Feature Pyramid Networks for Object Detection论文翻译——中英文对照

Feature Pyramid Networks for Object Detection论文翻译——中英文对照

文章作者：Tyan
博客：noahsnail.com | CSDN | 简书

翻译论文汇总：https://github.com/SnailTyan/deep-learning-papers-translation

Feature Pyramid Networks for Object Detection

Abstract

Feature pyramids are a basic component in recognition systems for detecting objects at different scales. But recent deep learning object detectors have avoided pyramid representations, in part because they are compute and memory intensive. In this paper, we exploit the inherent multi-scale, pyramidal hierarchy of deep convolutional networks to construct feature pyramids with marginal extra cost. A top-down architecture with lateral connections is developed for building high-level semantic feature maps at all scales. This architecture, called a Feature Pyramid Network (FPN), shows significant improvement as a generic feature extractor in several applications. Using FPN in a basic Faster R-CNN system, our method achieves state-of-the-art single-model results on the COCO detection benchmark without bells and whistles, surpassing all existing single-model entries including those from the COCO 2016 challenge winners. In addition, our method can run at 6 FPS on a GPU and thus is a practical and accurate solution to multi-scale object detection. Code will be made publicly available.

摘要

特征金字塔是识别系统中用于检测不同尺度目标的基本组件。但最近的深度学习目标检测器已经避免了金字塔表示，部分原因是它们是计算和内存密集型的。在本文中，我们利用深度卷积网络内在的多尺度、金字塔分级来构造具有很少额外成本的特征金字塔。开发了一种具有横向连接的自顶向下架构，用于在所有尺度上构建高级语义特征映射。这种称为特征金字塔网络（FPN）的架构在几个应用程序中作为通用特征提取器表现出了显著的改进。在一个基本的Faster R-CNN系统中使用FPN，没有任何不必要的东西，我们的方法可以在COCO检测基准数据集上取得最先进的单模型结果，结果超过了所有现有的单模型输入，包括COCO 2016挑战赛的获奖者。此外，我们的方法可以在GPU上以6FPS运行，因此是多尺度目标检测的实用和准确的解决方案。代码将公开发布。

1. Introduction

Recognizing objects at vastly different scales is a fundamental challenge in computer vision. Feature pyramids built upon image pyramids (for short we call these featurized image pyramids) form the basis of a standard solution [1] (Fig. 1(a)). These pyramids are scale-invariant in the sense that an object's scale change is offset by shifting its level in the pyramid. Intuitively, this property enables a model to detect objects across a large range of scales by scanning the model over both positions and pyramid levels.

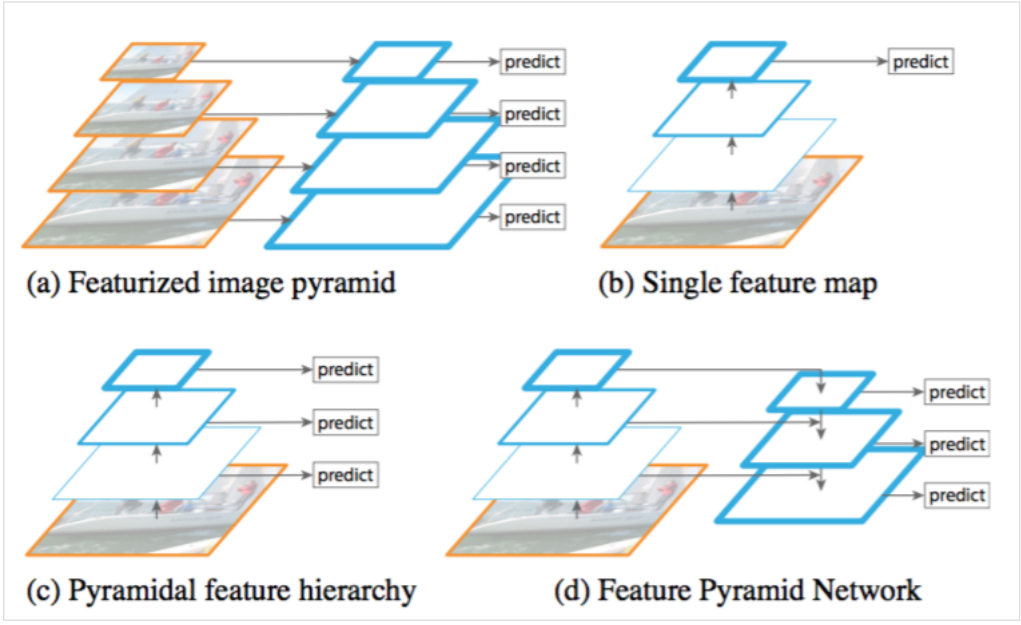

Figure 1. (a) Using an image pyramid to build a feature pyramid. Features are computed on each of the image scales independently, which is slow. (b) Recent detection systems have opted to use only single scale features for faster detection. (c) An alternative is to reuse the pyramidal feature hierarchy computed by a ConvNet as if it were a featurized image pyramid. (d) Our proposed Feature Pyramid Network (FPN) is fast like (b) and (c), but more accurate. In this figure, feature maps are indicate by blue outlines and thicker outlines denote semantically stronger features.

1. 引言

识别不同尺度的目标是计算机视觉中的一个基本挑战。建立在图像金字塔之上的特征金字塔（我们简称为特征化图像金字塔）构成了标准解决方案的基础[1]（图1（a））。这些金字塔是尺度不变的，因为目标的尺度变化是通过在金字塔中移动它的层级来抵消的。直观地说，该属性使模型能够通过在位置和金字塔等级上扫描模型来检测大范围尺度内的目标。

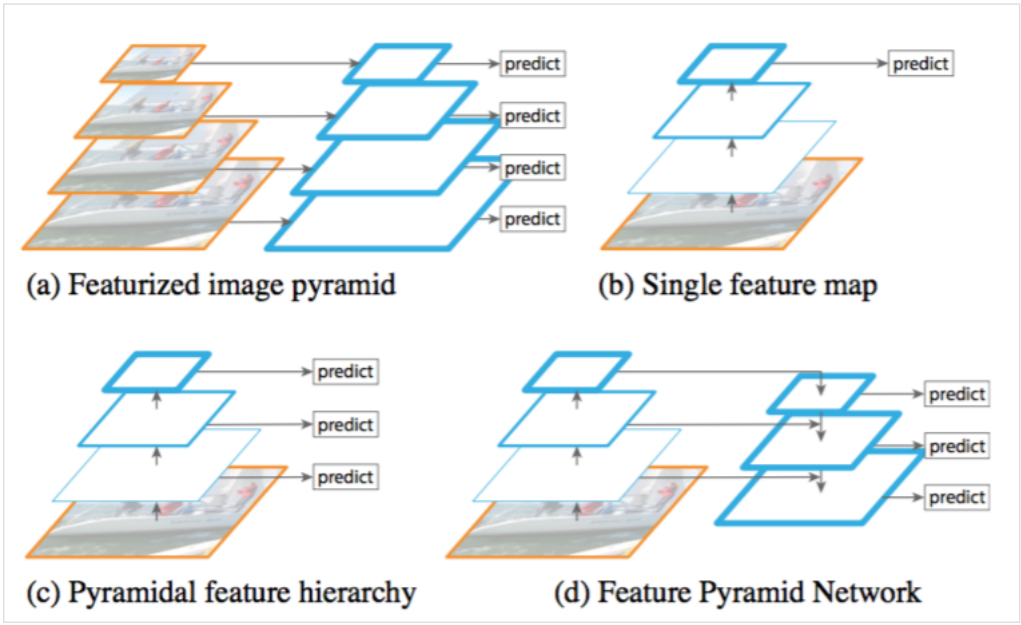

图1。（a）使用图像金字塔构建特征金字塔。每个图像尺度上的特征都是独立计算的，速度很慢。（b）最近的检测系统选择只使用单一尺度特征进行更快的检测。（c）另一种方法是重用ConvNet计算的金字塔特征层次结构，就好像它是一个特征化的图像金字塔。（d）我们提出的特征金字塔网络（FPN）与（b）和（c）类似，但更准确。在该图中，特征映射用蓝色轮廓表示，较粗的轮廓表示语义上较强的特征。

Featurized image pyramids were heavily used in the era of hand-engineered features [5, 25]. They were so critical that object detectors like DPM [7] required dense scale sampling to achieve good results (e.g., 10 scales per octave). For recognition tasks, engineered features have largely been replaced with features computed by deep convolutional networks (ConvNets) [19, 20]. Aside from being capable of representing higher-level semantics, ConvNets are also more robust to variance in scale and thus facilitate recognition from features computed on a single input scale [15, 11, 29] (Fig. 1(b)). But even with this robustness, pyramids are still needed to get the most accurate results. All recent top entries in the ImageNet [33] and COCO [21] detection challenges use multi-scale testing on featurized image pyramids (e.g., [16, 35]). The

principle advantage of featurizing each level of an image pyramid is that it produces a multi-scale feature representation in which all levels are semantically strong, including the high-resolution levels.

特征化图像金字塔在手工设计的时代被大量使用[5，25]。它们非常关键，以至于像DPM[7]这样的目标检测器需要密集的尺度采样才能获得好的结果（例如每组10个尺度，octave含义参考SIFT特征）。对于识别任务，工程特征大部分已经被深度卷积网络（ConvNets）[19，20]计算的特征所取代。除了能够表示更高级别的语义，ConvNets对于尺度变化也更加鲁棒，从而有助于从单一输入尺度上计算的特征进行识别[15，11，29]（图1（b））。但即使有这种鲁棒性，金字塔仍然需要得到最准确的结果。在ImageNet[33]和COCO[21]检测挑战中，最近的所有排名靠前的输入都使用了针对特征化图像金字塔的多尺度测试（例如[16，35]）。对图像金字塔的每个层次进行特征化的主要优势在于它产生了多尺度的特征表示，其中所有层次上在语义上都很强，包括高分辨率层。

Nevertheless, featurizing each level of an image pyramid has obvious limitations. Inference time increases considerably (e.g., by four times [11]), making this approach impractical for real applications. Moreover, training deep networks end-to-end on an image pyramid is infeasible in terms of memory, and so, if exploited, image pyramids are used only at test time [15, 11, 16, 35], which creates an inconsistency between train/test-time inference. For these reasons, Fast and Faster R-CNN [11, 29] opt to not use featurized image pyramids under default settings.

尽管如此，特征化图像金字塔的每个层次都具有明显的局限性。推断时间显著增加（例如，四倍[11]），使得这种方法在实际应用中不切实际。此外，在图像金字塔上端对端地训练深度网络在内存方面是不可行的，所以如果被采用，图像金字塔仅在测试时被使用[15，11，16，35]，这造成了训练/测试时推断的不一致性。出于这些原因，Fast和Faster R-CNN[11，29]选择在默认设置下不使用特征化图像金字塔。

However, image pyramids are not the only way to compute a multi-scale feature representation. A deep ConvNet computes a feature hierarchy layer by layer, and with subsampling layers the feature hierarchy has an inherent multi-scale, pyramidal shape. This in-network feature hierarchy produces feature maps of different spatial resolutions, but introduces large semantic gaps caused by different depths. The high-resolution maps have low-level features that harm their representational capacity for object recognition.

但是，图像金字塔并不是计算多尺度特征表示的唯一方法。深层ConvNet逐层计算特征层级，而对于下采样层，特征层级具有内在的多尺度金字塔形状。这种网内特征层级产生不同空间分辨率的特征映射，但引入了由不同深度引起的较大的语义差异。高分辨率映射具有损害其目标识别表示能力的低级特征。

The Single Shot Detector (SSD) [22] is one of the first attempts at using a ConvNet's pyramidal feature hierarchy as if it were a featurized image pyramid (Fig. 1(c)). Ideally, the SSD-style pyramid would reuse the multi-scale feature maps from different layers computed in the forward pass and thus come free of cost. But to avoid using low-level features SSD foregoes reusing already computed layers and instead builds the pyramid starting from high up in the network (e.g., conv4_3 of VGG nets [36]) and then by adding several new layers. Thus it misses the opportunity to reuse the higher-resolution maps of the feature hierarchy. We show that these are important for detecting small objects.

单次检测器（SSD）[22]是首先尝试使用ConvNet的金字塔特征层级中的一个，好像它是一个特征化的图像金字塔（图1（c））。理想情况下，SSD风格的金字塔将重用正向传递中从不同层中计算的多尺度特征映射，因此是零成本的。但为了避免使用低级特征，SSD放弃重用已经计算好的图层，而从网络中的最高层开始构建金字塔（例如，VGG网络的conv4_3[36]），然后添加几个新层。因此它错过了重用特征层级的更高分辨率映射的机会。我们证明这些对于检测小目标很重要。

The goal of this paper is to naturally leverage the pyramidal shape of a ConvNet's feature hierarchy while creating a feature pyramid that has strong semantics at all scales. To achieve this goal, we rely on an architecture that combines low-resolution, semantically strong features with high-resolution, semantically weak features via a top-down pathway and lateral connections (Fig. 1(d)). The result is a feature pyramid that has rich semantics at all levels and is built quickly from a single input image scale. In other words, we show how to create in-network feature pyramids that can be used to replace featurized image pyramids without sacrificing representational power, speed, or memory.

本文的目标是自然地利用ConvNet特征层级的金字塔形状，同时创建一个在所有尺度上都具有强大语义的特征金字塔。为了实现这个目标，我们所依赖的架构将低分辨率、强语义的特征与高分辨率、弱语义的特征通过自顶向下的路径和横向连接相结合。（图1（d））。其结果是一个特征金字塔，在所有级别都具有丰富的语义，并且可以从单个输入图像尺度上进行快速构建。换句话说，我们展示了如何创建网络中的特征金字塔，可以用来代替特征化的图像金字塔，而不牺牲表示能力，速度或内存。

Similar architectures adopting top-down and skip connections are popular in recent research [28, 17, 8, 26]. Their goals are to produce a single high-level feature map of a fine resolution on which the predictions are to be made (Fig. 2 top). On the contrary, our method leverages the architecture as a feature pyramid where predictions (e.g., object detections) are independently made on each level (Fig. 2 bottom). Our model echoes a featurized image pyramid, which has not been explored in these works.

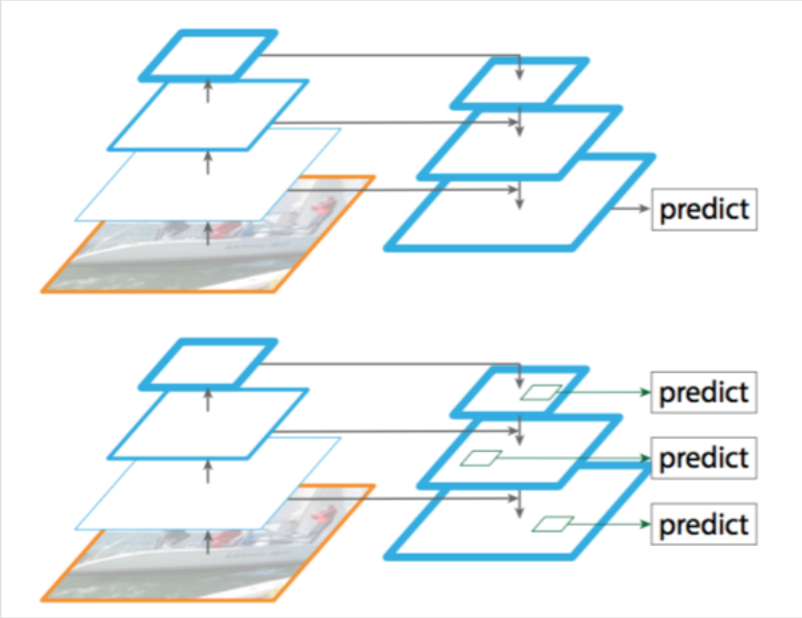

Figure 2. Top: a top-down architecture with skip connections, where predictions are made on the finest level (e.g., [28]). Bottom: our model that has a similar structure but leverages it as a feature pyramid, with predictions made independently at all levels.

最近的研究[28，17，8，26]中流行采用自顶向下和跳跃连接的类似架构。他们的目标是生成具有高分辨率的单个高级特征映射，并在其上进行预测（图2顶部）。相反，我们的方法利用这个架构作为特征金字塔，其中预测（例如目标检测）在每个级别上独立进行（图2底部）。我们的模型反映了一个特征化的图像金字塔，这在这些研究中还没有探索过。

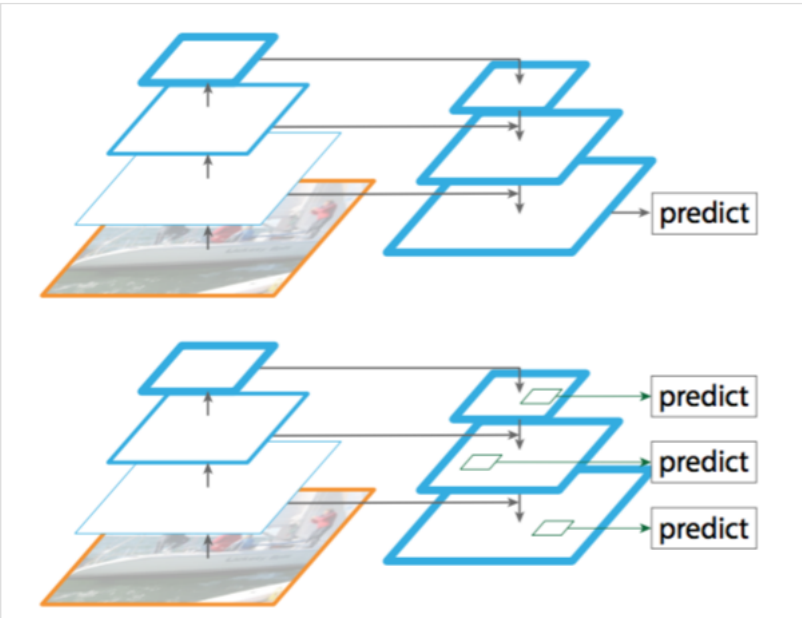

图2。顶部：带有跳跃连接的自顶向下的架构，在最好的级别上进行预测（例如，[28]）。底部：我们的模型具有类似的结构，但将其用作特征金字塔，并在各个层级上独立进行预测。

We evaluate our method, called a Feature Pyramid Network (FPN), in various systems for detection and segmentation [11, 29, 27]. Without bells and whistles, we report a state-of-the-art single-model result on the challenging COCO detection benchmark [21] simply based on FPN and a basic Faster R-CNN detector [29], surpassing all existing heavily-engineered single-model entries of competition winners. In ablation experiments, we find that for bounding box proposals, FPN significantly increases the Average Recall (AR) by 8.0 points; for object detection, it improves the COCO-style Average Precision (AP) by 2.3 points and PASCAL-style AP by 3.8 points, over a strong single-scale baseline of Faster R-CNN on ResNets [16]. Our method is also easily extended to mask proposals and improves both instance segmentation AR and speed over state-of-the-art methods that heavily depend on image pyramids.

我们评估了我们称为特征金字塔网络（FPN）的方法，其在各种系统中用于检测和分割[11，29，27]。没有任何不必要的东西，我们在具有挑战性的COCO检测基准数据集上报告了最新的单模型结果，仅仅基于FPN和基本的Faster R-CNN检测器[29]，就超过了竞赛获奖者所有现存的严重工程化的单模型竞赛输入。在消融实验中，我们发现对于边界框提议，FPN将平均召回率（AR）显著增加了8个百分点；对于目标检测，它将COCO型的平均精度（AP）提高了2.3个百分点，PASCAL型AP提高了3.8个百分点，超过了ResNet[16]上Faster R-CNN强大的单尺度基准线。我们的方法也很容易扩展掩模提议，改进实例分隔AR，加速严重依赖图像金字塔的最先进方法。

In addition, our pyramid structure can be trained end-to-end with all scales and is used consistently at train/test time, which would be memory-infeasible using image pyramids. As a result, FPNs are able to achieve higher accuracy than all existing state-of-the-art methods. Moreover, this improvement is achieved without increasing testing time over the single-scale baseline. We believe these advances will facilitate future research and applications. Our code will be made publicly available.

另外，我们的金字塔结构可以通过所有尺度进行端对端培训，并且在训练/测试时一致地使用，这在使用图像金字塔时是内存不可行的。因此，FPN能够比所有现有的最先进方法获得更高的准确度。此外，这种改进是在不增加单尺度基准测试时间的情况下实现的。我们相信这些进展将有助于未来的研究和应用。我们的代码将公开发布。

2. Related Work

Hand-engineered features and early neural networks. SIFT features [25] were originally extracted at scale-space extrema and used for feature point matching. HOG features [5], and later SIFT features as well, were computed densely over entire image pyramids. These HOG and SIFT pyramids have been used in numerous works for image classification, object detection, human pose estimation, and more. There has also been significant interest in computing featurized image pyramids quickly. Dollar et al.[6] demonstrated fast pyramid computation by first computing a sparsely sampled (in scale) pyramid and then interpolating missing levels. Before HOG and SIFT, early work on face detection with ConvNets [38, 32] computed shallow networks over image pyramids to detect faces across scales.

2. 相关工作

手工设计特征和早期神经网络。SIFT特征[25]最初是从尺度空间极值中提取的，用于特征点匹配。HOG特征[5]，以及后来的SIFT特征，都是在整个图像金字塔上密集计算的。这些HOG和SIFT金字塔已在许多工作中得到了应用，用于图像分类，目标检测，人体姿势估计等。这对快速计算特征化图像金字塔也很有意义。Dollar等人[6]通过先计算一个稀疏采样（尺度）金字塔，然后插入缺失的层级，从而演示了快速金字塔计算。在HOG和SIFT之前，使用ConvNet[38，32]的早期人脸检测工作计算了图像金字塔上的浅网络，以检测跨尺度的人脸。

Deep ConvNet object detectors. With the development of modern deep ConvNets [19], object detectors like OverFeat [34] and R-CNN [12] showed dramatic improvements in accuracy. OverFeat adopted a strategy similar to early neural network face detectors by applying a ConvNet as a sliding window detector on an image pyramid. R-CNN adopted a region proposal-based strategy [37] in which each proposal was scale-normalized before classifying with a ConvNet. SPPnet [15] demonstrated that such region-based detectors could be applied much more efficiently on feature maps extracted on a single image scale. Recent and more accurate detection methods like Fast R-CNN [11] and Faster R-CNN [29] advocate using features computed from a single scale, because it offers a good trade-off between accuracy and speed. Multi-scale detection, however, still performs better, especially for small objects.

Deep ConvNet目标检测器。随着现代深度卷积网络[19]的发展，像OverFeat[34]和R-CNN[12]这样的目标检测器在精度上显示出了显著的提高。OverFeat采用了一种类似于早期神经网络人脸检测器的策略，通过在图像金字塔上应用ConvNet作为滑动窗口检测器。R-CNN采用了基于区域提议的策略[37]，其中每个提议在用ConvNet进行分类之前都进行了尺度归一化。SPPnet[15]表明，这种基于区域的检测器可以更有效地应用于在单个图像尺度上提取的特征映射。最近更准确的检测方法，如Fast R-CNN[11]和Faster R-CNN[29]提倡使用从单一尺度计算出的特征，因为它提供了精确度和速度之间的良好折衷。然而，多尺度检测性能仍然更好，特别是对于小型目标。

Methods using multiple layers. A number of recent approaches improve detection and segmentation by using different layers in a ConvNet. FCN [24] sums partial scores for each category over multiple scales to compute semantic segmentations. Hypercolumns [13] uses a similar method for object instance segmentation. Several other approaches (HyperNet [18], ParseNet [23], and ION [2]) concatenate features of multiple layers before computing predictions, which is equivalent to summing transformed features. SSD [22] and MS-CNN [3] predict objects at multiple layers of the feature hierarchy without combining features or scores.

使用多层的方法。一些最近的方法通过使用ConvNet中的不同层来改进检测和分割。FCN[24]将多个尺度上的每个类别的部分分数相加以计算语义分割。Hypercolumns[13]使用类似的方法进行目标实例分割。在计算预测之前，其他几种方法（HyperNet[18]，ParseNet[23]和ION[2]）将多个层的特征连接起来，这相当于累加转换后的特征。SSD[22]和MS-CNN[3]可预测特征层级中多个层的目标，而不需要组合特征或分数。

There are recent methods exploiting lateral/skip connections that associate low-level feature maps across resolutions and semantic levels, including U-Net [31] and SharpMask [28] for segmentation, Recombinator networks [17] for face detection, and Stacked Hourglass networks [26] for keypoint estimation. Ghiasi et al. [8] present a Laplacian pyramid presentation for FCNs to progressively refine segmentation. Although these methods adopt architectures with pyramidal shapes, they are unlike featurized image pyramids [5, 7, 34] where predictions are made independently at all levels, see Fig. 2. In fact, for the pyramidal architecture in Fig. 2 (top), image pyramids are still needed to recognize objects across multiple scales [28].

最近有一些方法利用横向/跳跃连接将跨分辨率和语义层次的低级特征映射关联起来，包括用于分割的U-Net[31]和SharpMask[28]，Recombinator网络[17]用于人脸检测以及Stacked Hourglass网络[26]用于关键点估计。Ghiasi等人[8]为FCN提出拉普拉斯金字塔表示，以逐步细化分割。尽管这些方法采用的是金字塔形状的架构，但它们不同于特征化的图像金字塔[5，7，34]，其中所有层次上的预测都是独立进行的，参见图2。事实上，对于图2（顶部）中的金字塔结构，图像金字塔仍然需要跨多个尺度上识别目标[28]。

3. Feature Pyramid Networks

Our goal is to leverage a ConvNet's pyramidal feature hierarchy, which has semantics from low to high levels, and build a feature pyramid with high-level semantics throughout. The resulting Feature Pyramid Network is general-purpose and in this paper we focus on sliding window proposers (Region Proposal Network, RPN for short) [29] and region-based detectors (Fast R-CNN) [11]. We also generalize FPNs to instance segmentation proposals in Sec.6.

3. 特征金字塔网络

我们的目标是利用ConvNet的金字塔特征层级，该层次结构具有从低到高的语义，并在整个过程中构建具有高级语义的特征金字塔。由此产生的特征金字塔网络是通用的，在本文中，我们侧重于滑动窗口提议（Region Proposal Network，简称RPN）[29]和基于区域的检测器（Fast R-CNN）[11]。在第6节中我们还将FPN泛化到实例细分提议。

Our method takes a single-scale image of an arbitrary size as input, and outputs proportionally sized feature maps at multiple levels, in a fully convolutional fashion. This process is independent of the backbone convolutional architectures (e.g., [19, 36, 16]), and in this paper we present results using ResNets [16]. The construction of our pyramid involves a bottom-up pathway, a top-down pathway, and lateral connections, as introduced in the following.

我们的方法以任意大小的单尺度图像作为输入，并以全卷积的方式输出多层适当大小的特征映射。这个过程独立于主卷积体系结构（例如[19，36，16]），在本文中，我们呈现了使用ResNets[16]的结果。如下所述，我们的金字塔结构包括自下而上的路径，自上而下的路径和横向连接。

Bottom-up pathway. The bottom-up pathway is the feed-forward computation of the backbone ConvNet, which computes a feature hierarchy consisting of feature maps at several scales with a scaling step of 2. There are often many layers producing output maps of the same size and we say these layers are in the same network stage. For our feature pyramid, we define one pyramid level for each stage. We choose the output of the last layer of each stage as our reference set of feature maps, which we will enrich to create our pyramid. This choice is natural since the deepest layer of each stage should have the strongest features.

自下而上的路径。自下向上的路径是主ConvNet的前馈计算，其计算由尺度步长为2的多尺度特征映射组成的特征层级。通常有许多层产生相同大小的输出映射，并且我们认为这些层位于相同的网络阶段。对于我们的特征金字塔，我们为每个阶段定义一个金字塔层。我们选择每个阶段的最后一层的输出作为我们的特征映射参考集，我们将丰富它来创建我们的金字塔。这种选择是自然的，因为每个阶段的最深层应具有最强大的特征。

Specifically, for ResNets [16] we use the feature activations output by each stage's last residual block. We denote the output of these last residual blocks as $\{C_2, C_3, C_4, C_5\}$ for conv2, conv3, conv4, and conv5 outputs, and note that they have strides of {4, 8, 16, 32} pixels with respect to the input image. We do not include conv1 into the pyramid due to its large memory footprint.

具体而言，对于ResNets[16]，我们使用每个阶段的最后一个残差块输出的特征激活。对于conv2，conv3，conv4和conv5输出，我们将这些最后残差块的输出表示为$\{C_2, C_3, C_4, C_5\}$，并注意相对于输入图像它们的步长为{4，8，16，32}个像素。由于其庞大的内存占用，我们不会将conv1纳入金字塔。

Top-down pathway and lateral connections. The top-down pathway hallucinates higher resolution features by upsampling spatially coarser, but semantically stronger, feature maps from higher pyramid levels. These features are then enhanced with features from the bottom-up pathway via lateral connections. Each lateral connection merges feature maps of the same spatial size from the bottom-up pathway and the top-down pathway. The bottom-up feature map is of lower-level semantics, but its activations are more accurately localized as it was subsampled fewer times.

自顶向下的路径和横向连接。自顶向下的路径通过上采样空间上更粗糙但在语义上更强的来自较高金字塔等级的特征映射来幻化更高分辨率的特征。这些特征随后通过来自自下而上路径上的特征经由横向连接进行增强。每个横向连接合并来自自下而上路径和自顶向下路径的具有相同空间大小的特征映射。自下而上的特征映射具有较低级别的语义，但其激活可以更精确地定位，因为它被下采样的次数更少。

Fig. 3 shows the building block that constructs our top-down feature maps. With a coarser-resolution feature map, we upsample the spatial resolution by a factor of 2 (using nearest neighbor upsampling for simplicity). The upsampled map is then merged with the corresponding bottom-up map (which undergoes a 1×1 convolutional layer to reduce channel dimensions) by element-wise addition. This process is iterated until the finest resolution map is generated. To start the iteration, we simply attach a 1×1 convolutional layer on C_5 to produce the coarsest resolution map. Finally, we append a 3 × 3 convolution on each merged map to generate the final feature map, which is to reduce the aliasing effect of upsampling. This final set of feature maps is called $\{P_2, P_3, P_4, P_5\}$, corresponding to $\{C_2, C_3, C_4, C_5\}$ that are respectively of the same spatial sizes.

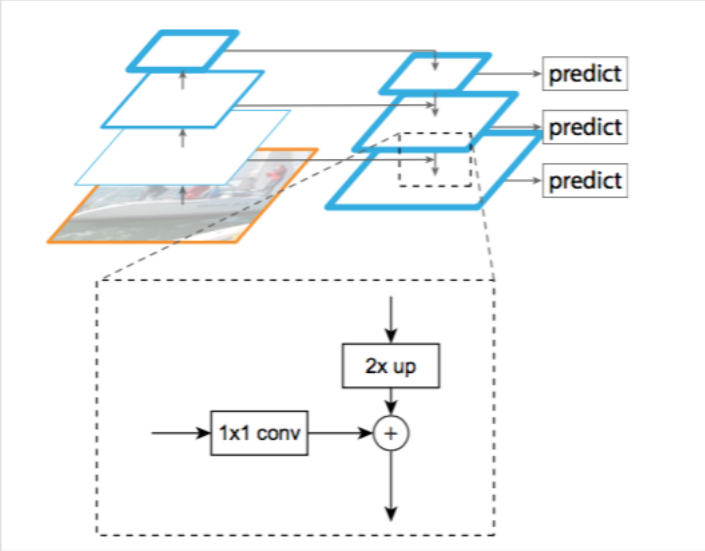

Figure 3. A building block illustrating the lateral connection and the top-down pathway, merged by addition.

图3显示了建造我们的自顶向下特征映射的构建块。使用较粗糙分辨率的特征映射，我们将空间分辨率上采样为2倍（为了简单起见，使用最近邻上采样）。然后通过按元素相加，将上采样映射与相应的自下而上映射（其经过1×1卷积层来减少通道维度）合并。迭代这个过程，直到生成最佳分辨率映射。为了开始迭代，我们只需在C_5上添加一个1×1卷积层来生成最粗糙分辨率映射。最后，我们在每个合并的映射上添加一个3×3卷积来生成最终的特征映射，这是为了减少上采样的混叠效应。这个最终的特征映射集称为$\{P_2, P_3, P_4, P_5\}$，对应于$\{C_2, C_3, C_4, C_5\}$，分别具有相同的空间大小。

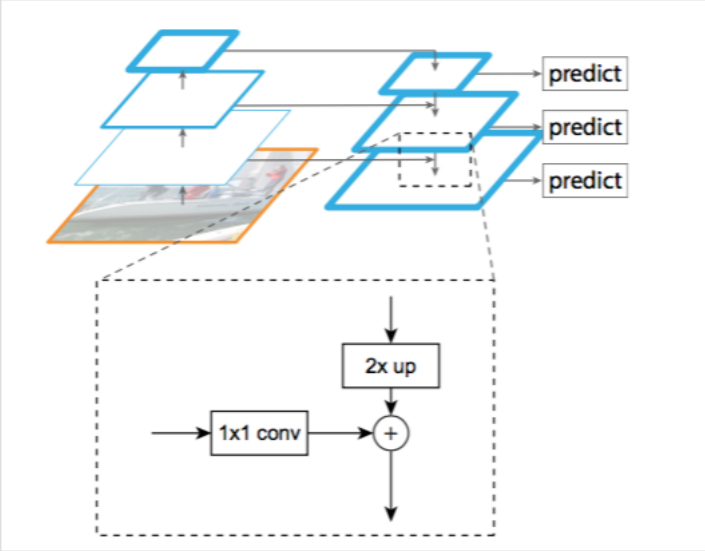

图3。构建模块说明了横向连接和自顶向下路径，通过加法合并。

Because all levels of the pyramid use shared classifiers/regressors as in a traditional featurized image pyramid, we fix the feature dimension (numbers of channels, denoted as d) in all the feature maps. We set $d = 256$ in this paper and thus all extra convolutional layers have 256-channel outputs. There are no non-linearities in these extra layers, which we have empirically found to have minor impacts.

由于金字塔的所有层都像传统的特征图像金字塔一样使用共享分类器/回归器，因此我们在所有特征映射中固定特征维度（通道数记为d）。我们在本文中设置$d = 256$，因此所有额外的卷积层都有256个通道的输出。在这些额外的层中没有非线性，我们在实验中发现这些影响很小。

Simplicity is central to our design and we have found that our model is robust to many design choices. We have experimented with more sophisticated blocks (e.g., using multi-layer residual blocks [16] as the connections) and observed marginally better results. Designing better connection modules is not the focus of this paper, so we opt for the simple design described above.

简洁性是我们设计的核心，我们发现我们的模型对许多设计选择都很鲁棒。我们已经尝试了更复杂的块（例如，使用多层残差块[16]作为连接）并观察到稍微更好的结果。设计更好的连接模块并不是本文的重点，所以我们选择上述的简单设计。

4. Applications

Our method is a generic solution for building feature pyramids inside deep ConvNets. In the following we adopt our method in RPN [29] for bounding box proposal generation and in Fast R-CNN [11] for object detection. To demonstrate the simplicity and effectiveness of our method, we make minimal modifications to the original systems of [29, 11] when adapting them to our feature pyramid.

4. 应用

我们的方法是在深度ConvNets内部构建特征金字塔的通用解决方案。在下面，我们采用我们的方法在RPN[29]中进行边界框提议生成，并在Fast R-CNN[11]中进行目标检测。为了证明我们方法的简洁性和有效性，我们对[29，11]的原始系统进行最小修改，使其适应我们的特征金字塔。

4.1. Feature Pyramid Networks for RPN

RPN [29] is a sliding-window class-agnostic object detector. In the original RPN design, a small subnetwork is evaluated on dense 3×3 sliding windows, on top of a single-scale convolutional feature map, performing object/non-object binary classification and bounding box regression. This is realized by a 3×3 convolutional layer followed by two sibling 1×1 convolutions for classification and regression, which we refer to as a network head. The object/non-object criterion and bounding box regression target are defined with respect to a set of reference boxes called anchors [29]. The anchors are of multiple pre-defined scales and aspect ratios in order to cover objects of different shapes.

4.1. RPN的特征金字塔网络

RPN[29]是一个滑动窗口类不可知的目标检测器。在原始的RPN设计中，一个小型子网络在密集的3×3滑动窗口，单尺度卷积特征映射上进行评估，执行目标/非目标的二分类和边界框回归。这是通过一个3×3的卷积层实现的，后面跟着两个用于分类和回归的1×1兄弟卷积，我们称之为网络头部。目标/非目标标准和边界框回归目标的定义是关于一组称为锚点的参考框的[29]。这些锚点具有多个预定义的尺度和长宽比，以覆盖不同形状的目标。

We adapt RPN by replacing the single-scale feature map with our FPN. We attach a head of the same design (3×3 conv and two sibling 1×1 convs) to each level on our feature pyramid. Because the head slides densely over all locations in all pyramid levels, it is not necessary to have multi-scale anchors on a specific level. Instead, we assign anchors of a single scale to each level. Formally, we define the anchors to have areas of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ pixels on $\{P_2, P_3, P_4, P_5, P_6\}$ respectively. As in [29] we also use anchors of multiple aspect ratios $\{1:2, 1:1, 2:1\}$ at each level. So in total there are 15 anchors over the pyramid.

我们通过用我们的FPN替换单尺度特征映射来适应RPN。我们在我们的特征金字塔的每个层级上附加一个相同设计的头部（3x3 conv和两个1x1兄弟convs）。由于头部在所有金字塔等级上的所有位置密集滑动，所以不需要在特定层级上具有多尺度锚点。相反，我们为每个层级分配单尺度的锚点。在形式上，我们定义锚点$\{P_2, P_3, P_4, P_5, P_6\}$分别具有$\{32^2, 64^2, 128^2, 256^2, 512^2\}$个像素的面积。正如在[29]中，我们在每个层级上也使用了多个长宽比$\{1:2, 1:1, 2:1\}$的锚点。所以在金字塔上总共有十五个锚点。

We assign training labels to the anchors based on their Intersection-over-Union (IoU) ratios with ground-truth bounding boxes as in [29]. Formally, an anchor is assigned a positive label if it has the highest IoU for a given ground-truth box or an IoU over 0.7 with any ground-truth box, and a negative label if it has IoU lower than 0.3 for all ground-truth boxes. Note that scales of ground-truth boxes are not explicitly used to assign them to the levels of the pyramid; instead, ground-truth boxes are associated with anchors, which have been assigned to pyramid levels. As such, we introduce no extra rules in addition to those in [29].

如[29]，我们根据锚点和实际边界框的交并比（IoU）比例将训练标签分配给锚点。形式上，如果一个锚点对于一个给定的实际边界框具有最高的IoU或者与任何实际边界框的IoU超过0.7，则给其分配一个正标签，如果其与所有实际边界框的IoU都低于0.3，则为其分配一个负标签。请注意，实际边界框的尺度并未明确用于将它们分配到金字塔的层级；相反，实际边界框与已经分配给金字塔等级的锚点相关联。因此，除了[29]中的内容外，我们不引入额外的规则。

We note that the parameters of the heads are shared across all feature pyramid levels; we have also evaluated the alternative without sharing parameters and observed similar accuracy. The good performance of sharing parameters indicates that all levels of our pyramid share similar semantic levels. This advantage is analogous to that of using a featurized image pyramid, where a common head classifier can be applied to features computed at any image scale.

我们注意到头部的参数在所有特征金字塔层级上共享；我们也评估了替代方案，没有共享参数并且观察到相似的准确性。共享参数的良好性能表明我们的金字塔的所有层级共享相似的语义级别。这个优点类似于使用特征图像金字塔的优点，其中可以将常见头部分类器应用于在任何图像尺度下计算的特征。

With the above adaptations, RPN can be naturally trained and tested with our FPN, in the same fashion as in [29]. We elaborate on the implementation details in the experiments.

通过上述改编，RPN可以自然地通过我们的FPN进行训练和测试，与[29]中的方式相同。我们在实验中详细说明实施细节。

4.2. Feature Pyramid Networks for Fast R-CNN

Fast R-CNN [11] is a region-based object detector in which Region-of-Interest (RoI) pooling is used to extract features. Fast R-CNN is most commonly performed on a single-scale feature map. To use it with our FPN, we need to assign RoIs of different scales to the pyramid levels.

4.2. Fast R-CNN的特征金字塔网络

Fast R-CNN[11]是一个基于区域的目标检测器，利用感兴趣区域（RoI）池化来提取特征。Fast R-CNN通常在单尺度特征映射上执行。要将其与我们的FPN一起使用，我们需要为金字塔等级分配不同尺度的RoI。

We view our feature pyramid as if it were produced from an image pyramid. Thus we can adapt the assignment strategy of region-based detectors [15, 11] in the case when they are run on image pyramids. Formally, we assign an RoI of width w and height h (on the input image to the network) to the level P_k of our feature pyramid by:

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor. \tag{1}$$

Here 224 is the canonical ImageNet pre-training size, and k_0 is the target level on which an RoI with $w \times h = 224^2$ should be mapped into. Analogous to the ResNet-based Faster R-CNN system [16] that uses C_4 as the single-scale feature map, we set k_0 to 4. Intuitively, Eqn.(1) means that if the RoI's scale becomes smaller (say, 1/2 of 224), it should be mapped into a finer-resolution level (say, $k = 3$).

我们将我们的特征金字塔看作是从图像金字塔生成的。因此，当它们在图像金字塔上运行时，我们可以适应基于区域的检测器的分配策略[15，11]。在形式上，我们通过以下公式将宽度为w和高度为h（在网络上的输入图像上）的RoI分配到特征金字塔的级别P_k上：

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor. \tag{1}$$

这里224是规范的ImageNet预训练大小，而k_0是大小为$w \times h = 224^2$的RoI应该映射到的目标级别。类似于基于ResNet的Faster R-CNN系统[16]使用C_4作为单尺度特征映射，我们将k_0设置为4。直觉上，方程（1）意味着如果RoI的尺寸变小了（比如224的1/2），它应该被映射到一个更精细的分辨率级别（比如$k = 3$）。

We attach predictor heads (in Fast R-CNN the heads are class-specific classifiers and bounding box regressors) to all RoIs of all levels. Again, the heads all share parameters, regardless of their levels. In [16], a ResNet's conv5 layers (a 9-layer deep subnetwork) are adopted as the head on top of the conv4 features, but our method has already harnessed conv5 to construct the feature pyramid. So unlike [16], we simply adopt RoI pooling to extract 7×7 features, and attach two hidden 1,024-d fully-connected (fc) layers (each followed by ReLU) before the final classification and bounding box regression layers. These layers are randomly initialized, as there are no pre-trained fc layers available in ResNets. Note that compared to the standard conv5 head, our 2-fc MLP head is lighter weight and faster.

我们在所有级别的所有RoI中附加预测器头部（在Fast R-CNN中，预测器头部是特定类别的分类器和边界框回归器）。再次，预测器头部都共享参数，不管他们在什么层级。在[16]中，ResNet的conv5层（9层深的子网络）被用作conv4特征之上的头部，但我们的方法已经利用了conv5来构建特征金字塔。因此，与[16]不同，我们只是采用RoI池化提取7×7特征，并在最终的分类层和边界框回归层之前附加两个隐藏单元为1024维的全连接（fc）层（每层后都接ReLU层）。这些层是随机初始化的，因为ResNets中没有预先训练好的fc层。请注意，与标准的conv5头部相比，我们的2-fc MLP头部更轻更快。

Based on these adaptations, we can train and test Fast R-CNN on top of the feature pyramid. Implementation details are given in the experimental section.

基于这些改编，我们可以在特征金字塔之上训练和测试Fast R-CNN。实现细节在实验部分给出。

5. Experiments on Object Detection

We perform experiments on the 80 category COCO detection dataset [21]. We train using the union of 80k train images and a 35k subset of val images (`trainval35k` [2]), and report ablations on a 5k subset of val images (`minival`). We also report final results on the standard test set (`test-std`) [21] which has no disclosed labels.

5. 目标检测实验

我们在80类的COCO检测数据集[21]上进行实验。我们训练使用80k张训练图像和35k大小的验证图像子集（`trainval35k`[2]）的联合，并报告了在5k大小的验证图像子集（`minival`）上的消融实验。我们还报告了在没有公开标签的标准测试集（`test-std`）[21]上的最终结果。

As is common practice [12], all network backbones are pre-trained on the ImageNet1k classification set [33] and then fine-tuned on the detection dataset. We use the pre-trained ResNet-50 and ResNet-101 models that are publicly available. Our code is a reimplementation of `py-faster-rcnn` using Caffe2.

正如通常的做法[12]，所有的网络骨干都是在ImageNet1k分类集[33]上预先训练好的，然后在检测数据集上进行微调。我们使用公开可用的预训练的ResNet-50和ResNet-101模型。我们的代码是使用Caffe2重新实现`py-faster-rcnn`。

5.1. Region Proposal with RPN

We evaluate the COCO-style Average Recall (AR) and AR on small, medium, and large objects (AR_s, AR_m, and AR_l) following the definitions in [21]. We report results for 100 and 1000 proposals per images (AR^{100} and AR^{1k}).

5.1. 区域提议与RPN

根据[21]中的定义，我们评估了COCO类型的平均召回率（AR）和在小型，中型和大型目标(AR$_s$, AR$_m$, and AR$_l$)上的AR。我们报告了每张图像使用100个提议和1000个提议的结果(AR^{100} and AR^{1k})。

Implementation details. All architectures in Table 1 are trained end-to-end. The input image is resized such that its shorter side has 800 pixels. We adopt synchronized SGD training on 8 GPUs. A mini-batch involves 2 images per GPU and 256 anchors per image. We use a weight decay of 0.0001 and a momentum of 0.9. The learning rate is 0.02 for the first 30k mini-batches and 0.002 for the next 10k. For all RPN experiments (including baselines), we include the anchor boxes that are outside the image for training, which is unlike [29] where these anchor boxes are ignored. Other implementation details are as in [29]. Training RPN with FPN on 8 GPUs takes about 8 hours on COCO.

RPN	feature	# anchors	lateral?	top-down?	AR^{100}	AR^{1k}	AR$_s^{1k}$	AR$_m^{1k}$	AR$_l^{1k}$
(a) baseline on conv4	C_4	47k			36.1	48.3	32.0	58.7	62.2
(b) baseline on conv5	C_5	12k			36.3	44.9	25.3	55.5	64.2
(c) **FPN**	$\{P_k\}$	200k	✓	✓	**44.0**	**56.3**	**44.9**	**63.4**	66.2
Ablation experiments follow:									
(d) bottom-up pyramid	$\{P_k\}$	200k	✓		37.4	49.5	30.5	59.9	**68.0**
(e) top-down pyramid, w/o lateral	$\{P_k\}$	200k		✓	34.5	46.1	26.5	57.4	64.7
(f) only finest level	P_2	750k	✓	✓	38.4	51.3	35.1	59.7	67.6

Table 1. Bounding box proposal results using RPN [29], evaluated on the COCO `minival` set. All models are trained on `trainval35k`. The columns "lateral" and "top-down" denote the presence of lateral and top-down connections, respectively. The column "feature" denotes the feature maps on which the heads are attached. All results are based on ResNet-50 and share the same hyper-parameters.

实施细节。表1中的所有架构都是端对端训练。输入图像的大小调整为其较短边有800像素。我们采用8个GPU进行同步SGD训练。小批量数据包括每个GPU上2张图像和每张图像上256个锚点。我们使用0.0001的权重衰减和0.9的动量。前30k次小批量数据的学习率为0.02，而下一个10k次的学习率为0.002。对于所有的RPN实验（包括基准数据集），我们都包含了图像外部的锚盒来进行训练，这不同于[29]中的忽略这些锚盒。其它实现细节如[29]中所述。使用具有FPN的RPN在8个GPU上训练COCO数据集需要约8小时。

RPN	feature	# anchors	lateral?	top-down?	AR^{100}	AR^{1k}	AR$_s^{1k}$	AR$_m^{1k}$	AR$_l^{1k}$
(a) baseline on conv4	C_4	47k			36.1	48.3	32.0	58.7	62.2
(b) baseline on conv5	C_5	12k			36.3	44.9	25.3	55.5	64.2
(c) **FPN**	$\{P_k\}$	200k	✓	✓	**44.0**	**56.3**	**44.9**	**63.4**	66.2
Ablation experiments follow:									
(d) bottom-up pyramid	$\{P_k\}$	200k	✓		37.4	49.5	30.5	59.9	**68.0**
(e) top-down pyramid, w/o lateral	$\{P_k\}$	200k		✓	34.5	46.1	26.5	57.4	64.7
(f) only finest level	P_2	750k	✓	✓	38.4	51.3	35.1	59.7	67.6

表1。使用RPN[29]的边界框提议结果，在COCO的`minival`数据集上进行评估。所有模型都是通过`trainval35k`训练的。列"lateral"和"top-down"分别表示横向连接和自上而下连接的存在。列"feature"表示附着头部的特征映射。所有结果都是基于ResNet-50的并共享相同的超参数。

5.1.1 Ablation Experiments

Comparisons with baselines. For fair comparisons with original RPNs[29], we run two baselines (Table 1(a, b)) using the single-scale map of C_4 (the same as [16]) or C_5, both using the same hyper-parameters as ours, including using 5 scale anchors of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$. Table 1 (b) shows no advantage over (a), indicating that a single higher-level feature map is not enough because there is a trade-off between coarser resolutions and stronger semantics.

5.1.1 消融实验

与基线进行比较。为了与原始RPNs[29]进行公平比较，我们使用C_4(与[16]相同)或C_5的单尺度映射运行了两个基线（表1（a，b））），都使用与我们相同的超参数，包括使用5种尺度锚点$\{32^2, 64^2, 128^2, 256^2, 512^2\}$。表1（b）显示没有优于（a），这表明单个更高级别的特征映射是不够的，因为存在在较粗分辨率和较强语义之间的权衡。

Placing FPN in RPN improves AR^{1k} to 56.3 (Table 1 (c)), which is **8.0** points increase over the single-scale RPN baseline (Table 1 (a)). In addition, the performance on small objects (AR$_s^{1k}$) is boosted by a large margin of 12.9 points. Our pyramid representation greatly improves RPN's robustness to object scale variation.

将FPN放在RPN中可将AR^{1k}提高到56.3（表1（c））），这比单尺度RPN基线（表1（a））增加了**8.0**个点。此外，在小型目标（AR$_s^{1k}$）上的性能也大幅上涨了12.9个点。我们的金字塔表示大大提高了RPN对目标尺度变化的鲁棒性。

How important is top-down enrichment? Table 1(d) shows the results of our feature pyramid without the top-down pathway. With this modification, the 1×1 lateral connections followed by 3×3 convolutions are attached to the bottom-up pyramid. This architecture simulates the effect of reusing the pyramidal feature hierarchy (Fig. 1(b)).

自上而下的改进有多重要？表1（d）显示了没有自上而下路径的特征金字塔的结果。通过这种修改，将1×1横向连接和后面的3×3卷积添加到自下而上的金字塔中。该架构模拟了重用金字塔特征层次结构的效果（图1（b））。

The results in Table 1(d) are just on par with the RPN baseline and lag far behind ours. We conjecture that this is because there are large semantic gaps between different levels on the bottom-up pyramid (Fig. 1(b)), especially for very deep ResNets. We have also evaluated a variant of Table 1(d) without sharing the parameters of the heads, but observed similarly degraded performance. This issue cannot be simply remedied by level-specific heads.

表1（d）中的结果与RPN基线相当，并且远远落后于我们的结果。我们推测这是因为自下而上的金字塔（图1（b））的不同层次之间存在较大的语义差距，尤其是对于非常深的ResNets。 我们还评估了表1（d）的一个变体，但没有分享磁头的参数，但观察到类似的性能下降。这个问题不能简单地由特定级别的负责人来解决。

How important are lateral connections? Table 1(e) shows the ablation results of a top-down feature pyramid without the 1×1 lateral connections. This top-down pyramid has strong semantic features and fine resolutions. But we argue that the locations of these features are not precise, because these maps have been downsampled and upsampled several times. More precise locations of features can be directly passed from the finer levels of the bottom-up maps via the lateral connections to the top-down maps. As a results, FPN has an AR^1k score 10 points higher than Table 1(e).

横向连接有多重要？表1（e）显示了没有1×1横向连接的自顶向下特征金字塔的消融结果。这个自顶向下的金字塔具有强大的语义特征和良好的分辨率。但是我们认为这些特征的位置并不精确，因为这些映射已经进行了多次下采样和上采样。更精确的特征位置可以通过横向连接直接从自下而上映射的更精细层级传递到自上而下的映射。因此，FPN的AR^1k的得分比表1（e）高10个点。

How important are pyramid representations? Instead of resorting to pyramid representations, one can attach the head to the highest-resolution, strongly semantic feature maps of P_2 (i.e., the finest level in our pyramids). Similar to the single-scale baselines, we assign all anchors to the P_2 feature map. This variant (Table 1(f)) is better than the baseline but inferior to our approach. RPN is a sliding window detector with a fixed window size, so scanning over pyramid levels can increase its robustness to scale variance.

金字塔表示有多重要？可以将头部附加到P_2的最高分辨率的强语义特征映射上（即我们金字塔中的最好层级），而不采用金字塔表示。与单尺度基线类似，我们将所有锚点分配给P_2特征映射。这个变体（表1（f））比基线要好，但不如我们的方法。RPN是一个具有固定窗口大小的滑动窗口检测器，因此在金字塔层级上扫描可以增加其对尺度变化的鲁棒性。

In addition, we note that using P_2 alone leads to more anchors (750k, Table 1(f)) caused by its large spatial resolution. This result suggests that a larger number of anchors is not sufficient in itself to improve accuracy.

另外，我们注意到由于P_2较大的空间分辨率，单独使用P_2会导致更多的锚点（750k，表1（f））。这个结果表明，大量的锚点本身并不足以提高准确率。

5.2. Object Detection with Fast/Faster R-CNN

Next we investigate FPN for region-based (non-sliding window) detectors. We evaluate object detection by the COCO-style Average Precision (AP) and PASCAL-style AP (at a single IoU threshold of 0.5). We also report COCO AP on objects of small, medium, and large sizes (namely, AP_s, AP_m, and AP_l) following the definitions in [21].

5.2. 使用Fast/Faster R-CNN的目标检测

接下来我们研究基于区域（非滑动窗口）检测器的FPN。我们通过COCO类型的平均精度（AP）和PASCAL类型的AP（单个IoU阈值为0.5）来评估目标检测。我们还按照[21]中的定义报告了在小尺寸，中尺寸和大尺寸（即AP_s，AP_m和AP_l）目标上的COCO AP。

Implementation details. The input image is resized such that its shorter side has 800 pixels. Synchronized SGD is used to train the model on 8 GPUs. Each mini-batch involves 2 image per GPU and 512 RoIs per image. We use a weight decay of 0.0001 and a momentum of 0.9. The learning rate is 0.02 for the first 60k mini-batches and 0.002 for the next 20k. We use 2000 RoIs per image for training and 1000 for testing. Training Fast R-CNN with FPN takes about 10 hours on the COCO dataset.

实现细节。调整大小输入图像，使其较短边为800像素。同步SGD用于在8个GPU上训练模型。每个小批量数据包括每个GPU2张图像和每张图像上512个RoI。我们使用0.0001的权重衰减和0.9的动量。前60k次小批量数据的学习率为0.02，而接下来的20k次迭代学习率为0.002。我们每张图像使用2000个RoIs进行训练，1000个RoI进行测试。使用FPN在COCO数据集上训练Fast R-CNN需要约10小时。

5.2.1 Fast R-CNN (on fixed proposals)

To better investigate FPN's effects on the region-based detector alone, we conduct ablations of Fast R-CNN on *a fixed set of proposals*. We choose to freeze the proposals as computed by RPN on FPN (Table 1(c)), because it has good performance on small objects that are to be recognized by the detector. For simplicity we do not share features between Fast R-CNN and RPN, except when specified.

5.2.1 Fast R-CNN(固定提议)

为了更好地调查FPN对仅基于区域的检测器的影响，我们在*一组固定的提议*上进行Fast R-CNN的消融。我们选择冻结RPN在FPN上计算的提议（表1（c）），因为它在能被检测器识别的小目标上具有良好的性能。为了简单起见，我们不在Fast R-CNN和RPN之间共享特征，除非指定。

As a ResNet-based Fast R-CNN baseline, following [16], we adopt RoI pooling with an output size of 14×14 and attach all conv5 layers as the hidden layers of the head. This gives an AP of 31.9 in Table 2(a). Table 2(b) is a baseline exploiting an MLP head with 2 hidden fc layers, similar to the head in our architecture. It gets an AP of 28.8, indicating that the 2-fc head does not give us any orthogonal advantage over the baseline in Table 2(a).

Fast R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(a) baseline on conv4	RPN, $\{P_k\}$	C_4	conv5			54.7	31.9	15.7	36.5	45.5
(b) baseline on conv5	RPN, $\{P_k\}$	C_5	2*fc*			52.9	28.8	11.9	32.4	43.4
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	**45.8**
Ablation experiments follow:										
(d) bottom-up pyramid	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓		44.9	24.9	10.9	24.4	38.5
(e) top-down pyramid, w/o lateral	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*		✓	54.0	31.3	13.3	35.2	45.3
(f) only finest level	RPN, $\{P_k\}$	P_2	2*fc*	✓	✓	56.3	33.4	17.3	37.3	45.6

Table 2. Object detection results using **Fast R-CNN** [11] on a fixed set of proposals (RPN, $\{P_k\}$, Table 1(c)), evaluated on the COCO `minival` set. Models are trained on the `trainval35k` set. All results are based on ResNet-50 and share the same hyper-parameters.

作为基于ResNet的Fast R-CNN基线，遵循[16]，我们采用输出尺寸为14×14的RoI池化，并将所有conv5层作为头部的隐藏层。这得到了31.9的AP，如表2（a）。表2（b）是利用MLP头部的基线，其具有2个隐藏的fc层，类似于我们的架构中的头部。它得到了28.8的AP，表明2-fc头部没有给我们带来任何超过表2（a）中基线的正交优势。

Fast R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(a) baseline on conv4	RPN, $\{P_k\}$	C_4	conv5			54.7	31.9	15.7	36.5	45.5
(b) baseline on conv5	RPN, $\{P_k\}$	C_5	2*fc*			52.9	28.8	11.9	32.4	43.4
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	**45.8**
Ablation experiments follow:										
(d) bottom-up pyramid	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓		44.9	24.9	10.9	24.4	38.5
(e) top-down pyramid, w/o lateral	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*		✓	54.0	31.3	13.3	35.2	45.3
(f) only finest level	RPN, $\{P_k\}$	P_2	2*fc*	✓	✓	56.3	33.4	17.3	37.3	45.6

表2。使用**Fast R-CNN**[11]在一组固定提议（RPN，$\{P_k\}$，表1（c））上的目标检测结果，在COCO的`minival`数据集上进行评估。模型在`trainval35k`数据集上训练。所有结果都基于ResNet-50并共享相同的超参数。

Table 2(c) shows the results of our FPN in Fast R-CNN. Comparing with the baseline in Table 2(a), our method improves AP by 2.0 points and small object AP by 2.1 points. Comparing with the baseline that also adopts a 2fc head (Table 2(b)), our method improves AP by 5.1 points. These comparisons indicate that our feature pyramid is superior to single-scale features for a region-based object detector.

表2（c）显示了Fast R-CNN中我们的FPN结果。与表2（a）中的基线相比，我们的方法将AP提高了2.0个点，小型目标AP提高了2.1个点。与也采用2fc头部的基线相比（表2（b）），我们的方法将AP提高了5.1个点。这些比较表明，对于基于区域的目标检测器，我们的特征金字塔优于单尺度特征。

Table 2(d) and (e) show that removing top-down connections or removing lateral connections leads to inferior results, similar to what we have observed in the above sub-section for RPN. It is noteworthy that removing top-down connections (Table 2(d)) significantly degrades the accuracy, suggesting that Fast R-CNN suffers from using the low-level features at the high-resolution maps.

表2（d）和（e）表明，去除自上而下的连接或去除横向连接会导致较差的结果，类似于我们在上面的RPN小节中观察到的结果。值得注意的是，去除自上而下的连接（表2（d））显著降低了准确性，表明Fast R-CNN在高分辨率映射中使用了低级特征。

In Table 2(f), we adopt Fast R-CNN on the single finest scale feature map of P_2. Its result (33.4 AP) is marginally worse than that of using all pyramid levels (33.9 AP, Table 2(c)). We argue that this is because RoI pooling is a warping-like operation, which is less sensitive to the region's scales. Despite the good accuracy of this variant, it is based on the RPN proposals of $\{P_k\}$ and has thus already benefited from the pyramid representation.

在表2（f）中，我们在P_2的单个最好的尺度特征映射上采用了Fast R-CNN。其结果（33.4 AP）略低于使用所有金字塔等级（33.9 AP，表2（c））的结果。我们认为这是因为RoI池化是一种扭曲式的操作，对区域尺度较不敏感。尽管这个变体具有很好的准确性，但它是基于$\{P_k\}$的RPN提议的，因此已经从金字塔表示中受益。

5.2.2 Faster R-CNN (on consistent proposals)

In the above we used a fixed set of proposals to investigate the detectors. But in a Faster R-CNN system [29], the RPN and Fast R-CNN must use *the same network backbone* in order to make feature sharing possible. Table 3 shows the comparisons between our method and two baselines, all using consistent backbone architectures for RPN and Fast R-CNN. Table 3(a) shows our reproduction of the baseline Faster R-CNN system as described in [16]. Under controlled settings, our FPN (Table 3(c)) is better than this strong baseline by 2.3 points AP and 3.8 points AP@0.5.

Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(*) baseline from He *et al.* [16]†	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	**47.1**
(b) baseline on conv5	RPN, C_5	C_5	2*fc*			51.7	28.0	9.6	31.9	43.1
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	45.8

Table 3. Object detection results using Faster R-CNN [29] evaluated on the COCO `minival` set. *The backbone network for RPN are consistent with Fast R-CNN.* Models are trained on the `trainval35k` set and use ResNet-50. †Provided by authors of [16].

5.2.2 Faster R-CNN(一致提议)

在上面我们使用了一组固定的提议来研究检测器。但是在Faster R-CNN系统中[29]，RPN和Fast R-CNN必须使用*相同的骨干网络*来实现特征共享。表3显示了我们的方法和两个基线之间的比较，所有这些RPN和Fast R-CNN都使用一致的骨干架构。表3（a）显示了我们再现[16]中描述的Faster R-CNN系统的基线。在受控的环境下，我们的FPN（表3（c））比这个强劲的基线要好2.3个点的AP和3.8个点的AP@0.5。

Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(*) baseline from He *et al.* [16]†	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	**47.1**
(b) baseline on conv5	RPN, C_5	C_5	2*fc*			51.7	28.0	9.6	31.9	43.1
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	45.8

表3。使用Faster R-CNN[29]在COCO `minival` 数据集上评估的目标检测结果。*RPN与Fast R-CNN的骨干网络是一致的。*模型在`trainval35k`数据集上训练并使用ResNet-50。†由[16]的作者提供。

Note that Table 3(a) and (b) are baselines that are much stronger than the baseline provided by He et al. [16] in Table 3(). *We find the following implementations contribute to the gap: (i) We use an image scale of 800 pixels instead of 600 in [11, 16]; (ii) We train with 512 RoIs per image which accelerate convergence, in contrast to 64 RoIs in [11, 16]; (iii) We use 5 scale anchors instead of 4 in [16] (adding 32^2); (iv) At test time we use 1000 proposals per image instead of 300 in [16]. So comparing with He et al.'s ResNet-50 Faster R-CNN baseline in Table 3()*, our method improves AP by 7.6 points and AP@0.5 by 9.6 points.

请注意，表3（a）和（b）的基线比He等人[16]在表3（*）中提供的基线强大得多。我们发现以下实现有助于缩小差距：（i）我们使用800像素的图像尺度，而不是[11，16]中的600像素；（ii）与[11，16]中的64个ROI相比，我们训练时每张图像有512个ROIs，可以加速收敛；（iii）我们使用5个尺度的锚点，而不是[16]中的4个（添加32^2）；（iv）在测试时，我们每张图像使用1000个提议，而不是[16]中的300个。因此，与表3（）中的He等人的ResNet-50 Faster R-CNN基线相比，我们的方法将AP提高了7.6点个并且将AP@0.5提高了9.6个点。

Sharing features. In the above, for simplicity we do not share the features between RPN and Fast R-CNN. In Table 5, we evaluate sharing features following the 4-step training described in [29]. Similar to [29], we find that sharing features improves accuracy by a small margin. Feature sharing also reduces the testing time.

	ResNet-50		ResNet-101	
share features?	$AP_{@0.5}$	AP	$AP_{@0.5}$	AP
no	56.9	33.9	58.0	35.0
yes	**57.2**	**34.3**	**58.2**	**35.2**

Table 5. More object detection results using Faster R-CNN and our FPNs, evaluated on `minival`. Sharing features increases train time by 1.5× (using 4-step training [29]), but reduces test time.

共享特征。在上面，为了简单起见，我们不共享RPN和Fast R-CNN之间的特征。在表5中，我们按照[29]中描述的4步训练评估了共享特征。与[29]类似，我们发现共享特征提高了一点准确率。特征共享也缩短了测试时间。

	ResNet-50		ResNet-101	
share features?	$AP_{@0.5}$	AP	$AP_{@0.5}$	AP
no	56.9	33.9	58.0	35.0
yes	**57.2**	**34.3**	**58.2**	**35.2**

表5。使用Faster R-CNN和我们的FPN在`minival`上的更多目标检测结果。共享特征将训练时间增加了1.5倍（使用4步训练[29]），但缩短了测试时间。

Running time. With feature sharing, our FPN-based Faster R-CNN system has inference time of 0.148 seconds per image on a single NVIDIA M40 GPU for ResNet-50, and 0.172 seconds for ResNet-101. As a comparison, the single-scale ResNet-50 baseline in Table 3(a) runs at 0.32 seconds. Our method introduces small extra cost by the extra layers in the FPN, but has a lighter weight head. Overall our system is faster than the ResNet-based Faster R-CNN counterpart. We believe the efficiency and simplicity of our method will benefit future research and applications.

运行时间。通过特征共享，我们的基于FPN的Faster R-CNN系统使用ResNet-50在单个NVIDIA M40 GPU上每张图像的推断时间为0.148秒，使用ResNet-101的时间为0.172秒。作为比较，表3（a）中的单尺度ResNet-50基线运行时间为0.32秒。我们的方法通过FPN中的额外层引入了较小的额外成本，但具有更轻的头部。总体而言，我们的系统比对应的基于ResNet的Faster R-CNN更快。我们相信我们方法的高效性和简洁性将有利于未来的研究和应用。

5.2.3 Comparing with COCO Competition Winners

We find that our ResNet-101 model in Table 5 is not sufficiently trained with the default learning rate schedule. So we increase the number of mini-batches by 2× at each learning rate when training the Fast R-CNN step. This increases AP on `minival` to 35.6, without sharing features. This model is the one we submitted to the COCO detection leaderboard, shown in Table 4. We have not evaluated its feature-sharing version due to limited time, which should be slightly better as implied by Table 5.

				test-dev					test-std				
method	backbone	competition	image pyramid	$AP_{@.5}$	AP	AP_s	AP_m	AP_l	$AP_{@.5}$	AP	AP_s	AP_m	AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on `minival`)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

Table 4. Comparisons of **single-model** results on the COCO detection benchmark. Some results were not available on the `test-std` set, so we also include the `test-dev` results (and for Multipath [40] on `minival`). †: http://image-net.org/challenges/ talks/2016/GRMI-COCO-slidedeck.pdf. ‡: http://mscoco.org/dataset/#detections-leaderboard. §: This entry of AttractioNet [10] adopts VGG-16 for proposals and Wide ResNet [39] for object detection, so is not strictly a single-model result.

5.2.3 与COCO竞赛获胜者的比较

我们发现表5中我们的ResNet-101模型在默认学习速率的情况下没有进行足够的训练。因此，在训练Fast R-CNN步骤时，我们将每个学习速率的小批量数据的数量增加了2倍。这将`minival`上的AP增加到了35.6，没有共享特征。该模型是我们提交给COCO检测排行榜的模型，如表4所示。由于时间有限，我们尚未评估其特征共享版本，这应该稍微好一些，如表5所示。

				test-dev					test-std				
method	backbone	competition	image pyramid	$AP_{@.5}$	AP	AP_s	AP_m	AP_l	$AP_{@.5}$	AP	AP_s	AP_m	AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on `minival`)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

表4。在COCO检测基线上**单模型**结果的比较。一些在`test-std`数据集上的结果是不可获得的，因此我们也包括了在`test-dev`上的结果（和Multipath[40]在`minival`上的结果）。†：http://image-net.org/challenges/ talks/2016/GRMI-COCO-slidedeck.pdf。‡：http://mscoco.org/dataset/#detections-leaderboard。§：AttractioNet[10]的输入采用VGG-16进行目标提议，用Wide ResNet[39]进行目标检测，因此它不是严格意义上的单模型。

Table 4 compares our method with the single-model results of the COCO competition winners, including the 2016 winner G-RMI and the 2015 winner Faster R-CNN+++. Without adding bells and whistles, our single-model entry has surpassed these strong, heavily engineered competitors. On the `test-dev` set, our method increases over

the existing best results by 0.5 points of AP (36.2 vs. 35.7) and 3.4 points of AP@0.5 (59.1 vs. 55.7). It is worth noting that our method does not rely on image pyramids and only uses a single input image scale, but still has outstanding AP on small-scale objects. This could only be achieved by high-resolution image inputs with previous methods.

表4将我们方法的单模型结果与COCO竞赛获胜者的结果进行了比较，其中包括2016年冠军G-RMI和2015年冠军Faster R-CNN+++。没有添加额外的东西，我们的单模型提交就已经超越了这些强大的，经过严格设计的竞争对手。在`test-dev`数据集中，我们的方法在现有最佳结果上增加了0.5个点的AP（36.2 vs.35.7）和3.4个点的AP@0.5（59.1 vs. 55.7）。值得注意的是，我们的方法不依赖图像金字塔，只使用单个输入图像尺度，但在小型目标上仍然具有出色的AP。这只能通过使用前面方法的高分辨率图像输入来实现。

Moreover, our method does not exploit many popular improvements, such as iterative regression [9], hard negative mining [35], context modeling [16], stronger data augmentation [22], etc. These improvements are complementary to FPNs and should boost accuracy further.

此外，我们的方法没有利用许多流行的改进，如迭代回归[9]，难例挖掘[35]，上下文建模[16]，更强大的数据增强[22]等。这些改进与FPN互补，应该会进一步提高准确度。

Recently, FPN has enabled new top results in all tracks of the COCO competition, including detection, instance segmentation, and keypoint estimation. See [14] for details.

最近，FPN在COCO竞赛的所有方面都取得了新的最佳结果，包括检测，实例分割和关键点估计。详情请参阅[14]。

6. Extensions: Segmentation Proposals

Our method is a generic pyramid representation and can be used in applications other than object detection. In this section we use FPNs to generate segmentation proposals, following the DeepMask/SharpMask framework [27, 28].

6. 扩展：分割提议

我们的方法是一种通用金字塔表示，可用于除目标检测之外的其他应用。在本节中，我们使用FPN生成分割建议，遵循DeepMask/SharpMask框架[27，28]。

DeepMask/SharpMask were trained on image crops for predicting instance segments and object/non-object scores. At inference time, these models are run convolutionally to generate dense proposals in an image. To generate segments at multiple scales, image pyramids are necessary [27, 28].

DeepMask/SharpMask在裁剪图像上进行训练，可以预测实例块和目标/非目标分数。在推断时，这些模型是卷积运行的，以在图像中生成密集的提议。为了在多个尺度上生成分割块，图像金字塔是必要的[27，28]。

It is easy to adapt FPN to generate mask proposals. We use a fully convolutional setup for both training and inference. We construct our feature pyramid as in Sec. 5.1 and set $d = 128$. On top of each level of the feature pyramid, we apply a small 5×5 MLP to predict 14×14 masks and object scores in a fully convolutional fashion, see Fig. 4. Additionally, motivated by the use of 2 scales per octave in the image pyramid of [27, 28], we use a second MLP of input size 7×7 to handle half octaves. The two MLPs play a similar role as anchors in RPN. The architecture is trained end-to-end; full implementation details are given in the appendix.

method	backbone	competition	image pyramid	test-dev $AP_{@.5}$	test-dev AP	test-dev AP_s	test-dev AP_m	test-dev AP_l	test-std $AP_{@.5}$	test-std AP	test-std AP_s	test-std AP_m	test-std AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on minival)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

Figure 4. FPN for object segment proposals. The feature pyramid is constructed with identical structure as for object detection. We apply a small MLP on 5x5 windows to generate dense object segments with output dimension of 14x14. Shown in orange are the size of the image regions the mask corresponds to for each pyramid level (levels $P_{3-5} are shown here). Both the corresponding image region size (light orange) and canonical object size (dark orange) are shown. Half octaves are handled by$ 7 \approx 5 \sqrt 2$), not shown here. Details are in the appendix.

改编FPN生成掩码提议很容易。我们对训练和推断都使用全卷积设置。我们在5.1小节中构造我们的特征金字塔并设置$d = 128$。在特征金字塔的每个层级上，我们应用一个小的5×5MLP以全卷积方式预测14×14掩码和目标分数，参见图4。此外，由于在[27,28]的图像金字塔中每组使用2个尺度，我们使用输入大小为7×7的第二个MLP来处理半个组。这两个MLP在RPN中扮演着类似于锚点的角色。该架构是端到端训练的，完整的实现细节在附录中给出。

method	backbone	competition	image pyramid	test-dev AP@.5	test-dev AP	test-dev AP_s	test-dev AP_m	test-dev AP_l	test-std AP@.5	test-std AP	test-std AP_s	test-std AP_m	test-std AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on minival)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

图4。目标分割提议的FPN。特征金字塔的构造结构与目标检测相同。我们在5x5窗口上应用一个小的MLP来生成输出尺寸为14x14的密集目标块。以橙色显示的掩码是每个金字塔层级所对应的图像区域的大小（此处显示的是层级P_{3-5}）。显示了相应的图像区域大小（浅橙色）和典型目标大小（深橙色）。半个组由MLP在$7x7$窗口（7 \ approx 5 \ sqrt 2 $）处理，此处未展示。详情见附录。

6.1. Segmentation Proposal Results

Results are shown in Table 6. We report segment AR and segment AR on small, medium, and large objects, always for 1000 proposals. Our baseline FPN model with a single 5×5 MLP achieves an AR of 43.4. Switching to a slightly larger 7×7 MLP leaves accuracy largely unchanged. Using both MLPs together increases accuracy to 45.7 AR. Increasing mask output size from 14×14 to 28×28 increases AR another point (larger sizes begin to degrade accuracy). Finally, doubling the training iterations increases AR to 48.1.

method	backbone	competition	image pyramid	test-dev AP@.5	test-dev AP	test-dev AP_s	test-dev AP_m	test-dev AP_l	test-std AP@.5	test-std AP	test-std AP_s	test-std AP_m	test-std AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on minival)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

Table 6. Instance segmentation proposals evaluated on the first 5k COCO `val` images. All models are trained on the `train` set. DeepMask, SharpMask, and FPN use ResNet-50 while Instance-FCN uses VGG-16. DeepMask and SharpMask performance is computed with models available from https://github.com/facebookresearch/deepmask (both are the 'zoom' variants). †Runtimes are measured on an NVIDIA M40 GPU, except the InstanceFCN timing which is based on the slower K40.

6.1. 分割提议结果

结果如表6所示。我们报告了分割AR和在小型，中型和大型目标上的分割AR，都是对于1000个提议而言的。我们的具有单个5×5MLP的基线FPN模型达到了43.4的AR。切换到稍大的7×7MLP，精度基本保持不变。同时使用两个MLP将精度提高到了45.7的AR。将掩码输出尺寸从14×14增加到28×28会增加AR另一个点（更大的尺寸开始降低准确度）。最后，加倍训练迭代将AR增加到48.1。

method	backbone	competition	image pyramid	test-dev AP@.5	test-dev AP	test-dev AP_s	test-dev AP_m	test-dev AP_l	test-std AP@.5	test-std AP	test-std AP_s	test-std AP_m	test-std AP_l
ours, Faster R-CNN on **FPN**	ResNet-101	-		**59.1**	**36.2**	**18.2**	**39.0**	48.2	**58.5**	**35.8**	**17.5**	**38.7**	47.8
*Competition-winning **single-model** results follow:*													
G-RMI†	Inception-ResNet	2016		-	34.7	-	-	-	-	-	-	-	-
AttractioNet‡ [10]	VGG16 + Wide ResNet§	2016	✓	53.4	35.7	15.6	38.0	**52.7**	52.9	35.3	14.7	37.6	**51.9**
Faster R-CNN +++ [16]	ResNet-101	2015	✓	55.7	34.9	15.6	38.7	50.9	-	-	-	-	-
Multipath [40] (on minival)	VGG-16	2015		49.6	31.5	-	-	-	-	-	-	-	-
ION‡ [2]	VGG-16	2015		53.4	31.2	12.8	32.9	45.2	52.9	30.7	11.8	32.8	44.8

表6。在前5k张COCO `val` 图像上评估的实例分割提议。所有模型都是在 `train` 数据集上训练的。DeepMask，SharpMask和FPN使用ResNet-50，而Instance-FCN使用VGG-16。DeepMask和SharpMask性能计算的模型是从https://github. com/facebookresearch/deepmask上获得的（都是'zoom'变体）。†运行时间是在NVIDIA M40 GPU上测量的，除了基于较慢的K40的InstanceFCN。

We also report comparisons to DeepMask [27], Sharp-Mask [28], and InstanceFCN [4], the previous state of the art methods in mask proposal generation. We outperform the accuracy of these approaches by over 8.3 points AR. In particular, we nearly double the accuracy on small objects.

我们还报告了与DeepMask[27]，Sharp-Mask[28]和InstanceFCN[4]的比较，这是以前的掩模提议生成中的先进方法。我们的准确度超过这些方法8.3个点的AR。尤其是我们几乎将小目标的精度提高了一倍。

Existing mask proposal methods [27, 28, 4] are based on densely sampled image pyramids (e.g., scaled by $2^{\{-2:0.5:1\}}$ in [27, 28]), making them computationally expensive. Our approach, based on FPNs, is substantially faster (our models run at 6 to 7 FPS). These results demonstrate that our model is a generic feature extractor and can replace image pyramids for other multi-scale detection problems.

现有的掩码提议方法[27，28，4]是基于密集采样的图像金字塔的（例如，[27，28]中的缩放为$2^{\{-2:0.5:1\}}$），使得它们是计算昂贵的。我们的方法基于FPN，速度明显加快（我们的模型运行速度为6至7FPS）。这些结果表明，我们的模型是一个通用的特征提取器，可以替代图像金字塔以用于其他多尺度检测问题。

7. Conclusion

We have presented a clean and simple framework for building feature pyramids inside ConvNets. Our method shows significant improvements over several strong baselines and competition winners. Thus, it provides a practical solution for research and applications of feature pyramids, without the need of computing image pyramids. Finally, our study suggests that despite the strong representational power of deep ConvNets and their implicit robustness to scale variation, it is still critical to explicitly address multi-scale problems using pyramid representations.

7. 结论

我们提出了一个干净而简单的框架，用于在ConvNets内部构建特征金字塔。我们的方法比几个强大的基线和竞赛获胜者显示出了显著的改进。因此，它为特征金字塔的研究和应用提供了一个实用的解决方案，而不需要计算图像金字塔。最后，我们的研究表明，尽管深层ConvNets具有强大的表示能力以及它们对尺度变化的隐式鲁棒性，但使用金字塔表示对于明确地解决多尺度问题仍然至关重要。

References

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing. RCA engineer, 1984.

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In CVPR, 2016.

[3] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In ECCV, 2016.

[4] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In ECCV, 2016.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In CVPR, 2005.

[6] P. Dollar, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. TPAMI, 2014.

[7] P.F.Felzenszwalb,R.B.Girshick,D.McAllester,andD.Ramanan. Object detection with discriminatively trained part-based models. TPAMI, 2010.

[8] G.GhiasiandC.C.Fowlkes.Laplacianpyramidreconstruction and refinement for semantic segmentation. In ECCV, 2016.

[9] S. Gidaris and N. Komodakis. Object detection via a multi-region & semantic segmentation-aware CNN model. In ICCV, 2015.

[10] S. Gidaris and N. Komodakis. Attend refine repeat: Active box proposal generation via in-out localization. In BMVC, 2016.

[11] R. Girshick. Fast R-CNN. In ICCV, 2015.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[13] B.Hariharan,P.Arbelaez,R.Girshick,andJ.Malik.Hypercolumns for object segmentation and fine-grained localization. In CVPR, 2015.

[14] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. arXiv:1703.06870, 2017.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV. 2014.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[17] S. Honari, J. Yosinski, P. Vincent, and C. Pal. Recombinator networks: Learning coarse-to-fine feature aggregation. In CVPR, 2016.

[18] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In CVPR, 2016.

[19] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012.

[20] 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.

[21] 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 ECCV, 2014.

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. SSD: Single shot multibox detector. In ECCV, 2016.

[23] W. Liu, A. Rabinovich, and A. C. Berg. ParseNet: Looking wider to see better. In ICLR workshop, 2016.

[24] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[25] D. G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 2004.

[26] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In ECCV, 2016.

[27] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to segment object candidates. In NIPS, 2015.

[28] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In ECCV, 2016.

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015.

[30] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun. Object detection networks on convolutional feature maps. PAMI, 2016.

[31] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In MIC- CAI, 2015.

[32] H. Rowley, S. Baluja, and T. Kanade. Human face detection in visual scenes. Technical Report CMU-CS-95-158R, Carnegie Mellon University, 1995.

[33] 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. IJCV, 2015.

[34] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In ICLR, 2014.

[35] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In CVPR, 2016.

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[37] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. IJCV, 2013.

[38] R. Vaillant, C. Monrocq, and Y. LeCun. Original approach for the localisation of objects in images. IEE Proc. on Vision, Image, and Signal Processing, 1994.

[39] S. Zagoruyko and N. Komodakis. Wide residual networks. In BMVC, 2016.

[40] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár. A multipath network for object detection. In BMVC, 2016. 10

Deep Learning

‹ Feature Pyramid Networks for Object Detection论文翻译——中文版

正则表达式 ›