窗口的中心，并与一个尺度和长宽比相关（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。对于大小为 W*H（通常约为 2400）的卷积特征映射，总共有 W*H*k 个锚点。

平移不变的 Anchors

我们的方法的一个重要特性是它是*平移不变*的，无论是在 Anchor 还是计算相对于 Anchor 的区域提议的函数。如果在图像中平移目标，提议区域应该平移，并且同样的函数应该能够在任一位置预测提议。平移不变特性是由我们的方法保证的。作为比较，MultiBox 方法[27]使用 k-means 生成 800 个锚点，

generate 800 anchors, which are not translation invariant. So MultiBox does not guarantee that the same proposal is generated if an object is translated.

这不是平移不变的。所以如果平移目标，MultiBox 不保证会生成相同的提议。

The translation-invariant property also reduces the model size. MultiBox has a (4+1)×800-dimensional fully-connected output layer, whereas our method has a (4+2)×9-dimensional convolutional output layer in the case of k=9 anchors. As a result, our output layer has 2.8e4 parameters (512×(4+2)×9 for VGG-16), two orders of magnitude fewer than MultiBox's output layer that has 6.1e6 parameters (1536×(4+1)×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. We expect our method to have less risk of overfitting on small datasets, like PASCAL VOC.

平移不变特性也减小了模型的大小。MultiBox 有(4+1)x800 维的全连接输出层，而我们的方法在 k=9 个锚点的情况下有(4+2)x9 维的卷积输出层。因此，对于 VGG-16，我们的输出层具有 2.8e4 个参数（对于 VGG-16 为 512×(4+2)×9），比 MultiBox 输出层的 6.1e6 个参数少了两个数量级（对于 MultiBox [27]中的 GoogleNet[34]为 1536×(4+1)×800）。如果考虑到特征投影层，我们的提议层仍然比 MultiBox 少一个数量级。我们期望我们的方法在 PASCAL VOC 等小数据集上有更小的过拟合风险。

Multi-Scale Anchors as Regression References

多尺度 Anchor 作为回归参考

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 5x7 and 7x5). 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].

我们的锚点设计提出了一个新的方案来解决多尺度（和长宽比）。如图 1 所示，多尺度预测有两种流行的方法。第一种方法是基于图像/特征金字塔，例如 DPM[8]和基于 CNN 的方法[9]，[1]，[2]中。图像在多个尺度上进行缩放，并且针对每个尺度（图 1（a））计算特征映射（HOG[8]或深卷积特征[9]，[1]，[2]）。这种方法通常是有用的，但是非常耗时。第二种方法是在特征映射上使用多尺度（和/或长宽比）的滑动窗口。例如，在 DPM[8]中，使用不同的滤波器大小（例如 5x7 和 7x5）分别对不同长宽比的模型进行训练。如果用这种方法来解决多尺度问题，可以把它看作是一个"滤波器金字塔"（图 1（b））。第二种方法通常与第一种方法联合采用[8]。

As a comparison, our anchor-based method is built on a pyramid of anchors, which is more

作为比较，我们的基于 Anchor 方法建立在 Anhcor 金字塔上，

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).

这是更具成本效益的。我们的方法参照多尺度和长宽比的 Anchor Box 来分类和回归边界框。它只依赖单一尺度的图像和特征映射，并使用单一尺寸的滤波器（特征映射上的滑动窗口）。我们通过实验来展示这个方案解决多尺度和尺寸的效果（表 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.

由于这种基于 Anchor 的多尺度设计，我们可以简单地使用在单尺度图像上计算的卷积特征，Fast R-CNN 检测器也是这样做的[2]。多尺度 Anchor 设计是共享特征的关键组件，不需要额外的成本来处理尺度。

3.1.2 Loss Function

3.1.2 损失函数

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 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.

为了训练 RPN，我们为每个 Anchor 分配一个二值类别标签（是目标或不是目标）。我们给两种 Anchor 分配一个正标签：（i）具有与实际边界框的重叠最高交并比（IoU）的锚点，或者（ii）具有与实际边界框的重叠超过 0.7 IoU 的锚点。注意，单个真实边界框可以为多个 Anchor 分配正标签。通常第二个条件足以确定正样本；但我们仍然采用第一个条件，因为在一些极少数情况下，第二个条件可能找不到正样本。对于所有的真实边界框，如果一个 Anchor 的 IoU 比率低于 0.3，我们给非正面的 Anchor 分配一个负标签。既不正面也不负面的 Anchor 不会有助于训练目标函数。

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:

根据这些定义，我们对目标函数 Fast R-CNN[2]中的多任务损失进行最小化。我们对图像的损失函数定义为：

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*)$$

$$+\lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

Here, **i** is the index of an anchor in a mini-batch and pi is the predicted probability of anchor **i** being an object. The ground-truth label pi* is 1 if the anchor is positive, and is 0 if the anchor is negative. ti is a vector representing the 4 parameterized coordinates of the predicted bounding box, and ti∗ is that of the ground-truth box associated with a positive anchor. The classification loss Lcls is log loss over two classes (object vs not object). For the regression loss, we use Lreg(ti, ti*)=R(ti − ti*) where R is the robust loss function (smooth L1) defined in [2]. The term pi*Lreg means the regression loss is activated only for positive anchors (pi*=1) and is disabled otherwise (pi*=0). The outputs of the *cls* and *reg* layers consist of pi and ti respectively.

The two terms are normalized by Ncls and Nreg 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 (ie, Ncls=256) and the reg term is normalized by the number of anchor locations (ie, Nreg∼2,400). By default we set λ=10 λ =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]:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*)$$

$$+\lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

其中，i 是一个小批量数据中锚点的索引，pi 是 Anchor i 作为目标的预测概率。如果 Anchor 为正，真实标签 pi*为 1，如果 Anchor 为负，则为 0。ti 是表示预测边界框 4 个参数化坐标的向量，而 ti*是与正 Anchor 相关的真实边界框的向量。分类损失 Lcls 是两个类别上（目标或不是目标）的对数损失。对于回归损失，我们使用 Lreg(ti, ti*)=R(ti – ti*)，其中 R 是在[2]中定义的鲁棒损失函数（smooth L1）。项 pi*Lreg 表示回归损失仅对于正 Anchor 激活，否则被禁用（pi*=0）。*cls* 和 *reg* 层的输出分别由 pi 和 ti 组成。

这两个项用 Ncls 和 Nreg 进行标准化，并由一个平衡参数 λ 加权。在我们目前的实现中（如在发布的代码中），方程（1）中的 cls 项通过小批量数据的大小（即 Ncls=256）进行归一化，reg 项根据锚点位置的数量（即，Nreg~24000）进行归一化。默认情况下，我们设置 λ=10 λ =10，因此 *cls* 和 *reg* 项的权重大致相等。我们通过实验显示，结果对宽范围的 λ 值不敏感(表 9)。我们还注意到，上面的归一化不是必需的，可以简化。

对于边界框回归，我们采用[5]中的 4 个坐标参数化：

$$t_{\mathrm{x}}=\left(x-x_{\mathrm{a}}\right)/w_{\mathrm{a}},\quad t_{\mathrm{y}}=\left(y-y_{\mathrm{a}}\right)/h_{\mathrm{a}}$$

$$t_{\mathrm{w}}=\log\left(w/w_{\mathrm{a}}\right),\quad t_{\mathrm{h}}=\log\left(h/h_{\mathrm{a}}\right)$$

$$t_{\mathrm{x}}^{*}=\left(x^{*}-x_{\mathrm{a}}\right)/w_{\mathrm{a}},\quad t_{\mathrm{y}}^{*}=\left(y^{*}-y_{\mathrm{a}}\right)/h_{\mathrm{a}}$$

$$t_{\mathrm{w}}^{*}=\log\left(w^{*}/w_{\mathrm{a}}\right),\quad t_{\mathrm{h}}^{*}=\log\left(h^{*}/h_{\mathrm{a}}\right)$$

where x, y, w, and h denote the box's center coordinates and its width and height. Variables x, xa, 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 (3x3) 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.

$$t_{\mathrm{x}}=\left(x-x_{\mathrm{a}}\right)/w_{\mathrm{a}},\quad t_{\mathrm{y}}=\left(y-y_{\mathrm{a}}\right)/h_{\mathrm{a}}$$

$$t_{\mathrm{w}}=\log\left(w/w_{\mathrm{a}}\right),\quad t_{\mathrm{h}}=\log\left(h/h_{\mathrm{a}}\right)$$

$$t_{\mathrm{x}}^{*}=\left(x^{*}-x_{\mathrm{a}}\right)/w_{\mathrm{a}},\quad t_{\mathrm{y}}^{*}=\left(y^{*}-y_{\mathrm{a}}\right)/h_{\mathrm{a}}$$

$$t_{\mathrm{w}}^{*}=\log\left(w^{*}/w_{\mathrm{a}}\right),\quad t_{\mathrm{h}}^{*}=\log\left(h^{*}/h_{\mathrm{a}}\right)$$

其中，x，y，w 和 h 表示边界框的中心坐标及其宽和高。变量 x，xa 和 x*分别表示预测边界框，Anchor Box 和实际边界框（类似于 y,w,h）。这可以被认为是从 Anchor Box 到邻近的实际边界框的回归。

然而，我们的方法通过与之前的基于 RoI（感兴趣区域）方法[1]，[2]不同的方式来实现边界框回归。在[1]，[2]中，对任意大小的 RoI 池化的特征执行边界框回归，并且回归权重由所有区域大小共享。在我们的公式中，用于回归的特征在特征映射上具有相同的空间大小（3x3）。为了说明不同的大小，学习一组 k 个边界框回归器。每个回归器负责一个尺度和一个长宽比，而 k 个回归器不共享权重。因此，由于 Anchor 的设计，即使特征具有固定的尺度/比例，仍然可以预测各种尺寸的边界框。

3.1.3 训练 RPN

RPN 可以通过反向传播和随机梯度下降（SGD）进行端对端训练[35]。我们遵循[2]的"以图像为中心"的采样策略来训练

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.

这个网络。每个小批量数据都从包含许多正负例 Anchor 的单张图像中产生。对所有 Anchor 的损失函数进行优化是可能的，但是这样会偏向于负样本，因为它们是占主导地位的。取而代之的是，我们在图像中随机采样 256 个 Anchor，计算一个小批量数据的损失函数，其中采样的正 Anchor 和负 Anchor 的比率可达 1:1。如果图像中的正样本少于 128 个，我们使用负样本填充小批量数据。

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].

我们通过从标准方差为 0.01 的零均值高斯分布中提取权重来随机初始化所有新层。所有其他层（即共享卷积层）通过预训练的 ImageNet 分类模型[36]来初始化，如同标准实践[5]。我们调整 ZF 网络的所有层，以及 VGG 网络的 conv3_1 及其之上的层以节省内存[2]。对于 60k 的小批量数据，我们使用 0.001 的学习率，对于 PASCAL VOC 数据集中的下一个 20k 小批量数据，使用 0.0001。我们使用 0.9 的动量和 0.0005 的重量衰减[37]。我们的实现使用 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).

3.2RPN 和 Fast R-CNN 共享特征

到目前为止，我们已经描述了如何训练用于区域提议生成的网络，没有考虑将利用这些提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]。接下来我们介绍一些算法，学习由 RPN 和 Fast R-CNN 组成的具有共享卷积层的统一网络（图 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

独立训练的 RPN 和 Fast R-CNN 将以不同的方式修改卷积层。因此，我们需要开发一种允许在两个网络之间共享卷积层的技

convolutional layers between the 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

术，而不是学习两个独立的网络。我们讨论三个方法来训练具有共享特征的网络：

(1) 交替训练。在这个解决方案中，我们首先训练 RPN，并使用这些提议来训练 Fast R-CNN。由 Fast R-CNN 微调的网络然后被用于初始化 RPN，并且重复这个过程。这是本文所有实验中使用的解决方案。

(2) 近似联合训练。在这个解决方案中，RPN 和 Fast R-CNN 网络在训练期间合并成一个网络，如图 2 所示。在每次 SGD 迭代中，前向传递生成区域提议，在训练 Fast R-CNN 检测器将这看作是固定的、预计算的提议。反向传播像往常一样进行，其中对于共享层，组合来自 RPN 损失和 Fast R-CNN 损失的反向传播信号。这个解决方案很容易实现。但是这个解决方案忽略了关于提议边界框的坐标（也是网络响应）的导数，因此是近似的。在我们的实验中，我们实验发现这个求解器产生了相当的结果，与交替训练相比，训练时间减少了大约 25–50%。这个求解器包含在我们发布的 Python 代码中。

(3) 非近似的联合训练。如上所述，由 RPN 预测的边界框也是输入的函数。Fast R-CNN 中的 RoI 池化层[2]接受卷积特征以及预测的边界框作为输入，所以理论上有效的反向传播求解器也应该包括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。在一个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分的 RoI 池化层。这是一个重要的问题，可以通过[15]中提出的“RoI 扭曲”层给出解决方案，

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.

这超出了本文的范围。

四步交替训练。在本文中，我们采用实用的四步训练算法，通过交替优化学习共享特征。在第一步中，我们按照 3.1.3 节的描述训练 RPN。该网络使用 ImageNet 的预训练模型进行初始化，并针对区域提议任务进行了端到端的微调。在第二步中，我们使用由第一步 RPN 生成的提议，由 Fast R-CNN 训练单独的检测网络。该检测网络也由 ImageNet 的预训练模型进行初始化。此时两个网络不共享卷积层。在第三步中，我们使用检测器网络来初始化 RPN 训练，但是我们修正共享的卷积层，并且只对 RPN 特有的层进行微调。现在这两个网络共享卷积层。最后，保持共享卷积层的固定，我们对 Fast R-CNN 的独有层进行微调。因此，两个网络共享相同的卷积层并形成统一的网络。类似的交替训练可以运行更多的迭代，但是我们只观察到可以忽略的改进。

3.3 实现细节

我们在单尺度图像上训练和测试区域提议和目标检测网络[1]，[2]。我们重新缩放图像，使得它们的短边是 s=600s=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，但不会表现出速度与精度的良好折衷[2]。在重新缩放的图像上，最后卷积层上的 ZF 和 VGG 网络的总步长为 16 个像素，因此在调整大小（～500*375）之前，典型的 PASCAL 图像上的总步长为～10 个像素。即使如此大的步长也能提供良好的效果，尽管步幅更小，精度可能会进一步提高。

For anchors, we use 3 scales with box areas of 128*128 256*256, and 512*512 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.

对于 Anchor，我们使用了 3 个尺度，边界框面积分别为 128*128，256*256 和 512*512 个像素，以及 1:1，1:2 和 2:1 的长宽比。这些超参数不是针对特定数据集仔细选择的，我们将在下一节中提供有关其作用的消融实验。如上所述，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个尺度的区域，节省了大量的运行时间。图 3（右）显示了我们的方法在广泛的尺度和长宽比方面的能力。表 1 显示了使用 ZF 网络的每个 Anchor 学习到的平均提议大小。我们注意到，我们的算法允许预测比基础感受野更大。这样的预测不是不可能的——如果只有目标的中间部分是可见的，那么仍然可以粗略地推断出目标的范围。

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 (≈60×40×9) 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.

超出图像边界的 Anchor Box 需要小心处理。在训练过程中，我们忽略了所有的越界锚点，所以不会造成损失。对于一个典型的 1000×600 的图片，总共将会有大约 20000（≈60×40×9）个锚点。越界 Anchor 被忽略，每张图像约有 6000 个 Anchor 用于训练。如果越界异常值在训练中不被忽略，则会在目标函数中引入大的，难以纠正的误差项，且训练不会收敛。但在测试过程中，我们仍然将全卷积 RPN 应用于整张图像。这可能会产生超出边界的提议边界框，我们剪切到图像边界。

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

一些 RPN 提议相互之间高度重叠。为了减少冗余，我们在提议区域根据他们的 *cls* 分数采取非极大值抑制（NMS）。我们将 NMS 的 IoU 阈值固定为 0.7，这就给每张图像留下了大约 2000 个提议区域。正如我们将要展示的那样，NMS 不会损害最终的检测准确性，但会大大减少提议的数量。在 NMS 之后，我们使

the following, we train Fast R-CNN using 2000 RPN proposals, but evaluate different numbers of proposals at test-time.

用前 N 个提议区域来进行检测。接下来，我们使用 2000 个 RPN 提议对 Fast R-CNN 进行训练，但在测试时评估不同数量的提议。