**Feature Pyramid Networks for Object Detection**

Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie

**Abstract**

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

피처 피라미드는 다양한 스케일의 객체를 검출하기 위한 인식 시스템의 기본 컴포넌트입니다. 그러나 최근 딥 러닝 객체 검출기는 컴퓨팅과 메모리 부하가 높기 때문에 피라미드 표현을 피하고 있습니다. 이 백서에서는 심층 컨볼루션 네트워크의 고유한 멀티 스케일 피라미드 계층을 이용하여 한계 추가 비용으로 피쳐 피라미드를 구축한다. 횡방향 접속이 있는 탑다운 아키텍처는 모든 척도에서 높은 수준의 의미적 특징 맵을 구축하기 위해 개발된다. Feature Pyramid Network(FPN; 기능 피라미드 네트워크)라고 불리는 이 아키텍처는 여러 애플리케이션에서 범용 기능 추출기로서 대폭 개선되었습니다. 기본 Faster R-CNN 시스템에서 FPN을 사용하여 벨과 휘파람 없이 COCO 탐지 벤치마크에서 최첨단 단일 모델 결과를 달성하여 COCO 2016 챌린지 우승자를 포함한 기존의 모든 단일 모델 엔트리를 능가합니다. 또한 GPU에서 6FPS로 실행할 수 있으므로 멀티스케일 물체 검출을 위한 실용적이고 정확한 솔루션입니다. 코드가 공개됩니다.

1. **Introduction**

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

매우 다른 스케일의 물체를 인식하는 것은 컴퓨터 비전의 근본적인 과제입니다. 이미지 피라미드 위에 구축된 특징 피라미드(이러한 특징을 가진 이미지 피라미드)는 표준 솔루션의 기초를 형성한다(그림 1(a)). 이러한 피라미드는 피라미드 내 레벨을 이동함으로써 물체의 스케일 변화가 상쇄된다는 점에서 스케일 불변입니다. 직관적으로 이 속성을 사용하면 모델이 위치와 피라미드 수준에서 모델을 스캔하여 광범위한 범위의 객체를 감지할 수 있습니다.

Featurized image pyramids were heavily used in the era of hand-engineered features. They were so critical that object detectors like DPM required dense scale sampling to achieve good results (e.g., 10 scales per octave). For recognition tasks, engineered features have largely been replaced with features computed by deep convolutional networks (ConvNets). Aside from being capable of representing higher-level semantics, ConvNets are also more robust to variance in scale and thus facilitate recognition from features computed on a single input scale (Fig. 1(b)). But even with this robustness, pyramids are still needed to get the most accurate results. All recent top entries in the ImageNet and COCO detection challenges use multi-scale testing on featurized image pyramids. The principle advantage of featurizing each level of an image pyramid is that it produces a multi-scale feature representation in which all levels are semantically strong, including the high-resolution levels.

피처라이즈된 이미지 피라미드는 수공예 기능 시대에 많이 사용되었습니다. DPM과 같은 물체 검출기는 매우 중요했기 때문에 좋은 결과를 얻기 위해 조밀한 척도 샘플링이 필요했다(예: 옥타브당 10 척도). 인식 작업의 경우, 엔지니어링된 기능은 주로 심층 컨볼루션 네트워크(ConvNets)에 의해 계산된 기능으로 대체되었습니다. ConvNets는 더 높은 수준의 의미론을 나타낼 수 있을 뿐만 아니라 규모의 분산에도 더 강력하며, 따라서 단일 입력 척도로 계산된 특징의 인식을 용이하게 한다(그림 1(b)). 그러나 이러한 견고함에도 불구하고 피라미드는 여전히 가장 정확한 결과를 얻기 위해 필요합니다. ImageNet 및 COCO 검출 과제에서 최근 상위 항목은 모두 피처화 이미지 피라미드에 대한 멀티스케일 테스트를 사용합니다. 이미지 피라미드의 각 레벨을 특징짓는 주요 장점은 고해상도 레벨을 포함한 모든 레벨이 의미적으로 강력한 다중 스케일 피쳐 표현을 생성한다는 것입니다.

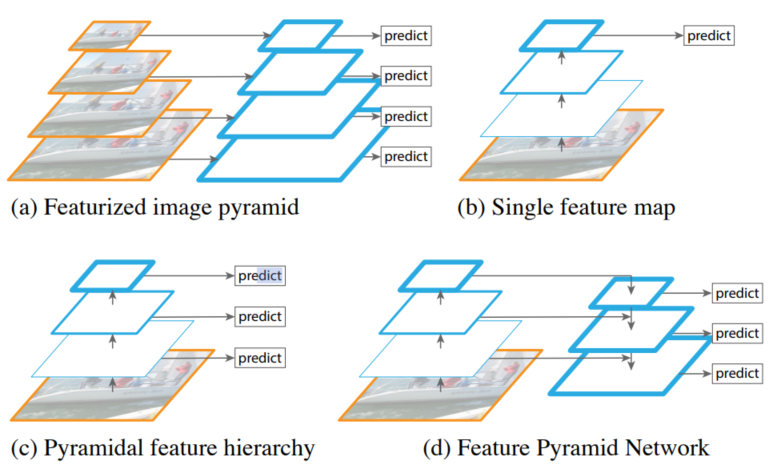


Figure 1. (a) Using an image pyramid to build a feature pyramid. Features are computed on each of the image scales independently, which is slow. (b) Recent detection systems have opted to use only single scale features for faster detection. (c) An alternative is to reuse the pyramidal feature hierarchy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

그림 1. (a) 이미지 피라미드를 사용한 피처 피라미드 구축 특징들은 각각의 이미지 스케일에서 독립적으로 계산되며, 속도는 느리다. (b) 최근의 검출 시스템은 빠른 검출을 위해 단일 스케일 특징만을 사용하는 것을 선택하고 있다. (c) 대안은 ConvNet에 의해 계산된 피라미드 특징 계층을 마치 피처라이즈된 이미지 피라미드인 것처럼 재사용하는 것이다. (d) 델이 제안하는 Feature Pyramid Network(FPN; 기능 피라미드 네트워크)는 (b)나 (c)와 같이 빠르지만 정확도가 더 높습니다. 이 그림에서 기능 맵은 파란색 윤곽선으로 나타나며, 굵은 윤곽선은 의미적으로 더 강력한 기능을 나타낸다.

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

그럼에도 불구하고 이미지 피라미드의 각 레벨을 특징짓는 데는 분명한 한계가 있다. 추론 시간이 상당히 증가하므로(예: 4배), 이 접근 방식은 실제 애플리케이션에 실용적이지 않다. 더욱이, 이미지 피라미드 상에서 엔드 투 엔드로 딥 네트워크를 훈련하는 것은 메모리 측면에서 실현 불가능하기 때문에, 이미지 피라미드는 테스트 시간에만 사용되므로, 열차/테스트 시간 추론 간에 불일치가 발생한다. 이러한 이유로 고속 및 고속 R-CNN에서는 기본 설정에서는 피처라이즈된 이미지 피라미드를 사용하지 않는다.

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

그러나 이미지 피라미드가 다중 스케일 피쳐 표현을 계산하는 유일한 방법은 아니다. 심층 ConvNet은 계층별로 특징 계층을 계산하고 하위 샘플링 계층을 사용하여 특징 계층은 고유한 멀티스케일, 피라미드 모양을 가진다. 이 네트워크 내 기능 계층은 다양한 공간 분해능의 기능 맵을 생성하지만 다른 깊이로 인해 발생하는 큰 의미적 갭을 초래한다. 고해상도 맵에는 객체 인식에 대한 표현 능력을 해치는 낮은 수준의 특징이 있다.

The Single Shot Detector (SSD) is one of the first attempts at using a ConvNet’s pyramidal feature hierarchy as if it were a featurized image pyramid (Fig. 1(c)). Ideally, the SSD-style pyramid would reuse the multi-scale feature maps from different layers computed in the forward pass and thus come free of cost. But to avoid using low-level features SSD foregoes reusing already computed layers and instead builds the pyramid starting from high up in the network (e.g., conv4 3 of VGG nets) and then by adding several new layers. Thus it misses the opportunity to reuse the higher-resolution maps of the feature hierarchy. We show that these are important for detecting small objects.

싱글샷 검출기(SSD)는 ConvNet의 피라미드 특징 계층을 마치 피처라이즈된 이미지 피라미드인 것처럼 사용하기 위한 첫 번째 시도 중 하나이다(그림 1(c)). SSD 스타일의 피라미드는 포워드 패스에서 계산된 여러 계층의 멀티스케일 기능 맵을 재사용하여 무료로 제공하는 것이 이상적입니다. 그러나 낮은 수준의 기능을 사용하지 않기 위해 SSD는 이미 계산된 계층을 재사용하지 않고 네트워크 상층(예: VGG 넷의 conv4 3)부터 시작하여 여러 개의 새로운 계층을 추가하여 피라미드를 구축합니다. 따라서 기능 계층의 고해상도 맵을 재사용할 기회를 놓칩니다. 우리는 이것들이 작은 물체를 감지하는 데 중요하다는 것을 보여준다.

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

이 백서의 목표는 ConvNet의 특징 계층 구조의 피라미드 형상을 자연스럽게 활용하는 동시에 모든 규모의 강력한 의미를 갖는 특징 피라미드를 만드는 것이다. 이 목표를 달성하기 위해 하향 경로와 측면 연결을 통해 저해상도, 의미론적으로 강한 특징과 고해상도, 의미론적으로 약한 특징을 결합한 아키텍처에 의존한다(그림 1(d)). 그 결과, 모든 레벨에서 풍부한 의미를 가지며, 단일 입력 이미지 스케일로 신속하게 구축되는 기능 피라미드가 실현됩니다. 즉, 피처라이즈된 이미지 피라미드를 대체하기 위해 사용할 수 있는 네트워크 내 피처 피라미드를 작성하는 방법에 대해 설명합니다.이 피라미드는 표현력, 속도 또는 메모리를 희생하지 않습니다.

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

톱다운 접속과 스킵 접속을 채택한 유사한 아키텍처가 최근 조사에서 인기를 끌고 있습니다. 이들의 목표는 예측이 이루어지는 고해상도의 단일 개략적 기능 지도를 만드는 것이다(그림 2 위). 반대로 우리의 방법은 각 수준에서 독립적으로 예측(예: 객체 탐지)이 이루어지는 특징 피라미드로서 아키텍처를 활용한다(그림 2 하단). 우리의 모델은 이 작품들에서는 탐험되지 않은 특징적인 이미지 피라미드를 반영하고 있다.

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

검출 및 분할을 위해 다양한 시스템에서 Feature Pyramid Network(FPN) 벨과 휘파람 없이, 우리는 단순히 FPN과 기본적인 Faster R-CNN 검출기를 기반으로 한 까다로운 COCO 검출 벤치마크에 대한 최첨단 단일 모델 결과를 보고하며, 기존의 모든 중공학적 단일 모델 수상자를 능가한다. 절제 실험에서 FPN은 경계 상자 제안의 경우 평균 회수(AR)를 8.0 포인트 크게 증가시키고, 개체 검출의 경우 Resnets의 강력한 단일 스케일 기준선에 비해 COCO 스타일 평균 정밀도(AP)를 2.3 포인트, PASCAL 스타일 AP를 3.8 포인트 향상시킨다는 것을 발견했습니다. 또한 우리의 방법은 마스크 제안으로 쉽게 확장되며 이미지 피라미드에 크게 의존하는 최첨단 방법보다 인스턴스 분할 AR과 속도를 향상시킨다.

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

또한 피라미드 구조는 모든 스케일로 엔드 투 엔드로 훈련할 수 있으며 열차/테스트 시 일관되게 사용되므로 이미지 피라미드를 사용하면 메모리 사용이 불가능합니다. 그 결과, FPN은 기존의 모든 최첨단 방식보다 높은 정확도를 달성할 수 있습니다. 또한 이 개선은 단일 스케일 기준선에 비해 테스트 시간을 늘리지 않고 달성할 수 있습니다. 이러한 진보는 향후의 연구 및 응용에 도움이 될 것으로 생각됩니다. 우리의 코드가 공개될 것이다.

1. **Related Work**

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

**수공학적 특징과 초기 신경 네트워크.** SIFT 피쳐는 원래 축척 공간 극단에서 추출되어 특징점 매칭에 사용되었다. HOG 피쳐와 이후 SIFT 피쳐는 전체 이미지 피라미드에 걸쳐 조밀하게 계산되었다. 이러한 HOG와 SIFT 피라미드는 이미지 분류, 물체 감지, 인체 자세 추정 등을 위한 수많은 작업에 사용되고 있습니다. 또한, 형상화된 이미지 피라미드를 신속하게 계산하는 데 상당한 관심이 있었습니다. 달러 등은 우선 희박하게 샘플링된 (규모로) 피라미드를 계산한 후 누락된 레벨을 보간함으로써 빠른 피라미드 계산을 보여주었다. HOG 및 SIFT 이전에는 ConvNets를 사용한 얼굴 검출에 관한 초기 작업이 이미지 피라미드 상의 얕은 네트워크를 계산하여 스케일 전체에 걸친 얼굴을 검출했습니다.

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

**Deep ConvNet 객체 디텍터.** 현대의 심층 ConvNets의 개발로 OverFeat 및 R-CNN과 같은 물체 감지기는 정확도가 극적으로 향상되었다. OverFeat는 이미지 피라미드에 슬라이딩 윈도우 검출기로 ConvNet을 적용하여 초기 뉴럴 네트워크 얼굴 검출기와 유사한 전략을 채택했다. R-CNN은 ConvNet으로 분류하기 전에 각 제안이 스케일 정규화되는 지역 제안 기반 전략을 채택했다. SPPnet은 그러한 지역 기반 검출기를 단일 이미지 척도에서 추출한 특징 지도에 훨씬 더 효율적으로 적용할 수 있음을 입증했다. Fast R-CNN 및 Fast R-CNN과 같은 최근 보다 정확한 검출 방법은 정확도와 속도 간의 균형을 잘 맞추기 때문에 단일 척도에서 계산된 기능을 사용하는 것을 지지한다. 그러나 멀티스케일 검출은 특히 작은 물체의 경우 여전히 더 나은 성능을 발휘합니다.

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

**여러 레이어를 사용하는 메서드.** ConvNet의 여러 레이어를 사용함으로써 검출과 분할이 향상됩니다. FCN은 의미 세분화를 계산하기 위해 여러 척도에 걸쳐 각 범주의 부분 점수를 집계합니다. 하이퍼컬럼에서는 오브젝트인스턴스 분할에도 비슷한 방법을 사용합니다. 그 외의 몇개의 어프로치(HyperNet, ParseNet, 및 ION)는, 예측치를 계산하기 전에 복수의 레이어의 기능을 접속합니다.이것은 변환된 피쳐의 합계와 같습니다. SSD와 MS-CNN은 특징이나 점수를 결합하지 않고 기능 계층의 여러 계층에서 개체를 예측합니다.

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

U-Net과 SharpMask, 얼굴검출용 재조합 네트워크, 키포인트 추정용 적층 모래시계 네트워크 등 해상도 및 시맨틱레벨에 걸쳐 저레벨 기능 맵을 관련짓는 횡방향/스킵 접속을 이용하는 최근 방법이 있습니다. Ghiasi 등은 분할을 점진적으로 개선하기 위해 FCN을 위한 라플라시안 피라미드 프레젠테이션을 제시한다. 이러한 방법들은 피라미드 형태의 건축물을 채택하고 있지만, 모든 수준에서 독립적으로 예측되는 형상화된 이미지 피라미드와는 다르다. 그림 2를 참조한다. 실제로 그림 2(위)의 피라미드 건축에서는 여러 척도의 물체를 인식하기 위해 이미지 피라미드가 여전히 필요합니다.

1. **Feature Pyramid Networks**

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

우리의 목표는 ConvNet의 피라미드 피쳐 계층을 활용하여 낮은 레벨에서 높은 레벨의 시멘틱스를 가진 피쳐 피라미드를 구축하는 것이다. 결과적인 기능 피라미드 네트워크는 범용적이며, 이 백서에서는 슬라이딩 윈도우 제안자(지역 제안 네트워크, 줄여서 RPN)와 지역 기반 검출기(Fast R-CNN)에 초점을 맞추고 있다. 또한 6항의 인스턴스 분할 제안으로 FPN을 일반화한다.

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

우리의 방법은 임의의 크기의 단일 스케일 이미지를 입력으로 받아들이고, 다양한 수준에서 비례적으로 크기가 조정된 피쳐 맵을 완전한 컨볼루션 방식으로 출력한다. 이 프로세스는 백본 컨볼루션 아키텍처와는 독립적이며, 본 백서에서는 ResNets를 사용하여 결과를 제시합니다. 피라미드의 건설은 아래에서 소개한 바와 같이 상향식 경로, 하향식 경로 및 측면 연결을 포함한다.

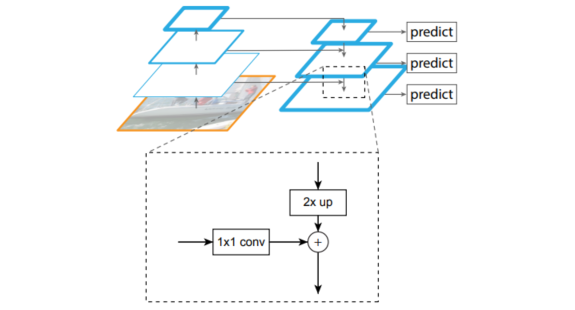


Figure 3. A building block illustrating the lateral connection and the top-down pathway, merged by addition.

그림 3 측면 연결과 하향식 경로를 나타내는 구성 블록으로, 추가로 병합됩니다.

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

**상향식 경로.** 상향 패스는 backbone ConvNet의 피드포워드 계산으로 스케일링 스텝2의 여러 스케일에서의 기능 맵으로 구성된 기능 계층을 계산합니다. 대부분의 경우 같은 크기의 출력 맵을 생성하는 레이어가 있으며, 이러한 레이어는 같은 네트워크 스테이지에 있다고 합니다. 피처 피라미드에서는 각 스테이지마다 1개의 피라미드 레벨을 정의합니다. 각 스테이지의 마지막 레이어의 출력을 피처 맵의 레퍼런스 세트로 선택하고, 이것을 풍부하게 해 피라미드를 작성합니다. 각 스테이지의 가장 깊은 레이어에는 가장 강한 특징이 있기 때문에 이 선택은 자연스러운 것입니다.

Specifically, for ResNets we use the feature activations output by each stage’s last residual block. We denote the output of these last residual blocks as {C2, C3, C4, C5} for conv2, conv3, conv4, and conv5 outputs, and note that they have strides of {4, 8, 16, 32} pixels with respect to the input image. We do not include conv1 into the pyramid due to its large memory footprint.

구체적으로는 ResNet의 경우 각 단계의 마지막 잔여 블록에 의한 기능 활성화 출력을 사용합니다. 이러한 마지막 잔차 블록의 출력을 conv2, conv3, conv4, conv5 출력에 대해 {C2, C3, C4, C5}로 나타내며 입력 이미지에 대해 {4, 8, 16, 32}픽셀의 진보를 보이고 있습니다. conv1은 메모리 용량이 크기 때문에 피라미드에 포함하지 않습니다.

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

**하향식 경로 및 측면 연결.** 하향식 경로는 공간적으로 거칠지만 의미적으로는 더 강한 피라미드 수준의 특징 맵을 상향 샘플링하여 고해상도 특징을 환각시킨다. 그런 다음 이러한 기능은 측면 연결을 통한 상향 경로에서 기능으로 강화됩니다. 각 횡방향 연결은 상향 경로와 하향 경로에서 동일한 공간 크기의 피쳐 맵을 병합한다. 상향식 기능 맵은 하위 수준의 의미론이지만 서브샘플링 횟수가 적기 때문에 액티베이션이 보다 정확하게 현지화됩니다.

Fig. 3 shows the building block that constructs our topdown feature maps. With a coarser-resolution feature map, we upsample the spatial resolution by a factor of 2 (using nearest neighbor upsampling for simplicity). The upsam3pled map is then merged with the corresponding bottom-up map (which undergoes a 1×1 convolutional layer to reduce channel dimensions) by element-wise addition. This process is iterated until the finest resolution map is generated. To start the iteration, we simply attach a 1×1 convolutional layer on C5 to produce the coarsest resolution map. Finally, we append a 3×3 convolution on each merged map to generate the final feature map, which is to reduce the aliasing effect of upsampling. This final set of feature maps is called {P2, P3, P4, P5}, corresponding to {C2, C3, C4, C5} that are respectively of the same spatial sizes.

그림 3은 하향식 피쳐 맵을 구성하는 구성 요소를 보여줍니다. 조잡한 해상도의 기능 맵에서는 공간 분해능을 2배로 업샘플링합니다(간단함을 위해 가장 가까운 인접 업샘플링 사용). 그런 다음 업샘플링된 맵은 요소별 추가에 의해 대응하는 상향식 맵(채널 치수를 줄이기 위해 1×1 컨볼루션 레이어를 거친다)과 병합됩니다. 이 프로세스는 최적의 해상도 맵이 생성될 때까지 반복됩니다. 반복을 시작하기 위해 C5에 1×1 컨볼루션 레이어를 부착하여 가장 거친 해상도 맵을 생성한다. 마지막으로 각 병합된 맵에 3×3 회전수를 추가하여 최종 피쳐 맵을 생성함으로써 업샘플링의 앨리어싱 효과를 감소시킨다. 이 기능 맵의 마지막 집합을 각각 공간 크기가 동일한 {C2, C3, C4, C5}에 해당하는 {P2, P3, P4, P5}이라고 합니다.

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

피라미드의 모든 레벨은 기존의 특징 이미지 피라미드와 같이 공유 분류자/레지스터를 사용하기 때문에 모든 특징 맵에서 특징 치수(채널 수, d로 표시)를 고정합니다. 이 문서에서는 d = 256으로 설정했으며, 따라서 모든 추가 컨볼루션 계층은 256 채널 출력을 가집니다. 이러한 추가 레이어에는 비선형성이 없습니다.이러한 레이어에는 경미한 영향이 있는 것이 실증적으로 판명되었습니다.

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

심플함은 설계의 핵심이며, 델의 모델은 많은 디자인 선택에 강하다는 것을 알 수 있었습니다. 보다 정교한 블록(예: 다층 잔여 블록을 연결로 사용)으로 실험하고 약간 더 나은 결과를 관찰했습니다. 이 문서에서는 보다 나은 연결 모듈을 설계하는 데 중점을 두지 않기 때문에 위에서 설명한 단순한 설계를 선택합니다.

1. **Applications**

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

당사의 방법은 심층 ConvNet 내부에 기능 피라미드를 구축하기 위한 일반적인 솔루션입니다. 다음에서는 경계 상자 제안 생성을 위해 RPN에서, 객체 검출을 위해 Fast R-CNN에서 우리의 방법을 채택한다. 방법의 단순성과 효과를 입증하기 위해 기능 피라미드에 적용할 때 [29, 11]의 원래 시스템을 최소한으로 수정합니다.

* 1. **Feature Pyramid Networks for RPN**

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

RPN은 슬라이딩 윈도우클래스에 구애받지 않는 오브젝트 디텍터입니다. 원래의 RPN 설계에서는 싱글스케일 컨볼루션 피쳐 맵 위에 조밀한 3×3 슬라이딩 윈도 상에서 작은 서브네트워크를 평가하여 오브젝트/비오브젝트 바이너리 분류 및 바운딩 박스 회귀를 실시한다. 이는 3×3 컨볼루션 층에 이어 분류 및 회귀를 위한 두 개의 형제 1×1 컨볼루션에 의해 실현되며, 이를 네트워크 헤드라고 한다. 객체/비객체 기준 및 경계 상자 회귀 대상은 앵커라고 하는 참조 상자 세트에 대해 정의됩니다. 앵커는 다양한 모양의 객체를 커버하기 위해 미리 정의된 여러 축척 및 가로 세로 비율입니다.

We adapt RPN by replacing the single-scale feature map with our FPN. We attach a head of the same design (3×3 conv and two sibling 1×1 convs) to each level on our feature pyramid. Because the head slides densely over all locations in all pyramid levels, it is not necessary to have multi-scale anchors on a specific level. Instead, we assign anchors of a single scale to each level. Formally, we define the anchors to have areas of {32^2, 64^2, 128^2, 256^2, 512^2} pixels on {P2, P3, P4, P5, P6} respectively.1 As in [29] we also use anchors of multiple aspect ratios {1:2, 1:1, 2:1} at each level. So in total there are 15 anchors over the pyramid.

싱글스케일 기능 맵을 FPN으로 대체함으로써 RPN을 적응시킵니다. 피처 피라미드의 각 레벨에 같은 디자인의 헤드(3×3 cv, 형제 1×1 cv 2개)를 부착합니다. 헤드는 모든 피라미드 레벨의 모든 위치에서 밀착되기 때문에 특정 레벨에 멀티스케일 앵커가 있을 필요는 없습니다. 대신 각 레벨에 단일 척도의 앵커를 할당합니다. 정식으로 앵커는 {P2, P3, P4, P5, P6}에 각각 {32^2, 64^2, 128^2, 256^2, 512^2}픽셀의 영역을 갖도록 정의한다.1 [29]와 같이 각 레벨에서 여러 종횡비 {1:2, 1:1, 2:1}의 앵커를 사용합니다. 그래서 피라미드 위에는 총 15개의 앵커가 있다.

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

우리는 [29]에서와 같이 지상 정보 경계 상자와 함께 조합 교차(Intersection-over-Union, IoU) 비율을 기반으로 앵커에 교육 라벨을 할당한다. 일반적으로 앵커는 주어진 지상 정보 상자에 대해 가장 높은 IoU를 가지거나 지상 정보 상자에 0.7 이상의 IoU를 가질 경우 양의 라벨이 할당되며, 모든 지상 정보 상자에 대해 0.3 미만의 IoU를 가질 경우 음의 라벨이 할당됩니다. 지표 참치 상자의 척도는 피라미드 수준에 할당하는 데 명시적으로 사용되지 않습니다. 대신 지표 참치 상자는 피라미드 수준에 할당된 앵커와 연결됩니다. 따라서, 우리는 [29]의 규칙 외에 추가 규칙을 도입하지 않습니다.

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

헤드의 파라미터는 모든 피처 피라미드 레벨에서 공유됩니다. 또한 파라미터를 공유하지 않고 대안을 평가하고 유사한 정확도를 관찰했습니다. 매개 변수 공유의 좋은 성과는 피라미드의 모든 수준이 유사한 의미 수준을 공유함을 나타냅니다. 이 장점은 공통 헤드 분류기를 모든 이미지 스케일로 계산된 피쳐에 적용할 수 있는 특징화 이미지 피라미드를 사용하는 것과 유사합니다.

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

위의 적응을 통해 RPN은 [29]와 같은 방법으로 당사의 FPN을 사용하여 자연스럽게 훈련 및 테스트할 수 있습니다. 우리는 실험에서 구현 세부 사항에 대해 자세히 설명합니다.

* 1. **Feature Pyramid Networks for Fast R-CNN**

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

Fast R-CNN은 관심 영역(RoI) 풀링을 사용하여 피쳐를 추출하는 지역 기반 객체 디텍터입니다. Fast R-CNN은 단일 스케일피처 맵에서 가장 일반적으로 실행됩니다. FPN에서 사용하려면 피라미드 레벨에 다른 스케일의 RoI를 할당해야 합니다.

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

피처 피라미드는 이미지 피라미드에서 생성된 것처럼 보입니다. 따라서 이미지 피라미드에서 실행되는 경우 지역 기반 검출기의 할당 전략을 적용할 수 있습니다. 정식으로 다음과 같이 피처 피라미드의 레벨 Pk에 폭 w와 높이 h의 RoI를 할당합니다.



Here 224 is the canonical ImageNet pre-training size, and k0 is the target level on which an RoI with w × h = 2242 should be mapped into. Analogous to the ResNet-based Faster R-CNN system that uses C4 as the single-scale feature map, we set k0 to 4. Intuitively, Eqn. (1) means that if the RoI’s scale becomes smaller (say, 1/2 of 224), it should be mapped into a finer-resolution level (say, k = 3).

여기서 224는 표준 ImageNet 사전 훈련 크기이고 k0은 w × h = 2242의 RoI를 매핑해야 하는 대상 수준입니다. C4를 단일 스케일피처 맵으로 사용하는 ResNet 기반의 고속 R-CNN 시스템과 마찬가지로 k0을 4로 설정합니다. 직관적으로, Eqn. (1)은 RoI의 눈금이 작아지면(예: 224의 1/2), 더 미세한 해상도 수준(예: k = 3)으로 매핑되어야 함을 의미합니다.

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

예측기 헤드(Fast R-CNN에서 헤드는 클래스별 분류기 및 경계 상자 회귀기)를 모든 수준의 모든 RoI에 부착한다. 레벨에 관계없이 모든 헤드는 파라미터를 공유합니다. [16]에서 ResNet의 con5 레이어 (9 레이어 딥 서브 네트워크)가 con4 기능 위에 헤드로서 채택되고 있지만, 우리의 방법은 기능 피라미드를 구성하기 위해 이미 conv5를 이용했습니다. 따라서 [16]과 달리 RoI 풀링을 채택하여 7×7 기능을 추출하고 최종 분류 및 경계 상자 회귀 레이어 전에 1,024-d 완전 연결(fc) 레이어 2개(각각 ReLU)를 부착하면 됩니다. ResNets에는 사전 교육을 받은 fc 계층이 없기 때문에 이러한 계층은 임의로 초기화됩니다. 표준 conv5 헤드와 비교하여 당사의 2-fc MLP 헤드는 무게가 가볍고 속도가 빠릅니다.

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

이러한 적응을 바탕으로 기능 피라미드 위에서 Fast RCNN을 훈련하고 테스트할 수 있습니다. 구현에 대한 자세한 내용은 실험 섹션에 나와 있습니다.

1. **Experiments on Object Detection**

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

80 카테고리 COCO 검출 데이터 세트에 대한 실험을 실시합니다. 80k의 열차 이미지와 35k의 발상 부분 집합(트레인밸35k)을 조합하여 훈련하고 발상 부분 집합(미니벌)의 5k에 대해 애블레이션을 보고한다. 또한 라벨이 공개되지 않은 표준 테스트 세트(test-std)에 대한 최종 결과를 보고한다.

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

일반적인 관행과 마찬가지로 모든 네트워크 백본은 ImageNet1k 분류 세트에 대해 사전 교육을 받은 후 검출 데이터 세트에 대해 미세 조정됩니다. 사전에 훈련을 받은 ResNet-50 및 ResNet-101 모델을 사용합니다. 우리의 코드는 Caffe2를 사용한 py-faster-rcn의 재실장입니다.

* 1. **Region Proposal with RPN**

We evaluate the COCO-style Average Recall (AR) and AR on small, medium, and large objects (ARs, ARm, and ARl) following the definitions in [21]. We report results for 100 and 1000 proposals per images (AR100 and AR1k).

우리는 [21]의 정의에 따라 소형, 중형, 대형 객체(AR, ARM, ARL)에 대한 COCO 스타일의 평균 리콜(AR)과 AR을 평가한다. 이미지(AR100 및 AR1k)당 100 및 1000건의 제안 결과를 보고합니다.

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

**구현 상세.** 표 1의 모든 아키텍처는 엔드 투 엔드로 훈련되어 있습니다. 입력 이미지의 사이즈가 짧아진 쪽이 800픽셀이 되도록 조정됩니다. 8대의 GPU에 동기화된 SGD 트레이닝을 채용하고 있습니다. 미니 배치에는 GPU당 2개의 이미지와 이미지당 256개의 앵커가 포함됩니다. 무게 감소는 0.0001이고 운동량은 0.9입니다. 학습률은 첫 번째 30k 미니배츠의 경우 0.02이고 다음 10k의 경우 0.002입니다. 모든 RPN 실험(베이스라인 포함)에 대해 이미지 외부에 있는 앵커박스를 트레이닝용으로 포함합니다.이는 이러한 앵커박스가 무시되는 [29]와는 다릅니다. 기타 구현 세부 사항은 [29]와 같습니다. 8개의 GPU에서 FPN을 사용한RPN 트레이닝은 COCO에서 약8시간 소요됩니다.

* + 1. **Ablation Experiments**

**Comparisons with baselines.** For fair comparisons with original RPNs [29], we run two baselines (Table 1(a, b)) using the single-scale map of C4 (the same as [16]) or C5, both using the same hyper-parameters as ours, including using 5 scale anchors of {32^2, 64^2, 128^2, 256^2, 512^2}. Table 1 (b) shows no advantage over (a), indicating that a single higherlevel feature map is not enough because there is a trade-off between coarser resolutions and stronger semantics.

**기준선과의 비교.** 원래의 RPN [29]과(와) 공평한 비교를 위해, 우리는 C4의 단일 스케일 맵 ([16]과 동일) 또는 C5를 사용하여 두 개의 기준선 (표 1(a, b))을 실행하며, 둘 다 {32^2, 64^2, 128^2, 256^2, 512^2의 5개의 스케일 앵커를 사용한다. 표 1 (b)는 (a)에 비해 장점이 없음을 보여주며, 이는 보다 거친 해상도와 보다 강력한 의미론 사이에 트레이드오프가 존재하기 때문에 단일 상위 수준의 기능 맵으로는 충분하지 않음을 나타낸다.

Placing FPN in RPN improves AR1k to 56.3 (Table 1(c)), which is 8.0 points increase over the single-scale RPN baseline (Table 1 (a)). In addition, the performance on small objects (AR1ks) is boosted by a large margin of 12.9 points. Our pyramid representation greatly improves RPN’s robustness to object scale variation.

FPN을 RPN에 배치하면 AR1k가 56.3으로 개선됩니다(표 1(c)).이것은 싱글스케일 RPN 베이스라인(표 1(a))보다 8.0 포인트 증가합니다. 또한 소형 객체(AR1ks)에 대한 성능은 12.9점이라는 큰 폭으로 향상되었습니다. 우리의 피라미드 표현은 객체 규모 변동에 대한 RPN의 견고성을 크게 향상시킨다.

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

**하향식 농축은 얼마나 중요합니까?** 표 1(d)은 톱다운 경로가 없는 특징 피라미드의 결과를 보여준다. 이 변경에 의해 1×1 가로방향 연결과 3×3 회전의 연속이 상향식 피라미드에 부착된다. 이 아키텍처는 피라미드 형상 계층을 재사용하는 효과를 시뮬레이션합니다(그림 1(b)).

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

표 1(d)의 결과는 RPN 기준과 동등하며 우리의 기준보다 훨씬 뒤떨어져 있습니다. 우리는 이것이 특히 매우 깊은 ResNets에 대해 상향식 피라미드(그림 1(b))의 서로 다른 수준 사이에 큰 의미적 차이가 있기 때문이라고 추측한다. 또한 헤드의 매개변수를 공유하지 않고 표 1(d)의 변형을 평가했지만 유사한 성능 저하를 관찰했다. 이 문제는 수준별 헤드로 간단히 해결할 수 없습니다.

**How important are lateral connections?** Table 1(e) shows the ablation results of a top-down feature pyramid without the 1×1 lateral connections. This top-down pyramid has strong semantic features and fine resolutions. But we argue that the locations of these features are not precise, because these maps have been downsampled and upsampled several times. More precise locations of features can be directly passed from the finer levels of the bottom-up maps via the lateral connections to the top-down maps. As a results, FPN has an AR1k score 10 points higher than Table 1(e).

**측면 연결부는 얼마나 중요합니까?** 표 1(e)은 1×1 가로 방향 연결이 없는 하향식 특징 피라미드의 절제 결과를 보여줍니다. 이 하향식 피라미드는 강한 의미적 특징과 훌륭한 해상도를 가지고 있다. 그러나 이러한 맵은 여러 번 다운샘플링 및 업샘플링이 되었기 때문에 이러한 기능의 위치가 정확하지 않다고 주장합니다. 보다 정밀한 기능의 위치는 측면 연결을 통해 상향식 맵의 미세한 수준에서 하향식 맵으로 직접 전달할 수 있습니다. 그 결과 FPN은 표 1(e)보다 AR1k 점수가 10점 더 높습니다.

**How important are pyramid representations?** Instead of resorting to pyramid representations, one can attach the head to the highest-resolution, strongly semantic feature maps of P2 (i.e., the finest level in our pyramids). Similar to the single-scale baselines, we assign all anchors to the P2 feature map. This variant (Table 1(f)) is better than the baseline but inferior to our approach. RPN is a sliding window detector with a fixed window size, so scanning over pyramid levels can increase its robustness to scale variance.

**피라미드 표현은 얼마나 중요합니까?** 피라미드 표현에 의존하는 대신, P2의 가장 고해상도의 강한 의미적 특징 지도(즉, 피라미드에서 가장 뛰어난 수준)에 머리를 부착할 수 있다. 단일 스케일 기준선과 마찬가지로 모든 앵커를 P2 기능 맵에 할당합니다. 이 변종(표 1(f))은 기준선보다 낫지만 우리의 접근 방식보다 못하다. RPN은 고정된 윈도우 크기를 가진 슬라이딩 윈도우 검출기이므로 피라미드 레벨에서 스캔하면 견고성을 높여 분산 크기를 조정할 수 있습니다.

In addition, we note that using P2 alone leads to more anchors (750k, Table 1(f)) caused by its large spatial resolution. This result suggests that a larger number of anchors is not sufficient in itself to improve accuracy.

또한 P2를 단독으로 사용하면 큰 공간 분해능으로 인해 더 많은 앵커(750k, 표 1(f))가 발생한다는 점에 주목한다. 이 결과는 앵커 수가 많을수록 정확도 향상에 충분하지 않다는 것을 나타냅니다.

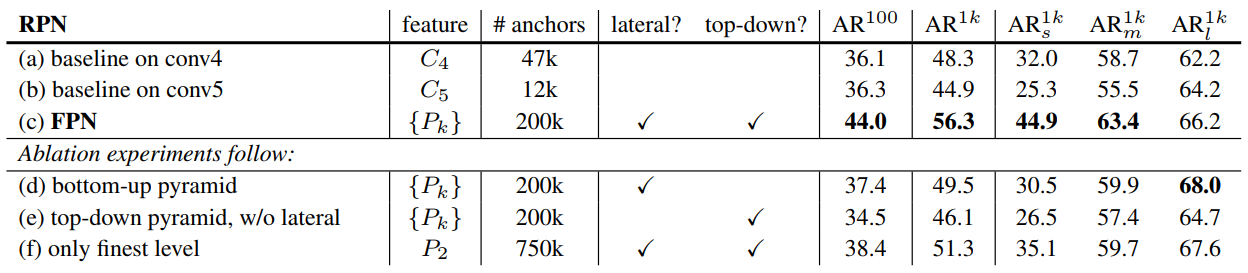


Table 1. Bounding box proposal results using RPN [29], evaluated on the COCO minival set. All models are trained on trainval35k. The columns “lateral” and “top-down” denote the presence of lateral and top-down connections, respectively. The column “feature” denotes the feature maps on which the heads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

표 1. COCO 미니벌 세트에서 평가된 RPN [29]를 사용한 경계 상자 제안 결과. 모든 모델은 trainval35k로 훈련됩니다. "측면" 열과 "상하향" 열은 각각 횡방향 연결과 하향식 연결의 존재를 나타냅니다. "feature" 열은 헤드가 부착된 피쳐 맵을 나타냅니다. 모든 결과는 ResNet-50을 기반으로 하며 동일한 하이퍼 파라미터를 공유합니다.

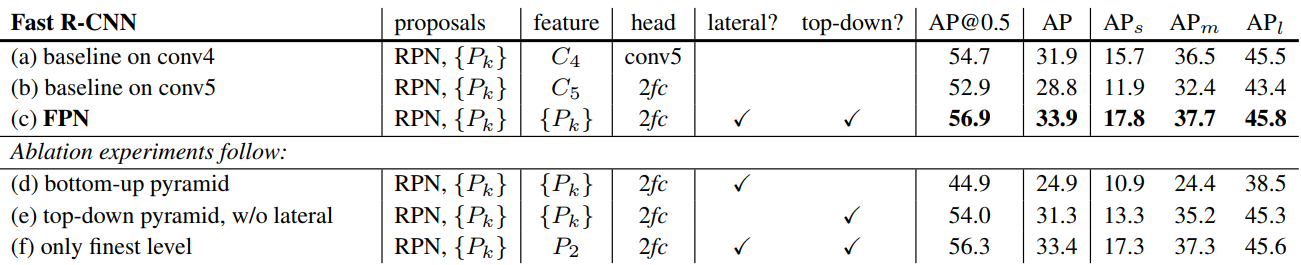


Table 2. Object detection results using Fast R-CNN [11] on a fixed set of proposals (RPN, {Pk}, Table 1(c)), evaluated on the COCO minival set. Models are trained on the trainval35k set. All results are based on ResNet-50 and share the same hyper-parameters.

표 2 COCO 미니벌 세트에서 평가된 고정 제안 세트(RPN, {Pk}, 표 1(c))에서 Fast R-CNN [11]을 사용한 객체 감지 결과. 모델은 trainval35k 세트로 훈련됩니다. 모든 결과는 ResNet-50을 기반으로 하며 동일한 하이퍼 파라미터를 공유합니다.

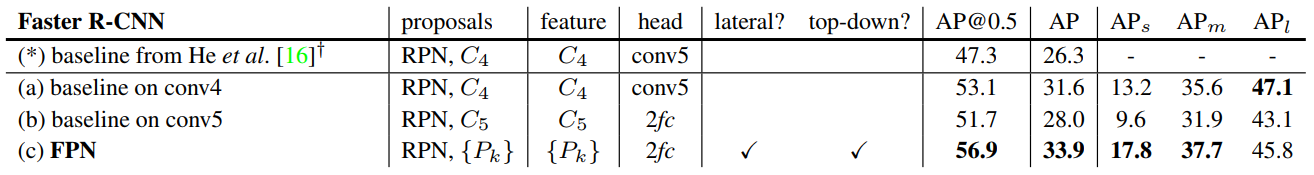


Table 3. Object detection results using Faster R-CNN [29] evaluated on the COCO minival set. The backbone network for RPN are consistent with Fast R-CNN. Models are trained on the trainval35k set and use ResNet-50. † Provided by authors of [16].

표 3 COCO 미니벌 세트에서 평가된 Faster R-CNN [29]를 사용한 물체 감지 결과. RPN의 백본네트워크는 Fast R-CNN과 일치합니다.모델은 trainval35k 세트로 훈련되어 ResNet-50을 사용합니다.[16]의 저자에 의해 제공됩니다.

**5.2. Object Detection with Fast/Faster R-CNN**

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

다음으로 지역 기반(비슬라이딩 윈도우) 디텍터의 FPN을 조사합니다. COCO 스타일의 Average Precision(AP)과 PASCAL 스타일의 AP(단일 IoTU 임계값 0.5)에 의한 객체 검출을 평가합니다. 또한 [21]의 정의에 따라 소형, 중형 및 대형 객체(AP, APm 및 APL)에 대해서도 COCO AP를 보고합니다.

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

**구현 상세.** 입력 이미지의 사이즈가 짧아진 쪽이 800픽셀이 되도록 조정됩니다. 동기화된 SGD를 사용하여 8개의 GPU에서 모델을 훈련합니다. 각 미니 배치에는 GPU당 2개의 이미지와 512개의 RoI가 포함됩니다. 무게 감소는 0.0001이고 운동량은 0.9입니다. 학습률은 최초 60k 미니배츠의 경우 0.02이고 다음 20k의 경우 0.002입니다. 트레이닝에는 이미지당 2000개의 RoI를, 테스트에는 1000개의 RoI를 사용합니다. FPN을 사용한 Fast R-CNN 트레이닝에는 COCO 데이터 세트에 약 10시간이 소요됩니다.

**5.2.1. Fast R-CNN (on fixed proposals)**

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

지역 기반 검출기에만 대한 FPN의 영향을 더 잘 조사하기 위해, 우리는 고정된 제안 세트에 대해 Fast R-CNN의 제거를 수행한다. 검출기에 의해 인식되는 작은 물체에서 우수한 성능을 발휘하기 때문에 FPN(표 1(c))의 RPN에 의해 계산된 제안을 동결하기로 선택한다. 알기 쉽게 하기 위해 Fast R-CNN과 RPN 간에는 지정된 경우를 제외하고 기능을 공유하지 않습니다.

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

ResNet 기반의 Fast R-CNN 베이스라인으로서 [16]에 이어 출력 사이즈가 14×14인 RoI 풀링을 채택하고 모든 conv5 레이어를 헤드의 숨겨진 레이어로 부착합니다. 이로써 표 2(a)의 AP는 31.9가 됩니다. 표 2(b)는 2개의 숨겨진 FC 레이어를 가진 MLP 헤드를 이용하는 베이스라인으로 아키텍처의 헤드와 유사합니다. 28.8의 AP를 얻어서 표 2(a)의 기준선에 비해 2-fc 헤드가 직교 이점을 제공하지 않음을 나타냅니다.

Table 2(c) shows the results of our FPN in Fast R-CNN. Comparing with the baseline in Table 2(a), our method improves AP by 2.0 points and small object AP by 2.1 points. Comparing with the baseline that also adopts a 2fc head (Table 2(b)), our method improves AP by 5.1 points.5 These comparisons indicate that our feature pyramid is superior to single-scale features for a region-based object detector.

표 2(c)는 Fast R-CNN에서의 FPN 결과를 나타내고 있습니다.표 2(a)의 베이스라인과 비교하여 이 방법은 AP를 2.0 포인트, 소형 객체 AP를 2.1 포인트 향상시킵니다. 2fc 헤드를 채택한 베이스라인(표 2(b))과 비교하여 이 방법은 AP를 5.1포인트 향상시킵니다.5 이러한 비교는 우리의 특징 피라미드가 지역 기반 물체 검출기의 단일 스케일 특징보다 우수하다는 것을 나타낸다.

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

표 2(d)와 (e)는 위에서 설명한 RPN 서브섹션에서 관찰한 것과 마찬가지로 하향식 연결 또는 횡방향 연결의 삭제가 열악함을 나타내고 있습니다. Top-down 접속(표 2(d))을 삭제하면 정확도가 크게 저하되어 고해상도 맵에서 Fast R-CNN이 낮은 수준의 기능을 사용하는 데 어려움을 겪는다는 것을 알 수 있습니다.

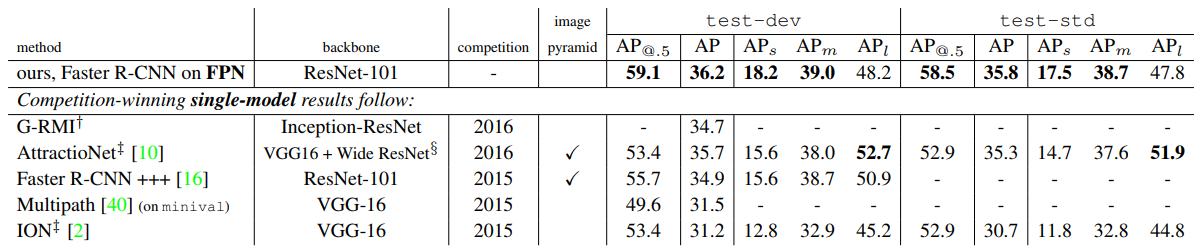


Table 4. Comparisons of single-model results on the COCO detection benchmark. Some results were not available on the test-std set, so we also include the test-dev results (and for Multipath [40] on minival). : This entry of AttractioNet [10] adopts VGG-16 for proposals and Wide ResNet [39] for object detection, so is not strictly a single-model result.

표 4 COCO 검출 벤치마크의 단일 모델 결과 비교. test-std 세트에서는 일부 결과를 사용할 수 없었기 때문에 test-dev 결과(및 미니밴드의 멀티패스[40]의 경우)도 포함되어 있습니다. : 이 AttractioNet [10]의 엔트리에서는 제안에는 VGG-16을, 객체 검출에는 Wide ResNet[39]를 채택하고 있기 때문에, 엄밀하게는 단일 모델의 결과는 아닙니다.

In Table 2(f), we adopt Fast R-CNN on the single finest scale feature map of P2. Its result (33.4 AP) is marginally worse than that of using all pyramid levels (33.9 AP, Table 2(c)). We argue that this is because RoI pooling is a warping-like operation, which is less sensitive to the region’s scales. Despite the good accuracy of this variant, it is based on the RPN proposals of {Pk} and has thus already benefited from the pyramid representation.

표 2(f)에서는 P2의 단일 정밀도 피쳐 맵에 Fast R-CNN을 채택하고 있다. 그 결과(33.4 AP)는 모든 피라미드 수준을 사용한 결과(33.9 AP, 표 2(c))보다 약간 더 나쁘다. 이는 RoI 풀링이 뒤틀리는 것과 같은 동작이기 때문에 지역의 규모에 덜 민감하기 때문이라고 우리는 주장한다. 이 변형은 정확도가 높지만 {Pk}의 RPN 제안을 기반으로 하기 때문에 피라미드 표현으로부터 이미 이점을 얻었습니다.

**5.2.2. Faster R-CNN (on consistent proposals)**

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

위에서는 검출기를 조사하기 위해 고정 제안 세트를 사용했다. 단, 고속 R-CNN 시스템[29]에서는 기능 공유를 가능하게 하기 위해 RPN과 고속 R-CNN이 같은 네트워크 백본을 사용해야 합니다. 표 3은 RPN과 고속 R-CNN을 위한 일관된 백본 아키텍처를 사용하는 방법과 두 개의 기준선 간의 비교를 보여줍니다. 표 3(a)은 [16]에서 설명된 바와 같이 기준선 고속 R-CNN 시스템의 재현을 보여줍니다. FPN(표 3(c))은 이 강력한 기준선보다 2.3포인트 AP 및 3.8포인트 AP@0.5로 더 우수합니다.

Note that Table 3(a) and (b) are baselines that are much stronger than the baseline provided by He et al. [16] in Table 3(\*). We find the following implementations contribute to the gap: (i) We use an image scale of 800 pixels instead of 600 in [11, 16]; (ii) We train with 512 RoIs per image which accelerate convergence, in contrast to 64 RoIs in [11, 16]; (iii) We use 5 scale anchors instead of 4 in [16] (adding 322); (iv) At test time we use 1000 proposals per image instead of 300 in [16]. So comparing with He et al.’s ResNet50 Faster R-CNN baseline in Table 3(\*), our method improves AP by 7.6 points and AP@0.5 by 9.6 points.

표 3(a)과 (b)는 He 등이 제공한 기준선보다 훨씬 강한 기준선이다. 표 3(\*)의 [16]. 갭의 원인이 되는 실장은 다음과 같습니다.(i) [11, 16]에서는 600픽셀이 아닌 800픽셀의 이미지 스케일을 사용하고 (ii) [11, 16]에서는 64개의 RoI를 사용하는 것과 대조적으로 (ii) [16](322)에서는 5개의 스케일 앵커를 사용하고 있습니다.[16]의 300 대신. 따라서 표 3(\*)에 있는 He 등의 ResNet50 Faster R-CNN 기준과 비교하여 이 방법은 AP를 7.6포인트, AP@0.5를 9.6포인트 향상시킨다.

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

**기능의 공유.** 위에서는 단순성을 위해 RPN과 Fast R-CNN 간에 기능을 공유하지 않습니다. 표 5에서는 [29]에 설명된 4단계 훈련에 따른 공유 기능을 평가합니다. [29]와 마찬가지로 기능을 공유하면 정확도가 약간 향상됩니다. 기능 공유를 통해 테스트 시간도 단축됩니다.

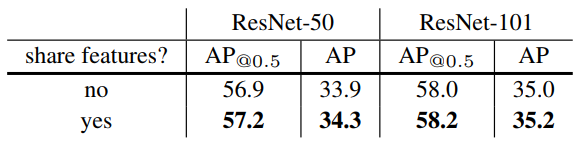


Table 5. More object detection results using Faster R-CNN and our FPNs, evaluated on minival. Sharing features increases train time by 1.5× (using 4-step training [29]), but reduces test time.

표 5 더 많은 객체 검출 결과가 고속 R-CNN 및 FPN을 사용하여 미니벌로 평가됩니다. 기능을 공유하면 열차 시간이 1.5배 증가하지만 (4단계 교육 사용 [29]), 테스트 시간은 감소합니다.

**Running time.** With feature sharing, our FPN-based Faster R-CNN system has inference time of 0.148 seconds per image on a single NVIDIA M40 GPU for ResNet-50, and 0.172 seconds for ResNet-101.6 As a comparison, the single-scale ResNet-50 baseline in Table 3(a) runs at 0.32 seconds. Our method introduces small extra cost by the extra layers in the FPN, but has a lighter weight head. Overall our system is faster than the ResNet-based Faster R-CNN counterpart. We believe the efficiency and simplicity of our method will benefit future research and applications.

**실행 시간** 기능 공유를 통해 FPN 기반의 고속 R-CNN 시스템은 ResNet-50의 경우 단일 NVIDIA M40 GPU에서 이미지당 추론 시간이 0.148초, ResNet-101.6의 경우 0.172초입니다.이것에 비해 표 3(a)의 싱글스케일 ResNet-50 베이스라인은 032초에 실행됩니다. 우리의 방법은 FPN의 추가 레이어에 의해 적은 추가 비용을 발생시키지만, 무게 헤드가 더 가볍습니다. 전체적으로 당사의 시스템은 ResNet 기반의 Faster R-CNN보다 더 빠릅니다. 델의 방법의 효율과 심플함은, 장래의 연구나 애플리케이션에 도움이 된다고 생각하고 있습니다.

**5.2.3. Comparing with COCO Competition Winners**

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

표 5의 ResNet-101 모델은 기본 학습 속도 일정으로 충분히 훈련되지 않았습니다. 따라서 Fast R-CNN 스텝을 교육할 때 각 학습 속도마다 미니 목욕 횟수를 2배 늘립니다. 따라서 기능을 공유하지 않고 최소값 AP를 35.6으로 늘립니다. 이 모델은 표 4와 같이 COCO 검출 리더보드에 제출한 모델입니다. 시간이 한정되어 있기 때문에 기능 공유 버전을 평가하지 않았습니다. 이것은 표 5에서 암시한 바와 같이 약간 개선될 것입니다.

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

표 4는 2016년 우승자 G-RMI와 2015년 우승자 Faster R-CNN++를 포함한 COCO 대회 우승자의 단일 모델 결과와 비교한 것입니다. 벨과 휘파람을 추가하지 않고도 NAT의 단일 모델 진입은 강력하고 고도로 설계된 경쟁업체를 능가합니다.

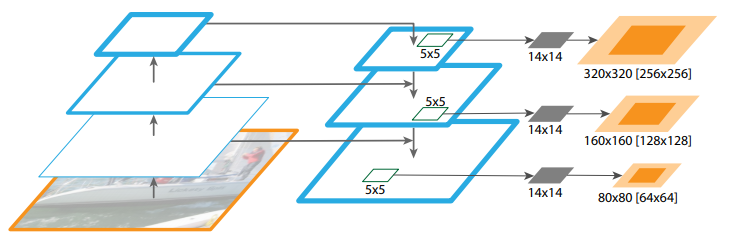


Figure 4. FPN for object segment proposals. The feature pyramid is constructed with identical structure as for object detection. We apply a small MLP on 5×5 windows to generate dense object segments with output dimension of 14×14. Shown in orange are the size of the image regions the mask corresponds to for each pyramid level (levels P3−5 are shown here). Both the corresponding image region size (light orange) and canonical object size (dark orange) are shown. Half octaves are handled by an MLP on 7x7 windows (7 ≈ 5√2), not shown here. Details are in the appendix.

그림 4. 객체 세그먼트 제안의 FPN. 피처 피라미드는 객체 검출과 동일한 구조로 구성되어 있습니다. 5×5 윈도우에 작은 MLP를 적용하여 출력 치수가 14×14인 고밀도 객체 세그먼트를 생성합니다. 주황색으로 표시된 것은 각 피라미드 레벨에 대응하는 이미지 영역의 크기입니다(레벨 P3-5는 여기에 표시되어 있습니다). 해당하는 영상 영역 크기(연한 주황색)와 표준 개체 크기(진한 주황색)가 모두 표시됩니다. 하프 옥타브는 7x7 윈도(7 µ 5 µ 2)에서 MLP에 의해 처리됩니다.여기서는 설명하지 않습니다. 자세한 내용은 부록을 참조하십시오.

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

test-dev 세트에서는 기존의 최선의 결과보다 AP의 0.5포인트(36.2 대 35.7)와 AP@0.5의 3.4포인트(59.1 대 55.7)가 증가합니다. 이 방법은 이미지 피라미드에 의존하지 않고 단일 입력 이미지 스케일만 사용하지만 소규모 객체에서는 여전히 뛰어난 AP를 가지고 있다는 점에 유의해야 합니다. 이는 이전 방법으로 고해상도 영상 입력을 통해서만 달성할 수 있습니다.

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

게다가, 우리의 방법은 반복 회귀, 하드 네거티브 마이닝, 컨텍스트 모델링, 더 강력한 데이터 확대 등과 같은 많은 일반적인 개선 사항을 활용하지 않는다. 이러한 개선 FPNs하고 정확도를 올릴 것 보완적이다.

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

최근 FPN은 검출, 인스턴스 분할, 키포인트 추정 등 COCO 경쟁의 모든 트랙에서 새로운 상위 결과를 가능하게 했습니다. 자세한 내용은 [14]를 참조하십시오.

1. **Extensions: Segmentation Proposals**

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

우리의 방법은 일반적인 피라미드 표현으로 객체 감지 이외의 응용 프로그램에서 사용할 수 있다. 이 섹션에서는 DeepMask/SharpMask 프레임워크에 따라 FPN을 사용하여 세그멘테이션프로포절 생성에 대해 설명합니다.

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

DeepMask/SharpMask는 인스턴스 세그먼트 및 객체/비개체 점수를 예측하기 위해 이미지 크롭에 대해 교육되었습니다. 추론 시 이들 모델은 이미지 내에서 조밀한 제안을 생성하기 위해 컨볼루션 방식으로 실행됩니다. 세그먼트를 여러 척도로 생성하려면 이미지 피라미드가 필요합니다.

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

마스크 프로포절 생성에 FPN을 쉽게 적용할 수 있습니다. 우리는 훈련과 추론 모두에 완전한 컨볼루션 설정을 사용한다. 5.1절과 같이 피처 피라미드를 구성하고 d=128로 설정합니다. 특징 피라미드의 각 레벨 위에 작은 5×5 MLP를 적용하여 14×14 마스크와 객체 점수를 완전 컨볼루션 방식으로 예측한다(그림 4 참조). 또한 [27, 28]의 이미지 피라미드에서 옥타브당 2개의 스케일을 사용하여 입력 크기 7×7의 두 번째 MLP를 사용하여 반 옥타브를 처리한다. 2개의 MLP는 RPN의 앵커와 같은 역할을 합니다. 아키텍처는 엔드 투 엔드로 트레이닝됩니다.실장 상세 내용은 부록에 기재되어 있습니다.

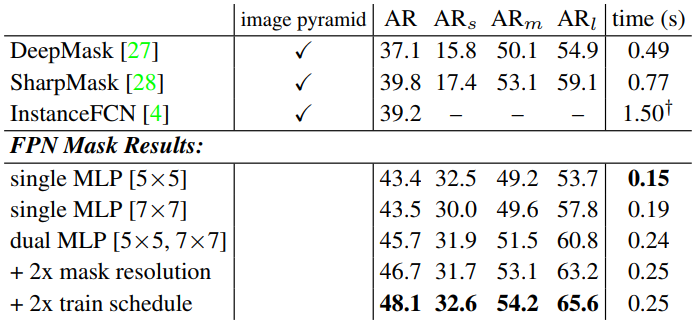


Table 6. Instance segmentation proposals evaluated on the first 5k COCO val images. All models are trained on the train set. DeepMask, SharpMask, and FPN use ResNet-50 while InstanceFCN uses VGG-16. DeepMask and SharpMask performance is computed with models available from https://github. com/facebookresearch/deepmask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

표 6 첫 번째 5k COCO val 영상에 대해 평가된 인스턴스 분할 제안. 모든 모델은 기차 세트로 훈련됩니다. DeepMask, SharpMask 및 FPN은 인스턴스 중에 ResNet-50을 사용합니다.FCN은 VGG-16을 사용합니다. DeepMask 및 SharpMask 성능은 https://github. com/facebookrearch/deepmask (둘 다 '줌' 변형)에서 사용할 수 있는 모델로 계산됩니다. § 런타임은 NVIDIA M40 GPU(인스턴스 제외)에서 측정됩니다. 느린 K40에 기반한 FCN 타이밍

* 1. **Segmentation Proposal Results**

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

결과는 표 6과 같다. 항상 1000개의 제안서에 대해 segment AR과 segment AR을 소형, 중형, 대형 객체에 대해 보고합니다. 단일 5×5 MLP를 사용하는 기본 FPN 모델은 43.4의 AR을 달성합니다. 약간 더 큰 7×7 MLP로 전환해도 정확도는 거의 변경되지 않습니다. 두 개의 MLP를 함께 사용하면 정확도가 45.7AR로 높아집니다. 마스크 출력 크기를 14×14에서 28×28로 늘리면 AR이 다른 점으로 증가합니다(크기가 클수록 정확도가 저하되기 시작합니다). 마지막으로, 교육 반복 횟수를 두 배로 늘리면 AR이 48.1로 증가합니다.

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

또한 DeepMask[27], SharpMask[28] 및 Instance와의 비교도 보고합니다.FCN [4] 마스크 제안 생성의 최신 기술 방식입니다. 우리는 이러한 접근법의 정확성을 8.3점 AR 이상 능가한다. 특히 작은 물체의 정확도는 거의 두 배입니다.

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

기존 마스크 제안 방법은 조밀하게 샘플링된 이미지 피라미드(예: [27, 28]의 2 {-2:0.5:1} 크기)를 기반으로 하므로 계산 비용이 많이 든다. FPN을 기반으로 한 접근 방식이 상당히 빠릅니다(6~7FPS로 동작). 이러한 결과는 우리의 모델이 일반적인 특징 추출기이며 다른 멀티 스케일 감지 문제에 대한 이미지 피라미드를 대체할 수 있음을 보여준다.

1. **Conclusion**

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

ConvNets 내부에 기능 피라미드를 구축하기 위한 깨끗하고 단순한 프레임워크를 제시했습니다. 델의 방법은, 복수의 강력한 베이스 라인과 경쟁 우위자에 비해 큰폭으로 개선되고 있습니다. 따라서 이미지 피라미드를 계산할 필요 없이 피처 피라미드의 연구와 적용을 위한 실용적인 솔루션을 제공합니다. 마지막으로, 우리의 연구는 심층 ConvNets의 강력한 표현력과 변화를 확장하기 위한 암묵적인 견고성에도 불구하고 피라미드 표현을 사용하여 멀티스케일 문제를 명시적으로 해결하는 것이 여전히 중요하다는 것을 시사한다.

**A. Implementation of Segmentation Proposals**

We use our feature pyramid networks to efficiently generate object segment proposals, adopting an image-centric training strategy popular for object detection. Our FPN mask generation model inherits many of the ideas and motivations from DeepMask/SharpMask. However, in contrast to these models, which were trained on image crops and used a densely sampled image pyramid for inference, we perform fully-convolutional training for mask prediction on a feature pyramid. While this requires changing many of the specifics, our implementation remains similar in spirit to DeepMask. Specifically, to define the label of a mask instance at each sliding window, we think of this window as being a crop on the input image, allowing us to inherit definitions of positives/negatives from DeepMask. We give more details next, see also Fig. 4 for a visualization.

피처 피라미드 네트워크를 사용하여 객체 세그먼트 제안을 효율적으로 생성하고 객체 검출에 널리 사용되는 이미지 중심 교육 전략을 채택합니다. 당사의 FPN 마스크 생성 모델은 DeepMask/SharpMask에서 많은 아이디어와 동기를 계승합니다. 그러나 이미지 작물에 대해 교육을 받고 추론을 위해 조밀하게 샘플링된 이미지 피라미드를 사용한 이러한 모델과 대조적으로, 우리는 특징 피라미드에서 마스크 예측을 위한 완전한 컨볼루션 훈련을 수행한다. 이를 위해서는 많은 세부 사항을 변경해야 하지만, NAT의 구현 방식은 DeepMask와 유사합니다. 구체적으로는 각 슬라이딩 창에서 마스크인스턴스의 라벨을 정의하기 위해 이 창을 입력 이미지의 자른 것으로 간주하여 DeepMask에서 긍정/부정 정의를 상속할 수 있습니다. 다음으로 자세한 내용을 설명하겠습니다.시각화에 대해서는 그림 4도 참조해 주십시오.

We construct the feature pyramid with P2−6 using the same architecture as described in Sec. 5.1. We set d = 128. Each level of our feature pyramid is used for predicting masks at a different scale. As in DeepMask, we define the scale of a mask as the max of its width and height. Masks with scales of {32, 64, 128, 256, 512} pixels map to {P2, P3, P4, P5, P6}, respectively, and are handled by a 5×5 MLP. As DeepMask uses a pyramid with half octaves, we use a second slightly larger MLP of size 7×7 (7 ≈ 5 √2) to handle half-octaves in our model (e.g., a 128√2 scale mask is predicted by the 7×7 MLP on P4). Objects at intermediate scales are mapped to the nearest scale in log space.

5.1항에서 설명한 것과 동일한 아키텍처를 사용하여 P2-6으로 피처 피라미드를 구성합니다. d = 128로 설정합니다. 우리의 특징 피라미드의 각 레벨은 다른 스케일로 마스크를 예측하는 데 사용됩니다. DeepMask에서와 같이 마스크의 스케일을 마스크의 폭과 높이의 최대값으로 정의합니다. 스케일 {32,64,128,256,512}픽셀의 마스크는 각각 {P2,P3,P4,P5,P6}에 매핑되어 5×5 MLP로 처리됩니다.DeepMask는 반옥타브의 피라미드를 사용하므로 크기가 7×7(775 half2 half5 half)인 MLP를 사용합니다. 중간 축척에 있는 개체는 로그 공간의 가장 가까운 축척에 매핑됩니다.

As the MLP must predict objects at a range of scales for each pyramid level (specifically a half octave range), some padding must be given around the canonical object size. We use 25% padding. This means that the mask output over {P2, P3, P4, P5, P6} maps to {40, 80, 160, 320, 640} sized image regions for the 5×5 MLP (and to √2 larger corresponding sizes for the 7×7 MLP).

MLP는 각 피라미드 레벨에 대해 스케일 범위(특히 0.5 옥타브 범위)에서 객체를 예측해야 하므로 표준 객체 크기 주위에 일부 패딩을 제공해야 한다. 패딩 25% 사용. 즉, {P2, P3, P4, P5, P6} 위의 마스크 출력이 5×5 MLP의 경우 {40, 80, 160, 320, 640} 크기의 이미지 영역(및 7×7 MLP의 경우 해당 크기가 2개 이상)에 매핑됩니다.

Each spatial position in the feature map is used to predict a mask at a different location. Specifically, at scale Pk, each spatial position in the feature map is used to predict the mask whose center falls within 2k pixels of that location (corresponding to ±1 cell offset in the feature map). If no object center falls within this range, the location is considered a negative, and, as in DeepMask, is used only for training the score branch and not the mask branch.

피쳐 맵의 각 공간 위치는 다른 위치의 마스크를 예측하는 데 사용됩니다. 구체적으로는, 스케일 Pk에서, 특징 맵내의 각 공간 위치는, 중심이 그 위치의 2 k픽셀 이내(특징 맵내의 ±1 셀 오프셋에 상당)인 마스크를 예측하기 위해서 사용된다. 이 범위 내에 객체 센터가 없는 경우 위치는 음으로 간주되며 DeepMask와 같이 마스크 분기가 아닌 점수 분기에만 사용됩니다.

The MLP we use for predicting the mask and score is fairly simple. We apply a 5×5 kernel with 512 outputs, followed by sibling fully connected layers to predict a 14×14 mask (142 outputs) and object score (1 output). The model is implemented in a fully convolutional manner (using 1×1 convolutions in place of fully connected layers). The 7×7 MLP for handling objects at half octave scales is identical to the 5×5 MLP except for its larger input region.

마스크와 점수를 예측하기 위해 사용하는 MLP는 매우 간단합니다. 512개의 출력이 있는 5×5 커널을 적용한 후 14×14 마스크(142개의 출력)와 객체 점수(1개의 출력)를 예측하기 위해 형제자매 완전 연결 레이어를 적용한다. 모델은 완전 컨볼루션 방식으로 구현됩니다(완전 연결된 레이어 대신 1×1 컨볼루션 사용). 반 옥타브 스케일로 물체를 다루기 위한 7×7 MLP는 입력 영역이 더 크다는 점을 제외하고 5×5 MLP와 동일합니다.

During training, we randomly sample 2048 examples per mini-batch (128 examples per image from 16 images) with a positive/negative sampling ratio of 1:3. The mask loss is given 10× higher weight than the score loss. This model is trained end-to-end on 8 GPUs using synchronized SGD (2 images per GPU). We start with a learning rate of 0.03 and train for 80k mini-batches, dividing the learning rate by 10 after 60k mini-batches. The image scale is set to 800 pixels during training and testing (we do not use scale jitter). During inference our fully-convolutional model predicts scores at all positions and scales and masks at the 1000 highest scoring locations. We do not perform any non-maximum suppression or post-processing.

트레이닝 중에 미니배치당 2048개의 샘플(16개의 이미지에서 이미지당 128개의 샘플)을 1:3의 양의 샘플/음성의 샘플 비율로 랜덤으로 추출했습니다. 마스크 손실은 점수 손실보다 10배 높은 가중치가 부여됩니다. 이 모델은 동기화된 SGD(GPU당 2개의 이미지)를 사용하여 8개의 GPU에서 엔드 투 엔드로 훈련됩니다. 0.03의 학습률로 시작하여 80,000회의 미니목욕 훈련을 실시하여 60,000회의 미니목욕 후 학습률을 10으로 나눕니다. 이미지 스케일은 훈련 및 테스트 중에 800픽셀로 설정됩니다(스케일 지터는 사용하지 않습니다). 추론하는 동안, 우리의 완전 컨볼루션 모델은 모든 위치에서 점수를 예측하고, 가장 높은 점수 1000개 위치에서 척도 및 마스크를 예측한다. 최대가 아닌 억제나 후처리는 실시하지 않습니다.