**Mask R-CNN**

Kaiming He,Georgia Gkioxari, Piotr Dollar, Ross Girshick

**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, boundingbox 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..

객체 인스턴스 분할을 위한 개념적으로 간단하고 유연하며 일반적인 프레임워크를 제시합니다. 우리의 접근 방식은 이미지에서 객체를 효율적으로 감지하는 동시에 각 인스턴스에 대해 고품질 분할 마스크를 생성합니다. Mask R-CNN이라고 하는 방법은 기존의 bounding box 인식 브랜치와 병렬로 객체 마스크를 예측하는 브랜치를 추가하여 Faster R-CNN을 확장합니다. Mask R-CNN은 훈련이 간단하고 5fps에서 실행되는 Faster R-CNN에 약간의 오버헤드만 추가합니다. 또한 Mask R-CNN은 다른 작업으로 일반화하기 쉽습니다. 예를 들어 동일한 프레임워크에서 사람의 자세를 추정할 수 있습니다. 인스턴스 분할, 경계 상자 객체 감지 및 사람 키포인트 감지를 포함하여 COCO 챌린지의 세 가지 트랙 모두에서 최고의 결과를 보여줍니다. 종소리와 휘파람이없는 Mask R-CNN은 COCO 2016 챌린지 우승자를 포함하여 모든 작업에서 기존의 모든 단일 모델 항목을 능가합니다. 우리의 간단하고 효과적인 접근 방식이 견고한 기준이 되고 인스턴스 수준 인식에 대한 향후 연구를 용이하게 하는 데 도움이 되기를 바랍니다. 코드는 https://github.com/

facebookresearch/Detectron..에서 사용할 수 있습니다.

**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 RCNN [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.

비전 커뮤니티는 단기간에 객체 감지 및 의미론적 세분화 결과를 빠르게 개선했습니다. 대부분 이러한 발전은 각각 객체 감지 및 의미론적 분할을 위한 Fast/Faster RCNN[12, 36] 및 FCN(Fully Convolutional Network)[30] 프레임워크와 같은 강력한 기준 시스템에 의해 주도되었습니다. 이러한 방법은 개념적으로 직관적이며 빠른 훈련 및 추론 시간과 함께 유연성과 견고성을 제공합니다. 이 작업에서 우리의 목표는 인스턴스 분할을 위한 비교적 가능한 프레임워크를 개발하는 것입니다.

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.

인스턴스 분할은 이미지의 모든 객체를 올바르게 감지하는 동시에 각 인스턴스를 정확하게 분할해야 하기 때문에 어렵습니다. 따라서 개체 감지의 고전적인 컴퓨터 비전 작업의 요소를 결합합니다. 여기서 목표는 경계 상자를 사용하여 개별 개체를 분류하고 각 개체를 지역화하는 것이며, 목표는 각 픽셀을 구별하지 않고 고정된 범주 집합으로 분류하는 것입니다. 개체 인스턴스.1 이를 감안할 때 좋은 결과를 얻으려면 복잡한 방법이 필요할 것으로 예상할 수 있습니다. 그러나 놀랍도록 간단하고 유연하며 빠른 시스템이 이전의 최첨단 인스턴스 분할 결과를 능가할 수 있음을 보여줍니다.

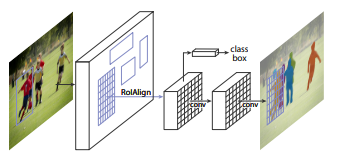


Figure 1. The Mask R-CNN framework for instance segmentation.

그림 1. 인스턴스 분할을 위한 Mask R-CNN 프레임워크.

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-topixel 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.

Mask R-CNN이라고 하는 우리의 방법은 분류 및 경계 상자 회귀를 위한 기존 분기와 병렬로 각 관심 영역(RoI)에서 분할 마스크를 예측하기 위한 분기를 추가하여 Faster R-CNN을 확장합니다(그림 1). . 마스크 분기는 각 RoI에 적용되는 작은 FCN으로, 픽셀 대 픽셀 방식으로 분할 마스크를 예측합니다. Mask R-CNN은 광범위한 유연한 아키텍처 설계를 용이하게 하는 Faster R-CNN 프레임워크를 감안할 때 구현 및 교육이 간단합니다. 또한 마스크 분기는 작은 계산 오버헤드만 추가하므로 빠른 시스템과 빠른 실험이 가능합니다.

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 RCNN 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 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.

원칙적으로 Mask R-CNN은 Faster R-CNN의 직관적인 확장이지만 좋은 결과를 얻으려면 마스크 분기를 적절하게 구성하는 것이 중요합니다. 가장 중요한 것은 Faster RCNN은 네트워크 입력과 출력 간의 픽셀 대 픽셀 정렬을 위해 설계되지 않았다는 것입니다. 이것은 인스턴스를 처리하기 위한 사실상의 핵심 연산인 RoIPool[18, 12]이 특징 추출을 위해 거친 공간 양자화를 수행하는 방법에서 가장 분명합니다. 오정렬을 수정하기 위해 정확한 공간 위치를 충실하게 보존하는 RoIAlign이라는 단순하고 양자화가 없는 레이어를 제안합니다. 겉보기에 사소한 변경에도 불구하고 RoIAlign은 큰 영향을 미칩니다. 마스크 정확도를 상대적으로 10%에서 50% 향상시켜 더 엄격한 현지화 측정 기준에서 더 큰 이득을 보여줍니다. 둘째, 마스크와 클래스 예측을 분리하는 것이 필수적이라는 것을 발견했습니다. 클래스 간의 경쟁 없이 독립적으로 각 클래스에 대한 이진 마스크를 예측하고 네트워크의 RoI 분류 분기에 의존하여 범주를 예측합니다. 대조적으로, FCN은 일반적으로 분할과 분류를 결합하는 픽셀당 다중 클래스 분류를 수행하며 우리의 실험에 따르면 인스턴스 분할에서는 제대로 작동하지 않습니다.

텍스트, 실외, 다채로운이(가) 표시된 사진

자동 생성된 설명

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.

그림 2. COCO 테스트 세트에서 마스크 R-CNN 결과. 이 결과는 ResNet-101[19]을 기반으로 하며 35.7의 마스크 AP를 달성하고 5fps에서 실행됩니다. 마스크는 색상으로 표시되며 경계 상자, 범주 및 신뢰도도 표시됩니다.

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 heavilyengineered 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.

종소리와 호루라기 없이 Mask R-CNN은 2016년 대회 우승자의 엔지니어링 항목을 포함하여 COCO 인스턴스 분할 작업[28]에서 이전의 모든 최첨단 단일 모델 결과를 능가합니다. 부산물로서 우리의 방법은 COCO 물체 감지 작업에서도 탁월합니다. 절제 실험에서 여러 기본 인스턴스화를 평가하여 견고성을 입증하고 핵심 요소의 효과를 분석할 수 있습니다.

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.

우리 모델은 GPU에서 프레임당 약 200ms로 실행할 수 있으며 COCO에 대한 교육은 단일 8-GPU 시스템에서 1~2일이 소요됩니다. 우리는 프레임워크의 유연성 및 정확성과 함께 빠른 훈련 및 테스트 속도가 인스턴스 분할에 대한 향후 연구에 도움이 되고 용이할 것이라고 믿습니다.

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.

마지막으로 COCO 키포인트 데이터 세트[28]에서 인간 포즈 추정 작업을 통해 프레임워크의 일반성을 보여줍니다. 각 키포인트를 원-핫 바이너리 마스크로 보고 최소한의 수정으로 마스크 R-CNN을 적용하여 인스턴스별 포즈를 감지할 수 있습니다. Mask R-CNN은 2016 COCO 키포인트 대회 우승자를 제치고 동시에 5fps로 실행됩니다. 따라서 Mask R-CNN은 인스턴스 수준 인식을 위한 유연한 프레임워크로 더 광범위하게 볼 수 있으며 더 복잡한 작업으로 쉽게 확장될 수 있습니다.

We have released code to facilitate future research.

향후 연구를 용이하게 하기 위해 코드를 릴리스했습니다.

**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.

R-CNN: 경계 상자 객체 감지에 대한 지역 기반 CNN(R-CNN) 접근 방식[13]은 관리 가능한 수의 후보 객체 영역에 주의를 기울이고[42, 20] 컨볼루션 네트워크[25, 24]를 독립적으로 평가하는 것입니다. 각 ROI에 대해 R-CNN이 확장되어 [18, 12] RoIPool을 사용하여 기능 맵에서 RoI에 주의를 기울일 수 있으므로 빠른 속도와 정확도가 향상됩니다. Faster R-CNN[36]은 RPN(Region Proposal Network)으로 주의 메커니즘을 학습하여 이 스트림을 발전시켰습니다. Faster R-CNN은 많은 후속 개선 사항(예: [38, 27, 21])에 대해 유연하고 강력하며 여러 벤치마크에서 현재 최고의 프레임워크입니다.

Instance Segmentation: Driven by the effectiveness of RCNN, 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.

인스턴스 분할: RCNN의 효율성에 힘입어 인스턴스 분할에 대한 많은 접근 방식은 세그먼트 제안을 기반으로 합니다. 이전 방법[13, 15, 16, 9]은 상향식 세그먼트[42, 2]에 의존했습니다. DeepMask [33] 및 다음 작업 [34, 8]은 세그먼트 후보를 제안한 다음 Fast R-CNN에 의해 ​​분류되는 방법을 학습합니다. 이러한 방법에서는 분할이 인식보다 우선하므로 느리고 정확도가 떨어집니다. 마찬가지로, Dai et al. [10]은 경계 상자 제안에서 세그먼트 제안을 예측한 후 분류하는 복잡한 다단계 캐스케이드를 제안했습니다. 대신 우리의 방법은 마스크와 클래스 레이블의 병렬 예측을 기반으로 하며 이는 더 간단하고 유연합니다.

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 positionsensitive 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.

가장 최근에는 Li et al. [26]은 "완전한 컨볼루션 인스턴스 분할"(FCIS)을 위해 [8]의 세그먼트 제안 시스템과 [11]의 객체 감지 시스템을 결합했습니다. [8, 11, 26]의 일반적인 아이디어는 위치 감지 출력 채널 세트를 완전히 컨볼루션으로 예측하는 것입니다. 이러한 채널은 객체 클래스, 상자 및 마스크를 동시에 처리하여 시스템을 빠르게 만듭니다. 그러나 FCIS는 중복되는 인스턴스에 대해 시스템적 오류를 나타내고 가상 가장자리를 생성하여(그림 6), 인스턴스를 분할하는 근본적인 어려움에 직면해 있음을 보여줍니다.

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.

인스턴스 분할에 대한 또 다른 솔루션 제품군[23, 4, 3, 29]은 의미론적 분할의 성공에 의해 주도됩니다. 픽셀별 분류 결과(예: FCN 출력)에서 시작하여 이러한 방법은 동일한 범주의 픽셀을 다른 인스턴스로 자르려고 시도합니다. 이러한 방법의 세분화 우선 전략과 달리 Mask R-CNN은 인스턴스 우선 전략을 기반으로 합니다. 앞으로 두 전략의 더 깊은 통합이 연구될 것으로 기대합니다.

**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.

Mask R-CNN은 개념적으로 간단합니다. Faster R-CNN은 각 후보 객체에 대해 두 개의 출력, 즉 클래스 레이블과 경계 상자 오프셋을 가지고 있습니다. 여기에 객체 마스크를 출력하는 세 번째 분기를 추가합니다. 따라서 Mask R-CNN은 자연스럽고 직관적인 아이디어입니다. 그러나 추가 마스크 출력은 클래스 및 상자 출력과 구별되므로 개체의 훨씬 더 미세한 공간 레이아웃을 추출해야 합니다. 다음으로 Fast/Faster R-CNN의 주요 누락 부분인 픽셀 대 픽셀 정렬을 포함하여 Mask R-CNN의 핵심 요소를 소개합니다.

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.

Faster R-CNN: Faster R-CNN 검출기[36]를 간략하게 검토하는 것으로 시작합니다. Faster R-CNN은 두 단계로 구성됩니다. RPN(Region Proposal Network)이라고 하는 첫 번째 단계에서는 후보 객체 경계 상자를 제안합니다. 본질적으로 Fast R-CNN[12]인 두 번째 단계는 각 후보 상자에서 RoIPool을 사용하여 특징을 추출하고 분류 및 경계 상자 회귀를 수행합니다. 더 빠른 추론을 위해 두 단계에서 사용되는 기능을 공유할 수 있습니다. Faster R-CNN과 다른 프레임워크 간의 포괄적인 최신 비교를 위해 [21]을 참조하십시오.

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]).

Mask R-CNN: Mask R-CNN은 동일한 첫 번째 단계(RPN)와 함께 동일한 2단계 절차를 채택합니다. 두 번째 단계에서는 클래스 및 상자 오프셋을 예측하는 것과 동시에 Mask R-CNN도 각 RoI에 대해 이진 마스크를 출력합니다. 이것은 분류가 마스크 예측에 의존하는 가장 최근의 시스템과 대조적입니다(예: [33, 10, 26]). 우리의 접근 방식은 경계 상자 분류 및 회귀를 병렬로 적용하는 Fast R-CNN[12]의 정신을 따릅니다(이는 원래 R-CNN의 다단계 파이프라인[13]을 크게 단순화하는 것으로 판명됨).

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

공식적으로 훈련 중에 샘플링된 각 RoI에 대한 다중 작업 손실을 로 정의합니다. 분류 손실 및 경계 상자 손실 는 [12]에서 정의한 것과 동일합니다. 마스크 분기는 각 RoI에 대한 Km^2 차원 출력을 가지며, 이는 K 클래스 각각에 대해 하나씩 해상도 m × m의 K 이진 마스크를 인코딩합니다. 이를 위해 픽셀당 시그모이드를 적용하고 를 평균 이진 교차 엔트로피 손실로 정의합니다. ground-truth class k와 관련된 RoI의 경우, 는 k번째 마스크에만 정의됩니다(다른 마스크 출력은 손실에 기여하지 않습니다).

Our definition of Lmask 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.

Lmask의 정의를 통해 네트워크는 클래스 간 경쟁 없이 모든 클래스에 대한 마스크를 생성할 수 있습니다. 출력 마스크를 선택하는 데 사용되는 클래스 레이블을 예측하기 위해 전용 분류 분기에 의존합니다. 이것은 마스크와 클래스 예측을 분리합니다. 이것은 FCN[30]을 의미론적 분할에 적용할 때 일반적인 관행과 다릅니다. 이는 일반적으로 픽셀당 softmax와 다항 교차 엔트로피 손실을 사용합니다. 이 경우 클래스 간에 마스크가 경쟁합니다. 우리의 경우 픽셀당 시그모이드와 이진 손실이 있는 경우에는 그렇지 않습니다. 우리는 이 공식이 좋은 인스턴스 분할 결과의 핵심이라는 것을 실험을 통해 보여줍니다.

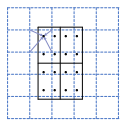


Figure 3. RoIAlign: The dashed grid represents a feature map, the solid lines an RoI (with 2×2 bins in this example), and the dots the 4 sampling points in each bin. RoIAlign computes the value of each sampling point by bilinear interpolation from the nearby grid points on the feature map. No quantization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

그림 3. RoIAlign: 점선 그리드는 기능 맵, 실선은 RoI(이 예에서는 2x2 빈 포함), 점은 각 빈의 4개 샘플링 지점을 나타냅니다. RoIAlign은 피쳐 맵의 가까운 그리드 점에서 쌍선형 보간법으로 각 샘플링 점의 값을 계산합니다. RoI, 해당 빈 또는 샘플링 지점과 관련된 좌표에 대해 양자화가 수행되지 않습니다.

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.

마스크 표현: 마스크는 입력 개체의 공간 레이아웃을 인코딩합니다. 따라서 완전 연결(fc) 레이어에 의해 불가피하게 짧은 출력 벡터로 축소되는 클래스 레이블 또는 상자 오프셋과 달리 마스크의 공간 구조 추출은 컨볼루션이 제공하는 픽셀 대 픽셀 대응에 의해 자연스럽게 처리될 수 있습니다.

Specifically, we predict an m × m mask from each RoI using an FCN [30]. This allows each layer in the mask branch to maintain the explicit m × 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.

특히 FCN을 사용하여 각 RoI에서 m × m 마스크를 예측합니다[30]. 이를 통해 마스크 분기의 각 레이어는 공간 차원이 없는 벡터 표현으로 축소하지 않고 명시적 m × m 객체 공간 레이아웃을 유지할 수 있습니다. 마스크 예측을 위해 fc 레이어에 의존하는 이전 방법[33, 34, 10]과 달리, 우리의 완전한 컨볼루션 표현은 더 적은 수의 매개변수를 필요로 하고 실험에 의해 입증된 바와 같이 더 정확합니다.

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.

이러한 픽셀 대 픽셀 동작을 수행하려면 작은 기능 맵인 RoI 기능이 픽셀당 명시적 공간 대응을 충실하게 보존하기 위해 잘 정렬되어야 합니다. 이것은 우리가 마스크 예측에서 핵심적인 역할을 하는 다음 RoIAlign 레이어를 개발하도록 동기를 부여했습니다.

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 [·] 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.

RoIAlign: RoIPool[12]은 각 RoI에서 작은 특징 맵(예: 7×7)을 추출하기 위한 표준 작업입니다. RoIPool은 먼저 부동 숫자 RoI를 기능 맵의 이산 단위로 양자화하고, 이 양자화된 RoI는 자체적으로 양자화되는 공간 빈으로 세분화되고, 마지막으로 각 빈에 포함된 기능 값이 집계됩니다(일반적으로 최대 풀링에 의해). 양자화는 예를 들어 [x/16]을 계산하여 연속 좌표 x에 대해 수행되며, 여기서 16은 특징 맵 스트라이드이고 [·]는 반올림입니다. 마찬가지로 bin(예: 7x7)으로 나눌 때 양자화가 수행됩니다. 이러한 양자화는 RoI와 추출된 특징 사이의 정렬 불량을 유발합니다. 이것은 작은 번역에 강력한 분류에 영향을 미치지 않을 수 있지만 픽셀 정확도 마스크를 예측하는 데는 큰 부정적인 영향을 미칩니다.

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 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.

이를 해결하기 위해 우리는 RoIPool의 거친 양자화를 제거하고 추출된 특징을 입력과 적절하게 정렬하는 RoIAlign 레이어를 제안합니다. 제안된 변경 사항은 간단합니다. RoI 경계 또는 bin의 양자화를 방지합니다(즉, [x/16] 대신 x/16을 사용). 쌍선형 보간법[22]을 사용하여 각 RoI 빈의 정기적으로 샘플링된 4개 위치에서 입력 기능의 정확한 값을 계산하고 결과를 집계합니다(최대 또는 평균 사용). 자세한 내용은 그림 3을 참조하세요. 양자화가 수행되지 않는 한 결과는 정확한 샘플링 위치 또는 샘플링된 포인트 수에 민감하지 않습니다.

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.

RoIAlign은 §4.2에서 볼 수 있듯이 크게 개선되었습니다. 또한 [10]에서 제안한 RoIWarp 연산과 비교합니다. RoIAlign과 달리 RoIWarp는 정렬 문제를 간과하고 RoIPool과 마찬가지로 RoI를 양자화하는 것으로 [10]에서 구현되었습니다. 따라서 RoIWarp도 [22]에 의해 동기 부여된 이중 선형 리샘플링을 채택하지만 실험(표 2c에 더 자세한 내용은 표 2c 참조)에서 볼 수 있듯이 RoIPool과 동등하게 수행하여 정렬의 중요한 역할을 보여줍니다.

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.

네트워크 아키텍처: 접근 방식의 일반성을 보여주기 위해 다중 아키텍처로 Mask R-CNN을 인스턴스화합니다. 명확성을 위해 (i) 전체 이미지에 대한 특징 추출에 사용되는 컨볼루션 백본 아키텍처와 (ii) 경계 상자 인식(분류 및 회귀) 및 각 RoI에 개별적으로 적용되는 마스크 예측을 위한 네트워크 헤드를 구분합니다.

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].

명명법 네트워크 깊이 기능을 사용하여 백본 아키텍처를 나타냅니다. 깊이 50 또는 101 레이어의 ResNet [19] 및 ResNeXt [45] 네트워크를 평가합니다. ResNets[19]를 사용한 Faster R-CNN의 원래 구현은 C4라고 하는 4단계의 최종 컨볼루션 계층에서 기능을 추출했습니다. 예를 들어 ResNet-50이 있는 이 백본은 ResNet-50-C4로 표시됩니다. 이것은 [19, 10, 21, 39]에서 사용되는 일반적인 선택입니다.

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 RCNN gives excellent gains in both accuracy and speed. For further details on FPN, we refer readers to [27].

우리는 또한 최근에 Lin et al.이 제안한 또 다른 효과적인 백본을 탐색합니다. FPN(Feature Pyramid Network)이라고 합니다. FPN은 측면 연결이 있는 하향식 아키텍처를 사용하여 단일 규모 입력에서 네트워크 내 기능 피라미드를 구축합니다. FPN 백본을 사용하는 더 빠른 R-CNN은 규모에 따라 기능 피라미드의 다양한 수준에서 RoI 기능을 추출하지만 나머지 접근 방식은 기본 ResNet과 유사합니다. Mask RCNN과 함께 특징 추출을 위해 ResNet-FPN 백본을 사용하면 정확도와 속도 모두에서 탁월한 이점을 얻을 수 있습니다. FPN에 대한 자세한 내용은 독자에게 [27]을 참조하십시오.

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 computeintensive. For FPN, the backbone already includes res5 and thus allows for a more efficient head that uses fewer filters.

네트워크 헤드의 경우 우리는 완전 컨볼루션 마스크 예측 분기를 추가하는 이전 작업에서 제시된 아키텍처를 밀접하게 따릅니다. 특히 ResNet [19] 및 FPN [27] 논문에서 Faster R-CNN 상자 헤드를 확장합니다. 세부 사항은 그림 4에 나와 있습니다. ResNet-C4 백본의 헤드에는 계산 집약적인 ResNet의 5단계(즉, 9계층 'res5'[19])가 포함됩니다. FPN의 경우 백본에는 이미 res5가 포함되어 있으므로 더 적은 수의 필터를 사용하는 더 효율적인 헤드가 가능합니다.

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.

우리의 마스크 브랜치는 간단한 구조를 가지고 있습니다. 더 복잡한 디자인은 성능을 향상시킬 가능성이 있지만 이 작업의 초점은 아닙니다.

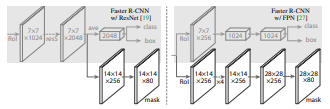


Figure 4. Head Architecture: We extend two existing Faster RCNN heads [19, 27]. Left/Right panels show the heads for the ResNet C4 and FPN backbones, from [19] and [27], respectively, to which a mask branch is added. Numbers denote spatial resolution and channels. Arrows denote either conv, deconv, or 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.

그림 4. 헤드 아키텍처: 기존 Faster RCNN 헤드 2개를 확장합니다[19, 27]. 왼쪽/오른쪽 패널은 마스크 분기가 추가된 각각 [19] 및 [27]의 ResNet C4 및 FPN 백본에 대한 헤드를 보여줍니다. 숫자는 공간 해상도와 채널을 나타냅니다. 화살표는 컨텍스트에서 추론할 수 있는 conv, deconv 또는 fc 레이어를 나타냅니다(conv는 공간 차원을 유지하지만 deconv는 공간 차원을 증가시킴). 출력 conv는 1x1이고 deconv는 2x2이고 stride 2이며 숨겨진 레이어에서 ReLU [31]를 사용하는 것을 제외하고 모든 변환은 3x3입니다. 왼쪽: 'res5'는 ResNet의 다섯 번째 단계를 나타냅니다. 단순화를 위해 첫 번째 전환이 보폭 1([19]에서와 같이 14×14 / 보폭 2 대신)이 있는 7×7 RoI에서 작동하도록 변경했습니다. 오른쪽: '×4'는 4회의 연속 전환 스택을 나타냅니다.

**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.

기존 Fast/Faster R-CNN 작업[12, 36, 27]에 따라 하이퍼파라미터를 설정했습니다. 이러한 결정은 원본 논문[12, 36, 27]에서 객체 감지에 대해 이루어졌지만 인스턴스 분할 시스템이 이에 대해 강력하다는 것을 알았습니다.

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 is defined only on positive RoIs. The mask target is the intersection between an RoI and its associated ground-truth mask.

훈련: Fast R-CNN에서와 같이, RoI에 최소 0.5의 ground-truth box가 있는 IoU가 있으면 양수로 간주되고 그렇지 않으면 음수로 간주됩니다. 마스크 손실 는 양의 RoI에서만 정의됩니다. 마스크 타겟은 RoI와 관련 실측 마스크 사이의 교차점입니다.

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 minibatch 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.

우리는 이미지 중심 교육을 채택합니다[12]. 이미지는 크기(짧은 가장자리)가 800픽셀이 되도록 크기가 조정됩니다[27]. 각 미니 배치에는 GPU당 2개의 이미지가 있고 각 이미지에는 N 샘플링된 RoI가 있으며 포지티브 대 네거티브 비율이 1:3입니다[12]. N은 C4 백본의 경우 64([12, 36]에서와 같이)이고 FPN의 경우([27]에서와 같이) 512입니다. 우리는 160,000번 반복에 대해 8개의 GPU(따라서 효과적인 미니 배치 크기는 16)에서 훈련하고 120,000번 반복에서 10만큼 감소하는 0.02의 학습률을 사용합니다. 0.0001의 가중치 감쇠와 0.9의 운동량을 사용합니다. ResNeXt[45]를 사용하여 GPU당 1개의 이미지와 0.01의 시작 학습률로 동일한 반복 횟수로 훈련합니다.

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.

RPN 앵커는 [27]에 따라 5개의 스케일과 3개의 종횡비에 걸쳐 있습니다. 편리한 절제를 위해 RPN은 별도로 교육되며 지정되지 않는 한 Mask R-CNN과 기능을 공유하지 않습니다. 이 문서의 모든 항목에 대해 RPN과 Mask R-CNN은 동일한 백본을 가지고 있으므로 공유 가능합니다.

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 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×m floating-number mask output is then resized to the RoI size, and binarized at a threshold of 0.5.

추론: 테스트 시간에 제안 번호는 C4 백본의 경우 300([36]), FPN의 경우 1000([27])입니다. 우리는 이러한 제안에 대해 상자 예측 분기를 실행한 다음 최대가 아닌 억제를 실행합니다[14]. 그런 다음 마스크 분기가 가장 높은 점수를 받은 100개의 감지 상자에 적용됩니다. 이는 훈련에 사용되는 병렬 계산과 다르지만 추론 속도를 높이고 정확도를 향상시킵니다(더 적고 더 정확한 RoI를 사용하기 때문에). 마스크 분기는 RoI당 K개의 마스크를 예측할 수 있지만 k번째 마스크만 사용합니다. 여기서 k는 분류 분기에 의해 예측된 클래스입니다. 그런 다음 m×m 부동 숫자 마스크 출력이 RoI 크기로 조정되고 임계값 0.5에서 이진화됩니다.

텍스트, 다른, 다채로운, 다양한이(가) 표시된 사진

자동 생성된 설명

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).

그림 5. ResNet-101-FPN을 사용하고 35.7 마스크 AP로 5fps로 실행한 COCO 테스트 이미지에 대한 Mask R-CNN의 추가 결과(표 1).

테이블이(가) 표시된 사진

자동 생성된 설명

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.

표 1. COCO test-dev의 인스턴스 분할 마스크 AP MNC [10] 및 FCIS [26]는 각각 COCO 2015 및 2016 분할 챌린지의 승자입니다. 종소리와 휘파람이 없으면 Mask R-CNN은 다중 스케일 훈련/테스트, 수평 플립 테스트 및 OHEM을 포함하는 더 복잡한 FCIS+++보다 성능이 뛰어납니다[38]. 모든 항목은 단일 모델 결과입니다.

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).

상위 100개의 탐지 상자에서만 마스크를 계산하기 때문에 Mask R-CNN은 Faster R-CNN에 약간의 오버헤드를 추가합니다(예: 일반 모델의 경우 ~20%).

**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 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].

우리는 COCO 데이터 세트[28]에 대한 포괄적인 절제와 함께 Mask R-CNN을 최신 기술과 철저히 비교합니다. AP(IoU 임계값 AP\_75 및 AP\_S,AP\_M,AP\_L(다른 척도의 AP)에 대해 평균)를 포함한 표준 COCO 메트릭을 보고합니다. 명시되지 않는 한 AP는 마스크 IoU를 사용하여 평가합니다. 이전 작업 [5, 27]에서와 같이 우리는 훈련합니다. 80k 기차 이미지와 35k val 이미지의 하위 집합(trainval35k)의 합집합을 사용하고 나머지 5k val 이미지(minival)에 대한 절제를 보고합니다. 또한 test-dev [28]에 대한 결과도 보고합니다.

**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.

우리는 Mask R-CNN을 표 1의 인스턴스 분할에서 최첨단 방법과 비교합니다. 우리 모델의 모든 인스턴스화는 이전 최신 모델의 기준 변형보다 성능이 뛰어납니다. 여기에는 COCO 2015 및 2016 세분화 챌린지에서 각각 우승한 MNC[10]와 FCIS[26]가 포함됩니다. 종소리 없이 ResNet-101-FPN 백본을 사용한 Mask R-CNN은 다중 스케일 학습/테스트, 수평 플립 테스트 및 OHEM(온라인 하드 예제 마이닝)[38]을 포함하는 FCIS+++[26]보다 성능이 뛰어납니다. 이 작업의 범위를 벗어나지만 이러한 개선 사항이 많이 적용될 것으로 기대합니다.

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.

Mask R-CNN 출력은 그림 2와 5에 시각화되어 있습니다. Mask R-CNN은 어려운 조건에서도 좋은 결과를 얻습니다. 그림 6에서는 Mask R-CNN 기준선과 FCIS+++를 비교합니다[26]. FCIS+++는 중복되는 인스턴스에 대한 체계적인 아티팩트를 보여 인스턴스 세분화의 근본적인 어려움이 문제임을 시사합니다. Mask R-CNN은 그러한 아티팩트를 보여주지 않습니다.

텍스트, 다른, 다양한, 다채로운이(가) 표시된 사진

자동 생성된 설명

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

그림 6. FCIS+++ [26](위)와 Mask R-CNN(아래, ResNet-101-FPN). FCIS는 겹치는 물체에 체계적인 인공물을 전시합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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

(a) 백본 아키텍처: 더 나은 백본은 예상되는 이득을 가져옵니다. 더 깊은 네트워크는 더 나은 성능을 제공하고, FPN은 C4 기능을 능가하며, ResNeXt는 ResNet에서 개선됩니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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

(b) 다항식 대 독립 마스크(ResNet-50-C4): 클래스별 이진 마스크(sigmoid)를 통한 분리는 다항식 마스크(softmax)보다 큰 이득을 제공합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

(c) RoIAlign (ResNet-50-C4): Mask results with various RoI layers. Our RoIAlign layer improves AP by ∼3 points and AP75 by ∼5 points. Using proper alignment is the only factor that contributes to the large gap between RoI layers.

(c) RoIAlign(ResNet-50-C4): 다양한 RoI 레이어로 결과를 마스킹합니다. RoIAlign 레이어는 AP를 ~3포인트, AP75를 ~5포인트 향상시킵니다. 적절한 정렬을 사용하는 것이 RoI 레이어 간의 큰 간격에 기여하는 유일한 요소입니다.

테이블이(가) 표시된 사진

자동 생성된 설명

(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.

(d) RoIAlign(ResNet-50-C5, stride 32): Large stride 기능을 사용하는 마스크 수준 및 상자 수준 AP. 정렬 불량은 stride-16 기능(표 2c)보다 더 심하여 정확도 격차가 커집니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

(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.

(e) 마스크 분기(ResNet-50-FPN): 마스크 예측을 위한 완전 컨벌루션 네트워크(FCN) 대 다층 퍼셉트론(MLP, 완전 연결). FCN은 공간 레이아웃을 명시적으로 인코딩하므로 결과를 개선합니다.

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

표 2. 절제. 우리는 trainval35k에서 훈련하고, minival에서 테스트하고, 달리 명시되지 않는 한 마스크 AP를 보고합니다.

**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.

우리는 Mask R-CNN을 분석하기 위해 많은 절제를 실행합니다. 결과는 표 2에 나와 있으며 다음에 자세히 설명합니다.

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]).

아키텍처: 표 2a는 다양한 백본이 있는 Mask R-CNN을 보여줍니다. 더 깊은 네트워크(50 vs. 101)와 FPN 및 ResNeXt를 포함한 고급 설계의 이점이 있습니다. 우리는 모든 프레임워크가 더 깊은 또는 고급 네트워크의 이점을 자동으로 얻는 것은 아니라는 점에 주목합니다([21]의 벤치마킹 참조).

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.

다항식 대 독립 마스크: 마스크 R-CNN은 마스크와 클래스 예측을 분리합니다. 기존 상자 분기가 클래스 레이블을 예측하므로 클래스 간 경쟁 없이 각 클래스에 대한 마스크를 생성합니다(픽셀당 시그모이드 및 이진 손실에 의해). 표 2b에서 픽셀당 softmax 및 다항 손실(FCN[30]에서 일반적으로 사용됨)을 사용하는 것과 이를 비교합니다. 이 대안은 마스크와 클래스 예측 작업을 결합하고 마스크 AP(5.5포인트)에서 심각한 손실을 초래합니다. 이것은 인스턴스가 전체로 분류되면(상자 분기에 의해) 범주에 대한 걱정 없이 이진 마스크를 예측하는 것으로 충분하므로 모델을 더 쉽게 훈련할 수 있습니다.

Class-Specific vs. Class-Agnostic Masks: Our default instantiation predicts class-specific masks, i.e., one m×m mask per class. Interestingly, Mask R-CNN with classagnostic masks (i.e., predicting a single m×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.

클래스별 마스크 대 클래스 불가지론적 마스크: 기본 인스턴스화는 클래스별 마스크, 즉 클래스당 하나의 m×m 마스크를 예측합니다. 흥미롭게도, Classagnostic Mask를 사용하는 Mask R-CNN(즉, 클래스에 관계없이 단일 m×m 출력 예측)은 거의 효과적입니다. ResNet-50-C4의 클래스별 대응물에 대해 29.7 마스크 AP 대 30.3입니다. 이것은 분류와 세분화를 크게 분리하는 우리의 접근 방식에서 노동 분업을 더욱 강조합니다.

RoIAlign: An evaluation of our proposed RoIAlign layer is shown in Table 2c. For this experiment we use the ResNet50-C4 backbone, which has stride 16. RoIAlign improves AP by about 3 points over RoIPool, with much of the gain coming at high IoU (AP75). RoIAlign is insensitive to max/average pool; we use average in the rest of the paper.

RoIAlign: 제안된 RoIAlign 레이어의 평가는 표 2c에 나와 있습니다. 이 실험을 위해 보폭이 16인 ResNet50-C4 백본을 사용합니다. RoIAlign은 RoIPool보다 AP를 약 3포인트 향상시키며 대부분의 이득은 높은 IoU(AP75)에서 발생합니다. RoIAlign은 최대/평균 풀에 둔감합니다. 나머지 논문에서는 평균을 사용합니다.

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.

또한 이중 선형 샘플링을 채택한 MNC[10]에서 제안한 RoIWarp와 비교합니다. §3에서 논의된 바와 같이 RoIWarp는 여전히 RoI를 양자화하여 입력과의 정렬을 잃습니다. 표 2c에서 볼 수 있듯이 RoIWarp는 RoIPool과 동등하고 RoIAlign보다 훨씬 나쁜 성능을 보입니다. 이것은 적절한 정렬이 핵심임을 강조합니다.

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 AP75 by 10.5 points (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.

우리는 또한 32픽셀의 훨씬 더 큰 보폭을 가진 ResNet-50-C5 백본으로 RoIAlign을 평가합니다. res5 헤드가 적용되지 않기 때문에 그림 4(오른쪽)와 동일한 헤드를 사용합니다. 표 2d는 RoIAlign이 마스크 AP를 7.3포인트, 마스크 AP75를 10.5포인트(상대적으로 50% 개선) 향상시키는 것을 보여줍니다. 또한 RoIAlign을 사용하면 stride-32 C5 기능(30.9 AP)을 사용하는 것이 stride-16 C4 기능(30.3 AP, 표 2c)을 사용하는 것보다 더 정확합니다. RoIAlign은 탐지 및 세분화를 위해 대규모 기능을 사용하는 오랜 문제를 크게 해결합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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 ), multitask training (+0.9 ), and ResNeXt-101 (+1.6 )

표 3. 객체 감지 단일 모델 결과(경계 상자 AP)와 test-dev의 최신 기술 비교 ResNet-101-FPN을 사용하는 Mask R-CNN은 이전의 모든 최첨단 모델의 기본 변형을 능가합니다(이 실험에서는 마스크 출력이 무시됨). [27]에 비해 Mask R-CNN의 이점은 RoIAlign(+1.1 AP^bb), 멀티태스킹 교육(+0.9 AP^bb) 및 ResNeXt-101(+1.6 AP^bb) 사용에서 비롯됩니다.

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).

마지막으로 RoIAlign은 더 미세한 다중 레벨 보폭을 갖는 FPN과 함께 사용할 때 1.5 마스크 AP 및 0.5 박스 AP의 이득을 보여줍니다. 더 정밀한 정렬이 필요한 키포인트 감지의 경우 RoIAlign은 FPN에서도 큰 이득을 보입니다(표 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.

마스크 분기: 분할은 픽셀 대 픽셀 작업이며 FCN을 사용하여 마스크의 공간 레이아웃을 활용합니다. 표 2e에서는 ResNet-50-FPN 백본을 사용하여 다층 퍼셉트론(MLP)과 FCN을 비교합니다. FCN을 사용하면 MLP에 비해 2.1 마스크 AP 이득을 얻을 수 있습니다. MLP와 공정한 비교를 위해 FCN 헤드의 변환 레이어가 사전 훈련되지 않도록 이 백본을 선택했습니다.

**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-ofthe-art models, including the single-model variant of GRMI [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).

우리는 표 3에서 Mask R-CNN을 최신 COCO bounding-box object detection과 비교합니다. 이 결과를 위해 전체 Mask R-CNN 모델이 학습되더라도 분류 및 box 출력만 사용됩니다. 추론(마스크 출력은 무시됨). ResNet-101-FPN을 사용하는 Mask R-CNN은 COCO 2016 Detection Challenge의 우승자인 GRMI[21]의 단일 모델 변형을 포함하여 이전의 모든 최첨단 모델의 기본 변형을 능가합니다. ResNeXt-101-FPN을 사용하는 Mask R-CNN은 [39](Inception-ResNet-v2-TDM 사용)의 이전 단일 모델 항목보다 3.0포인트 상자 AP의 여백으로 결과를 더욱 향상시킵니다.

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.

추가 비교를 위해 우리는 Mask R-CNN 버전을 훈련했지만 마스크 분기는 없으며 표 3에서 "Faster R-CNN, RoIAlign"으로 표시됩니다. 이 모델은 RoIAlign으로 인해 [27]에 제시된 모델보다 성능이 좋습니다. 반면 Mask R-CNN보다 0.9포인트 낮은 박스 AP이다. 따라서 상자 감지에서 Mask R-CNN의 이러한 격차는 전적으로 다중 작업 교육의 이점 때문입니다.

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.

마지막으로 Mask R-CNN은 마스크와 상자 AP 사이에 작은 간격(예: 37.1(마스크, 표 1)과 39.8(상자, 표 3) 사이의 2.7 포인트)에 도달합니다. 이것은 우리의 접근 방식이 객체 감지와 더 까다로운 인스턴스 분할 작업 사이의 격차를 크게 좁혀준다는 것을 나타냅니다.

**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.

추론: Faster R-CNN의 4단계 교육에 따라 RPN과 Mask R-CNN 단계 간에 기능을 공유하는 ResNet-101-FPN 모델을 교육합니다[36]. 이 모델은 Nvidia Tesla M40 GPU에서 이미지당 195ms로 실행되며(출력을 원래 해상도로 크기 조정하는 15ms CPU 시간 추가) 통계적으로 공유되지 않은 것과 동일한 마스크 AP를 달성합니다. 또한 ResNet-101-C4 변형은 박스 헤드가 더 무거워서 ~400ms가 걸린다고 보고하므로(그림 4) 실제로 C4 변형을 사용하지 않는 것이 좋습니다.

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.

Mask R-CNN은 빠르지만 우리의 디자인은 속도에 최적화되어 있지 않으며 더 나은 속도/정확도 절충안을 달성할 수 있습니다[21], 예를 들어 이 문서의 범위를 벗어난 다양한 이미지 크기와 제안 번호를 사용합니다.

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.

훈련: Mask R-CNN은 훈련 속도도 빠릅니다. COCO trainval35k에서 ResNet-50-FPN을 사용한 교육은 동기화된 8-GPU 구현(16개 이미지 미니 배치당 0.72초)에서 32시간이 걸리고 ResNet-101-FPN으로 44시간이 걸립니다. 실제로 기차 세트에서 훈련하면 빠른 프로토타이핑이 하루도 채 안 되어 완료될 수 있습니다. 우리는 이러한 신속한 교육이 이 분야의 주요 장애물을 제거하고 더 많은 사람들이 이 도전적인 주제에 대한 연구를 수행하도록 장려하기를 바랍니다.

**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.

우리의 프레임워크는 인간 포즈 추정으로 쉽게 확장될 수 있습니다. 우리는 키포인트의 위치를 ​​원-핫 마스크로 모델링하고 마스크 R-CNN을 채택하여 K개의 키포인트 유형(예: 왼쪽 어깨, 오른쪽 팔꿈치)마다 하나씩 K 마스크를 예측합니다. 이 작업은 Mask R-CNN의 유연성을 입증하는 데 도움이 됩니다.

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.

우리는 실험이 주로 Mask R-CNN 프레임워크의 일반성을 입증하기 위한 것이기 때문에 인간 포즈에 대한 최소한의 도메인 지식이 우리 시스템에 의해 이용된다는 점에 주목합니다. 도메인 지식(예: 모델링 구조[6])이 간단한 접근 방식을 보완할 것으로 기대합니다.

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 × 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 m2 -way softmax output (which encourages a single point to be detected). We note that as in instance segmentation, the K keypoints are still treated independently.

구현 세부 정보: 키포인트에 맞게 조정할 때 분할 시스템을 약간 수정합니다. 인스턴스의 각 K 키포인트에 대해 훈련 대상은 단일 픽셀만 전경으로 레이블되는 원-핫 m × m 이진 마스크입니다. 훈련하는 동안 각각의 가시적 진실 키포인트에 대해 m2-way softmax 출력(단일 포인트가 감지되도록 권장)에 대한 교차 엔트로피 손실을 최소화합니다. 인스턴스 분할에서와 같이 K 키포인트는 여전히 독립적으로 처리됩니다.

다른, 다양한, 다채로운, 같은이(가) 표시된 사진

자동 생성된 설명

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.

그림 7. Mask R-CNN(ResNet-50-FPN)을 사용한 COCO 테스트의 키포인트 감지 결과와 동일한 모델에서 예측된 사람 세분화 마스크. 이 모델은 63.1의 키포인트 AP를 가지며 5fps로 실행됩니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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).

표 4. COCO test-dev의 키포인트 감지 AP 우리는 5fps로 실행되는 단일 모델(ResNet-50-FPN)입니다. CMU-Pose+++[6]는 다중 스케일 테스트, CPM을 사용한 후처리[44] 및 객체 감지기로 필터링을 사용하여 누적 ~5포인트를 추가하는 2016년 대회 우승자입니다(개인 커뮤니케이션에서 명시). † : G-RMI는 두 가지 모델(경계 상자 감지용 Inception-ResNet-v2 및 키포인트용 ResNet-101)을 사용하여 COCO + MPII[1](25k 이미지)에 대해 교육되었습니다.

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.

ResNet-FPN 변형을 채택하고 키포인트 헤드 아키텍처는 그림 4(오른쪽)와 유사합니다. 키포인트 헤드는 8개의 3×3 512-d conv 레이어 스택으로 구성되며, 그 다음 deconv 레이어와 2× 쌍선형 업스케일링으로 구성되어 56×56의 출력 해상도를 생성합니다. 우리는 (마스크와 비교하여) 상대적으로 높은 해상도의 출력이 키포인트 수준의 로컬라이제이션 정확도를 위해 필요하다는 것을 발견했습니다.

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.

모델은 주석이 달린 키포인트가 포함된 모든 COCO trainval35k 이미지에서 훈련됩니다. 과적합을 줄이기 위해 이 훈련 세트가 더 작기 때문에 [640, 800] 픽셀에서 무작위로 샘플링된 이미지 스케일을 사용하여 훈련합니다. 추론은 800픽셀의 단일 스케일에 있습니다. 0.02의 학습률에서 시작하여 60,000 및 80,000회 반복에서 10으로 줄이는 90,000회 반복을 위해 훈련합니다. 임계값이 0.5인 경계 상자 NMS를 사용합니다. 기타 사항은 3.1항과 동일합니다.

Main Results and Ablations: We evaluate the person keypoint AP (APkp) and experiment with a ResNet-50-FPN backbone; more backbones will be studied in the appendix. Table 4 shows that our result (62.7 APkp) 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.

주요 결과 및 절제: 사람 키포인트 AP(APkp)를 평가하고 ResNet-50-FPN 백본으로 실험합니다. 부록에서 더 많은 백본을 연구할 것입니다. 표 4는 우리의 결과(62.7 APkp)가 다단계 처리 파이프라인을 사용하는 COCO 2016 키포인트 탐지 승자[6]보다 0.9포인트 높음을 보여줍니다(표 4의 캡션 참조). 우리의 방법은 훨씬 더 간단하고 빠릅니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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 ResNet50-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).

표 5. 사람 범주에 대한 상자, 마스크 및 키포인트의 다중 작업 학습, 미니얼 평가. 모든 항목은 공정한 비교를 위해 동일한 데이터에 대해 학습됩니다. 백본은 ResNet50-FPN입니다. minival에서 64.2 및 64.7 AP가 있는 항목의 테스트 개발 AP는 각각 62.7 및 63.1입니다(표 4 참조).

텍스트이(가) 표시된 사진

자동 생성된 설명

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

표 6. Minival에서 키포인트 감지를 위한 RoIAlign 대 RoIPool. 백본은 ResNet-50-FPN입니다.

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 APkp 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).

더 중요한 것은 5fps로 실행하면서 상자, 세그먼트 및 키포인트를 동시에 예측할 수 있는 통합 모델이 있다는 것입니다. 세그먼트 분기(사람 범주용)를 추가하면 test-dev에서 APkp가 63.1(표 4)로 향상됩니다. minival에 대한 다중 작업 학습의 더 많은 제거는 표 5에 있습니다. box-only(즉, Faster R-CNN) 또는 keypoint-only 버전에 마스크 분기를 추가하면 이러한 작업이 일관되게 향상됩니다. 그러나 키포인트 분기를 추가하면 상자/마스크 AP가 약간 줄어들어 키포인트 감지가 멀티태스킹 교육의 이점을 얻지만 다른 작업에는 도움이 되지 않습니다. 그럼에도 불구하고 세 가지 작업을 모두 학습하면 통합 시스템이 모든 출력을 동시에 효율적으로 예측할 수 있습니다(그림 7).

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 APkp 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.

우리는 또한 키포인트 탐지에 대한 RoIAlign의 효과를 조사합니다(표 6). 이 ResNet-50-FPN 백본이 더 미세하지만(예: 가장 정밀한 수준에서 4픽셀) RoIAlign은 여전히 ​​RoIPool에 비해 상당한 개선을 보여주고 APkp를 4.4포인트 증가시킵니다. 이는 키포인트 감지가 현지화 정확도에 더 민감하기 때문입니다. 이것은 마스크와 키포인트를 포함하여 픽셀 수준 지역화에 정렬이 필수적임을 다시 나타냅니다.

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.

객체 경계 상자, 마스크 및 키포인트를 추출하기 위한 Mask R-CNN의 효율성을 감안할 때 다른 인스턴스 수준 작업을 위한 효과적인 프레임워크가 될 것으로 기대합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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

표 7. Cityscapes val('AP [val]' 열) 및 테스트(나머지 열) 집합에 대한 결과. 우리의 방법은 ResNet-50-FPN을 사용합니다.

**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:

Cityscapes [7] 데이터 세트에 대한 인스턴스 세분화 결과를 추가로 보고합니다. 이 데이터 세트에는 2975개의 기차, 500개 발 및 1525개의 테스트 이미지에 대한 미세 주석이 있습니다. 우리가 사용하지 않는 인스턴스 주석이 없는 20k의 거친 훈련 이미지가 있습니다. 모든 이미지는 2048×1024 픽셀입니다. 인스턴스 분할 작업에는 8개의 객체 범주가 포함되며, 미세 훈련 세트의 인스턴스 수는 다음과 같습니다.



Instance segmentation performance on this task is measured by the COCO-style mask AP (averaged over IoU thresholds); AP50 (i.e., mask AP at an IoU of 0.5) is also reported.

이 작업에 대한 인스턴스 분할 성능은 COCO 스타일 마스크 AP(IoU 임계값에 대한 평균)로 측정됩니다. AP50(즉, 0.5의 IoU에서 마스크 AP)도 보고됩니다.

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.

구현: ResNet-FPN-50 백본과 함께 Mask R-CNN 모델을 적용합니다. 우리는 101-레이어 대응물이 작은 데이터 세트 크기로 인해 유사하게 수행된다는 것을 발견했습니다. [800, 1024]에서 무작위로 샘플링된 이미지 스케일(짧은 쪽)로 훈련하여 과적합을 줄입니다. 추론은 1024픽셀의 단일 스케일에 있습니다. GPU당 이미지 1개의 미니 배치 크기(GPU 8개에서 8개)를 사용하고 학습률 0.01에서 시작하여 18,000 반복에서 0.001로 줄이는 24,000 반복에 대해 모델을 훈련합니다. 이 설정에서는 단일 8-GPU 머신에서 약 4시간의 훈련이 필요합니다.

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.

결과: 표 7은 우리의 결과를 val 및 테스트 세트에 대한 최신 기술과 비교합니다. 거친 훈련 세트를 사용하지 않고 우리의 방법은 테스트에서 26.2 AP를 달성합니다. 이는 이전 최고 항목(DIN[3])에 비해 30% 이상 상대적으로 개선되었으며 SGN의 25.0[29]의 동시 작업보다 우수합니다. DIN과 SGN은 모두 미세 + 거친 데이터를 사용합니다. 미세 데이터만 사용하는 최상의 항목(17.4 AP)과 비교하여 ~50% 개선을 달성합니다.

7.4 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.

7.4 AP), 우리는 ~50% 개선을 달성했습니다. 사람 및 자동차 카테고리의 경우 Cityscapes 데이터 세트는 카테고리 내 중복 인스턴스가 많이 나타납니다(이미지당 평균 6명의 사람과 9대의 자동차). 우리는 범주 내 중복이 인스턴스 분할의 핵심 어려움이라고 주장합니다. 우리의 방법은 거친 데이터를 활용하지 않더라도 다른 최고의 항목에 비해 이 두 범주에서 엄청난 개선을 보여줍니다(사람의 경우 21.8에서 30.5로 ~40% 개선 및 자동차에서 39.4에서 46.9로 ~20% 개선).

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.

Cityscapes 데이터 세트의 주요 과제는 특히 각각 약 200-500개의 훈련 샘플이 있는 트럭, 버스 및 기차 범주에 대해 낮은 데이터 영역에서 모델을 훈련시키는 것입니다. 이 문제를 부분적으로 해결하기 위해 COCO 사전 교육을 사용하여 결과를 추가로 보고합니다. 이를 위해 사전 훈련된 COCO Mask R-CNN 모델에서 Cityscapes의 해당 7개 범주를 초기화합니다(라이더는 무작위로 초기화됨). 학습률이 3k 반복에서 감소하는 4k 반복에 대해 이 모델을 미세 조정하며, COCO 모델이 주어지면 훈련에 ~1시간이 걸립니다.

텍스트, 다른, 다채로운, 여러개이(가) 표시된 사진

자동 생성된 설명

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

그림 8. Cityscapes 테스트의 Mask R-CNN 결과(32.0 AP). 오른쪽 하단 이미지는 실패 예측을 보여줍니다.

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 lowshot learning performance. We show that using COCO pretraining is an effective strategy on this dataset.

COCO 사전 훈련된 Mask R-CNN 모델은 테스트에서 32.0 AP를 달성했으며 이는 fine-only 모델보다 거의 6포인트 향상되었습니다. 이는 훈련 데이터의 양이 중요한 역할을 함을 나타냅니다. 또한 Cityscapes의 방법은 낮은 수준의 학습 성능에 영향을 받을 수 있음을 시사합니다. COCO 사전 훈련을 사용하는 것이 이 데이터 세트에 대한 효과적인 전략임을 보여줍니다.

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).

마지막으로 [23, 4, 29]의 결과에서도 관찰된 바와 같이 val과 test AP 사이의 편향을 관찰했습니다. 우리는 이 편향이 주로 트럭, 버스 및 기차 범주에 의해 발생한다는 것을 발견했으며, 미세 전용 모델의 val/test AP는 각각 28.8/22.8, 53.5/32.2 및 33.0/18.6입니다. 이는 훈련 데이터가 거의 없는 이러한 범주에 대한 도메인 이동이 있음을 시사합니다. COCO 사전 교육은 이러한 범주에서 결과를 가장 개선하는 데 도움이 됩니다. 그러나 도메인 이동은 각각 38.0/30.1, 57.5/40.9 및 41.2/30.9 val/test AP에서 지속됩니다. 사람 및 자동차 범주의 경우 그러한 편향이 표시되지 않습니다(val/test AP는 ±1 포인트 이내).

Example results on Cityscapes are shown in Figure 8.

Cityscapes에 대한 예시 결과는 그림 8에 나와 있습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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.

표 8. COCO minival에서 Mask R-CNN의 향상된 탐지 결과. 각 행은 위의 행에 추가 구성요소를 추가합니다. 표기의 간결함을 위해 ResNeXt 모델을 'X'로 표시합니다.

**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.

일반적인 프레임워크로서 Mask R-CNN은 Fast/Faster R-CNN 및 FCN에 대한 개선 사항을 포함하여 탐지/분할을 위해 개발된 보완 기술과 호환됩니다. 이 부록에서는 원래 결과보다 개선된 몇 가지 기술을 설명합니다. 일반성과 유연성 덕분에 Mask R-CNN은 COCO 2017 인스턴스 세분화 경쟁에서 우승한 세 팀의 프레임워크로 사용되었으며 모두 이전 기술 상태를 크게 능가했습니다.

**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.

우리는 표 8에서 Mask R-CNN의 개선된 결과를 보고합니다. 전반적으로 개선은 마스크 AP 5.1 포인트(36.7에서 41.8로) 및 박스 AP 7.7 포인트(39.6에서 47.3으로)를 증가시킵니다. 각 모델 개선은 마스크 AP와 박스 AP를 일관되게 증가시켜 Mask R-CNN 프레임워크의 좋은 일반화를 보여줍니다. 개선 사항은 다음에 자세히 설명합니다. 이 결과는 향후 업데이트와 함께 https://github.com/facebookresearch/Detectron에서 릴리스된 코드로 재현할 수 있으며 향후 연구를 위한 더 높은 기준선 역할을 할 수 있습니다.

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.

업데이트된 기준선: 다른 하이퍼 매개변수 세트로 업데이트된 기준선으로 시작합니다. 학습 속도가 120k 및 160k 반복에서 10만큼 감소하는 180k 반복으로 훈련을 연장합니다. 또한 NMS 임계값을 0.5(기본값 0.3에서)로 변경합니다. 업데이트된 베이스라인에는 37.0 마스크 AP와 40.5 박스 AP가 있습니다.

End-to-end training: All previous results used stagewise training, i.e., training RPN as the first stage and Mask R-CNN as the second. Following [37], we evaluate endto-end (‘e2e’) training that jointly trains RPN and Mask RCNN. 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.

End-to-end training: 이전의 모든 결과는 stagewise training, 즉 첫 번째 단계로 RPN을 훈련하고 두 번째 단계로 Mask R-CNN을 사용했습니다. [37]에 이어 RPN과 Mask RCNN을 공동으로 훈련하는 end-to-end('e2e') 훈련을 평가합니다. [37]에서 기울기 w.r.t를 무시하여 RoIAlign 레이어의 부분 기울기만 계산하는 '근사치' 버전을 채택합니다. ROI 좌표. 표 8은 e2e 훈련이 마스크 AP를 0.6, 박스 AP를 1.2 향상시키는 것을 보여줍니다.

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.

ImageNet-5k 사전 훈련: [45]에 따라 ImageNet의 5k 클래스 하위 집합에서 사전 훈련된 모델을 실험합니다(표준 1k 클래스 하위 집합과 대조). 사전 훈련 데이터가 5배 증가하면 마스크와 박스 1 AP가 모두 향상됩니다. 참고로 [40]은 ~250배 더 ​​많은 이미지(300M)를 사용하고 기준선에서 2-3 상자 AP 개선을 보고했습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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

표 9. COCO minival에서 Mask R-CNN의 향상된 키포인트 결과. 각 행은 위의 행에 추가 구성요소를 추가합니다. 여기서는 키포인트 주석만 사용하고 마스크 주석은 사용하지 않습니다. 간결함을 위해 ResNet을 'R'로 표시하고 ResNeXt를 'X'로 표시합니다.

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.

기차 시간 증대: 기차 시간의 스케일 증대는 결과를 더욱 향상시킵니다. 훈련하는 동안 [640, 800] 픽셀에서 스케일을 무작위로 샘플링하고 반복 횟수를 260k로 늘립니다(학습 속도는 200k 및 240k 반복에서 10 감소). Train-time Augmentation은 마스크 AP를 0.6, 박스 AP를 0.8 향상시킵니다.

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.

모델 아키텍처: 101-layer ResNeXt를 152-layer 대응물로 업그레이드함으로써, 우리는 0.5 마스크 AP와 0.6 박스 AP의 증가를 관찰했습니다. 이것은 더 깊은 모델이 여전히 COCO에 대한 결과를 개선할 수 있음을 보여줍니다.

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.

최근 제안된 NL(non-local) 모델[43]을 사용하여 40.3 마스크 AP와 45.0 박스 AP를 달성합니다. 이 결과는 테스트 시간 증가가 없으며 이 방법은 테스트 시간에 Nvidia Tesla P100 GPU에서 3fps로 실행됩니다.

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 singlemodel result of 41.8 mask AP and 47.3 box AP

테스트 시간 증대: [400, 1200] 픽셀의 스케일을 사용하여 평가된 모델 결과를 100단계와 수평 대칭으로 결합합니다. 이것은 41.8 마스크 AP와 47.3 박스 AP의 단일 모델 결과를 제공합니다.

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.

위의 결과는 COCO 2017 대회(여기에서 논의되지 않은 앙상블도 사용됨)에 제출하는 기반입니다. 인스턴스 분할 작업에서 처음 세 개의 우승 팀은 모두 Mask R-CNN 프레임워크의 확장을 기반으로 한 것으로 알려졌습니다.

**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 by about 1 point. Replacing ResNet-50 with ResNet-101 and ResNeXt-101 increases to 66.1 and 67.3, respectively.

우리는 표 9에 키포인트 감지의 향상된 결과를 보고합니다. 업데이트된 기준으로서 학습 속도가 100,000 및 120,000 반복에서 10만큼 감소하는 130,000 반복으로 교육 일정을 확장합니다. 이것은 를 약 1포인트 향상시킵니다. ResNet-50을 ResNet-101 및 ResNeXt-101로 교체하면 가 각각 66.1 및 67.3으로 증가합니다.

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 APkp 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.

데이터 증류[35]라는 최근 방법을 사용하여 COCO에서 제공하는 추가 120k 레이블이 지정되지 않은 이미지를 활용할 수 있습니다. 간단히 말해서, 데이터 증류는 레이블이 지정된 데이터에 대해 훈련된 모델을 사용하여 레이블이 지정되지 않은 이미지의 주석을 예측하고 이러한 새 주석으로 모델을 업데이트하는 자가 학습 전략입니다. Mask R-CNN은 이러한 자기 훈련 전략을 위한 효과적인 프레임워크를 제공합니다. 데이터 증류를 통해 Mask R-CNN APkp는 69.1로 1.8포인트 향상됩니다. Mask R-CNN은 데이터에 레이블이 지정되지 않은 경우에도 추가 데이터의 이점을 얻을 수 있습니다.

By using the same test-time augmentation as used for instance segmentation, we further boost APkp to 70.4.

인스턴스 분할에 사용된 것과 동일한 테스트 시간 증대를 사용하여 APkp를 70.4로 추가로 높입니다.

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

감사의 말: 코드 릴리스 및 향상된 결과에 대한 공헌에 대해 Ilija Radosavovic과 엔지니어링 지원에 대한 Caffe2 팀에 감사드립니다.