Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick

Facebook AI Research (FAIR)

Abstract

We present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in parallel *with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks,* e.g.*, allowing us to estimate human poses in the same framework. We show top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection. Without bells and whistles, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners. We hope our simple and effective approach will serve as a solid baseline and help ease future research in instance-level recognition. Code has been made available at:* https://github.com/facebookresearch/Detectron.

1. Introduction

The vision community has rapidly improved object detection and semantic segmentation results over a short period of time. In large part, these advances have been driven by powerful baseline systems, such as the Fast/Faster R-CNN [12, 36] and Fully Convolutional Network (FCN) [30] frameworks for object detection and semantic segmentation, respectively. These methods are conceptually intuitive and offer flexibility and robustness, together with fast training and inference time. Our goal in this work is to develop a comparably enabling framework for *instance segmentation*.

Instance segmentation is challenging because it requires the correct detection of all objects in an image while also precisely segmenting each instance. It therefore combines elements from the classical computer vision tasks of *object detection*, where the goal is to classify individual objects and localize each using a bounding box, and *semantic segmentation*, where the goal is to classify each pixel into a fixed set of categories without differentiating object instances.[1] Given this, one might expect a complex method is required to achieve good results. However, we show that a surprisingly simple, flexible, and fast system can surpass prior state-of-the-art instance segmentation results.

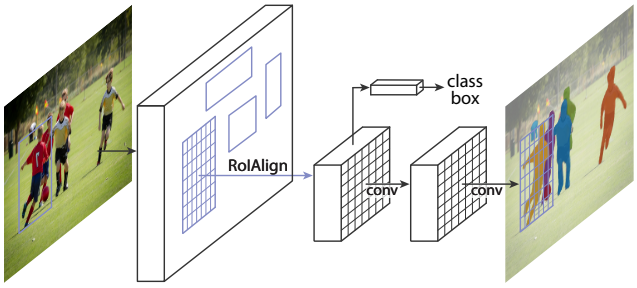

Figure 1. The **Mask R-CNN** framework for instance segmentation.

Our method, called *Mask R-CNN*, extends Faster R-CNN [36] by adding a branch for predicting segmentation masks on each Region of Interest (RoI), in *parallel* with the existing branch for classification and bounding box regression (Figure 1). The mask branch is a small FCN applied to each RoI, predicting a segmentation mask in a pixel-to-pixel manner. Mask R-CNN is simple to implement and train given the Faster R-CNN framework, which facilitates a wide range of flexible architecture designs. Additionally, the mask branch only adds a small computational overhead, enabling a fast system and rapid experimentation.

In principle Mask R-CNN is an intuitive extension of Faster R-CNN, yet constructing the mask branch properly is critical for good results. Most importantly, Faster R-CNN was not designed for pixel-to-pixel alignment between network inputs and outputs. This is most evident in how *RoIPool* [18, 12], the *de facto* core operation for attending to instances, performs coarse spatial quantization for feature extraction. To fix the misalignment, we propose a simple, quantization-free layer, called *RoIAlign*, that faithfully preserves exact spatial locations. Despite being

[1] Following common terminology, we use *object detection* to denote detection via *bounding boxes*, not masks, and *semantic segmentation* to denote per-pixel classification without differentiating instances. Yet we note that *instance segmentation* is both semantic and a form of detection.

Mask R-CNN

何恺明 乔治亚·吉奥克萨里 彼得·多拉尔 罗斯·吉斯克

Facebook AI 研究院（FAIR）

摘要

We present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in 并行 *with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks,* 例如 *., allowing us to estimate human poses in the same framework. We show top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection. Without bells and whistles, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners. We hope our simple and effective approach will serve as a solid baseline and help ease future research in instance-level recognition. Code has been made available at:* https://github.com/ facebook research/Detectron.

1. 引言

视觉社区在短时间内迅速提升了物体检测和语义分割的效果。这些进步很大程度上得益于强大的基线系统，例如分别用于物体检测和语义分割的Fast/Faster R-CNN [12, 36] 与全卷积网络（FCN）[30] 框架。这些方法在概念上直观，兼具灵活性与鲁棒性，同时保证了快速的训练与推理速度。本工作的目标是为{v*}构建一个具备类似赋能效果的框架。

实例分割具有挑战性，因为它需要正确检测图像中的所有物体，同时精确分割每个实例。因此，它结合了经典计算机视觉任务中的元素：*ob-ject detection*（目标是对单个物体进行分类并用边界框定位每个物体）与*semantic*。

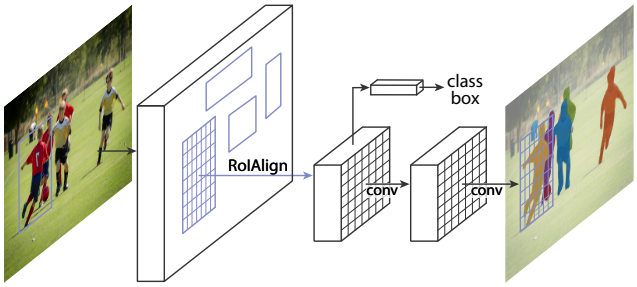

图1. 用于实例分割的Mask R-CNN框架。

*segmentation*其目标是将每个像素分类到一个固定的类别集合中，而不区分对象实例。[1] 鉴于此，人们可能会认为需要复杂的方法才能取得良好效果。然而，我们证明，一个出人意料地简单、灵活且快速的系统能够超越先前最先进的实例分割结果。

我们的方法名为*Mask R-CNN*，它通过在每个感兴趣区域（RoI）上增加一个预测分割掩码的分支来扩展Faster R-CNN[36]，该分支与现有的分类和边界框回归分支并行运行（图1）。掩码分支是一个应用于每个RoI的小型全卷积网络（FCN），以像素到像素的方式预测分割掩码。基于Faster R-CNN框架，Mask R-CNN易于实现和训练，这为广泛的灵活架构设计提供了便利。此外，掩码分支仅增加了少量计算开销，使得系统能够快速运行并支持高效实验。

原则上，Mask R-CNN是Faster R-CNN的直观扩展，但正确构建掩码分支对获得良好结果至关重要。最重要的是，Faster R-CNN并非为网络输入与输出之间的像素级对齐而设计。这一点在*RoIPool*[18, 12]（处理实例的核心操作*de facto*）执行特征提取时的粗粒度空间量化中体现得尤为明显。为解决这种错位问题，我们提出了一种简单、无量化的层——*RoIAlign*，它能精确保持空间位置信息。尽管

[1]Following common terminology, we use *object detection* to denote detection via *bounding boxes*, not masks, and *semantic segmentation* to denote per-pixel classification without differentiating instances. Yet we note that *instance segmentation* is both semantic and a form of detection.

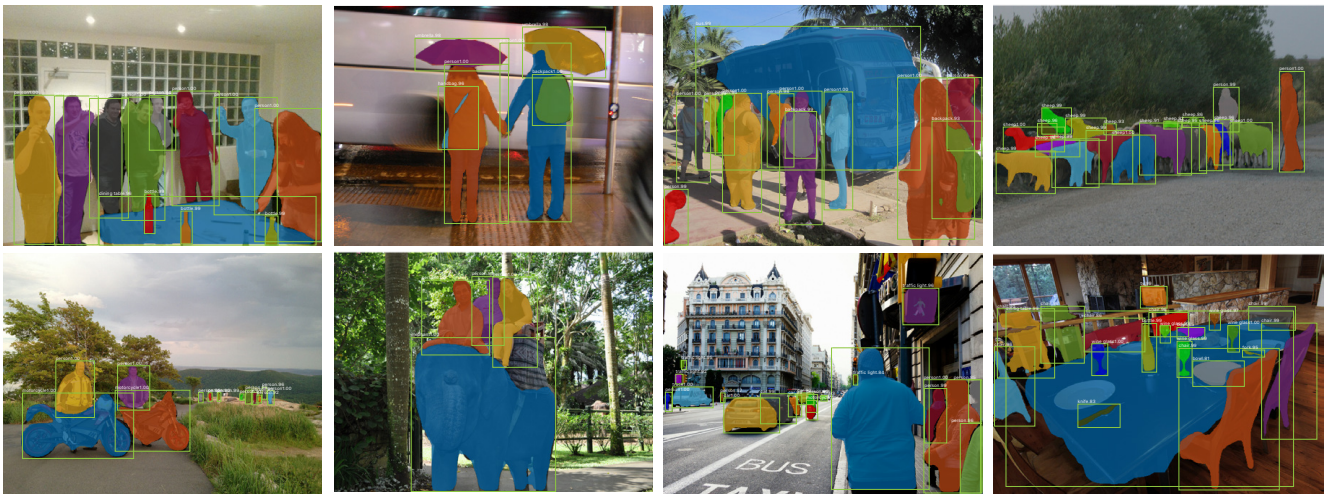

Figure 2. **Mask R-CNN** results on the COCO test set. These results are based on ResNet-101 [19], achieving a *mask* AP of 35.7 and running at 5 fps. Masks are shown in color, and bounding box, category, and confidences are also shown.

a seemingly minor change, RoIAlign has a large impact: it improves mask accuracy by relative 10% to 50%, showing bigger gains under stricter localization metrics. Second, we found it essential to *decouple* mask and class prediction: we predict a binary mask for each class independently, without competition among classes, and rely on the network's RoI classification branch to predict the category. In contrast, FCNs usually perform per-pixel multi-class categorization, which couples segmentation and classification, and based on our experiments works poorly for instance segmentation.

Without bells and whistles, Mask R-CNN surpasses all previous state-of-the-art single-model results on the COCO instance segmentation task [28], including the heavily-engineered entries from the 2016 competition winner. As a by-product, our method also excels on the COCO object detection task. In ablation experiments, we evaluate multiple basic instantiations, which allows us to demonstrate its robustness and analyze the effects of core factors.

Our models can run at about 200ms per frame on a GPU, and training on COCO takes one to two days on a single 8-GPU machine. We believe the fast train and test speeds, together with the framework's flexibility and accuracy, will benefit and ease future research on instance segmentation.

Finally, we showcase the generality of our framework via the task of human pose estimation on the COCO keypoint dataset [28]. By viewing each keypoint as a one-hot binary mask, with minimal modification Mask R-CNN can be applied to detect instance-specific poses. Mask R-CNN surpasses the winner of the 2016 COCO keypoint competition, and at the same time runs at 5 fps. Mask R-CNN, therefore, can be seen more broadly as a flexible framework for *instance-level recognition* and can be readily extended to more complex tasks.

We have released code to facilitate future research.

2. Related Work

R-CNN: The Region-based CNN (R-CNN) approach [13] to bounding-box object detection is to attend to a manageable number of candidate object regions [42, 20] and evaluate convolutional networks [25, 24] independently on each RoI. R-CNN was extended [18, 12] to allow attending to RoIs on feature maps using RoIPool, leading to fast speed and better accuracy. Faster R-CNN [36] advanced this stream by learning the attention mechanism with a Region Proposal Network (RPN). Faster R-CNN is flexible and robust to many follow-up improvements (*e.g.*, [38, 27, 21]), and is the current leading framework in several benchmarks.

Instance Segmentation: Driven by the effectiveness of R-CNN, many approaches to instance segmentation are based on *segment proposals*. Earlier methods [13, 15, 16, 9] resorted to bottom-up segments [42, 2]. DeepMask [33] and following works [34, 8] learn to propose segment candidates, which are then classified by Fast R-CNN. In these methods, segmentation *precedes* recognition, which is slow and less accurate. Likewise, Dai *et al*. [10] proposed a complex multiple-stage cascade that predicts segment proposals from bounding-box proposals, followed by classification. Instead, our method is based on *parallel* prediction of masks and class labels, which is simpler and more flexible.

Most recently, Li *et al*. [26] combined the segment proposal system in [8] and object detection system in [11] for "fully convolutional instance segmentation" (FCIS). The common idea in [8, 11, 26] is to predict a set of position-sensitive output channels fully convolutionally. These channels simultaneously address object classes, boxes, and masks, making the system fast. But FCIS exhibits systematic errors on overlapping instances and creates spurious edges (Figure 6), showing that it is challenged by the fundamental difficulties of segmenting instances.

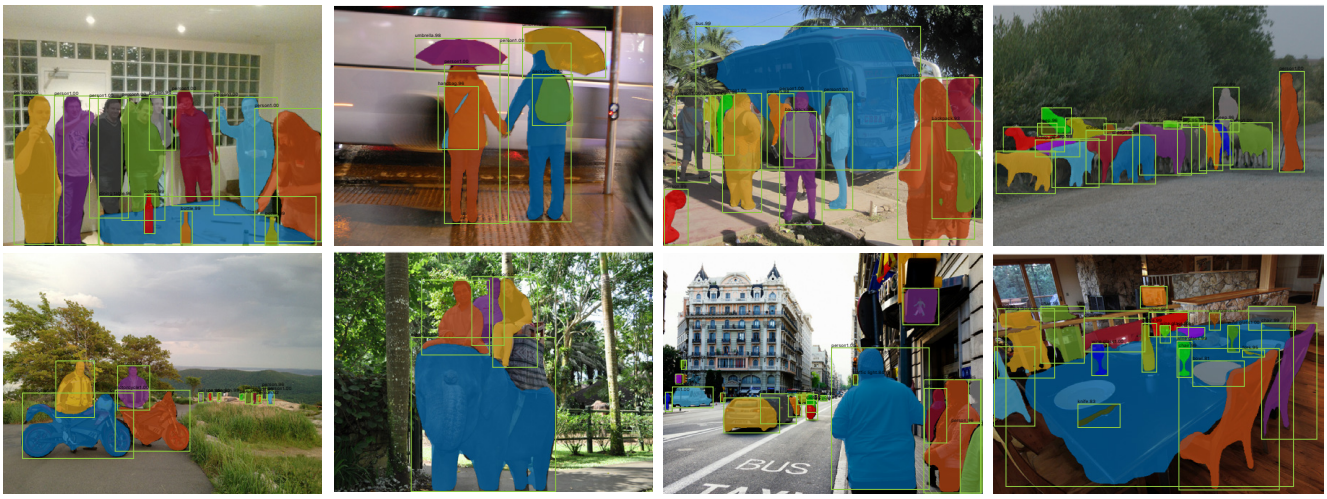

图2. Mask R-CNN在COCO测试集上的结果。这些结果基于ResNet-101[19]，实现了*mask* AP为35.7，并以5 fps的速度运行。掩码以彩色显示，同时显示了边界框、类别和置信度。

一个看似微小的改动，RoIAlign 却带来了巨大影响：它将掩码准确率相对提升了 10% 至 50%，在更严格的定位指标下显示出更大的增益。其次，我们发现将 {v*} 掩码与类别预测分离至关重要：我们为每个类别独立预测二值掩码，无需类别间竞争，并依靠网络的 RoI 分类分支来预测类别。相比之下，FCN 通常执行逐像素的多类别分类，这会将分割与分类耦合在一起，而根据我们的实验，这种方法在实例分割中效果不佳。

无需任何花哨的技巧，Mask R-CNN就在COCO实例分割任务[28]上超越了之前所有最先进的单模型结果，包括2016年竞赛冠军精心设计的方案。作为副产品，我们的方法在COCO目标检测任务上也表现出色。在消融实验中，我们评估了多种基础实现方案，这使我们能够证明其鲁棒性并分析核心因素的影响。

我们的模型在GPU上每帧运行时间约为200毫秒，在单台8-GPU机器上使用COCO数据集进行训练需要一至两天。我们相信，快速的训练和测试速度，加上框架的灵活性和准确性，将有益于并简化未来实例分割的研究。

最后，我们通过在COCO关键点数据集[28]上进行人体姿态估计任务，展示了我们框架的通用性。通过将每个关键点视为一个独热二进制掩码，只需最小程度的修改，Mask R-CNN即可应用于检测实例特定的姿态。Mask R-CNN超越了2016年COCO关键点竞赛的优胜者，同时以每秒5帧的速度运行。因此，Mask R-CNN可以被更广泛地视为一个灵活的*instance-level recognition*框架，并能轻松扩展到更复杂的任务。

我们已发布代码以促进未来的研究。

2. 相关工作

R-CNN：基于区域的CNN（R-CNN）方法[13]通过处理有限数量的候选目标区域[42, 20]，并在每个感兴趣区域（RoI）上独立评估卷积网络[25, 24]来实现边界框目标检测。后续研究[18, 12]对R-CNN进行扩展，通过RoIPool实现在特征图上处理RoI，从而提升速度与精度。Faster R-CNN[36]通过区域提议网络（RPN）学习注意力机制，推动了该方向的发展。Faster R-CNN具备灵活性，能稳健兼容多种后续改进（{v*}., [38, 27, 21]），目前在多项基准测试中保持领先框架地位。

实例分割：受R-CNN有效性的推动，许多实例分割方法基于*segment proposals*。早期方法[13, 15, 16, 9]依赖于自底向上的分割[42, 2]。DeepMask[33]及后续研究[34, 8]学习生成分割候选区域，再通过Fast R-CNN进行分类。这些方法中分割*precedes*识别速度较慢且精度较低。类似地，Dai*et al*[10]提出了一个复杂的多级级联结构，从边界框建议中预测分割建议，再进行分类。相比之下，我们的方法基于*parallel*的掩膜和类别标签预测，更简单且更灵活。

最近，Li *et al*等人[26]将[8]中的片段提议系统与[11]中的目标检测系统相结合，提出了“全卷积实例分割”（FCIS）。[8, 11, 26]中的共同思路是通过全卷积方式预测一组位置敏感的输出通道。这些通道同时处理目标类别、边界框和掩码，使系统运行迅速。但FCIS在重叠实例上表现出系统性错误，并产生虚假边缘（图6），这表明其在应对实例分割的根本性难题时仍面临挑战。

Another family of solutions [23, 4, 3, 29] to instance segmentation are driven by the success of semantic segmentation. Starting from per-pixel classification results (*e.g.*, FCN outputs), these methods attempt to cut the pixels of the same category into different instances. In contrast to the *segmentation-first* strategy of these methods, Mask R-CNN is based on an *instance-first* strategy. We expect a deeper incorporation of both strategies will be studied in the future.

3. Mask R-CNN

Mask R-CNN is conceptually simple: Faster R-CNN has two outputs for each candidate object, a class label and a bounding-box offset; to this we add a third branch that outputs the object mask. Mask R-CNN is thus a natural and intuitive idea. But the additional mask output is distinct from the class and box outputs, requiring extraction of much *finer* spatial layout of an object. Next, we introduce the key elements of Mask R-CNN, including pixel-to-pixel alignment, which is the main missing piece of Fast/Faster R-CNN.

Faster R-CNN: We begin by briefly reviewing the Faster R-CNN detector [36]. Faster R-CNN consists of two stages. The first stage, called a Region Proposal Network (RPN), proposes candidate object bounding boxes. The second stage, which is in essence Fast R-CNN [12], extracts features using RoIPool from each candidate box and performs classification and bounding-box regression. The features used by both stages can be shared for faster inference. We refer readers to [21] for latest, comprehensive comparisons between Faster R-CNN and other frameworks.

Mask R-CNN: Mask R-CNN adopts the same two-stage procedure, with an identical first stage (which is RPN). In the second stage, *in parallel* to predicting the class and box offset, Mask R-CNN also outputs a binary mask for each RoI. This is in contrast to most recent systems, where classification *depends* on mask predictions (*e.g.* [33, 10, 26]). Our approach follows the spirit of Fast R-CNN [12] that applies bounding-box classification and regression in *parallel* (which turned out to largely simplify the multi-stage pipeline of original R-CNN [13]).

Formally, during training, we define a multi-task loss on each sampled RoI as $L = L_{cls} + L_{box} + L_{mask}$. The classification loss L_{cls} and bounding-box loss L_{box} are identical as those defined in [12]. The mask branch has a Km^2-dimensional output for each RoI, which encodes K binary masks of resolution $m \times m$, one for each of the K classes. To this we apply a per-pixel sigmoid, and define L_{mask} as the average binary cross-entropy loss. For an RoI associated with ground-truth class k, L_{mask} is only defined on the k-th mask (other mask outputs do not contribute to the loss).

Our definition of L_{mask} allows the network to generate masks for every class without competition among classes; we rely on the dedicated classification branch to predict the class label used to select the output mask. This *decouples* mask and class prediction. This is different from common practice when applying FCNs [30] to semantic segmentation, which typically uses a per-pixel *softmax* and a *multinomial* cross-entropy loss. In that case, masks across classes compete; in our case, with a per-pixel *sigmoid* and a *binary* loss, they do not. We show by experiments that this formulation is key for good instance segmentation results.

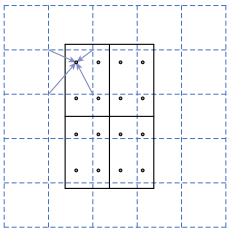

Figure 3. **RoIAlign:** The dashed grid represents a feature map, the solid lines an RoI (with 2×2 bins in this example), and the dots the 4 sampling points in each bin. RoIAlign computes the value of each sampling point by bilinear interpolation from the nearby grid points on the feature map. No quantization is performed on any coordinates involved in the RoI, its bins, or the sampling points.

Mask Representation: A mask encodes an input object's *spatial* layout. Thus, unlike class labels or box offsets that are inevitably collapsed into short output vectors by fully-connected (*fc*) layers, extracting the spatial structure of masks can be addressed naturally by the pixel-to-pixel correspondence provided by convolutions.

Specifically, we predict an $m \times m$ mask from each RoI using an FCN [30]. This allows each layer in the mask branch to maintain the explicit $m \times m$ object spatial layout without collapsing it into a vector representation that lacks spatial dimensions. Unlike previous methods that resort to *fc* layers for mask prediction [33, 34, 10], our fully convolutional representation requires fewer parameters, and is more accurate as demonstrated by experiments.

This pixel-to-pixel behavior requires our RoI features, which themselves are small feature maps, to be well aligned to faithfully preserve the explicit per-pixel spatial correspondence. This motivated us to develop the following *RoIAlign* layer that plays a key role in mask prediction.

RoIAlign: RoIPool [12] is a standard operation for extracting a small feature map (*e.g.*, 7×7) from each RoI. RoIPool first *quantizes* a floating-number RoI to the discrete granularity of the feature map, this quantized RoI is then subdivided into spatial bins which are themselves quantized, and finally feature values covered by each bin are aggregated (usually by max pooling). Quantization is performed, *e.g.*, on a continuous coordinate x by computing $[x/16]$, where 16 is a feature map stride and $[\cdot]$ is rounding; likewise, quantization is performed when dividing into bins (*e.g.*, 7×7). These quantizations introduce misalignments between the RoI and the extracted features. While this may not impact classification, which is robust to small translations, it has a large negative effect on predicting pixel-accurate masks.

To address this, we propose an *RoIAlign* layer that removes the harsh quantization of RoIPool, properly *aligning* the extracted features with the input. Our proposed change is simple: we avoid any quantization of the RoI boundaries

实例分割的另一类解决方案[23, 4, 3, 29]受语义分割成功的推动。这些方法从逐像素分类结果（*e.g*，如FCN输出）出发，尝试将同一类别的像素分割为不同实例。与这些方法采用的*segmentation-first*策略不同，Mask R-CNN基于*instance-first*策略。我们预计未来将出现对这两种策略更深度融合的研究。

3. 掩码区域卷积神经网络

Mask R-CNN在概念上很简单：Faster R-CNN对每个候选对象有两个输出，即类别标签和边界框偏移量；我们在此基础上增加了第三个分支，用于输出对象掩码。因此，Mask R-CNN是一个自然而直观的想法。但额外的掩码输出与类别和边界框输出不同，需要提取对象更精细的*finer*空间布局。接下来，我们将介绍Mask R-CNN的关键要素，包括像素到像素的对齐——这正是Fast/Faster R-CNN缺失的核心环节。

Faster R-CNN：我们首先简要回顾Faster R-CNN检测器[36]。Faster R-CNN包含两个阶段。第一阶段称为区域提议网络（RPN），负责生成候选目标边界框。第二阶段本质上是Fast R-CNN[12]，通过RoIPool从每个候选框中提取特征，并执行分类与边界框回归。两个阶段可采用共享特征以实现更快推理。关于Faster R-CNN与其他框架的最新全面对比，建议读者参阅文献[21]。

Mask R-CNN：Mask R-CNN采用相同的两阶段流程，第一阶段与RPN完全相同。在第二阶段，除了预测类别和边界框偏移量外，Mask R-CNN还会为每个RoI输出一个二值掩码。这与当前多数系统形成对比——那些系统通常依赖掩码预测进行分类。我们的方法遵循Fast R-CNN的思想，将边界框分类和回归并行处理，这显著简化了原始R-CNN的多阶段流程。

在训练过程中，我们正式将每个采样的RoI上的多任务损失定义为$L = L_{cls} + L_{box} + L_{mask}$。分类损失$L_{cls}$和边界框损失$L_{box}$与文献[12]中的定义相同。掩码分支为每个RoI生成一个Km^2维的输出，该输出编码了分辨率为$m \times m$的K个二值掩码，每个类别对应一个掩码（共K个类别）。我们对此应用逐像素sigmoid函数，并将L_{mask}定义为平均二值交叉熵损失。对于与真实类别k相关联的RoI，L_{mask}仅在第k个掩码上定义（其他掩码输出对损失没有贡献）。

我们对L_{mask}的定义允许网络为每个类别生成掩码，而无需类别间的竞争；我们依靠专用的分类分支来预测

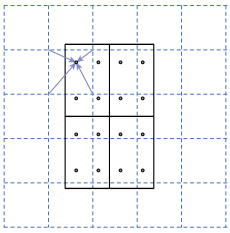

图3. RoIAlign：虚线网格表示特征图，实线表示RoI（本例中为2×2个单元），点表示每个单元中的4个采样点。RoIAlign通过特征图上邻近网格点的双线性插值计算每个采样点的值。RoI、其单元或采样点所涉及的坐标均未进行量化处理。

用于选择输出掩码的类别标签。这个 *decouples* 掩码与类别预测。这与将FCN [30] 应用于语义分割时的常见做法不同，后者通常使用逐像素的 *softmax* 和 *multinomial* 交叉熵损失。在那种情况下，不同类别的掩码会相互竞争；而在我们的方法中，使用逐像素的 *sigmoid* 和 *binary* 损失，它们则不会竞争。我们通过实验证明，这种形式对于获得良好的实例分割结果至关重要。

掩码表示：掩码编码了输入对象的*spatial*布局。因此，与类别标签或边界框偏移（这些信息不可避免地会被全连接层*fc*压缩为短输出向量）不同，提取掩码的空间结构可以通过卷积提供的像素到像素对应关系自然解决。

具体来说，我们使用全卷积网络（FCN）[30]从每个感兴趣区域（RoI）预测一个$m \times m$掩码。这使得掩码分支中的每一层都能保持明确的$m \times m$对象空间布局，而不会将其压缩为缺乏空间维度的向量表示。与以往依赖*fc*层进行掩码预测的方法[33, 34, 10]不同，我们的全卷积表示所需参数更少，且实验证明其精度更高。

这种像素到像素的行为要求我们的RoI特征（它们本身是小特征图）必须良好对齐，以忠实保留显式的逐像素空间对应关系。这促使我们开发了以下在掩模预测中起关键作用的*RoIAlign*层。

RoIAlign：RoIPool [12] 是一种从每个RoI中提取小特征图（*e.g*，如7×7）的标准操作。RoIPool首先将浮点数RoI量化到特征图的离散粒度，接着将量化后的RoI细分为空间单元（这些单元本身也被量化），最后聚合每个单元所覆盖的特征值（通常通过最大池化）。量化操作会在连续坐标x上执行，例如通过计算$[x/16]$，其中16是特征图步长，$[\cdot]$表示取整；同样，在划分为单元时（*e.g*，如7×7）也会进行量化。这些量化会导致RoI与提取的特征之间出现错位。虽然这可能不影响对微小平移具有鲁棒性的分类任务，但对预测像素级精确的分割掩码会产生显著的负面影响。

为此，我们提出了一个*RoIAlign*层，它消除了RoIPool的硬量化，并正确地将提取的特征与输入*aligning*。我们提出的改动很简单：避免对RoI边界进行任何量化。

or bins (*i.e.*, we use $x/16$ instead of $[x/16]$). We use bilinear interpolation [22] to compute the exact values of the input features at four regularly sampled locations in each RoI bin, and aggregate the result (using max or average), see Figure 3 for details. We note that the results are not sensitive to the exact sampling locations, or how many points are sampled, *as long as* no quantization is performed.

RoIAlign leads to large improvements as we show in §4.2. We also compare to the RoIWarp operation proposed in [10]. Unlike RoIAlign, RoIWarp overlooked the alignment issue and was implemented in [10] as quantizing RoI just like RoIPool. So even though RoIWarp also adopts bilinear resampling motivated by [22], it performs on par with RoIPool as shown by experiments (more details in Table 2c), demonstrating the crucial role of alignment.

Network Architecture: To demonstrate the generality of our approach, we instantiate Mask R-CNN with multiple architectures. For clarity, we differentiate between: (i) the convolutional *backbone* architecture used for feature extraction over an entire image, and (ii) the network *head* for bounding-box recognition (classification and regression) and mask prediction that is applied separately to each RoI.

We denote the *backbone* architecture using the nomenclature *network-depth-features*. We evaluate ResNet [19] and ResNeXt [45] networks of depth 50 or 101 layers. The original implementation of Faster R-CNN with ResNets [19] extracted features from the final convolutional layer of the 4-th stage, which we call C4. This backbone with ResNet-50, for example, is denoted by ResNet-50-C4. This is a common choice used in [19, 10, 21, 39].

We also explore another more effective backbone recently proposed by Lin *et al*. [27], called a Feature Pyramid Network (FPN). FPN uses a top-down architecture with lateral connections to build an in-network feature pyramid from a single-scale input. Faster R-CNN with an FPN backbone extracts RoI features from different levels of the feature pyramid according to their scale, but otherwise the rest of the approach is similar to vanilla ResNet. Using a ResNet-FPN backbone for feature extraction with Mask R-CNN gives excellent gains in both accuracy and speed. For further details on FPN, we refer readers to [27].

For the network *head* we closely follow architectures presented in previous work to which we add a fully convolutional mask prediction branch. Specifically, we extend the Faster R-CNN box heads from the ResNet [19] and FPN [27] papers. Details are shown in Figure 4. The head on the ResNet-C4 backbone includes the 5-th stage of ResNet (namely, the 9-layer 'res5' [19]), which is compute-intensive. For FPN, the backbone already includes res5 and thus allows for a more efficient head that uses fewer filters.

We note that our mask branches have a straightforward structure. More complex designs have the potential to improve performance but are not the focus of this work.

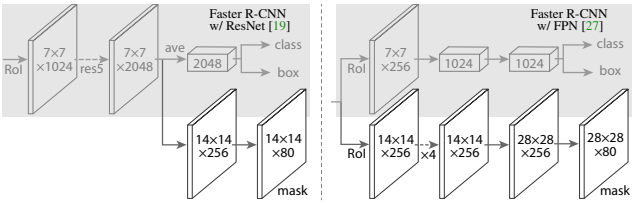

Figure 4. **Head Architecture**: We extend two existing Faster R-CNN heads [19, 27]. Left/Right panels show the heads for the ResNet C4 and FPN backbones, from [19] and [27], respectively, to which a mask branch is added. Numbers denote spatial resolution and channels. Arrows denote either conv, deconv, or *fc* layers as can be inferred from context (conv preserves spatial dimension while deconv increases it). All convs are 3×3, except the output conv which is 1×1, deconvs are 2×2 with stride 2, and we use ReLU [31] in hidden layers. *Left*: 'res5' denotes ResNet's fifth stage, which for simplicity we altered so that the first conv operates on a 7×7 RoI with stride 1 (instead of 14×14 / stride 2 as in [19]). *Right*: '×4' denotes a stack of four consecutive convs.

3.1. Implementation Details

We set hyper-parameters following existing Fast/Faster R-CNN work [12, 36, 27]. Although these decisions were made for object detection in original papers [12, 36, 27], we found our instance segmentation system is robust to them.

Training: As in Fast R-CNN, an RoI is considered positive if it has IoU with a ground-truth box of at least 0.5 and negative otherwise. The mask loss L_{mask} is defined only on positive RoIs. The mask target is the intersection between an RoI and its associated ground-truth mask.

We adopt image-centric training [12]. Images are resized such that their scale (shorter edge) is 800 pixels [27]. Each mini-batch has 2 images per GPU and each image has N sampled RoIs, with a ratio of 1:3 of positive to negatives [12]. N is 64 for the C4 backbone (as in [12, 36]) and 512 for FPN (as in [27]). We train on 8 GPUs (so effective mini-batch size is 16) for 160k iterations, with a learning rate of 0.02 which is decreased by 10 at the 120k iteration. We use a weight decay of 0.0001 and momentum of 0.9. With ResNeXt [45], we train with 1 image per GPU and the same number of iterations, with a starting learning rate of 0.01.

The RPN anchors span 5 scales and 3 aspect ratios, following [27]. For convenient ablation, RPN is trained separately and does not share features with Mask R-CNN, unless specified. For every entry in this paper, RPN and Mask R-CNN have the same backbones and so they are shareable.

Inference: At test time, the proposal number is 300 for the C4 backbone (as in [36]) and 1000 for FPN (as in [27]). We run the box prediction branch on these proposals, followed by non-maximum suppression [14]. The mask branch is then applied to the highest scoring 100 detection boxes. Although this differs from the parallel computation used in training, it speeds up inference and improves accuracy (due to the use of fewer, more accurate RoIs). The mask branch

或分箱（*i.e*，我们使用$x/16$而非$[x/16]$）。我们采用双线性插值[22]计算每个RoI分箱中四个规则采样位置处输入特征的确切值，并汇总结果（使用最大值或平均值），详见图3。我们注意到，结果对具体的采样位置或采样点数量并不敏感，*as long as*且未执行量化操作。

RoIAlign带来了显著的改进，正如我们在§4.2节中所展示的。我们还将与文献[10]提出的RoIWarp操作进行比较。与RoIAlign不同，RoIWarp忽视了对齐问题，且在[10]中实现时像RoIPool一样对RoI进行了量化处理。因此，尽管RoIWarp同样采用了文献[22]启发的双线性重采样技术，但实验表明其性能仅与RoIPool相当（更多细节见表2c），这证明了对齐的关键作用。

网络架构：为了展示我们方法的通用性，我们使用多种架构实例化了Mask R-CNN。为明确起见，我们区分以下两点：(i) 用于在整个图像上进行特征提取的卷积架构*backbone*，以及(ii) 分别应用于每个感兴趣区域（RoI）的边界框识别（分类与回归）和掩码预测网络*head*。

我们使用术语*network-depth-features*来表示*backbone*架构。我们评估了深度为50或101层的ResNet[19]和ResNeXt[45]网络。使用ResNet[19]的原始Faster R-CNN实现从第四阶段的最终卷积层提取特征，我们称之为C4。例如，采用ResNet-50的主干网络被标记为ResNet-50-C4。这是[19, 10, 21, 39]中常用的选择。

我们还探索了另一种更有效的骨干网络，即最近由Lin *et al*等人[27]提出的特征金字塔网络（FPN）。FPN采用自上而下的架构和横向连接，从单尺度输入构建网络内的特征金字塔。采用FPN骨干的Faster R-CNN根据目标尺度从特征金字塔的不同层级提取RoI特征，而其余部分与原始ResNet方法类似。将ResNet-FPN骨干网络用于Mask R-CNN的特征提取，在精度和速度上均取得了显著提升。关于FPN的更多细节，我们建议读者参阅文献[27]。

对于网络*head*，我们严格遵循先前工作中提出的架构，并在此基础上增加了一个全卷积掩码预测分支。具体而言，我们扩展了ResNet[19]和FPN[27]论文中提出的Faster R-CNN边界框头部。详细信息如图4所示。基于ResNet-C4骨干网络的头部包含了ResNet的第五阶段（即9层的‘res5’[19]），其计算量较大。对于FPN，其骨干网络已包含res5，因此可以采用更高效的头部设计，使用更少的滤波器。

我们注意到，我们的掩码分支结构简洁明了。虽然更复杂的设计有可能提升性能，但这并非本工作的研究重点。

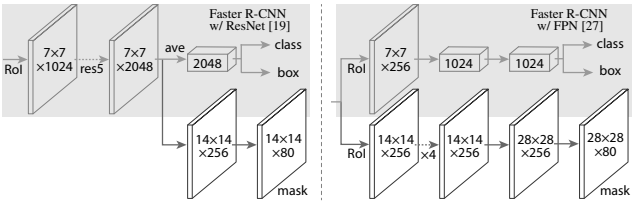

图4. 头部架构：我们扩展了两种现有的Faster R-CNN头部结构[19, 27]。左/右侧面板分别展示了基于ResNet C4和FPN骨干网络的头部结构（分别源自[19]和[27]），并为其添加了掩码分支。数字表示空间分辨率与通道数。箭头表示卷积层、反卷积层或*fc*层（根据上下文可推断：卷积层保持空间维度，反卷积层则扩大维度）。所有卷积层均为3×3，输出卷积层除外（采用1×1）；反卷积层为2×2且步长为2；隐藏层使用ReLU激活函数[31]。*Left*：“res5”表示ResNet的第五阶段，为简化设计我们调整了该结构，使其首层卷积以步长1处理7×7的RoI（而非[19]中采用的14×14/步长2）。*Right*：“×4”表示四个连续卷积层的堆叠。

3.1. 实现细节

我们根据现有的Fast/Faster R-CNN研究[12, 36, 27]设置超参数。尽管这些决策在原论文[12, 36, 27]中是为目标检测而制定的，但我们发现我们的实例分割系统对此具有鲁棒性。

训练：与Fast R-CNN相同，若某个RoI与真实标注框的交并比（IoU）不低于0.5则视为正样本，否则为负样本。掩码损失L_{mask}仅针对正样本RoI计算。掩码目标为RoI与其对应真实标注掩码的交集区域。

我们采用以图像为中心的训练方法[12]。图像被调整大小，使其尺度（较短边）为800像素[27]。每个小批次每GPU包含2张图像，每张图像有N个采样的感兴趣区域（RoIs），正负样本比例为1:3[12]。对于C4骨干网络，N为64（如[12, 36]所述），对于FPN则为512（如[27]所述）。我们在8个GPU上训练16万次迭代（实际小批次大小为16），初始学习率为0.02，在第12万次迭代时降低为原值的十分之一。权重衰减设置为0.0001，动量为0.9。使用ResNeXt[45]时，每GPU训练1张图像并保持相同迭代次数，初始学习率为0.01。

RPN锚点遵循[27]的设计，涵盖5种尺度与3种宽高比。为便于消融实验，除非特别说明，RPN均采用独立训练且不与Mask R-CNN共享特征。本文所有实验中，RPN与Mask R-CNN均采用相同主干网络，因此二者具备特征共享能力。

推理：在测试阶段，对于C4骨干网络（如[36]所述）的提议数量为300，对于FPN（如[27]所述）则为1000。我们在这些提议上运行边界框预测分支，随后进行非极大值抑制[14]。接着将掩码分支应用于得分最高的100个检测框。尽管这与训练时使用的并行计算方式不同，但它加快了推理速度并提高了准确性（因为使用了更少、更精确的RoIs）。掩码分支

Figure 5. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN	ResNeXt-101-FPN	**37.1**	**60.0**	**39.4**	**16.9**	**39.9**	**53.5**

Table 1. **Instance segmentation** *mask* AP on COCO `test-dev`. MNC [10] and FCIS [26] are the winners of the COCO 2015 and 2016 segmentation challenges, respectively. Without bells and whistles, Mask R-CNN outperforms the more complex FCIS+++, which includes multi-scale train/test, horizontal flip test, and OHEM [38]. All entries are *single-model* results.

can predict K masks per RoI, but we only use the k-th mask, where k is the predicted class by the classification branch. The $m \times m$ floating-number mask output is then resized to the RoI size, and binarized at a threshold of 0.5.

Note that since we only compute masks on the top 100 detection boxes, Mask R-CNN adds a small overhead to its Faster R-CNN counterpart (*e.g.*, ~20% on typical models).

4. Experiments: Instance Segmentation

We perform a thorough comparison of Mask R-CNN to the state of the art along with comprehensive ablations on the COCO dataset [28]. We report the standard COCO metrics including AP (averaged over IoU thresholds), AP_{50}, AP_{75}, and AP_S, AP_M, AP_L (AP at different scales). Unless noted, AP is evaluating using *mask* IoU. As in previous work [5, 27], we train using the union of 80k train images and a 35k subset of val images (`trainval35k`), and report ablations on the remaining 5k val images (`minival`). We also report results on `test-dev` [28].

4.1. Main Results

We compare Mask R-CNN to the state-of-the-art methods in instance segmentation in Table 1. All instantiations of our model outperform baseline variants of previous state-of-the-art models. This includes MNC [10] and FCIS [26], the winners of the COCO 2015 and 2016 segmentation challenges, respectively. Without bells and whistles, Mask R-CNN with ResNet-101-FPN backbone outperforms FCIS+++ [26], which includes multi-scale train/test, horizontal flip test, and online hard example mining (OHEM) [38]. While outside the scope of this work, we expect many such improvements to be applicable to ours.

Mask R-CNN outputs are visualized in Figures 2 and 5. Mask R-CNN achieves good results even under challenging conditions. In Figure 6 we compare our Mask R-CNN baseline and FCIS+++ [26]. FCIS+++ exhibits systematic artifacts on overlapping instances, suggesting that it is challenged by the fundamental difficulty of instance segmentation. Mask R-CNN shows no such artifacts.

图5. 在COCO测试图像上使用ResNet-101-FPN以5帧/秒速度运行的Mask R-CNN更多结果，其掩码AP为35.7（见表1）。

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN	ResNeXt-101-FPN	**37.1**	**60.0**	**39.4**	**16.9**	**39.9**	**53.5**

表1. COCO test-dev上的实例分割*mask* AP。MNC [10]和FCIS [26]分别是COCO 2015和2016分割挑战赛的优胜者。在不使用任何技巧的情况下，Mask R-CNN超越了更复杂的FCIS+++（后者包含了多尺度训练/测试、水平翻转测试和OHEM [38]）。所有条目均为*single-model*结果。

可以预测每个RoI的K个掩码，但我们只使用第k个掩码，其中k是分类分支预测的类别。随后将$m \times m$浮点数掩码输出调整至RoI尺寸，并以0.5为阈值进行二值化处理。

请注意，由于我们仅对前100个检测框计算掩码，Mask R-CNN 相较于其 Faster R-CNN 版本仅增加了少量计算开销（例如在典型模型中约为20%）。

4. 实验：实例分割

我们对Mask R-CNN与现有先进技术进行了全面比较，并在COCO数据集[28]上进行了详尽的消融实验。我们报告了标准的COCO评估指标，包括AP（在IoU阈值上取平均值）、AP_{50}、AP_{75}、AP_S、AP_M、AP_L（以及不同尺度下的AP)。除非特别说明，AP均采用*mask* IoU进行评估。遵循先前工作[5, 27]的做法，我们使用80k训练图像与35k验证图像子集（trainval35k）的联合集进行训练，并在剩余的5k验证图像（minival）上报告消融实验结果。同时，我们也报告了在test-dev数据集[28]上的结果。

4.1. 主要结果

我们在表1中将Mask R-CNN与实例分割领域的最先进方法进行了比较。我们模型的所有实例均超越了先前最先进模型的基线变体，这包括分别获得COCO 2015和2016分割挑战赛冠军的MNC[10]和FCIS[26]。在不使用任何额外技巧的情况下，采用ResNet-101-FPN骨干网络的Mask R-CNN超越了包含多尺度训练/测试、水平翻转测试和在线难例挖掘（OHEM）[38]的FCIS+++[26]。尽管这些改进超出了本工作的范畴，我们预计许多此类改进也适用于我们的模型。

Mask R-CNN 的输出结果在图 2 和图 5 中进行了可视化展示。即使在具有挑战性的条件下，Mask R-CNN 也能取得良好的效果。在图 6 中，我们将我们的 Mask R-CNN 基线模型与 FCIS+++ [26] 进行了比较。FCIS +++ 在重叠实例上表现出系统性的伪影，这表明它受到了实例分割这一根本性难题的挑战。而 Mask R-CNN 则没有出现此类伪影。

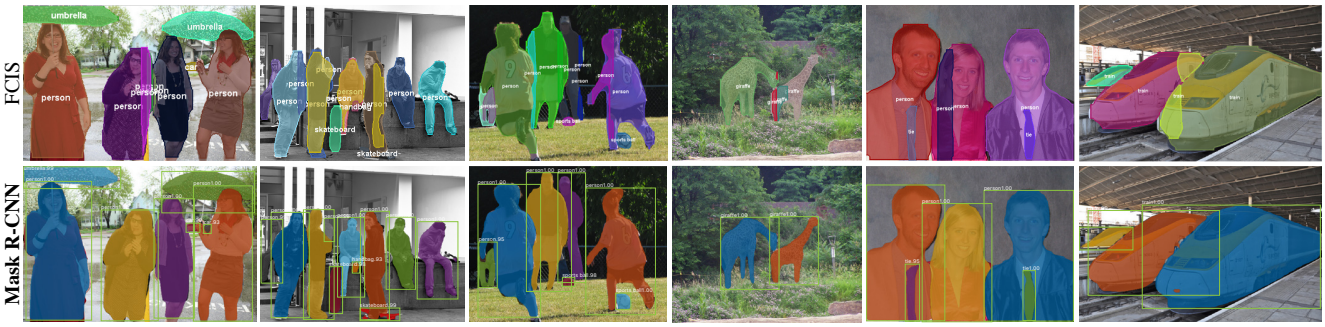

Figure 6. FCIS+++ [26] (top) *vs*. Mask R-CNN (bottom, ResNet-101-FPN). FCIS exhibits systematic artifacts on overlapping objects.

net-depth-features	AP	AP_{50}	AP_{75}
ResNet-50-C4	30.3	51.2	31.5
ResNet-101-C4	32.7	54.2	34.3
ResNet-50-FPN	33.6	55.2	35.3
ResNet-101-FPN	35.4	57.3	37.5
ResNeXt-101-FPN	**36.7**	**59.5**	**38.9**

(a) **Backbone Architecture**: Better backbones bring expected gains: deeper networks do better, FPN outperforms C4 features, and ResNeXt improves on ResNet.

	AP	AP_{50}	AP_{75}
softmax	24.8	44.1	25.1
sigmoid	**30.3**	**51.2**	**31.5**
	+5.5	*+7.1*	*+6.4*

(b) **Multinomial *vs*. Independent Masks** (ResNet-50-C4): *Decoupling* via per-class binary masks (sigmoid) gives large gains over multinomial masks (softmax).

	align?	bilinear?	agg.	AP	AP_{50}	AP_{75}
RoIPool [12]			max	26.9	48.8	26.4
RoIWarp [10]		✓	max	27.2	49.2	27.1
		✓	ave	27.1	48.9	27.1
RoIAlign	✓	✓	max	**30.2**	**51.0**	**31.8**
	✓	✓	ave	**30.3**	**51.2**	**31.5**

(c) **RoIAlign** (ResNet-50-C4): Mask results with various RoI layers. Our RoIAlign layer improves AP by ~3 points and AP_{75} by ~5 points. Using proper alignment is the only factor that contributes to the large gap between RoI layers.

	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
RoIPool	23.6	46.5	21.6	28.2	52.7	26.9
RoIAlign	**30.9**	**51.8**	**32.1**	**34.0**	**55.3**	**36.4**
	+7.3	*+ 5.3*	*+10.5*	*+5.8*	*+2.6*	*+9.5*

(d) **RoIAlign** (ResNet-50-**C5**, *stride 32*): Mask-level and box-level AP using *large-stride* features. Misalignments are more severe than with stride-16 features (Table 2c), resulting in big accuracy gaps.

	mask branch	AP	AP_{50}	AP_{75}
MLP	fc: 1024→1024→80·28^2	31.5	53.7	32.8
MLP	fc: 1024→1024→1024→80·28^2	31.5	54.0	32.6
FCN	conv: 256→256→256→256→256→80	**33.6**	**55.2**	**35.3**

(e) **Mask Branch** (ResNet-50-FPN): Fully convolutional networks (FCN) *vs*. multi-layer perceptrons (MLP, fully-connected) for mask prediction. FCNs improve results as they take advantage of explicitly encoding spatial layout.

Table 2. **Ablations**. We train on `trainval35k`, test on `minival`, and report *mask* AP unless otherwise noted.

4.2. Ablation Experiments

We run a number of ablations to analyze Mask R-CNN. Results are shown in Table 2 and discussed in detail next.

Architecture: Table 2a shows Mask R-CNN with various backbones. It benefits from deeper networks (50 *vs*. 101) and advanced designs including FPN and ResNeXt. We note that *not* all frameworks automatically benefit from deeper or advanced networks (see benchmarking in [21]).

Multinomial *vs*. Independent Masks: Mask R-CNN *decouples* mask and class prediction: as the existing box branch predicts the class label, we generate a mask for each class without competition among classes (by a per-pixel *sigmoid* and a *binary* loss). In Table 2b, we compare this to using a per-pixel *softmax* and a *multinomial* loss (as commonly used in FCN [30]). This alternative *couples* the tasks of mask and class prediction, and results in a severe loss in mask AP (5.5 points). This suggests that once the instance has been classified as a whole (by the box branch), it is sufficient to predict a binary mask without concern for the categories, which makes the model easier to train.

Class-Specific *vs*. Class-Agnostic Masks: Our default instantiation predicts class-specific masks, *i.e*., one $m \times m$ mask per class. Interestingly, Mask R-CNN with class-agnostic masks (*i.e*., predicting a single $m \times m$ output regardless of class) is nearly as effective: it has 29.7 mask AP *vs*. 30.3 for the class-specific counterpart on ResNet-50-C4. This further highlights the division of labor in our approach which largely decouples classification and segmentation.

RoIAlign: An evaluation of our proposed *RoIAlign* layer is shown in Table 2c. For this experiment we use the ResNet-50-C4 backbone, which has stride 16. RoIAlign improves AP by about 3 points over RoIPool, with much of the gain coming at high IoU (AP_{75}). RoIAlign is insensitive to max/average pool; we use average in the rest of the paper.

Additionally, we compare with *RoIWarp* proposed in MNC [10] that also adopt bilinear sampling. As discussed in §3, RoIWarp still quantizes the RoI, losing alignment with the input. As can be seen in Table 2c, RoIWarp performs on par with RoIPool and much worse than RoIAlign. This highlights that proper alignment is key.

We also evaluate RoIAlign with a *ResNet-50-C5* backbone, which has an even larger stride of 32 pixels. We use the same head as in Figure 4 (right), as the res5 head is not applicable. Table 2d shows that RoIAlign improves mask AP by a massive 7.3 points, and mask AP_{75} by 10.5 points

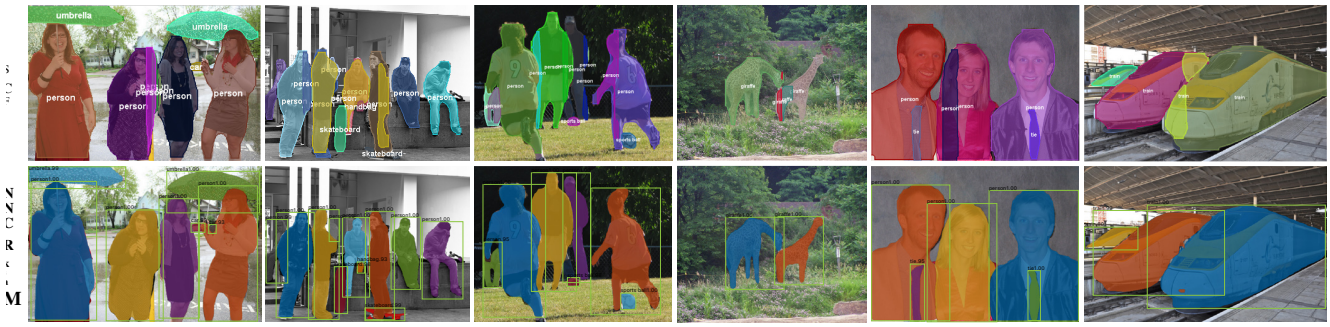

图6. FCIS+++ [26]（上）*vs*。 Mask R-CNN（底部，ResNet-101-FPN）。FCIS表现出系统性的a重叠物体上的伪影。

net-depth-features	AP	AP_{50}	AP_{75}
ResNet-50-C4	30.3	51.2	31.5
ResNet-101-C4	32.7	54.2	34.3
ResNet-50-FPN	33.6	55.2	35.3
ResNet-101-FPN	35.4	57.3	37.5
ResNeXt-101-FPN	**36.7**	**59.5**	**38.9**

(a) 骨干网络架构：更优的骨干网络带来预期增益：更深的网络表现更好，FPN优于C4特征，且ResNeXt在ResNet基础上有所改进。

	AP	AP_{50}	AP_{75}
softmax	24.8	44.1	25.1
sigmoid	**30.3**	**51.2**	**31.5**
	+5.5	*+7.1*	*+6.4*

(b) 多项分布 ***vs***。独立掩码(ResNet-50-C4)：通过逐类二元掩码（sigmoid）的 {v*} 相比多项式掩码（softmax）带来了显著提升。

	align?	bilinear?	agg.	AP	AP_{50}	AP_{75}
RoIPool [12]			max	26.9	48.8	26.4
RoIWarp [10]		✓	max	27.2	49.2	27.1
		✓	ave	27.1	48.9	27.1
RoIAlign	✓	✓	max	**30.2**	**51.0**	**31.8**
	✓	✓	ave	**30.3**	**51.2**	**31.5**

(c) RoIAlign（ResNet-50-C4）：使用不同RoI层的掩码结果。我们的RoIAlign层将AP提升了~3个百分点，AP_{75}提升了~5个百分点。采用正确的对齐方式是导致RoI层之间巨大差异的唯一因素。

	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
RoIPool	23.6	46.5	21.6	28.2	52.7	26.9
RoIAlign	**30.9**	**51.8**	**32.1**	**34.0**	**55.3**	**36.4**
	+7.3	*+ 5.3*	*+10.5*	*+5.8*	*+2.6*	*+9.5*

(d) RoIAlign（ResNet-50-C5，*stride 32*）：使用*large-stride*特征的掩码级和边界框级AP。错位问题比步幅16特征更严重（表2c），导致准确率差距较大。

	mask branch	AP	AP_{50}	AP_{75}
MLP	fc: 1024→1024→80·28^2	31.5	53.7	32.8
MLP	fc: 1024→1024→1024→80·28^2	31.5	54.0	32.6
FCN	conv: 256→256→256→256→256→80	**33.6**	**55.2**	**35.3**

(e) 掩码分支（ResNet-50-FPN）：全卷积网络（FCN）*vs*。使用多层感知机（MLP，全连接层）进行掩码预测。FCN通过显式编码空间布局的优势，提升了预测结果。

表2. 消融实验。除非另有说明，我们均在trainval35k上训练，在minival上测试，并报告*mask* AP。

4.2. 消融实验

我们进行了多项消融实验来分析Mask R-CNN。结果如表2所示，接下来将详细讨论。

架构：表2a展示了带有不同骨干网络的Mask R-CNN。它受益于更深的网络（50 {v*} 101）以及包括FPN和ResNeXt在内的先进设计。我们注意到，所有框架都会自动从更深层或更先进的网络中受益（参见[21]中的基准测试）。

多项式 ***vs***。独立掩码：Mask R-CNN *de- couples* 的掩码与类别预测：由于现有的边界框分支已预测类别标签，我们为每个类别生成掩码，无需在类别间进行竞争（通过逐像素的 *sig- moid* 和 *binary* 损失函数实现）。在表2b中，我们将其与使用逐像素 *softmax* 和 *multinomial* 损失函数的方法（如FCN[30]中常用方案）进行对比。这种替代方案 *couples* 了掩码与类别预测的任务，导致掩码AP大幅下降（降低5.5个点）。这表明当实例已通过边界框分支完成整体分类后，只需预测二值掩码而无需考虑具体类别，这使模型更易于训练。

类别特定***vs***。类别无关掩码：我们的默认实例化预测类别特定掩码，*i.e*，即一个$m \times m$每个类别的掩码。有趣的是，使用类别无关掩码的Mask R-CNN（即无论类别如何都预测单个掩码输出）几乎同样有效：在ResNet-50-C4上，其掩码AP为29.7，而类别特定版本的掩码AP为30.3。这进一步凸显了我们方法中的分工机制——该方法在很大程度上将分类与分割任务解耦。

RoIAlign：我们提出的*RoIAlign*层的评估结果如表2c所示。在此实验中，我们采用步长为16的ResNet-50-C4骨干网络。RoIAlign相比RoIPool将AP提升了约3个百分点，其中大部分提升来自高IoU阈值下的表现（AP_{75}）。RoIAlign对最大/平均池化不敏感；本文后续部分均采用平均池化。

此外，我们与MNC[10]中提出的同样采用双线性采样的*RoIWarp*进行了比较。如§3中所讨论的，RoIWarp仍然对RoI进行量化，导致与输入的对齐丢失。从表2c可以看出，RoIWarp的表现与RoIPool相当，但远不如RoIAlign。这突显了正确的对齐是关键。

我们还使用步幅更大（32像素）的*ResNet-50-C5*骨干网络评估了RoIAlign。由于res5头部不适用，我们采用了与图4（右）相同的头部结构。表2d显示，RoIAlign将掩码AP大幅提升了7.3个百分点，并将掩码AP_{75}提升了10.5个百分点。

	backbone	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [41]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [39]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	**39.8**	**62.3**	**43.4**	**22.1**	**43.2**	51.2

Table 3. **Object detection** *single-model* results (bounding box AP), *vs.* state-of-the-art on `test-dev`. Mask R-CNN using ResNet-101-FPN outperforms the base variants of all previous state-of-the-art models (the mask output is ignored in these experiments). The gains of Mask R-CNN over [27] come from using RoIAlign (+1.1 AP^{bb}), multitask training (+0.9 AP^{bb}), and ResNeXt-101 (+1.6 AP^{bb}).

(*50% relative improvement*). Moreover, we note that with RoIAlign, using *stride-32* C5 features (30.9 AP) is more accurate than using stride-16 C4 features (30.3 AP, Table 2c). RoIAlign largely resolves the long-standing challenge of using large-stride features for detection and segmentation.

Finally, RoIAlign shows a gain of 1.5 mask AP and 0.5 box AP when used with FPN, which has finer multi-level strides. For keypoint detection that requires finer alignment, RoIAlign shows large gains even with FPN (Table 6).

Mask Branch: Segmentation is a pixel-to-pixel task and we exploit the spatial layout of masks by using an FCN. In Table 2e, we compare multi-layer perceptrons (MLP) and FCNs, using a ResNet-50-FPN backbone. Using FCNs gives a 2.1 mask AP gain over MLPs. We note that we choose this backbone so that the conv layers of the FCN head are not pre-trained, for a fair comparison with MLP.

4.3. Bounding Box Detection Results

We compare Mask R-CNN to the state-of-the-art COCO *bounding-box* object detection in Table 3. For this result, even though the full Mask R-CNN model is trained, only the classification and box outputs are used at inference (the mask output is ignored). Mask R-CNN using ResNet-101-FPN outperforms the base variants of all previous state-of-the-art models, including the single-model variant of G-RMI [21], the winner of the COCO 2016 Detection Challenge. Using ResNeXt-101-FPN, Mask R-CNN further improves results, with a margin of 3.0 points box AP over the best previous single model entry from [39] (which used Inception-ResNet-v2-TDM).

As a further comparison, we trained a version of Mask R-CNN but *without* the mask branch, denoted by "Faster R-CNN, RoIAlign" in Table 3. This model performs better than the model presented in [27] due to RoIAlign. On the other hand, it is 0.9 points box AP lower than Mask R-CNN. This gap of Mask R-CNN on box detection is therefore due solely to the benefits of multi-task training.

Lastly, we note that Mask R-CNN attains a small gap between its mask and box AP: *e.g.*, 2.7 points between 37.1 (mask, Table 1) and 39.8 (box, Table 3). This indicates that our approach largely closes the gap between object detection and the more challenging instance segmentation task.

4.4. Timing

Inference: We train a ResNet-101-FPN model that shares features between the RPN and Mask R-CNN stages, following the 4-step training of Faster R-CNN [36]. This model runs at 195ms per image on an Nvidia Tesla M40 GPU (plus 15ms CPU time resizing the outputs to the original resolution), and achieves statistically the same mask AP as the unshared one. We also report that the ResNet-101-C4 variant takes ~400ms as it has a heavier box head (Figure 4), so we do not recommend using the C4 variant in practice.

Although Mask R-CNN is fast, we note that our design is not optimized for speed, and better speed/accuracy tradeoffs could be achieved [21], *e.g.*, by varying image sizes and proposal numbers, which is beyond the scope of this paper.

Training: Mask R-CNN is also fast to train. Training with ResNet-50-FPN on COCO `trainval35k` takes 32 hours in our synchronized 8-GPU implementation (0.72s per 16-image mini-batch), and 44 hours with ResNet-101-FPN. In fact, fast prototyping can be completed in *less than one day* when training on the `train` set. We hope such rapid training will remove a major hurdle in this area and encourage more people to perform research on this challenging topic.

5. Mask R-CNN for Human Pose Estimation

Our framework can easily be extended to human pose estimation. We model a keypoint's location as a one-hot mask, and adopt Mask R-CNN to predict K masks, one for each of K keypoint types (*e.g.*, left shoulder, right elbow). This task helps demonstrate the flexibility of Mask R-CNN.

We note that *minimal* domain knowledge for human pose is exploited by our system, as the experiments are mainly to demonstrate the generality of the Mask R-CNN framework. We expect that domain knowledge (*e.g.*, modeling structures [6]) will be complementary to our simple approach.

Implementation Details: We make minor modifications to the segmentation system when adapting it for keypoints. For each of the K keypoints of an instance, the training target is a one-hot $m \times m$ binary mask where only a *single* pixel is labeled as foreground. During training, for each visible ground-truth keypoint, we minimize the cross-entropy loss over an m^2-way softmax output (which encourages a

	backbone	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [41]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [39]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	**39.8**	**62.3**	**43.4**	**22.1**	**43.2**	51.2

表3. 目标检测 *single-model* 结果（边界框AP），*vs*。在test-dev集上的最新技术水平。使用ResNet-101-FPN的Mask R-CNN超越了所有先前最先进模型的基础变体（在这些实验中忽略了掩码输出）。Mask R-CNN相对于[27]的增益来自使用RoIAlign（+1.1 AP^{bb}）、多任务训练（+0.9 AP^{bb}）以及ResNeXt-101（+1.6 AP^{bb}）。

(*50% relative improvement*)。此外，我们注意到，使用RoIAlign时，采用*stride-32* C5特征（30.9 AP）比使用步长为16的C4特征（30.3 AP，表2c）更准确。RoIAlign在很大程度上解决了长期以来在检测和分割任务中使用大跨度特征的难题。

最后，当RoIAlign与具有更精细多级步长的FPN一起使用时，在掩码AP上提升了1.5，在边界框AP上提升了0.5。对于需要更精细对齐的关键点检测任务，即使使用FPN，RoIAlign也带来了显著提升（见表6）。

掩码分支：分割是一项像素到像素的任务，我们通过使用全卷积网络（FCN）来利用掩码的空间布局。在表2e中，我们使用ResNet-50-FPN骨干网络，比较了多层感知机（MLP）和FCN。使用FCN相比MLP带来了2.1的掩码AP提升。我们选择这一骨干网络是为了确保FCN头部的卷积层未经预训练，以便与MLP进行公平比较。

4.3. 边界框检测结果

我们在表3中将Mask R-CNN与最先进的COCO *bounding-box*目标检测方法进行了比较。在此结果中，尽管训练了完整的Mask R-CNN模型，但在推理时仅使用了分类和边界框输出（掩码输出被忽略）。采用ResNet-101-FPN的Mask R-CNN超越了所有先前最先进模型的基础变体，包括COCO 2016检测挑战赛冠军G-RMI [21]的单模型变体。使用ResNeXt-101-FPN时，Mask R-CNN进一步提升了性能，其边界框AP比先前最佳单模型方案[39]（采用Inception-ResNet-v2-TDM）高出3.0个点。

作为进一步的比较，我们训练了一个Mask R-CNN的变体，但移除了掩码分支，在表3中标记为“Faster R-CNN, RoIAlign”。由于采用了RoIAlign，该模型的表现优于文献[27]中的模型。另一方面，其边界框AP比Mask R-CNN低0.9个百分点。因此，Mask R-CNN在边界框检测上的优势完全得益于多任务训练带来的益处。

最后，我们注意到Mask R-CNN在其掩码AP与边界框AP之间的差距很小：*e.g*，即37.1（掩码AP，表1）与39.8（边界框AP，表3）之间相差2.7个百分点。这表明我们的方法在很大程度上弥合了目标检测与更具挑战性的实例分割任务之间的差距。

4.4. 时序

推理：我们训练了一个ResNet-101-FPN模型，该模型在RPN和Mask R-CNN阶段共享特征，遵循Faster R-CNN的四步训练法[36]。该模型在Nvidia Tesla M40 GPU上每张图像的处理时间为195毫秒（外加15毫秒的CPU时间用于将输出调整至原始分辨率），并在统计上获得了与非共享特征模型相同的掩码AP。我们还报告了ResNet-101-C4变体需要~400毫秒，因为其边界框头部更重（图4），因此我们不建议在实践中使用C4变体。

尽管Mask R-CNN速度很快，但我们注意到我们的设计并未针对速度进行优化，通过调整图像尺寸和候选框数量，可以实现更好的速度/精度权衡[v15]，但这已超出本文的讨论范围。

训练：Mask R-CNN的训练速度也很快。在我们同步的8-GPU实现中，使用ResNet-50-FPN在COCO trainval35k上训练需要32小时（每16张图像的小批量处理时间为0.72秒），使用ResNet-101-FPN则需要44小时。实际上，在训练集上进行训练时，快速原型制作可以在*less than one day*内完成。我们希望这种快速的训练能消除该领域的一个主要障碍，并鼓励更多人投身于这一具有挑战性的课题研究。

5. 用于人体姿态估计的掩码区域卷积神经网络

我们的框架可以轻松扩展到人体姿态估计。我们将关键点的位置建模为一个独热掩码，并采用Mask R-CNN来预测K个掩码，每个掩码对应K种关键点类型之一（*e.g*，例如左肩、右肘）。这项任务有助于展示Mask R-CNN的灵活性。

我们注意到，由于实验主要是为了展示Mask R-CNN框架的通用性，我们的系统并未利用人体姿态的*minimal*领域知识。我们预计领域知识（*e.g*，例如建模结构[6]）将与我们简单的方法形成互补。

实现细节：在将分割系统适配于关键点时，我们进行了细微调整。对于每个实例的K个关键点，训练目标是一个独热编码的$m \times m$二进制掩码，其中仅有一个*single*像素被标记为前景。在训练过程中，针对每个可见的真实关键点，我们通过最小化交叉熵损失来优化一个m^2路softmax输出（这有助于

Figure 7. Keypoint detection results on COCO `test` using Mask R-CNN (ResNet-50-FPN), with person segmentation masks predicted from the same model. This model has a keypoint AP of 63.1 and runs at 5 fps.

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
CMU-Pose+++ [6]	61.8	84.9	67.5	57.1	68.2
G-RMI [32]†	62.4	84.0	68.5	**59.1**	68.1
Mask R-CNN, keypoint-only	62.7	87.0	68.4	57.4	71.1
Mask R-CNN, keypoint & mask	**63.1**	**87.3**	**68.7**	57.8	**71.4**

Table 4. **Keypoint detection** AP on COCO `test-dev`. Ours is a single model (ResNet-50-FPN) that runs at 5 fps. CMU-Pose+++ [6] is the 2016 competition winner that uses multi-scale testing, post-processing with CPM [44], and filtering with an object detector, adding a cumulative ~5 points (clarified in personal communication). †: G-RMI was trained on COCO *plus* MPII [1] (25k images), using two models (Inception-ResNet-v2 for bounding box detection and ResNet-101 for keypoints).

single point to be detected). We note that as in instance segmentation, the K keypoints are still treated independently.

We adopt the ResNet-FPN variant, and the keypoint head architecture is similar to that in Figure 4 (right). The keypoint head consists of a stack of eight 3×3 512-d conv layers, followed by a deconv layer and 2× bilinear upscaling, producing an output resolution of 56×56. We found that a relatively high resolution output (compared to masks) is required for keypoint-level localization accuracy.

Models are trained on all COCO `trainval35k` images that contain annotated keypoints. To reduce overfitting, as this training set is smaller, we train using image scales randomly sampled from [640, 800] pixels; inference is on a single scale of 800 pixels. We train for 90k iterations, starting from a learning rate of 0.02 and reducing it by 10 at 60k and 80k iterations. We use bounding-box NMS with a threshold of 0.5. Other details are identical as in §3.1.

Main Results and Ablations: We evaluate the person keypoint AP (AP^{kp}) and experiment with a ResNet-50-FPN backbone; more backbones will be studied in the appendix. Table 4 shows that our result (62.7 AP^{kp}) is 0.9 points higher than the COCO 2016 keypoint detection winner [6] that uses a multi-stage processing pipeline (see caption of Table 4). Our method is considerably simpler and faster.

More importantly, we have *a unified model that can simultaneously predict boxes, segments, and keypoints* while running at 5 fps. Adding a segment branch (for the person category) improves the AP^{kp} to 63.1 (Table 4) on `test-dev`. More ablations of multi-task learning on `minival` are in Table 5. Adding the *mask* branch to the box-only (*i.e.*, Faster R-CNN) or keypoint-only versions consistently improves these tasks. However, adding the keypoint branch reduces the box/mask AP slightly, suggesting that while keypoint detection benefits from multitask training, it does not in turn help the other tasks. Nevertheless, learning all three tasks jointly enables a unified system to efficiently predict all outputs simultaneously (Figure 7).

	AP^{bb}_{person}	AP^{mask}_{person}	AP^{kp}
Faster R-CNN	52.5	-	-
Mask R-CNN, mask-only	**53.6**	**45.8**	-
Mask R-CNN, keypoint-only	50.7	-	64.2
Mask R-CNN, keypoint & mask	52.0	45.1	**64.7**

Table 5. **Multi-task learning** of box, mask, and keypoint about the *person* category, evaluated on `minival`. All entries are trained on the same data for fair comparisons. The backbone is ResNet-50-FPN. The entries with 64.2 and 64.7 AP on `minival` have `test-dev` AP of 62.7 and 63.1, respectively (see Table 4).

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
RoIPool	59.8	86.2	66.7	55.1	67.4
RoIAlign	**64.2**	**86.6**	**69.7**	**58.7**	**73.0**

Table 6. **RoIAlign *vs.* RoIPool** for keypoint detection on `minival`. The backbone is ResNet-50-FPN.

We also investigate the effect of *RoIAlign* on keypoint detection (Table 6). Though this ResNet-50-FPN backbone has finer strides (*e.g.*, 4 pixels on the finest level), RoIAlign still shows significant improvement over RoIPool and increases AP^{kp} by 4.4 points. This is because keypoint detections are more sensitive to localization accuracy. This again indicates that alignment is essential for pixel-level localization, including masks and keypoints.

Given the effectiveness of Mask R-CNN for extracting object bounding boxes, masks, and keypoints, we expect it be an effective framework for other instance-level tasks.

图7. 使用Mask R-CNN（ResNet-50-FPN）在COCO测试集上的关键点检测结果，人物分割掩码由同一模型预测得出。该模型的关键点AP为63.1，运行速度为5帧/秒。

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
CMU-Pose+++ [6]	61.8	84.9	67.5	57.1	68.2
G-RMI [32]†	62.4	84.0	68.5	**59.1**	68.1
Mask R-CNN, keypoint-only	62.7	87.0	68.4	57.4	71.1
Mask R-CNN, keypoint & mask	**63.1**	**87.3**	**68.7**	57.8	**71.4**

表4. COCO test-dev上的关键点检测AP。我们的模型是单模型（ResNet-50-FPN），运行速度为5 fps。CMU-Pose+++ [6]是2016年竞赛冠军，采用多尺度测试、CPM[44]后处理以及目标检测器过滤，累计提升~5分（经个人沟通确认）。†：G-RMI在COCO*plus* MPII[1]（25k图像）上训练，使用两个模型（Inception-ResNet-v2用于边界框检测，ResNet-101用于关键点）。

	AP^{bb}_{person}	AP^{mask}_{person}	AP^{kp}
Faster R-CNN	52.5	-	-
Mask R-CNN, mask-only	**53.6**	**45.8**	-
Mask R-CNN, keypoint-only	50.7	-	64.2
Mask R-CNN, keypoint & mask	52.0	45.1	**64.7**

表5. 关于*person*类别的框、掩码和关键点多任务学习，在minival上评估。所有条目均使用相同数据训练以确保公平比较。主干网络为ResNet-50-FPN。在minival上获得64.2和64.7 AP的条目，其test-dev AP分别为62.7和63.1（见表4）。

	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
RoIPool	59.8	86.2	66.7	55.1	67.4
RoIAlign	**64.2**	**86.6**	**69.7**	**58.7**	**73.0**

表6. 用于minival关键点检测的RoIAlign ***vs***与RoIPool对比。骨干网络为ResNet-50-FPN。

单点检测）。我们注意到，与实例分割类似，K关键点仍被独立处理。

我们采用ResNet-FPN变体，关键点头部架构与图4（右）类似。关键点头部由八个3×3 512维卷积层堆叠而成，随后是一个反卷积层和2×倍双线性上采样，最终输出分辨率为56×56。我们发现，要实现关键点级别的定位精度，需要相对较高分辨率的输出（与掩码相比）。

模型在所有包含标注关键点的COCO trainval35k图像上进行训练。由于训练集规模较小，为减少过拟合，我们采用随机从[640, 800]像素范围内采样的图像尺度进行训练；推理时则使用800像素的单一尺度。训练共进行90k次迭代，初始学习率为0.02，并在60k和80k迭代时将其降低10倍。我们使用阈值为0.5的边界框非极大值抑制。其他细节与§3.1节所述保持一致。

主要结果与消融实验：我们评估了人体关键点AP（AP^{kp}），并采用ResNet-50-FPN骨干网络进行实验；更多骨干网络将在附录中讨论。表4显示，我们的结果（62.7 AP^{kp}）比采用多阶段处理流程的COCO 2016关键点检测冠军方法[6]高出0.9分（见表4说明）。我们的方法明显更简洁、更快速。

更重要的是，我们拥有 *a unified model that can simultaneously predict boxes, segments, and keypoints* 以每秒5帧运行时。添加一个分割分支（针对人物类别）将测试开发集上的AP^{kp}提升至63.1（表4）。表5展示了在minival数据集上多任务学习的更多消融实验。在仅检测框（*i.e*，即Faster R-CNN）或仅关键点检测的版本中添加*mask*分支，均能持续提升这些任务的性能。然而，添加关键点分支会略微降低检测框/掩码的AP值，这表明虽然关键点检测受益于多任务训练，但其本身并未对其他任务产生助益。尽管如此，联合学习所有三项任务使得统一系统能够高效地同时预测全部输出结果（图7）。

我们还研究了*RoIAlign*对关键点检测的影响（表6）。尽管这个ResNet-50-FPN骨干网络具有更精细的步幅（*e.g*，在最精细层级上为4像素），RoIAlign仍然显示出相对于RoIPool的显著改进，并将AP^{kp}提高了4.4个百分点。这是因为关键点检测对定位精度更为敏感。这再次表明对齐对于像素级定位（包括掩码和关键点）至关重要。

鉴于Mask R-CNN在提取物体边界框、掩码和关键点方面的有效性，我们预期它将成为其他实例级任务的有效框架。

	training data	AP [val]	AP	AP_{50}	person	rider	car	truck	bus	train	mcycle	bicycle
InstanceCut [23]	fine + coarse	15.8	13.0	27.9	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
DWT [4]	fine	19.8	15.6	30.0	15.1	11.7	32.9	17.1	20.4	15.0	7.9	4.9
SAIS [17]	fine	-	17.4	36.7	14.6	12.9	35.7	16.0	23.2	19.0	10.3	7.8
DIN [3]	fine + coarse	-	20.0	38.8	16.5	16.7	25.7	20.6	30.0	23.4	17.1	10.1
SGN [29]	fine + coarse	29.2	25.0	44.9	21.8	20.1	39.4	24.8	33.2	30.8	17.7	12.4
Mask R-CNN	fine	31.5	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
Mask R-CNN	fine + COCO	**36.4**	**32.0**	**58.1**	**34.8**	**27.0**	**49.1**	**30.1**	**40.9**	**30.9**	**24.1**	**18.7**

Table 7. Results on Cityscapes val ('AP [val]' column) and test (remaining columns) sets. Our method uses ResNet-50-FPN.

Appendix A: Experiments on Cityscapes

We further report instance segmentation results on the Cityscapes [7] dataset. This dataset has fine annotations for 2975 train, 500 val, and 1525 test images. It has 20k coarse training images without instance annotations, which we do *not* use. All images are 2048×1024 pixels. The instance segmentation task involves 8 object categories, whose numbers of instances on the fine training set are:

person	rider	car	truck	bus	train	mcycle	bicycle
17.9k	1.8k	26.9k	0.5k	0.4k	0.2k	0.7k	3.7k

Instance segmentation performance on this task is measured by the COCO-style mask AP (averaged over IoU thresholds); AP_{50} (*i.e.*, mask AP at an IoU of 0.5) is also reported.

Implementation: We apply our Mask R-CNN models with the ResNet-FPN-50 backbone; we found the 101-layer counterpart performs similarly due to the small dataset size. We train with image scale (shorter side) randomly sampled from [800, 1024], which reduces overfitting; inference is on a single scale of 1024 pixels. We use a mini-batch size of 1 image per GPU (so 8 on 8 GPUs) and train the model for 24k iterations, starting from a learning rate of 0.01 and reducing it to 0.001 at 18k iterations. It takes ~4 hours of training on a single 8-GPU machine under this setting.

Results: Table 7 compares our results to the state of the art on the val and test sets. *Without* using the coarse training set, our method achieves 26.2 AP on test, which is over 30% relative improvement over the previous best entry (DIN [3]), and is also better than the concurrent work of SGN's 25.0 [29]. Both DIN and SGN use fine + coarse data. Compared to the best entry using fine data only (17.4 AP), we achieve a ~50% improvement.

For the *person* and *car* categories, the Cityscapes dataset exhibits a large number of *within*-category overlapping instances (on average 6 people and 9 cars per image). We argue that *within-category overlap* is a core difficulty of instance segmentation. Our method shows massive improvement on these two categories over the other best entries (relative ~40% improvement on *person* from 21.8 to 30.5 and ~20% improvement on *car* from 39.4 to 46.9), even though our method does not exploit the coarse data.

A main challenge of the Cityscapes dataset is training models in a *low-data* regime, particularly for the categories of *truck*, *bus*, and *train*, which have about 200-500 training samples each. To partially remedy this issue, we further report a result using COCO pre-training. To do this, we initialize the corresponding 7 categories in Cityscapes from a pre-trained COCO Mask R-CNN model (*rider* being randomly initialized). We fine-tune this model for 4k iterations in which the learning rate is reduced at 3k iterations, which takes ~1 hour for training given the COCO model.

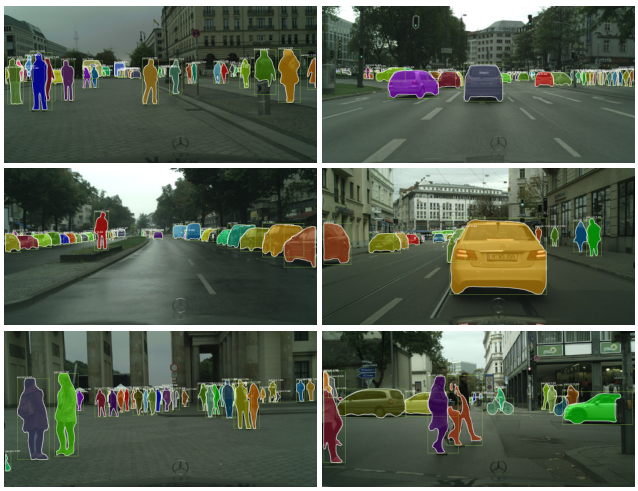

Figure 8. Mask R-CNN results on Cityscapes test (32.0 AP). The bottom-right image shows a failure prediction.

The COCO pre-trained Mask R-CNN model achieves 32.0 AP on test, almost a 6 point improvement over the fine-only counterpart. This indicates the important role the amount of training data plays. It also suggests that methods on Cityscapes might be influenced by their *low-shot* learning performance. We show that using COCO pre-training is an effective strategy on this dataset.

Finally, we observed a bias between the val and test AP, as is also observed from the results of [23, 4, 29]. We found that this bias is mainly caused by the *truck*, *bus*, and *train* categories, with the fine-only model having val/test AP of 28.8/22.8, 53.5/32.2, and 33.0/18.6, respectively. This suggests that there is a *domain shift* on these categories, which also have little training data. COCO pre-training helps to improve results the most on these categories; however, the domain shift persists with 38.0/30.1, 57.5/40.9, and 41.2/30.9 val/test AP, respectively. Note that for the *person* and *car* categories we do not see any such bias (val/test AP are within ±1 point).

Example results on Cityscapes are shown in Figure 8.

	training data	AP [val]	AP	AP_{50}	person	rider	car	truck	bus	train	mcycle	bicycle
InstanceCut [23]	fine + coarse	15.8	13.0	27.9	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
DWT [4]	fine	19.8	15.6	30.0	15.1	11.7	32.9	17.1	20.4	15.0	7.9	4.9
SAIS [17]	fine	-	17.4	36.7	14.6	12.9	35.7	16.0	23.2	19.0	10.3	7.8
DIN [3]	fine + coarse	-	20.0	38.8	16.5	16.7	25.7	20.6	30.0	23.4	17.1	10.1
SGN [29]	fine + coarse	29.2	25.0	44.9	21.8	20.1	39.4	24.8	33.2	30.8	17.7	12.4
Mask R-CNN	fine	31.5	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
Mask R-CNN	fine + COCO	**36.4**	**32.0**	**58.1**	**34.8**	**27.0**	**49.1**	**30.1**	**40.9**	**30.9**	**24.1**	**18.7**

表7. 在Cityscapes验证集（'AP [val]'列）和测试集（其余列）上的结果。我们的方法使用ResNet-50-FPN。

附录A：Cityscapes数据集上的实验

我们进一步报告了在Cityscapes [7]数据集上的实例分割结果。该数据集包含2975张训练图像、500张验证图像和1525张测试图像的精细标注。此外，还有2万张未标注实例的粗略训练图像，我们并未*not*使用。所有图像的分辨率均为2048×1024像素。实例分割任务涵盖8个对象类别，其在精细训练集上的实例数量如下：

行人	骑行者	汽车	卡车	公交车	火车	摩托车	自行车
17.9千	1.8千	26.9千	0.5千	0.4千	0.2千	0.7千	3.7千

在此任务中，实例分割性能通过COCO风格的掩码AP（在多个IoU阈值上取平均值）来衡量；同时报告了AP_{50} (*i.e*，以及IoU为0.5时的掩码AP)。

实现：我们采用ResNet-FPN-50作为主干网络的Mask R-CNN模型；由于数据集规模较小，我们发现101层的对应模型表现相似。训练时图像尺度（短边）从[800, 1024]中随机采样，这有助于减少过拟合；推理时采用1024像素的单尺度。我们使用每个GPU处理1张图像的小批量（因此在8个GPU上共8张），并训练模型24k次迭代，初始学习率为0.01，在18k次迭代时降至0.001。在此设置下，单台8-GPU机器需要~4小时的训练时间。

结果：表7将我们的结果与验证集和测试集上的最先进技术进行了比较。*Without*使用粗粒度训练集，我们的方法在测试集上达到了26.2 AP，相比之前的最佳方案（DIN [3]）有超过30%的相对提升，同时也优于同期SGN的25.0 AP [29]。DIN和SGN均使用了细粒度+粗粒度数据。与仅使用细粒度数据的最佳方案（17.4 AP）相比，我们实现了~50%的提升。

对于*person*和*car*类别，Cityscapes数据集存在大量*within*类别的重叠实例（平均每张图像包含6个人和9辆车）。我们认为*within-category overlap*是实例分割的核心难点。尽管我们的方法未利用粗糙数据，但在这两个类别上相比其他最佳方案仍展现出显著提升（*person*相对提升~40%，从21.8提高至30.5；*car*相对提升~20%，从39.4提高至46.9）。

Cityscapes数据集的一个主要挑战是在*low-data*机制下训练模型，特别是对于*truck*、*bus*和*train*这些类别，它们仅有约200-500张训练

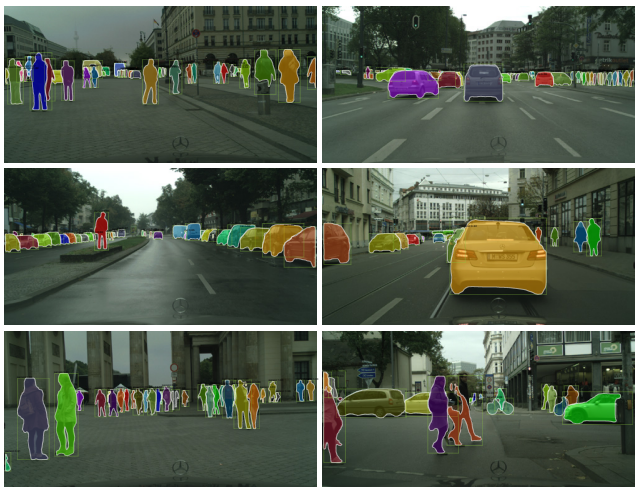

图8. Mask R-CNN在Cityscapes测试集上的结果（32.0 AP）。右下角图像展示了一个预测失败的案例。

每个样本。为了部分解决这个问题，我们进一步报告了使用COCO预训练的结果。为此，我们从预训练的COCO Mask R-CNN模型（*rider*为随机初始化）中初始化Cityscapes中对应的7个类别。我们将该模型微调4k次迭代，其中学习率在3k次迭代时降低，在给定COCO模型的情况下，训练耗时~1小时。

COCO预训练的Mask R-CNN模型在测试集上达到32.0 AP，比仅使用精细标注的模型提升近6个点。这表明训练数据量具有重要作用，同时也意味着Cityscapes数据集上的方法性能可能受其*low-shot*学习表现影响。我们证明在该数据集上采用COCO预训练是有效的策略。

最后，我们观察到验证集与测试集AP之间存在偏差，这也与[23, 4, 29]的结果一致。我们发现这种偏差主要由*truck*、*bus*和*train*类别引起，仅使用精细标注的模型在这些类别上的验证/测试AP分别为28.8/22.8、53.5/32.2和33.0/18.6。这表明这些类别存在*domain shift*现象，且它们对应的训练数据也较少。COCO预训练对这些类别的提升效果最显著，但域偏移问题依然存在，其验证/测试AP分别为38.0/30.1、57.5/40.9和41.2/30.9。值得注意的是，在*person*和*car*类别中我们未观察到此类偏差（验证/测试AP差异在±1个百分点以内）。

Cityscapes上的示例结果如图8所示。

description	backbone	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
original baseline	X-101-FPN	36.7	59.5	38.9	39.6	61.5	43.2
+ updated baseline	X-101-FPN	37.0	59.7	39.0	40.5	63.0	43.7
+ e2e training	X-101-FPN	37.6	60.4	39.9	41.7	64.1	45.2
+ ImageNet-5k	X-101-FPN	38.6	61.7	40.9	42.7	65.1	46.6
+ train-time augm.	X-101-FPN	39.2	62.5	41.6	43.5	65.9	47.2
+ deeper	X-152-FPN	39.7	63.2	42.2	44.1	66.4	48.4
+ Non-local [43]	X-152-FPN-NL	**40.3**	**64.4**	**42.8**	**45.0**	**67.8**	**48.9**
+ test-time augm.	X-152-FPN-NL	**41.8**	**66.0**	**44.8**	**47.3**	**69.3**	**51.5**

Table 8. **Enhanced detection results** of Mask R-CNN on COCO `minival`. Each row adds an extra component to the above row. We denote ResNeXt model by 'X' for notational brevity.

description	backbone	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
original baseline	R-50-FPN	64.2	86.6	69.7	58.7	73.0
+ updated baseline	R-50-FPN	65.1	86.6	70.9	59.9	73.6
+ deeper	R-101-FPN	66.1	87.7	71.7	60.5	75.0
+ ResNeXt	X-101-FPN	67.3	88.0	73.3	62.2	75.6
+ data distillation [35]	X-101-FPN	**69.1**	**88.9**	**75.3**	**64.1**	**77.1**
+ test-time augm.	X-101-FPN	**70.4**	**89.3**	**76.8**	**65.8**	**78.1**

Table 9. **Enhanced keypoint results** of Mask R-CNN on COCO `minival`. Each row adds an extra component to the above row. Here we use only keypoint annotations but no mask annotations. We denote ResNet by 'R' and ResNeXt by 'X' for brevity.

Appendix B: Enhanced Results on COCO

As a general framework, Mask R-CNN is compatible with complementary techniques developed for detection/segmentation, including improvements made to Fast/Faster R-CNN and FCNs. In this appendix we describe some techniques that improve over our original results. Thanks to its generality and flexibility, Mask R-CNN was used as the framework by the three winning teams in the COCO 2017 instance segmentation competition, which all significantly outperformed the previous state of the art.

Instance Segmentation and Object Detection

We report some enhanced results of Mask R-CNN in Table 8. Overall, the improvements increase mask AP 5.1 points (from 36.7 to 41.8) and box AP 7.7 points (from 39.6 to 47.3). Each model improvement increases both mask AP and box AP consistently, showing good generalization of the Mask R-CNN framework. We detail the improvements next. These results, along with future updates, can be reproduced by our released code at `https://github.com/facebookresearch/Detectron`, and can serve as higher baselines for future research.

Updated baseline: We start with an updated baseline with a different set of hyper-parameters. We lengthen the training to 180k iterations, in which the learning rate is reduced by 10 at 120k and 160k iterations. We also change the NMS threshold to 0.5 (from a default value of 0.3). The updated baseline has 37.0 mask AP and 40.5 box AP.

End-to-end training: All previous results used stage-wise training, *i.e.*, training RPN as the first stage and Mask R-CNN as the second. Following [37], we evaluate end-to-end ('e2e') training that jointly trains RPN and Mask R-CNN. We adopt the 'approximate' version in [37] that only computes partial gradients in the RoIAlign layer by ignoring the gradient w.r.t. RoI coordinates. Table 8 shows that e2e training improves mask AP by 0.6 and box AP by 1.2.

ImageNet-5k pre-training: Following [45], we experiment with models pre-trained on a 5k-class subset of ImageNet (in contrast to the standard 1k-class subset). This 5× increase in pre-training data improves both mask and box 1 AP. As a reference, [40] used ~250× more images (300M) and reported a 2-3 box AP improvement on their baselines.

Train-time augmentation: Scale augmentation at train time further improves results. During training, we randomly sample a scale from [640, 800] pixels and we increase the number of iterations to 260k (with the learning rate reduced by 10 at 200k and 240k iterations). Train-time augmentation improves mask AP by 0.6 and box AP by 0.8.

Model architecture: By upgrading the 101-layer ResNeXt to its 152-layer counterpart [19], we observe an increase of 0.5 mask AP and 0.6 box AP. This shows a deeper model can still improve results on COCO.

Using the recently proposed *non-local* (NL) model [43], we achieve 40.3 mask AP and 45.0 box AP. This result is without test-time augmentation, and the method runs at 3fps on an Nvidia Tesla P100 GPU at test time.

Test-time augmentation: We combine the model results evaluated using scales of [400, 1200] pixels with a step of 100 and on their horizontal flips. This gives us a single-model result of 41.8 mask AP and 47.3 box AP.

The above result is the foundation of our submission to the COCO 2017 competition (which also used an ensemble, not discussed here). The first three winning teams for the instance segmentation task were all reportedly based on an extension of the Mask R-CNN framework.

Keypoint Detection

We report enhanced results of keypoint detection in Table 9. As an updated baseline, we extend the training schedule to 130k iterations in which the learning rate is reduced by 10 at 100k and 120k iterations. This improves AP^{kp} by about 1 point. Replacing ResNet-50 with ResNet-101 and ResNeXt-101 increases AP^{kp} to 66.1 and 67.3, respectively.

With a recent method called *data distillation* [35], we are able to exploit the additional 120k *unlabeled* images provided by COCO. In brief, data distillation is a self-training strategy that uses a model trained on labeled data to predict annotations on unlabeled images, and in turn updates the model with these new annotations. Mask R-CNN provides an effective framework for such a self-training strategy. With data distillation, Mask R-CNN AP^{kp} improve by 1.8 points to 69.1. We observe that Mask R-CNN can benefit from extra data, even if that data is *unlabeled*.

By using the same test-time augmentation as used for instance segmentation, we further boost AP^{kp} to 70.4.

description	backbone	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
original baseline	X-101-FPN	36.7	59.5	38.9	39.6	61.5	43.2
+ updated baseline	X-101-FPN	37.0	59.7	39.0	40.5	63.0	43.7
+ e2e training	X-101-FPN	37.6	60.4	39.9	41.7	64.1	45.2
+ ImageNet-5k	X-101-FPN	38.6	61.7	40.9	42.7	65.1	46.6
+ train-time augm.	X-101-FPN	39.2	62.5	41.6	43.5	65.9	47.2
+ deeper	X-152-FPN	39.7	63.2	42.2	44.1	66.4	48.4
+ Non-local [43]	X-152-FPN-NL	**40.3**	**64.4**	**42.8**	**45.0**	**67.8**	**48.9**
+ test-time augm.	X-152-FPN-NL	**41.8**	**66.0**	**44.8**	**47.3**	**69.3**	**51.5**

表8. Mask R-CNN在COCO minival上的增强检测结果。每一行都在上一行的基础上增加了一个额外组件。为简洁起见，我们用“X”表示ResNeXt模型。

附录B：COCO数据集上的增强结果

作为一个通用框架，Mask R-CNN 兼容为检测/分割开发的互补技术，包括对 Fast/Faster R-CNN 和 FCNs 的改进。在本附录中，我们描述了一些超越我们原始结果的技术。得益于其通用性和灵活性，Mask R-CNN 被 COCO 2017 实例分割竞赛的三支获胜团队用作框架，这些团队的表现均显著超越了以往的最先进水平。

实例分割与目标检测

我们在表8中报告了Mask R-CNN的一些增强结果。总体而言，改进使掩码AP提升了5.1点（从36.7到41.8），边界框AP提升了7.7点（从39.6到47.3）。每项模型改进都持续提升了掩码AP和边界框AP，显示了Mask R-CNN框架良好的泛化能力。接下来我们将详细介绍这些改进。这些结果以及未来的更新，均可通过我们发布的代码（https://github.com/facebookresearch/Detectron）复现，并可作为未来研究的更高基准。

Updated baseline: 我们从一个更新后的基线开始，采用了一组不同的超参数。我们将训练延长至18万次迭代，其中学习率在第12万次和第16万次迭代时降低为原来的十分之一。同时，我们将非极大值抑制（NMS）的阈值调整为0.5（默认值为0.3）。更新后的基线达到了37.0的掩码平均精度（mask AP）和40.5的边界框平均精度（box AP）。

End-to-end training: 所有先前的结果都采用分阶段训练，*i.e*，即第一阶段训练RPN，第二阶段训练Mask R-CNN。遵循[37]的方法，我们评估了端到端（'e2e'）训练，该训练联合训练RPN和Mask R-CNN。我们采用[37]中的“近似”版本，该版本在RoIAlign层中仅计算部分梯度，忽略RoI坐标的梯度。表8显示，e2e训练将掩码AP提高了0.6，边界框AP提高了1.2。

ImageNet-5k pre-training: 根据[45]的研究，我们尝试使用在ImageNet的5k类别子集（相较于标准的1k类别子集）上预训练的模型进行实验。预训练数据量提升至5×倍，同时提升了掩码和边界框1 AP指标。作为参考，[40]使用了~250×倍更多的图像（3亿张），并在其基线模型上报告了2-3边界框AP的改进。

description	backbone	AP^{kp}	AP^{kp}_{50}	AP^{kp}_{75}	AP^{kp}_{M}	AP^{kp}_{L}
original baseline	R-50-FPN	64.2	86.6	69.7	58.7	73.0
+ updated baseline	R-50-FPN	65.1	86.6	70.9	59.9	73.6
+ deeper	R-101-FPN	66.1	87.7	71.7	60.5	75.0
+ ResNeXt	X-101-FPN	67.3	88.0	73.3	62.2	75.6
+ data distillation [35]	X-101-FPN	**69.1**	**88.9**	**75.3**	**64.1**	**77.1**
+ test-time augm.	X-101-FPN	**70.4**	**89.3**	**76.8**	**65.8**	**78.1**

表9. Mask R-CNN在COCO minival上增强的关键点检测结果。每一行在上一行的基础上增加一个组件。此处我们仅使用关键点标注，不使用掩码标注。为简洁起见，我们用“R”表示ResNet，用“X”表示ResNeXt。

Train-time augmentation: 训练时的尺度增强进一步改善了结果。在训练过程中，我们随机从[640, 800]像素范围内采样一个尺度，并将迭代次数增加到260k（学习率在200k和240k迭代时降低10倍）。训练时的增强使掩码AP提升了0.6，边界框AP提升了0.8。

Model architecture: 通过将101层的ResNeXt升级至其152层的对应版本[19]，我们观察到掩码AP提升了0.5，边界框AP提升了0.6。这表明更深的模型仍能在COCO数据集上提升效果。

使用最近提出的*non-local* (NL)模型[43]，我们实现了40.3的掩码AP和45.0的边界框AP。该结果未使用测试时增强，且该方法在测试时于Nvidia Tesla P100 GPU上以3fps的速度运行。

Test-time augmentation: 我们将模型在[400, 1200]像素尺度范围内以100为步长评估的结果与其水平翻转版本的结果相结合，由此得到单一模型下的评估指标：掩码AP为41.8，边界框AP为47.3。

上述结果是我们提交给COCO 2017竞赛的基础（该提交也使用了集成方法，此处不作讨论）。据悉，实例分割任务的前三名获胜团队均基于Mask R-CNN框架的扩展实现。

关键点检测

我们在表9中报告了关键点检测的增强结果。作为更新的基线，我们将训练计划延长至13万次迭代，其中学习率在第10万次和第12万次迭代时降低为原来的十分之一。这使得AP^{kp}提升了约1个百分点。将ResNet-50替换为ResNet-101和ResNeXt-101后，AP^{kp}分别提升至66.1和67.3。

借助最近提出的*data distillation*方法[35]，我们得以利用COCO提供的额外12万张*unlabeled*图像。简而言之，数据蒸馏是一种自训练策略：首先使用已标注数据训练的模型预测未标注图像的注释，再利用这些新注释更新模型。Mask R-CNN为此类自训练策略提供了高效框架。通过数据蒸馏，Mask R-CNN的AP^{kp}提升了1.8个点至69.1。我们观察到，即使数据是*unlabeled*，Mask R-CNN仍能从额外数据中获益。

通过采用与实例分割相同的测试时增强方法，我们进一步将AP^{kp}提升至70.4。

Acknowledgements: We would like to acknowledge Ilija Radosavovic for contributions to code release and enhanced results, and the Caffe2 team for engineering support.

References

[1] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 8

[2] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014. 2

[3] A. Arnab and P. H. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *CVPR*, 2017. 3, 9

[4] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017. 3, 9

[5] 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. 5

[6] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 7, 8

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 9

[8] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016. 2

[9] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015. 2

[10] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016. 2, 3, 4, 5, 6

[11] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 2

[12] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 1, 2, 3, 4, 6

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2, 3

[14] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *CVPR*, 2015. 4

[15] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*. 2014. 2

[16] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015. 2

[17] Z. Hayder, X. He, and M. Salzmann. Shape-aware instance segmentation. In *CVPR*, 2017. 9

[18] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*. 2014. 1, 2

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 4, 7, 10

[20] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *PAMI*, 2015. 2

[21] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017. 2, 3, 4, 6, 7

[22] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015. 4

[23] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: from edges to instances with multicut. In *CVPR*, 2017. 3, 9

[24] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 2

[25] 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. 2

[26] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017. 2, 3, 5, 6

[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 4, 5, 7

[28] 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. 2, 5

[29] S. Liu, J. Jia, S. Fidler, and R. Urtasun. SGN: Sequential grouping networks for instance segmentation. In *ICCV*, 2017. 3, 9

[30] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 3, 6

[31] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 4

[32] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multi-person pose estimation in the wild. In *CVPR*, 2017. 8

[33] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to segment object candidates. In *NIPS*, 2015. 2, 3

[34] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016. 2, 3

[35] I. Radosavovic, P. Dollár, R. Girshick, G. Gkioxari, and K. He. Data distillation: Towards omni-supervised learning. *arXiv:1712.04440*, 2017. 10

[36] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2, 3, 4, 7

[37] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *TPAMI*, 2017. 10

[38] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016. 2, 5

[39] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv:1612.06851*, 2016. 4, 7

[40] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017. 10

致谢：我们要感谢Ilija Radosavovic在代码发布和结果优化方面的贡献，以及Caffe2团队提供的工程支持。

参考文献

[1] M. Andriluka, L. Pishchulin, P. Gehler, 与 B. Schiele。二维人体姿态估计：新基准与前沿分析。发表于 *CVPR*, 2014. 8[2] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, 与 J. Malik。多尺度组合分组。发表于 *CVPR*, 2014. 2[3] A. Arnab 与 P. H. Torr。使用动态实例化网络的逐像素实例分割。发表于 *CVPR*, 2017. 3, 9[4] M. Bai 与 R. Urtasun。用于实例分割的深度分水岭变换。发表于 *CVPR*, 2017. 3, 9[5] S. Bell, C. L. Zitnick, K. Bala, 与 R. Girshick。内外网：通过跳跃池化和循环神经网络在上下文中检测物体。发表于 *CVPR*, 2016. 5[6] Z. Cao, T. Simon, S.-E. Wei, 与 Y. Sheikh。使用部位亲和力场的实时多人二维姿态估计。发表于 *CVPR*, 2017. 7, 8[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, 与 B. Schiele。用于语义城市场景理解的 Cityscapes 数据集。发表于 *CVPR*, 2016. 9[8] J. Dai, K. He, Y. Li, S. Ren, 与 J. Sun。实例敏感的完全卷积网络。发表于 *ECCV*, 2016. 2[9] J. Dai, K. He, 与 J. Sun。用于联合物体与背景分割的卷积特征掩蔽。发表于 *CVPR*, 2015. 2[10] J. Dai, K. He, 与 J. Sun。通过多任务网络级联的实例感知语义分割。发表于 *CVPR*, 2016. 2, 3, 4, 5, 6[11] J. Dai, Y. Li, K. He, 与 J. Sun。R-FCN：基于区域的完全卷积网络进行物体检测。发表于 *NIPS*, 2016. 2[12] R. Girshick。Fast R-CNN。发表于 *ICCV*, 2015. 1, 2, 3, 4, 6[13] R. Girshick, J. Donahue, T. Darrell, 与 J. Malik。用于精确物体检测与语义分割的丰富特征层次结构。发表于 *CVPR*, 2014. 2, 3[14] R. Girshick, F. Iandola, T. Darrell, 与 J. Malik。可变形部件模型即卷积神经网络。发表于 *CVPR*, 2015. 4[15] B. Hariharan, P. Arbeláez, R. Girshick, 与 J. Malik。同步检测与分割。发表于 *ECCV*. 2014. 2[16] B. Hariharan, P. Arbeláez, R. Girshick, 与 J. Malik。用于物体分割与细粒度定位的超列。发表于 *CVPR*, 2015. 2[17] Z. Hayder, X. He, 与 M. Salzmann。形状感知的实例分割。发表于 *CVPR*, 2017. 9[18] K. He, X. Zhang, S. Ren, 与 J. Sun。深度卷积网络中用于视觉识别的空间金字塔池化。发表于 *ECCV*. 2014. 1, 2[19] K. He, X. Zhang, S. Ren, 与 J. Sun。用于图像识别的深度残差学习。发表于 *CVPR*, 2016. 2, 4, 7, 10[20] J. Hosang, R. Benenson, P. Dollár, 与 B. Schiele。是什么造就了有效的检测提议？*PAMI*, 2015. 2

[21] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, 等。现代卷积目标检测器的速度/精度权衡。发表于 *CVPR*，2017年。2, 3, 4, 6, 7[22] M. Jaderberg, K. Simonyan, A. Zisserman, 和 K. Kavukcuoglu。空间变换网络。发表于 *NIPS*，2015年。4[23] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, 和 C. Rother。Instancecut：从边缘到实例的多割方法。发表于 *CVPR*，2017年。3, 9[24] A. Krizhevsky, I. Sutskever, 和 G. Hinton。使用深度卷积神经网络进行ImageNet分类。发表于 *NIPS*，2012年。2[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, 和 L. D. Jackel。反向传播应用于手写邮政编码识别。*Neural compu- tation*，1989年。2[26] Y. Li, H. Qi, J. Dai, X. Ji, 和 Y. Wei。全卷积实例感知语义分割。发表于 *CVPR*，2017年。2, 3, 5, 6[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, 和 S. Belongie。用于目标检测的特征金字塔网络。发表于 *CVPR*，2017年。2, 4, 5, 7[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, 和 C. L. Zitnick。Microsoft COCO：上下文中的常见物体。发表于 *ECCV*，2014年。2, 5[29] S. Liu, J. Jia, S. Fidler, 和 R. Urtasun。SGN：用于实例分割的顺序分组网络。发表于 *ICCV*，2017年。3, 9[30] J. Long, E. Shelhamer, 和 T. Darrell。用于语义分割的全卷积网络。发表于 *CVPR*，2015年。1, 3, 6[31] V. Nair 和 G. E. Hinton。修正线性单元改进受限玻尔兹曼机。发表于 *ICML*，2010年。4[32] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Thompson, C. Bregler, 和 K. Murphy。面向野外精确多人姿态估计。发表于 *CVPR*，2017年。8[33] P. O. Pinheiro, R. Collobert, 和 P. Dollar。学习分割候选物体。发表于 *NIPS*，2015年。2, 3[34] P. O. Pinheiro, T.-Y. Lin, R. Collobert, 和 P. Dollár。学习优化物体分割。发表于 *ECCV*，2016年。2, 3[35] I. Radosavovic, P. Dollár, R. Girshick, G. Gkioxari, 和 K. He。数据蒸馏：迈向全监督学习。*arXiv:1712.04440*，2017年。10[36] S. Ren, K. He, R. Girshick, 和 J. Sun。Faster R-CNN：利用区域提议网络实现实时目标检测。发表于 *NIPS*，2015年。1, 2, 3, 4, 7[37] S. Ren, K. He, R. Girshick, 和 J. Sun。Faster R-CNN：利用区域提议网络实现实时目标检测。发表于 *TPAMI*，2017年。10[38] A. Shrivastava, A. Gupta, 和 R. Girshick。使用在线难例挖掘训练基于区域的目标检测器。发表于 *CVPR*，2016年。2, 5[39] A. Shrivastava, R. Sukthankar, J. Malik, 和 A. Gupta。超越跳跃连接：用于目标检测的自顶向下调制。*arXiv:1612.06851*，2016年。4, 7[40] C. Sun, A. Shrivastava, S. Singh, 和 A. Gupta。重探深度学习时代数据的不合理有效性。发表于 *ICCV*，2017年。10

[41] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. In *ICLR Workshop*, 2016. 7

[42] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 2013. 2

[43] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. *arXiv:1711.07971*, 2017. 10

[44] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016. 8

[45] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 4, 10

[41] C. Szegedy, S. Ioffe, V. Vanhoucke. Inception-v4, Inception -ResNet与残差连接对学习的影响。发表于*ICLR Workshop*, 20 16. 7[42] J. R. Uijlings, K. E. van de Sande, T. Gevers, A. W. Sm eulders. 面向物体识别的选择性搜索。发表于*IJCV*, 2013. 2[43] X. Wang, R. Girshick, A. Gupta, K. He. 非局部神经网络。发 表于*arXiv:1711.07971*, 2017. 10[44] S.-E. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh. 卷积姿态机。发表于*CVPR*, 2016. 8[45] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He. 深度神经网络的聚 合残差变换。发表于*CVPR*, 2017. 4, 10