**Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition**

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun

**Abstract**

Existing deep convolutional neural networks (CNNs) require a fixed-size (e.g., 224×224) input image. This requirement is “artificial” and may reduce the recognition accuracy for the images or sub-images of an arbitrary size/scale. In this work, we equip the networks with another pooling strategy, “spatial pyramid pooling”, to eliminate the above requirement. The new network structure, called SPP-net, can generate a fixed-length representation regardless of image size/scale. Pyramid pooling is also robust to object deformations. With these advantages, SPP-net should in general improve all CNN-based image classification methods. On the ImageNet 2012 dataset, we demonstrate that SPP-net boosts the accuracy of a variety of CNN architectures despite their different designs. On the Pascal VOC 2007 and Caltech101 datasets, SPP-net achieves state-of-the-art classification results using a single full-image representation and no fine-tuning.

기존의 심층 컨볼루션 뉴럴 네트워크(CNN)에는 고정 크기(예: 224×224)의 입력 이미지가 필요하다. 이 요구 사항은 "인위적"이며 임의 크기/스케일의 영상 또는 하위 영상의 인식 정확도를 저하시킬 수 있습니다. 이 작업에서는 상기 요건을 제거하기 위해 네트워크에 또 다른 풀링 전략인 "공간 피라미드 풀링"을 설치한다. SPP-net이라고 불리는 새로운 네트워크 구조는 이미지 크기/스케일에 관계없이 고정 길이 표현을 생성할 수 있습니다. 피라미드 풀링은 객체 변형에도 강력합니다. 이러한 장점을 활용하여 SPP-net은 일반적으로 모든 CNN 기반 이미지 분류 방법을 개선해야 한다. ImageNet 2012 데이터셋에서는 SPP-Net이 다양한 설계에도 불구하고 다양한 CNN 아키텍처의 정확성을 향상시킨다는 것을 시연합니다. Pascal VOC 2007 및 Caltech101 데이터셋에서 SPP-net은 미세 조정 없이 단일 전체 이미지 표현을 사용하여 최첨단 분류 결과를 달성합니다.

1. **Introduction**

We are witnessing a rapid, revolutionary change in our vision community, mainly caused by deep convolutional neural networks (CNNs) and the availability of large scale training data. Deep-networks based approaches have recently been substantially improving upon the state of the art in image classification, object detection, many other recognition tasks, and even non-recognition tasks.

우리는 주로 심층 컨볼루션 뉴럴 네트워크(CNN)와 대규모 훈련 데이터의 가용성에 의해 야기된 비전 커뮤니티의 빠르고 혁명적인 변화를 목격하고 있다. 딥 네트워크 기반 접근법은 최근 이미지 분류, 객체 감지, 기타 많은 인식 작업, 심지어 비인식 작업에 있어 최첨단 기술을 통해 크게 개선되고 있습니다.

However, there is a technical issue in the training and testing of the CNNs: the prevalent CNNs require a fixed input image size (e.g., 224×224), which limits both the aspect ratio and the scale of the input image. When applied to images of arbitrary sizes, current methods mostly fit the input image to the fixed size, either via cropping or via warping, as shown in Figure 1 (top). But the cropped region may not contain the entire object, while the warped content may result in unwanted geometric distortion. Recognition accuracy can be compromised due to the content loss or distortion. Besides, a pre-defined scale may not be suitable when object scales vary. Fixing input sizes overlooks the issues involving scales.

그러나 CNN의 훈련과 테스트에는 기술적인 문제가 있다. 일반적인 CNN은 고정 입력 이미지 크기(예: 224×224)를 필요로 하며, 이는 입력 이미지의 가로 세로 비율과 크기를 모두 제한한다. 임의의 크기의 영상에 적용할 경우, 현재 방법은 그림 1(위)에서와 같이 자르기 또는 뒤틀림을 통해 입력 이미지를 대부분 고정 크기에 맞춥니다. 그러나 잘린 영역은 개체 전체를 포함하지 않을 수 있으며 내용물이 왜곡되면 원치 않는 기하학적 왜곡이 발생할 수 있습니다. 컨텐츠 손실 또는 왜곡으로 인해 인식 정확도가 저하될 수 있습니다. 또한 오브젝트 스케일이 다른 경우에는 미리 정의된 스케일이 적합하지 않을 수 있습니다. 입력 크기를 수정하면 척도와 관련된 문제를 간과할 수 있습니다.

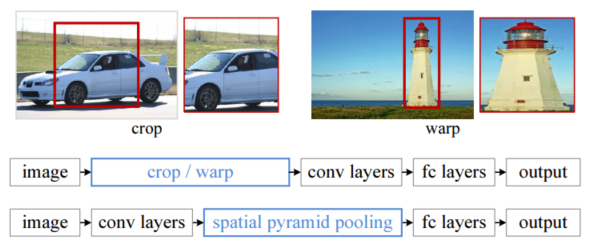


Figure 1: Top: cropping or warping to fit a fixed size. Middle: a conventional CNN. Bottom: our spatial pyramid pooling network structure.

그림 1: 상단: 일정한 크기에 맞게 자르기 또는 뒤틀리기 중간: 전통적인 CNN. 아래쪽: 우리의 공간 피라미드 풀링 네트워크 구조.

So why do CNNs require a fixed input size? A CNN mainly consists of two parts: convolutional layers, and fully-connected layers that follow. The convolutional layers operate in a sliding-window manner and output feature maps which represent the spatial arrangement of the activations (Figure 2). In fact, convolutional layers do not require a fixed image size and can generate feature maps of any sizes. On the other hand, the fully-connected layers need to have fixedsize/length input by their definition. Hence, the fixedsize constraint comes only from the fully-connected layers, which exist at a deeper stage of the network.

그렇다면 CNN은 왜 고정된 입력 크기를 필요로 할까요? CNN은 주로 두 부분으로 구성됩니다. 컨볼루션 계층과 그에 이은 완전히 연결된 계층입니다. 컨볼루션 층은 슬라이딩 윈도우 방식으로 작동하며 활성화의 공간적 배치를 나타내는 특징 맵을 출력한다(그림 2). 실제로 컨볼루션 레이어는 고정 이미지 크기를 필요로 하지 않으며 모든 크기의 피쳐 맵을 생성할 수 있습니다. 한편, 완전히 연결된 레이어에는 정의에 따라 고정 크기/길이 입력이 필요합니다. 따라서 고정 크기 제약은 네트워크의 더 깊은 단계에 존재하는 완전히 연결된 계층에서만 발생합니다.

In this paper, we introduce a spatial pyramid pooling (SPP) layer to remove the fixed-size constraint of the network. Specifically, we add an arXiv:1406.4729v4 [cs.CV] 23 Apr 20152 SPP layer on top of the last convolutional layer. The SPP layer pools the features and generates fixedlength outputs, which are then fed into the fullyconnected layers (or other classifiers). In other words, we perform some information “aggregation” at a deeper stage of the network hierarchy (between convolutional layers and fully-connected layers) to avoid the need for cropping or warping at the beginning. Figure 1 (bottom) shows the change of the network architecture by introducing the SPP layer. We call the new network structure SPP-net.

이 문서에서는 네트워크의 고정 크기 제약을 제거하기 위해 공간 피라미드 풀링(SPP) 계층을 소개한다. 구체적으로는 arXiv:1406.4729v4 [cs]를 추가합니다.CV] 2015년 4월 23일 마지막 컨볼루션 레이어 위에 SPP 레이어 2개. SPP 계층은 기능을 풀링하고 고정 길이 출력을 생성하여 완전히 연결된 계층(또는 기타 분류기)으로 공급합니다. 즉, 네트워크 계층의 보다 깊은 단계(컨볼루션레이어와 완전 접속 레이어 사이)에서 정보의 「집약」을 실시해, 최초로 크롭이나 왜곡을 회피합니다. 그림 1(아래)은 SPP 계층을 도입함으로써 네트워크 아키텍처의 변화를 보여줍니다. 새로운 네트워크 구조를 SPP-net이라고 부릅니다.

Spatial pyramid pooling (popularly known as spatial pyramid matching or SPM), as an extension of the Bag-of-Words (BoW) model, is one of the most successful methods in computer vision. It partitions the image into divisions from finer to coarser levels, and aggregates local features in them. SPP has long been a key component in the leading and competition-winning systems for classification and detection before the recent prevalence of CNNs. Nevertheless, SPP has not been considered in the context of CNNs. We note that SPP has several remarkable properties for deep CNNs: 1) SPP is able to generate a fixedlength output regardless of the input size, while the sliding window pooling used in the previous deep networks cannot; 2) SPP uses multi-level spatial bins, while the sliding window pooling uses only a single window size. Multi-level pooling has been shown to be robust to object deformations; 3) SPP can pool features extracted at variable scales thanks to the flexibility of input scales. Through experiments we show that all these factors elevate the recognition accuracy of deep networks.

BoW(Bag-of-Words) 모델의 확장으로서 공간 피라미드 풀링(일반적으로 공간 피라미드 매칭)은 컴퓨터 비전 분야에서 가장 성공적인 방법 중 하나입니다. 이미지를 미세 레벨에서 거친 레벨로 분할하여 로컬피처를 집약합니다. SPP는 최근 CNN이 보급되기 전까지 오랫동안 선도적이고 경쟁 우위를 점한 시스템의 핵심 구성요소였다. 그럼에도 불구하고 SPP는 CNN의 맥락에서 고려되지 않았다. SPP는 심층 CNN에 대해 몇 가지 주목할 만한 특성을 가지고 있습니다. 1) SPP는 입력 크기에 관계없이 고정 길이 출력을 생성할 수 있지만, 이전의 심층 네트워크에서 사용된 슬라이딩 윈도우 풀링은 그렇지 않습니다. 2) SPP는 다단계 공간 빈을 사용하는 반면 슬라이딩 윈도우 풀링은 단일 윈도우 크기만 사용합니다. 다단계 풀링은 객체 변형에 강한 것으로 나타났다. 3) SPP는 입력 척도의 유연성 덕분에 가변 척도로 추출된 특징을 풀링할 수 있다. 실험을 통해 이러한 모든 요소가 심층 네트워크의 인식 정확도를 높인다는 것을 알 수 있습니다.

SPP-net not only makes it possible to generate representations from arbitrarily sized images/windows for testing, but also allows us to feed images with varying sizes or scales during training. Training with variable-size images increases scale-invariance and reduces over-fitting. We develop a simple multi-size training method. For a single network to accept variable input sizes, we approximate it by multiple networks that share all parameters, while each of these networks is trained using a fixed input size. In each epoch we train the network with a given input size, and switch to another input size for the next epoch. Experiments show that this multi-size training converges just as the traditional single-size training, and leads to better testing accuracy.

SPP-net은 테스트를 위해 임의의 크기의 이미지/창에서 표현을 생성할 수 있을 뿐만 아니라 교육 중에 다양한 크기 또는 크기의 이미지를 제공할 수 있습니다. 가변 크기 이미지로 교육하면 스케일 불변성이 증가하고 오버핏이 줄어듭니다. 심플한 멀티 사이즈 트레이닝 방법을 개발합니다. 1개의 네트워크에서 다양한 입력 사이즈를 받아들이기 위해서는 모든 파라미터를 공유하는 복수의 네트워크에 의해 대략적인 값을 산출합니다.이러한 네트워크는 각각 고정 입력 사이즈를 사용하여 훈련됩니다. 각 에폭에서는, 특정의 입력 사이즈로 네트워크를 트레이닝 해, 다음의 에폭에서는 다른 입력 사이즈로 전환합니다. 실험에 따르면 이 멀티사이즈 트레이닝은 기존의 싱글사이즈 트레이닝과 동일하게 수렴되어 테스트 정확도가 향상됩니다.

The advantages of SPP are orthogonal to the specific CNN designs. In a series of controlled experiments on the ImageNet 2012 dataset, we demonstrate that SPP improves four different CNN architectures in existing publications (or their modifications), over the no-SPP counterparts. These architectures have various filter numbers/sizes, strides, depths, or other designs. It is thus reasonable for us to conjecture that SPP should improve more sophisticated (deeper and larger) convolutional architectures. SPP-net also shows state-of-the-art classification results on Caltech101 and Pascal VOC 2007 using only a single full-image representation and no fine-tuning.

SPP의 장점은 특정 CNN 설계와 직교합니다. ImageNet 2012 데이터 세트에 대한 일련의 통제된 실험에서 SPP는 기존 출판물(또는 그 수정 사항)에서 SPP가 SPP가 아닌 것에 비해 4개의 다른 CNN 아키텍처를 개선한다는 것을 입증했다. 이러한 아키텍처에는 다양한 필터 번호/사이즈, 단계, 깊이 또는 기타 설계가 있습니다. 따라서 SPP가 보다 정교한(더 작고 더 큰) 컨볼루션 아키텍처를 개선해야 한다고 추측하는 것은 타당합니다. SPP-net은 또한 Caltech101 및 Pascal VOC 2007에 대한 최신 분류 결과를 단일 전체 이미지 표시만으로 표시하며 미세 조정은 하지 않습니다.

SPP-net also shows great strength in object detection. In the leading object detection method R-CNN, the features from candidate windows are extracted via deep convolutional networks. This method shows remarkable detection accuracy on both the VOC and ImageNet datasets. But the feature computation in RCNN is time-consuming, because it repeatedly applies the deep convolutional networks to the raw pixels of thousands of warped regions per image. In this paper, we show that we can run the convolutional layers only once on the entire image (regardless of the number of windows), and then extract features by SPP-net on the feature maps. This method yields a speedup of over one hundred times over R-CNN. Note that training/running a detector on the feature maps (rather than image regions) is actually a more popular idea. But SPP-net inherits the power of the deep CNN feature maps and also the flexibility of SPP on arbitrary window sizes, which leads to outstanding accuracy and efficiency. In our experiment, the SPP-net-based system (built upon the R-CNN pipeline) computes features 24-102× faster than R-CNN, while has better or comparable accuracy. With the recent fast proposal method of EdgeBoxes, our system takes 0.5 seconds processing an image (including all steps). This makes our method practical for real-world applications.

SPP-net은 물체 감지에도 큰 힘을 발휘합니다. 선두 객체 검출 방법 R-CNN은 후보 윈도로부터의 특징을 심층 컨볼루션 네트워크를 통해 추출한다. 이 방법은 VOC 및 ImageNet 데이터 세트 모두에서 놀라운 탐지 정확도를 보여 줍니다. 그러나 RCNN의 기능 계산은 이미지당 수천 개의 왜곡된 영역의 원시 픽셀에 심층 컨볼루션 네트워크를 반복적으로 적용하기 때문에 시간이 많이 걸립니다. 이 백서에서는 전체 이미지에서 컨볼루션 레이어를 한 번만 실행하고(윈도우 수에 관계없이) 피쳐 맵에서 SPP-net으로 피쳐를 추출할 수 있음을 보여 준다. 이 방법을 사용하면 R-CNN보다 속도가 100배 이상 향상됩니다. 실제로 (이미지 영역이 아닌) 피쳐 맵에서 디텍터를 교육/실행하는 것이 더 일반적인 아이디어입니다. 그러나 SPP-net은 심층 CNN 기능 맵의 기능과 임의의 창 크기에 대한 SPP의 유연성을 계승하여 탁월한 정확성과 효율성을 제공합니다. 이 실험에서 SPP-net 기반 시스템(R-CNN 파이프라인에 기반)은 R-CNN보다 24-102배 더 빠르게 특징을 계산하면서도 정확도가 더 우수하거나 그에 필적하는 성능을 갖췄다. 최근 EdgeBoxes의 빠른 제안 방법을 사용하면 이미지 처리(모든 단계 포함)에 0.5초가 소요됩니다. 이것에 의해, 실제의 애플리케이션에 있어서, 델의 방법이 실용화됩니다.

A preliminary version of this manuscript has been published in ECCV 2014. Based on this work, we attended the competition of ILSVRC 2014, and ranked **#2 in object detection and #3 in image classification** (both are provided-data-only tracks) among all 38 teams. There are a few modifications made for ILSVRC 2014. We show that the SPP-nets can boost various networks that are deeper and larger (Sec. 3.1.2-3.1.4) over the no-SPP counterparts. Further, driven by our detection framework, we find that multi-view testing on feature maps with flexibly located/sized windows (Sec. 3.1.5) can increase the classification accuracy. This manuscript also provides the details of these modifications.

We have released the code to facilitate future research (http://research.microsoft.com/en-us/um/people/kahe/).

이 원고의 예비 버전은 ECCV 2014에 게재되었습니다. 이 작업을 바탕으로 ILSVRC 2014 대회에 참가하여 38개 팀 중 **객체 검출 2위, 이미지 분류 3위**(모두 데이터 전용 트랙)를 차지하였습니다. ILSVRC 2014에 대해 몇 가지 수정 사항이 있습니다. SPP-net은 SPP가 아닌 다른 네트워크보다 더 깊고 더 큰 다양한 네트워크(3.1.2-3.1.4항)를 지원할 수 있다는 것을 보여준다. 또한 탐지 프레임워크에 따라 유연하게 위치/크기 창을 가진 기능 지도(3.1.5항)에 대한 멀티 뷰 테스트를 통해 분류 정확도를 높일 수 있다. 이 원고는 또한 이러한 수정에 대한 자세한 내용을 제공한다.

향후 연구를 용이하게 하기 위해 코드를 공개했습니다(http://research.microsoft.com/en-us/um/people/kahe/)).

1. **Deep Networks with Spatial Pyramid Pooling**
   1. **Convolutional Layers and Future Maps**

Consider the popular seven-layer architectures. The first five layers are convolutional, some of which are followed by pooling layers. These pooling layers can also be considered as “convolutional”, in the sense that they are using sliding windows. The last two layers are fully connected, with an N-way softmax as the output, where N is the number of categories.

일반적인 7층 아키텍처에 대해 생각해 봅시다. 첫 번째 5개의 레이어는 컨볼루션이며, 그 중 일부는 풀링 레이어입니다. 이러한 풀링 레이어는 슬라이딩 윈도우를 사용한다는 점에서 "컨볼루션"으로 간주될 수도 있다. 마지막 2개의 레이어는 완전히 연결되어 있으며 출력은 N-way softmax입니다.여기서 N은 카테고리의 수입니다.

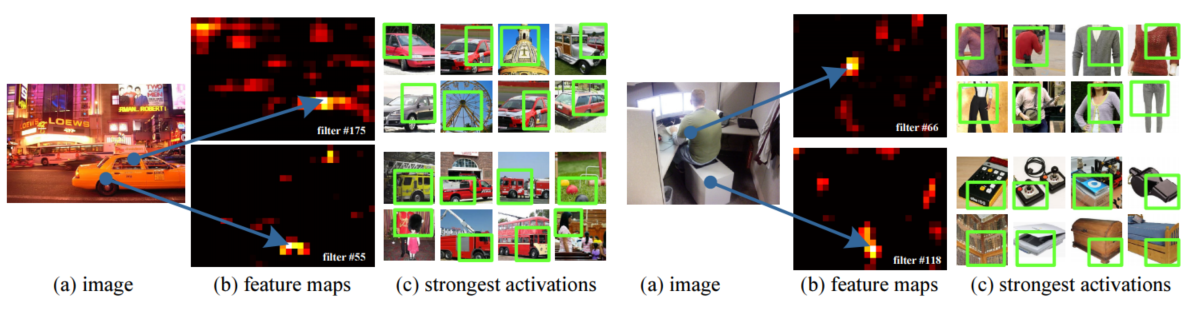


Figure 2: Visualization of the feature maps. (a) Two images in Pascal VOC 2007. (b) The feature maps of some conv5 filters. The arrows indicate the strongest responses and their corresponding positions in the images. (c) The ImageNet images that have the strongest responses of the corresponding filters. The green rectangles mark the receptive fields of the strongest responses.

그림 2: 피처 맵의 시각화 (a) Pascal VOC 2007의 2개의 이미지 (b) 일부 conv5 필터의 피처 맵. 화살표는 영상에서 가장 강력한 응답과 해당 위치를 나타냅니다. (c) 대응하는 필터의 응답이 가장 강한 ImageNet 영상. 녹색 직사각형은 가장 강력한 반응의 수용 필드를 나타냅니다.

The deep network described above needs a fixed image size. However, we notice that the requirement of fixed sizes is only due to the fully-connected layers that demand fixed-length vectors as inputs. On the other hand, the convolutional layers accept inputs of arbitrary sizes. The convolutional layers use sliding filters, and their outputs have roughly the same aspect ratio as the inputs. These outputs are known as feature maps - they involve not only the strength of the responses, but also their spatial positions.

위에서 설명한 딥 네트워크에는 고정 이미지 크기가 필요합니다. 그러나 고정 크기의 요건은 입력으로 고정 길이 벡터를 요구하는 완전히 연결된 계층에만 있다는 것을 알 수 있습니다. 한편, 컨볼루션 층은 임의의 크기의 입력을 받아들인다. 컨볼루션 층은 슬라이딩 필터를 사용하며, 그 출력은 입력과 거의 동일한 가로 세로 비율을 가진다. 이러한 출력은 기능 맵으로 알려져 있으며, 응답 강도뿐만 아니라 공간 위치도 포함됩니다.

In Figure 2, we visualize some feature maps. They are generated by some filters of the conv5 layer. Figure 2(c) shows the strongest activated images of these filters in the ImageNet dataset. We see a filter can be activated by some semantic content. For example, the 55-th filter (Figure 2, bottom left) is most activated by a circle shape; the 66-th filter (Figure 2, top right) is most activated by a ∧-shape; and the 118-th filter (Figure 2, bottom right) is most activated by a ∨-shape. These shapes in the input images (Figure 2(a)) activate the feature maps at the corresponding positions (the arrows in Figure 2).

그림 2에서는 몇 가지 기능 맵을 시각화하고 있습니다. conv5 레이어의 일부 필터에 의해 생성됩니다. 그림 2(c)는, ImageNet 데이터 세트내의 이러한 필터의 가장 강력한 액티브 이미지를 나타내고 있습니다. 필터는, 의미 있는 컨텐츠에 의해서 액티브하게 되는 것을 알 수 있습니다. 예를 들어 55번째 필터(그림 2, 왼쪽 하단)는 원 모양으로 가장 활성화되고 66번째 필터(그림 2, 오른쪽 상단)는 θ 모양으로 가장 활성화되며 118번째 필터(그림 2, 오른쪽 하단)는 θ 모양으로 가장 활성화됩니다. 입력 이미지(그림 2(a))의 이러한 모양은 해당 위치(그림 2의 화살표)에서 피쳐 맵을 활성화합니다.

It is worth noticing that we generate the feature maps in Figure 2 without fixing the input size. These feature maps generated by deep convolutional layers are analogous to the feature maps in traditional methods. In those methods, SIFT vectors or image patches are densely extracted and then encoded, e.g., by vector quantization, sparse coding, or Fisher kernels. These encoded features consist of the feature maps, and are then pooled by Bag-of-Words (BoW) or spatial pyramids. Analogously, the deep convolutional features can be pooled in a similar way.

그림 2의 기능 맵은 입력 크기를 변경하지 않고 생성된다는 점에 유의하십시오. 딥 컨볼루션레이어에 의해 생성된 이러한 기능 맵은 기존 방법의 기능 맵과 유사합니다. 이러한 방법에서 SIFT 벡터 또는 이미지 패치는 조밀하게 추출된 후 예를 들어 벡터 양자화, 스파스 코딩 또는 피셔 커널에 의해 부호화된다. 이러한 부호화된 기능은 기능 맵으로 구성되어 Bag-of-Words(BoW; 단어 가방) 또는 공간 피라미드에 의해 풀링됩니다. 이와 유사하게, 심층 컨볼루션 피쳐는 유사한 방법으로 풀링될 수 있다.

* 1. **The Spatial Pyramid Pooling Layer**

The convolutional layers accept arbitrary input sizes, but they produce outputs of variable sizes. The classifiers (SVM/softmax) or fully-connected layers require fixed-length vectors. Such vectors can be generated by the Bag-of-Words (BoW) approach that pools the features together. Spatial pyramid pooling improves BoW in that it can maintain spatial information by pooling in local spatial bins. These spatial bins have sizes proportional to the image size, so the number of bins is fixed regardless of the image size. This is in contrast to the sliding window pooling of the previous deep networks, where the number of sliding windows depends on the input size.

컨볼루션 레이어는 임의의 입력 크기를 받아들이지만 다양한 크기의 출력을 생성합니다. 분류자(SVM/softmax) 또는 완전히 연결된 계층에는 고정 길이 벡터가 필요합니다. 이러한 벡터는 기능을 풀링하는 Bag-of-Words(BoW) 접근법에 의해 생성될 수 있습니다. 공간 피라미드 풀링은 로컬 공간 빈에 풀링하여 공간 정보를 유지할 수 있다는 점에서 BoW를 개선합니다. 이러한 공간 빈은 이미지 크기에 비례하는 크기를 가지므로 이미지 크기에 관계없이 빈 수가 고정됩니다. 이는 슬라이딩 윈도우의 수가 입력 크기에 따라 달라지는 이전 딥 네트워크의 슬라이딩 윈도우 풀링과는 대조적입니다.

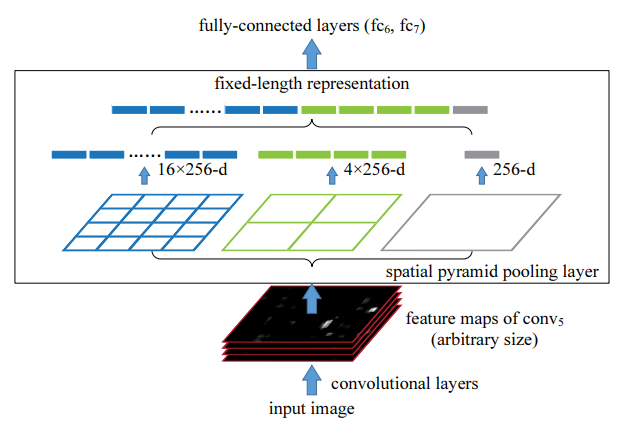


Figure 3: A network structure with a spatial pyramid pooling layer. Here 256 is the filter number of the conv5 layer, and conv5 is the last convolutional layer.

그림 3: 공간 피라미드 풀 레이어가 있는 네트워크 구조 여기서 256은 conv5층의 필터 번호이며 conv5는 마지막 convolution층이다.

To adopt the deep network for images of arbitrary sizes, we replace the last pooling layer (e.g., pool5, after the last convolutional layer) with a spatial pyramid pooling layer. Figure 3 illustrates our method. In each spatial bin, we pool the responses of each filter (throughout this paper we use max pooling). The outputs of the spatial pyramid pooling are kMdimensional vectors with the number of bins denoted as M (k is the number of filters in the last convolutional layer). The fixed-dimensional vectors are the input to the fully-connected layer.

임의의 크기의 이미지를 위해 심층 네트워크를 채택하기 위해 마지막 풀링 계층(예: 마지막 컨볼루션 계층 다음 pool5)을 공간 피라미드 풀링 계층으로 대체한다. 그림 3은 우리의 방법을 보여줍니다. 각 공간 빈에서는 각 필터의 응답을 풀링합니다(이 문서에서는 최대 풀링 사용). 공간 피라미드 풀링의 출력은 빈의 수가 M으로 표시된 kMdimensional 벡터이다(k는 마지막 컨볼루션 층의 필터 수입니다). 고정 차원 벡터는 완전히 연결된 레이어에 대한 입력입니다.

With spatial pyramid pooling, the input image can4 be of any sizes. This not only allows arbitrary aspect ratios, but also allows arbitrary scales. We can resize the input image to any scale (e.g., min(w, h)=180, 224, ...) and apply the same deep network. When the input image is at different scales, the network (with the same filter sizes) will extract features at different scales. The scales play important roles in traditional methods, e.g., the SIFT vectors are often extracted at multiple scales (determined by the sizes of the patches and Gaussian filters). We will show that the scales are also important for the accuracy of deep networks.

공간 피라미드 풀링의 경우 입력 이미지는 모든 크기를 가질 수 있습니다. 이렇게 하면 임의의 가로 세로 비율이 허용될 뿐만 아니라 임의의 배율도 허용됩니다. 입력 이미지의 크기를 임의의 스케일(예: min(w, h)=180, 224, ...)로 조정하여 동일한 딥 네트워크를 적용할 수 있습니다. 입력 이미지의 스케일이 다른 경우, 네트워크(같은 필터 사이즈의)는 다른 스케일로 피쳐를 추출합니다. 척도는 전통적인 방법에서 중요한 역할을 한다. 예를 들어, SIFT 벡터는 (패치와 가우스 필터의 크기에 따라) 여러 척도로 추출된다. 스케일도 딥 네트워크의 정확성에 중요하다는 것을 알 수 있습니다.

Interestingly, the coarsest pyramid level has a single bin that covers the entire image. This is in fact a “global pooling” operation, which is also investigated in several concurrent works. In [31], [32] a global average pooling is used to reduce the model size and also reduce overfitting; in [33], a global average pooling is used on Interestingly, the coarsest pyramid level has a single bin that covers the entire image. This is in fact a “global pooling” operation, which is also investigated in several concurrent works. In [31], [32] a global average pooling is used to reduce the model size and also reduce overfitting; in [33], a global average pooling is used on the testing stage after all fc layers to improve accuracy; in [34], a global max pooling is used for weakly supervised object recognition. The global pooling operation corresponds to the traditional Bag-of-Words method.

흥미롭게도, 가장 거친 피라미드 층은 전체 이미지를 덮는 하나의 빈을 가지고 있다. 이는 사실상 "글로벌 풀링" 작업이며, 여러 동시 작업에서도 조사됩니다. [31], [32]에서는 모델 크기를 줄이고 과적합도 줄이기 위해 글로벌 평균 풀링을 사용합니다. [33]에서는 흥미롭게도 가장 거친 피라미드 수준에는 전체 이미지를 포함하는 단일 빈이 있습니다. 이는 사실상 "글로벌 풀링" 작업이며, 여러 동시 작업에서도 조사됩니다. [31], [32]에서는 모델 크기를 줄이고 과적합도 줄이기 위해 글로벌 평균 풀링을 사용한다.[33]에서는 정확도를 향상시키기 위해 테스트 단계에서 글로벌 평균 풀링을 사용한다.[34]에서는 글로벌 최대 풀링을 약하게 감독되는 객체 인식에 사용한다. 글로벌 풀링 조작은 기존의 Bag-of-Words 메서드에 대응합니다.

* 1. **Training the Network**

Theoretically, the above network structure can be trained with standard back-propagation, regardless of the input image size. But in practice the GPU implementations (such as cuda-convnet and Caffe) are preferably run on fixed input images. Next we describe our training solution that takes advantage of these GPU implementations while still preserving the spatial pyramid pooling behaviors.

이론적으로 위의 네트워크 구조는 입력 이미지 크기에 관계없이 표준 역전파를 사용하여 훈련할 수 있습니다. 그러나 실제로는 GPU 구현(cuda-convnet 및 Caffe 등)은 고정 입력 이미지로 실행하는 것이 좋습니다. 다음으로 공간 피라미드 풀링 동작을 유지하면서 이러한 GPU 구현을 활용하는 트레이닝 솔루션에 대해 설명합니다.

*Single-size training*

As in previous works, we first consider a network taking a fixed-size input (224×224) cropped from images. The cropping is for the purpose of data augmentation. For an image with a given size, we can pre-compute the bin sizes needed for spatial pyramid pooling. Consider the feature maps after conv5 that have a size of a×a (e.g., 13×13). With a pyramid level of n×n bins, we implement this pooling level as a sliding window pooling, where the window size win = da/ne and stride str = ba/nc with d·e and b·c denoting ceiling and floor operations. With an l-level pyramid, we implement l such layers. The next fully-connected layer (fc6) will concatenate the l outputs. Figure 4 shows an example configuration of 3-level pyramid pooling (3×3, 2×2, 1×1) in the cuda-convnet style. The main purpose of our single-size training is to enable the multi-level pooling behavior. Experiments show that this is one reason for the gain of accuracy.

전작과 마찬가지로 우선 이미지에서 잘라낸 고정 사이즈 입력(224×224)을 취하는 네트워크를 검토한다. 크롭핑은 데이터 확대를 목적으로 합니다. 지정된 크기의 이미지의 경우 공간 피라미드 풀링에 필요한 빈 크기를 미리 계산할 수 있습니다. a×a(예: 13×13) 크기의 conv5 이후의 피쳐 맵을 검토합니다. 피라미드 레벨의 n×n bin을 사용하여 이 풀링 레벨을 슬라이딩 윈도우 풀링으로 구현하며, 여기서 d·e와 b·c가 천장 및 바닥 작동을 나타내는 str = da/ne, strid str = ba/nc이다. 우리는 l-level 피라미드를 사용하여 l개의 레이어를 구현한다. 다음 완전 접속 레이어(fc6)는 l 출력을 연결합니다. 그림 4는 cuda-convnet 스타일의 3단계 피라미드 풀링(3×3, 2×2, 1×1)의 구성 예를 보여준다. 단일 규모 교육의 주요 목적은 다단계 풀링 동작을 활성화하는 것입니다. 실험 결과 이것이 정확성을 높이는 한 가지 이유인 것으로 나타났습니다.

*Multi-size training*

Our network with SPP is expected to be applied on images of any sizes. To address the issue of varying image sizes in training, we consider a set of predefined sizes. We consider two sizes: 180×180 in addition to 224×224. Rather than crop a smaller 180×180 region, we resize the aforementioned 224×224 region to 180×180. So the regions at both scales differ only in resolution but not in content/layout. For the network to accept 180×180 inputs, we implement another fixed-size-input (180×180) network. The feature map size after conv5 is a×a = 10×10 in this case. Then we still use win = da/ne and str = ba/nc to implement each pyramid pooling level. The output of the spatial pyramid pooling layer of this 180-network has the same fixed length as the 224-network. As such, this 180-network has exactly the same parameters as the 224-network in each layer. In other words, during training we implement the varying-input-size SPP-net by two fixed-size networks that share parameters.

SPP와의 네트워크는 모든 크기의 이미지에 적용됩니다. 트레이닝에서 다양한 이미지 크기의 문제에 대처하기 위해 사전 정의된 크기 세트를 고려합니다. 224×224 사이즈 외에 180×180 사이즈도 생각하고 있습니다. 180×180의 작은 영역을 잘라내는 것이 아니라 앞서 말한 224×224의 영역을 180×180으로 크기를 조정합니다. 따라서 두 스케일 모두 해상도만 다를 뿐 콘텐츠/레이아웃은 다릅니다. 네트워크가 180×180 입력을 받아들이기 위해, 우리는 다른 고정 크기 입력(180×180) 네트워크를 구현합니다. conv5 이후의 피쳐 맵 크기는 a×a = 10×10입니다. 그런 다음 win = da/ne 및 str = ba/nc를 사용하여 각 피라미드 풀링 레벨을 구현합니다. 이 180 네트워크의 공간 피라미드 풀링 레이어의 출력은 224 네트워크의 고정 길이와 동일합니다. 이와 같이 이 180-네트워크는 각 레이어의 224-네트워크와 완전히 동일한 파라미터를 가집니다. 즉, 교육 중에 매개 변수를 공유하는 두 개의 고정 크기 네트워크를 통해 다양한 입력 크기의 SPP-net을 구현합니다.

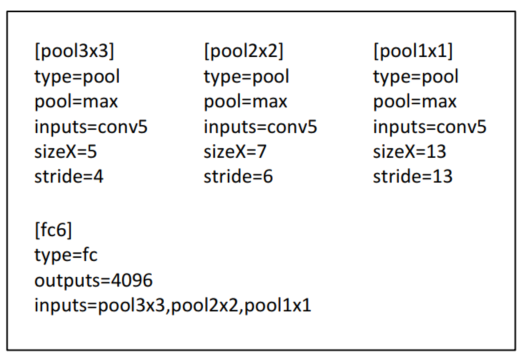


Figure 4: An example 3-level pyramid pooling in the cuda-convnet style. Here sizeX is the size of the pooling window. This configuration is for a network whose feature map size of conv5 is 13×13, so the pool3×3, pool2×2, and pool1×1 layers will have 3×3, 2×2, and 1×1 bins respectively.

그림 4: cuda-convnet 스타일의 3단계 피라미드 풀링 예시. 여기서 sizeX는 풀링 창의 크기입니다. 이 설정은 conv5의 피쳐 맵사이즈가 13×13인 네트워크용입니다.따라서 pool3×3, pool2×2, pool1×1 레이어에는 각각 3×3, 2×2, 1×1의 bin이 있습니다.

To reduce the overhead to switch from one network (e.g., 224) to the other (e.g., 180), we train each full epoch on one network, and then switch to the other one (keeping all weights) for the next full epoch. This is iterated. In experiments, we find the convergence rate of this multi-size training to be similar to the above single-size training.

한 네트워크(예를 들어 224)에서 다른 네트워크(예를 들어 180)로 전환하기 위한 오버헤드를 줄이기 위해 각 에폭을 한 네트워크 상에서 훈련한 후 다음 에폭을 위해 다른 에폭으로 전환합니다(모든 가중치를 유지). 반복됩니다. 실험에서, 우리는 이 다중 크기 훈련의 수렴률이 위의 단일 크기 훈련과 유사하다는 것을 발견했습니다.

The main purpose of our multi-size training is to simulate the varying input sizes while still leveraging the existing well-optimized fixed-size implementations. Besides the above two-scale implementation, we have also tested a variant using s × s as input where s is randomly and uniformly sampled from [180, 224] at each epoch. We report the results of both variants in the experiment section.

Note that the above single/multi-size solutions are for training only. At the testing stage, it is straightforward to apply SPP-net on images of any sizes.

멀티 사이즈 트레이닝의 주된 목적은 다양한 입력 사이즈를 시뮬레이트하면서 기존의 적절하게 최적화된 고정 사이즈 실장을 활용하는 것입니다. 위의 두 스케일 구현 외에도, 각 에폭에서 [180, 224]에서 s가 랜덤하고 균일하게 샘플링되는 입력으로 s × s를 사용하여 변형을 테스트했다. 우리는 실험 섹션에서 두 변종 모두의 결과를 보고한다.

위의 싱글/멀티 사이즈 솔루션은 트레이닝 전용입니다. 테스트 단계에서는 모든 크기의 이미지에 SPP-net을 적용하는 것이 간단합니다.

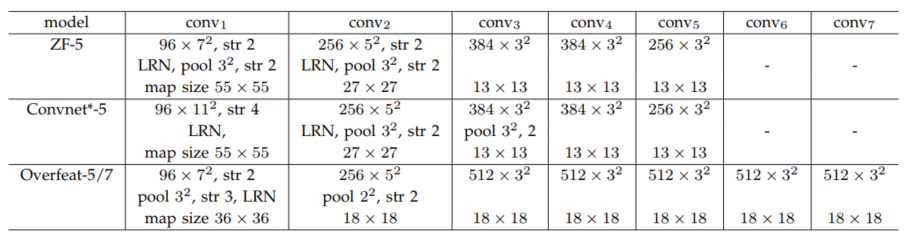


Table 1: Network architectures: filter number×filter size (e.g., 96×7^2), filter stride (e.g., str 2), pooling window size (e.g., pool 3^2), and the output feature map size (e.g., map size 55 × 55). LRN represents Local Response Normalization. The padding is adjusted to produce the expected output feature map size.

표 1: 네트워크 아키텍처: 필터 번호×필터 크기(예: 96×7^2), 필터 스트라이드(예: str 2), 풀링 윈도우 크기(예: 풀 3^2), 출력 피쳐 맵 크기(예: 맵 크기 55×55) LRN은 로컬 응답 정규화를 나타냅니다. 패딩은 예상되는 출력 피쳐 맵사이즈를 생성하도록 조정됩니다.

1. **SPP-Net for Image Classification**
   1. **Experiments on ImageNet 2012 Classification**

We train the networks on the 1000-category training set of ImageNet 2012. Our training algorithm follows the practices of previous work. The images are resized so that the smaller dimension is 256, and a 224×224 crop is picked from the center or the four corners from the entire image1. The data are augmented by horizontal flipping and color altering. Dropout is used on the two fully-connected layers. The learning rate starts from 0.01, and is divided by 10 (twice) when the error plateaus. Our implementation is based on the publicly available code of cuda-convnet and Caffe. All networks in this paper can be trained on a single GeForce GTX Titan GPU (6 GB memory) within two to four weeks.

델은 ImageNet 2012의 1000 카테고리 트레이닝 세트를 사용하여 네트워크를 훈련하고 있습니다. 우리의 훈련 알고리즘은 이전 작업의 관행을 따릅니다. 작은 치수가 256이 되도록 화상의 사이즈를 조정하고, 화상 전체로부터 중앙 또는 네 모서리에서 224×224의 크롭을 추출한다. 데이터는 수평 플립 및 색상 변경을 통해 증강됩니다. 드롭아웃은 완전히 연결된2개의 레이어에서 사용됩니다. 학습 속도는 0.01부터 시작되며 오류 평판일 때 10(2회)으로 나눕니다. 우리의 구현은 cuda-convnet과 Cafe의 공개 코드를 기반으로 한다. 이 문서의 모든 네트워크는 1개의 GeForce GTX Titan GPU(6GB 메모리)로 2~4주 이내에 트레이닝할 수 있습니다.

* + 1. ***Baseline Network Architectures***

The advantages of SPP are independent of the convolutional network architectures used. We investigate four different network architectures in existing publications (or their modifications), and we show SPP improves the accuracy of all these architectures. These baseline architectures are in Table 1 and briefly introduced below:

SPP의 장점은 사용되는 컨볼루션 네트워크 아키텍처와는 무관합니다. NAT은 기존 출판물에 있는 4가지 네트워크 아키텍처(또는 그 수정 사항)를 조사하여 SPP가 이러한 모든 아키텍처의 정확성을 향상시킨다는 것을 보여 줍니다. 이러한 베이스라인 아키텍처는 표 1에 기재되어 있으며, 아래에 간략히 소개되어 있습니다.

• **ZF-5**: this architecture is based on Zeiler and Fergus’s (ZF) “fast” (smaller) model. The number indicates five convolutional layers.

**• Convnet\*-5**: this is a modification on Krizhevsky et al.’s network [3]. We put the two pooling layers after conv2 and conv3 (instead of after conv1 and conv2). As a result, the feature maps after each layer have the same size as ZF-5.

• **Overfeat-5/7**: this architecture is based on the Overfeat paper, with some modifications as in [6]. In contrast to ZF-5/Convnet\*-5, this architecture produces a larger feature map (18×18 instead of 13 × 13) before the last pooling layer. A larger filter number (512) is used in conv3 and the following convolutional layers. We also investigate a deeper architecture with 7 convolutional layers, where conv3 to conv7 have the same structures.

• **ZF-5**: 이 아키텍처는 Zeiler 및 Pergus(ZF)의 고속(소형) 모델을 기반으로 합니다. 이 숫자는 5개의 컨볼루션 층을 나타냅니다.

• **Convnet\*-5**: 이것은 Krizhevsky 등의 네트워크에 대한 수정사항입니다[3]. 2개의 풀링 레이어를 conv1과 conv2 뒤에 배치했습니다. 그 결과 각 레이어 뒤의 기능 맵은 ZF-5와 같은 사이즈가 됩니다.

• **Overfeat-5/7**: 이 아키텍처는 Overfeat 백서를 기반으로 하며 [6]과 같이 일부 수정을 가합니다. ZF-5/Convnet\*-5와 달리 이 아키텍처는 마지막 풀링 레이어 전에 더 큰 기능 맵(13×13이 아닌 18×18)을 생성합니다. 더 큰 필터 번호(512)는 conv3 및 그 이후의 convolutional 층에서 사용된다. 또한 conv3에서 conv7까지의 구조가 동일한 7개의 convolution 층이 있는 더 깊은 아키텍처에 대해서도 조사한다.

In the baseline models, the pooling layer after the last convolutional layer generates 6×6 feature maps, with two 4096-d fc layers and a 1000-way softmax layer following. Our replications of these baseline networks are in Table 2 (a). We train 70 epochs for ZF-5 and 90 epochs for the others. Our replication of ZF-5 is better than the one reported in [4]. This gain is because the corner crops are from the entire image, as is also reported in [36].

베이스라인 모델에서 마지막 컨볼루션레이어 뒤의 풀링레이어는 6×6 기능 맵을 생성하고, 2개의 4096-d fc레이어와 1000방향 소프트맥스레이어가 그 뒤를 잇습니다. 이러한 베이스라인 네트워크의 레플리케이션은 표 2(a)에 나타냅니다. ZF-5는 70세기를, 그 외는 90세기를 훈련하고 있습니다. ZF-5의 복제는 [4]에서 보고된 복제보다 우수합니다. 이러한 이득은 [36]에서도 보고되었듯이 모서리 작물이 전체 이미지에서 나온 것이기 때문이다.

* + 1. ***Multi-level Pooling Improves Accuracy***

In Table 2 (b) we show the results using singlesize training. The training and testing sizes are both 224×224. In these networks, the convolutional layers have the same structures as the corresponding baseline models, whereas the pooling layer after the final convolutional layer is replaced with the SPP layer. For the results in Table 2, we use a 4-level pyramid. The pyramid is {6×6, 3×3, 2×2, 1×1} (totally 50 bins). For fair comparison, we still use the standard 10-view prediction with each view a 224×224 crop. Our results in Table 2 (b) show considerable improvement over the no-SPP baselines in Table 2 (a). Interestingly, the largest gain of top-1 error (1.65%) is given by the most accurate architecture. Since we are still using the same 10 cropped views as in (a), these gains are solely because of multi-level pooling.

표 2 (b)에서는 싱글 사이즈 트레이닝을 사용한 결과를 나타내고 있다. 교육 및 테스트 크기는 모두 224×224입니다. 이러한 네트워크에서, 컨볼루션 계층은 대응하는 기준선 모델과 동일한 구조를 가지고 있는 반면, 최종 컨볼루션 계층 이후의 풀링 계층은 SPP 계층으로 대체된다. 표 2의 결과에는 4단계 피라미드를 사용합니다. 피라미드는 {6×6, 3×3, 2×2, 1×1}(총 50개의 빈)입니다. 공정한 비교를 위해 표준 10뷰 예측을 사용하여 각 뷰당 224×224 크롭을 사용합니다. 표 2 (b)의 결과는 표 2 (a)의 no-SPP 기준선에 비해 상당히 개선된 것을 보여준다. 흥미롭게도 상위 1위 오류(1.65%)의 가장 큰 이점은 가장 정확한 아키텍처에 있습니다. 우리는 여전히 (a)와 같은 10개의 잘라낸 뷰를 사용하고 있기 때문에, 이러한 이점은 단지 다단계 풀링에 의한 것입니다.

It is worth noticing that the gain of multi-level pooling is not simply due to more parameters; rather, it is because the multi-level pooling is robust to the variance in object deformations and spatial layout. To show this, we train another ZF-5 network with a different 4-level pyramid: {4×4, 3×3, 2×2, 1×1} (totally 30 bins). This network has fewer parameters than its no-SPP counterpart, because its fc6 layer has 30×256-d inputs instead of 36×256-d. The top-1/top5 errors of this network are 35.06/14.04. This result is similar to the 50-bin pyramid above (34.98/14.14), but considerably better than the no-SPP counterpart (35.99/14.76).

다단계 풀링의 이득은 단순히 더 많은 매개변수에 의한 것이 아니라 다단계 풀링이 객체 변형과 공간 배치의 변화에 강하기 때문이다. 이를 보여주기 위해 다른 4단계 피라미드인 {4×4, 3×3, 2×2, 1×1}(총 30개의 빈)을 가진 또 다른 ZF-5 네트워크를 훈련한다. fc6 레이어의 입력은 36×256-d가 아닌 30×256-d이기 때문에 이 네트워크에는 SPP가 없는 네트워크보다 파라미터가 적습니다. 이 네트워크의 상위 1/5 에러는 35.06/14.04 입니다. 이 결과는 위의 50빈 피라미드(34.98/14.14)와 비슷하지만 SPP가 없는 피라미드(35.99/14.76)보다 훨씬 우수합니다.

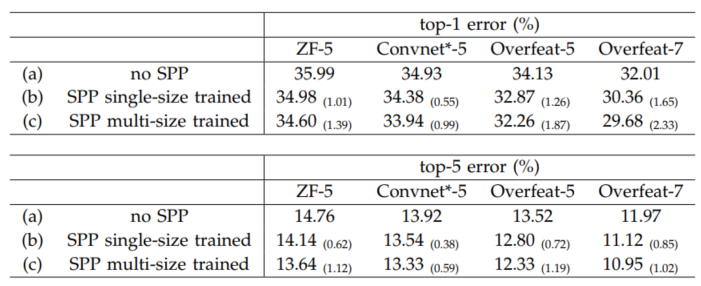


Table 2: Error rates in the validation set of ImageNet 2012. All the results are obtained using standard 10-view testing. In the brackets are the gains over the “no SPP” baselines.

표 2: ImageNet 2012 검증 세트의 오류율 모든 결과는 표준 10 뷰 테스트를 사용하여 얻을 수 있습니다. 괄호 안은 "SPP 없음" 기준선에 대한 이점입니다.

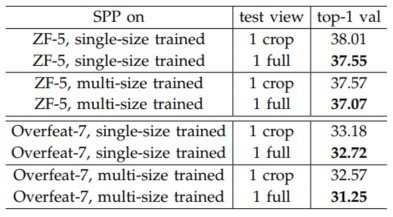


Table 3: Error rates in the validation set of ImageNet 2012 using a single view. The images are resized so min(w, h) = 256. The crop view is the central 224×224 of the image.

표 3: 단일 뷰를 사용한ImageNet 2012 검증 세트의 에러율 영상의 크기가 min(w, h) = 256이 되도록 조정됩니다. 크롭 뷰는 이미지의 중앙 224×224입니다.

* + 1. ***Multi-size Training Improves Accuracy***

Table 2 (c) shows our results using multi-size training. The training sizes are 224 and 180, while the testing size is still 224. We still use the standard 10-view prediction. The top-1/top-5 errors of all architectures further drop. The top-1 error of SPP-net (Overfeat-7) drops to 29.68%, which is 2.33% better than its noSPP counterpart and 0.68% better than its single-size trained counterpart.

표 2 (c)는 멀티 사이즈 트레이닝을 사용한 결과를 나타내고 있습니다. 트레이닝 사이즈는 224와 180이지만 테스트 사이즈는 224입니다. 우리는 여전히 표준 10뷰 예측을 사용합니다. 모든 아키텍처의 상위 1/5개 오류는 더욱 감소합니다. SPP-net(Overfeat-7)의 상위 1개 오류는 29.68%로 떨어졌는데, 이는 noSPP에 비해 2.33%, 단일 크기 교육을 받은 것에 비해 0.68% 향상된 것입니다.

Besides using the two discrete sizes of 180 and 224, we have also evaluated using a random size uniformly sampled from [180, 224]. The top-1/5 error of SPP-net (Overfeat-7) is 30.06%/10.96%. The top1 error is slightly worse than the two-size version, possibly because the size of 224 (which is used for testing) is visited less. But the results are still better the single-size version.

180과 224의 두 가지 이산 크기를 사용하는 것 외에도, 우리는 [180, 224]에서 균일하게 표본 추출한 무작위 크기를 사용하여 평가하였다. SPP-net(Overfeat-7)의 상위 1/5 오차는 30.06%/10.96%입니다. top1 에러는, 2 사이즈 버전보다 약간 악화하고 있습니다.이는 테스트에 사용되는 224 의 사이즈의 방문수가 적기 때문일 가능성이 있습니다. 하지만 결과는 싱글사이즈 버전보다 더 좋습니다.

There are previous CNN solutions that deal with various scales/sizes, but they are mostly based on testing. In Overfeat and Howard’s method, the single network is applied at multiple scales in the testing stage, and the scores are averaged. Howard further trains two different networks on low/highresolution image regions and averages the scores. To our knowledge, our method is the first one that trains a single network with input images of multiple sizes.

다양한 규모/사이즈를 다루는 이전 CNN 솔루션이 있지만 대부분 테스트를 기반으로 합니다. Overfeat 및 Howard의 방법에서는 테스트 단계에서 단일 네트워크를 여러 척도로 적용하여 점수를 평균화합니다. Howard는 또한 저해상도/고해상도 이미지 영역에서 두 개의 다른 네트워크를 교육하고 점수를 평균화합니다. 우리가 아는 한, 우리의 방법은 여러 크기의 입력 이미지를 사용하여 단일 네트워크를 훈련하는 첫 번째 방법입니다.

* + 1. ***Full-image Representations Improve Accuracy***

Next we investigate the accuracy of the full-image views. We resize the image so that min(w, h)=256 while maintaining its aspect ratio. The SPP-net is applied on this full image to compute the scores of the full view. For fair comparison, we also evaluate the accuracy of the single view in the center 224×224 crop (which is used in the above evaluations). The comparisons of single-view testing accuracy are in Table 3. Here we evaluate ZF-5/Overfeat-7. The top-1 error rates are all reduced by the full-view representation. This shows the importance of maintaining the complete content. Even though our network is trained using square images only, it generalizes well to other aspect ratios.

다음으로 전체 이미지 보기의 정확성을 조사합니다. 종횡비를 유지하면서 min(w, h)=256이 되도록 이미지의 크기를 조정합니다. SPP-net이 전체 이미지에 적용되어 전체 뷰의 점수를 계산합니다. 또한 공정한 비교를 위해 중앙 224×224 크롭(위 평가에서 사용)의 단일 뷰의 정확성을 평가합니다. 싱글 뷰 테스트 정확도의 비교는 표 3에 나와 있습니다. 여기에서는 ZF-5/Overfeat-7을 평가합니다. 상위 1개의 오류율은 모두 풀뷰 표시에 의해 감소합니다. 이는 전체 내용을 유지하는 것의 중요성을 보여줍니다. 네트워크는 정사각형 이미지만을 사용하여 훈련되지만 다른 애스펙트 비에 대해서는 잘 일반화되어 있습니다.

Comparing Table 2 and Table 3, we find that the combination of multiple views is substantially better than the single full-image view. However, the fullimage representations are still of good merits. First, we empirically find that (discussed in the next subsection) even for the combination of dozens of views, the additional two full-image views (with flipping) can still boost the accuracy by about 0.2%. Second, the full-image view is methodologically consistent with the traditional methods where the encoded SIFT vectors of the entire image are pooled together. Third, in other applications such as image retrieval, an image representation, rather than a classification score, is required for similarity ranking. A full-image representation can be preferred.

표 2와 표 3을 비교하면 여러 뷰의 조합이 단일 전체 이미지 뷰보다 훨씬 낫다는 것을 알 수 있다. 그러나 전체 이미지 표현은 여전히 좋은 장점이 있습니다. 첫째, 우리는 경험적으로 (다음 하위 섹션에서 논의) 수십 개의 보기 조합에 대해서도 (플립을 통해) 두 개의 전체 이미지 보기를 추가로 사용해도 정확도가 약 0.2% 향상될 수 있다는 것을 발견했다. 둘째, 전체 이미지 보기는 전체 이미지의 부호화된 SIFT 벡터가 함께 풀링되는 기존 방법과 방법론적으로 일치한다. 셋째, 이미지 검색과 같은 다른 응용 프로그램에서는 유사도 순위를 매기기 위해 분류 점수가 아닌 이미지 표현이 필요하다. 전체 이미지 표현을 선호할 수 있습니다.

* + 1. ***Multi-view Testing on Feature Maps***

Inspired by our detection algorithm (described in the next section), we further propose a multi-view testing method on the feature maps. Thanks to the flexibility of SPP, we can easily extract the features from windows (views) of arbitrary sizes from the convolutional feature maps.

검출 알고리즘(다음 섹션에서 설명)에서 영감을 얻어 기능 맵에서 멀티 뷰 테스트 방법을 제안합니다. SPP의 유연성 덕분에 컨볼루션 기능 맵에서 임의의 크기의 창(뷰)에서 기능을 쉽게 추출할 수 있습니다.

On the testing stage, we resize an image so min(w, h) = s where s represents a predefined scale (like 256). Then we compute the convolutional feature maps from the entire image. For the usage of flipped views, we also compute the feature maps of the flipped image. Given any view (window) in the image, we map this window to the feature maps (the way of mapping is in Appendix), and then use SPP to pool the features from this window (see Figure 5). The pooled features are then fed into the fc layers to compute the softmax score of this window. These scores are averaged for the final prediction. For the standard 10-view, we use s = 256 and the views are 224×224 windows on the corners or center. Experiments show that the top-5 error of the 10-view prediction on feature maps is within 0.1% around the original 10-view prediction on image crops.

테스트 단계에서는 최소(w, h) = s가 사전 정의된 스케일(예: 256)을 나타내도록 영상의 크기를 조정합니다. 그런 다음 전체 이미지에서 컨볼루션 피쳐 맵을 계산합니다. 플립 뷰를 사용하기 위해 플립된 이미지의 피쳐 맵도 계산합니다. 이미지의 뷰(창)를 지정하면 이 창을 기능 맵에 매핑한 다음(매핑 방법은 부록에 있음) SPP를 사용하여 이 창에서 기능을 풀링합니다(그림 5 참조). 풀링된 기능은 이 창의 softmax 점수를 계산하기 위해 fc 계층에 공급됩니다. 이러한 점수는 최종 예측에 대한 평균입니다. 표준 10뷰의 경우 s = 256을 사용하며, 뷰는 모서리 또는 중앙의 224×224 창입니다. 실험 결과, 피쳐 맵의 10 뷰 예측의 상위 5개의 오차는 이미지 크롭의 원래 10 뷰 예측 부근에서 0.1% 이내인 것으로 나타났습니다.

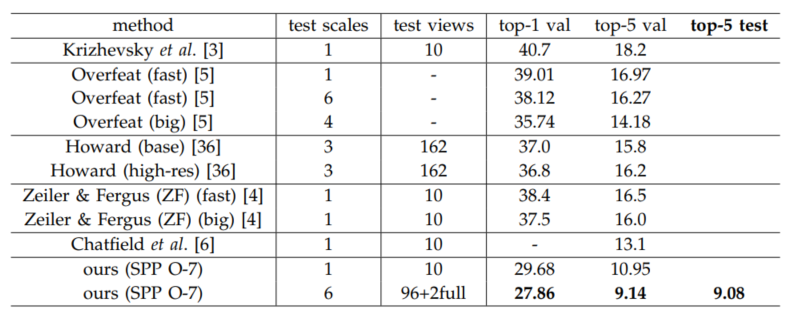


Table 4: Error rates in ImageNet 2012. All the results are based on a single network. The number of views in Overfeat depends on the scales and strides, for which there are several hundreds at the finest scale.

표 4: ImageNet 2012 오류율 모든 결과는 단일 네트워크를 기반으로 합니다. Overfeat의 조회수는 음계와 진전에 따라 달라지는데, 그 음계에는 수백 개의 최고 음계가 있습니다.

We further apply this method to extract multiple views from multiple scales. We resize the image to six scales s ∈ {224, 256, 300, 360, 448, 560} and compute the feature maps on the entire image for each scale. We use 224 × 224 as the view size for any scale, so these views have different relative sizes on the original image for different scales. We use 18 views for each scale: one at the center, four at the corners, and four on the middle of each side, with/without flipping (when s = 224 there are 6 different views). The combination of these 96 views reduces the top-5 error from 10.95% to 9.36%. Combining the two fullimage views (with flipping) further reduces the top-5 error to 9.14%.

또한 이 방법을 적용하여 여러 척도에서 여러 뷰를 추출한다. 이미지 크기를 6개 축척 { {224, 256, 300, 360, 448, 560}(으)로 조정하고 각 축척에 대한 전체 이미지에서 기능 맵을 계산합니다. 224 × 224를 임의의 스케일의 뷰 사이즈로 사용하고 있기 때문에, 이러한 뷰는, 스케일에 따라 원래의 이미지상의 상대 사이즈가 다릅니다. 각 축척에 대해 18개의 뷰를 사용합니다. 즉, 가운데에 하나씩, 모서리에 4개, 양쪽에 4개씩 플립 유무(s = 224일 경우 6개의 다른 뷰가 있음). 이러한 96개의 뷰를 조합하면 상위 5개의 오류가 10.95%에서 9.36%로 감소합니다. 두 개의 전체 이미지 보기(플립 포함)를 결합하면 상위 5개 오류가 9.14%로 감소합니다.

In the Overfeat paper [5], the views are also extracted from the convolutional feature maps instead of image crops. However, their views cannot have arbitrary sizes; rather, the windows are those where the pooled features match the desired dimensionality. We empirically find that these restricted windows are less beneficial than our flexibly located/sized windows.

Overfeat 논문 [5]에서는 이미지 크롭 대신 컨볼루션 피쳐 맵에서 뷰를 추출한다. 그러나 뷰는 임의의 크기를 가질 수 없으며 풀링된 피쳐가 원하는 치수와 일치하는 창이 됩니다. 이러한 제한된 창은 유연하게 배치/크기 조정 가능한 창보다 유리하지 않다는 것을 경험적으로 알게 되었습니다.

* + 1. ***Summaray and Results for ILSVRC 2014***

In Table 4 we compare with previous state-of-the-art methods. Krizhevsky et al.’s [3] is the winning method in ILSVRC 2012; Overfeat, Howard’s, and Zeiler and Fergus’s [4] are the leading methods in ILSVRC 2013. We only consider single-network performance for manageable comparisons.

표 4에서 우리는 이전의 최첨단 방법과 비교합니다. Krizhevsky et al.의 [3]은 ILSVRC 2012에서 승리한 방법입니다. Overfeat, Howard's, Zeiler and Fergus's[4]는 ILSVRC 2013의 주요 방법입니다. 우리는 관리 가능한 비교를 위해 단일 네트워크 성능만 고려합니다.

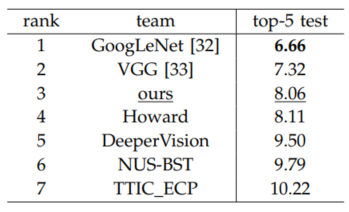


Table 5: The competition results of ILSVRC 2014 classification. The best entry of each team is listed.

표 5: ILSVRC 2014 분류의 경기 결과. 각 팀의 최우수 엔트리가 나열됩니다.

Our best single network achieves 9.14% top-5 error on the validation set. This is exactly the single-model entry we submitted to ILSVRC 2014. The top-5 error is 9.08% on the testing set (ILSVRC 2014 has the same training/validation/testing data as ILSVRC 2012). After combining eleven models, our team’s result (8.06%) is ranked #3 among all 38 teams attending ILSVRC 2014 (Table 5). Since the advantages of SPPnet should be in general independent of architectures, we expect that it will further improve the deeper and larger convolutional architectures.

델 최고의 단일 네트워크는 검증 세트에서 9.14%의 상위 5위 오류를 달성합니다. ILSVRC 2014에 제출한 단일 모델 엔트리입니다. 테스트 세트의 상위 5개 오류는 9.08%입니다(ILSVRC 2014에는 ILSVRC 2012와 동일한 교육/검증/테스트 데이터가 있습니다). 11개 모델을 조합한 결과, ILSVRC 2014 참가 38개 팀 중 3위(8.06%)를 차지했습니다(표 5). SPPnet의 장점은 일반적으로 아키텍처와 독립적이어야 하므로, 우리는 SPPnet이 더 깊고 큰 컨볼루션 아키텍처를 더욱 개선할 것으로 기대합니다.

* 1. **Experiments on VOC 2007 Classification**

Our method can generate a full-view image representation. With the above networks pre-trained on ImageNet, we extract these representations from the images in the target datasets and re-train SVM classifiers. In the SVM training, we intentionally do not use any data augmentation (flip/multi-view). We l2-normalize the features for SVM training.

이 방법은 전체 이미지 표현을 생성할 수 있습니다. ImageNet에서 사전 교육을 받은 위의 네트워크를 통해 타깃 데이터셋의 이미지에서 이러한 표현을 추출하고 SVM 분류기를 다시 교육합니다. SVM 트레이닝에서는 의도적으로 데이터 증강(플립/멀티 뷰)을 사용하지 않습니다. SVM 트레이닝의 기능을 정규화합니다.

The classification task in Pascal VOC 2007 involves 9,963 images in 20 categories. 5,011 images are for training, and the rest are for testing. The performance is evaluated by mean Average Precision (mAP). Table 6 summarizes the results.

Pascal VOC 2007의 분류 작업은 20개 카테고리 9,963개의 이미지를 포함하고 있으며 5,011개의 이미지는 훈련용이고 나머지는 테스트용입니다. 퍼포먼스는 평균 정밀도(mAP)로 평가됩니다. 표 6은 결과를 요약한 것입니다.

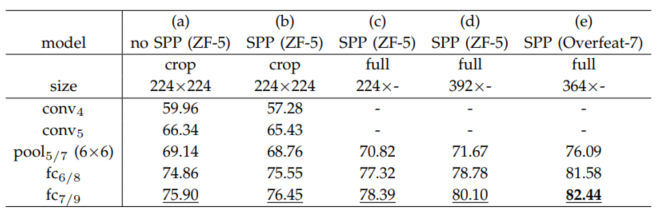


Table 6: Classification mAP in Pascal VOC 2007. For SPP-net, the pool5/7 layer uses the 6×6 pyramid level.

표 6: Pascal VOC 2007의 분류 mAP. SPP-net의 경우 풀5/7 계층은 6×6 피라미드 레벨을 사용합니다.

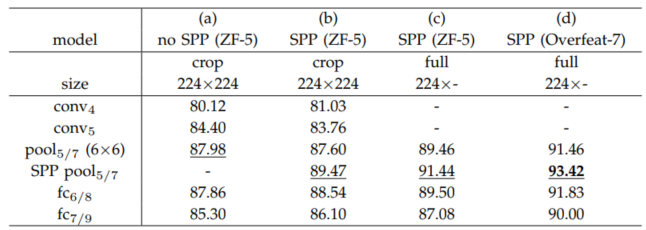


Table 7: Classification accuracy in Caltech101. For SPP-net, the pool5/7 layer uses the 6×6 pyramid level.

표 7: Caltech101의 분류 정확도. SPP-net의 경우 풀5/7 계층은 6×6 피라미드 레벨을 사용합니다.

We start from a baseline in Table 6 (a). The model is ZF-5 without SPP. To apply this model, we resize the image so that its smaller dimension is 224, and crop the center 224×224 region. The SVM is trained via the features of a layer. On this dataset, the deeper the layer is, the better the result is. In Table 6 (b), we replace the no-SPP net with our SPP-net. As a first-step comparison, we still apply the SPP-net on the center 224×224 crop. The results of the fc layers improve. This gain is mainly due to multi-level pooling.

표 6(a)의 기준선부터 시작한다. 모델은 ZF-5(SPP 제외)입니다. 이 모델을 적용하기 위해서, 이미지의 사이즈가 224가 되도록 크기를 조정해, 중앙의 224×224 영역을 잘라냅니다. SVM은 레이어의 기능을 통해 훈련됩니다. 이 데이터 집합에서는 계층이 깊을수록 더 나은 결과를 얻을 수 있습니다. 표 6(b)에서는 SPP-net으로 SPP-net을 교체합니다. 첫 단계 비교로 SPP-net을 중앙 224×224 크롭에 적용하고 있습니다. fc 계층의 결과가 개선됩니다. 이러한 이익은 주로 다단계 풀링에 기인합니다.

Table 6 (c) shows our results on full images, where the images are resized so that the shorter side is 224. We find that the results are considerably improved (78.39% vs. 76.45%). This is due to the full-image representation that maintains the complete content.

표 6(c)은 전체 이미지에 대한 결과를 보여 줍니다. 여기서 짧은 쪽이 224가 되도록 이미지의 크기가 조정됩니다. 결과는 상당히 개선되었습니다(78.39% 대 76.45%). 이는 전체 콘텐츠를 유지하는 전체 이미지 표현 때문입니다.

Because the usage of our network does not depend on scale, we resize the images so that the smaller dimension is s and use the same network to extract features. We find that s = 392 gives the best results (Table 6 (d)) based on the validation set. This is mainly because the objects occupy smaller regions in VOC 2007 but larger regions in ImageNet, so the relative object scales are different between the two sets. These results indicate scale matters in the classification tasks, and SPP-net can partially address this “scale mismatch” issue.

네트워크의 사용량은 규모에 따라 달라지지 않기 때문에 작은 치수가 되도록 이미지의 크기를 조정하고 동일한 네트워크를 사용하여 기능을 추출합니다. 우리는 s = 392가 검증 세트를 기반으로 최고의 결과 (표 6 (d))를 제공한다는 것을 발견했다. 이는 주로 객체가 VOC 2007에서는 더 작은 영역을 차지하지만 ImageNet에서는 더 큰 영역을 차지하기 때문에 두 세트의 상대적 객체 스케일이 다르기 때문입니다. 이러한 결과는 분류 작업에서 규모 문제를 나타내며, SPP-net은 이 "규모 불일치" 문제를 부분적으로 해결할 수 있다.

In Table 6 (e) the network architecture is replaced with our best model (Overfeat-7, multi-size trained), and the mAP increases to 82.44%. Table 8 summarizes our results and the comparisons with the state-of-theart methods. Among these methods, VQ, LCC, and FK are all based on spatial pyramids matching, and are based on deep networks. In these results, Oquab et al.’s (77.7%) and Chatfield et al.’s (82.42%) are obtained by network fine-tuning and multi-view testing. Our result is comparable with the state of the art, using only a single full-image representation and without fine-tuning.

표 6 (e)에서는 네트워크 아키텍처가 델의 베스트모델(오버피팅-7, 멀티사이즈 트레이닝 완료)로 대체되어 mAP가 82.44%까지 증가합니다. 표 8은 당사의 결과와 최신 방법과의 비교를 요약한 것입니다. 이러한 방법 중 VQ, LCC 및 FK는 모두 공간 피라미드 매칭을 기반으로 하며 딥 네트워크를 기반으로 합니다. 이 결과, 네트워크 미세 조정 및 멀티 뷰 테스트를 통해 Oquab 등(77.7%)과 Chatfield 등(82.42%)을 얻었다. 우리의 결과는 미세 조정 없이 하나의 전체 이미지 표현만을 사용하여 최첨단 기술과 비교할 수 있습니다.

* 1. **Experiments on Caltech101**

The Caltech101 dataset contains 9,144 images in 102 categories (one background). We randomly sample 30 images per category for training and up to 50 images per category for testing. We repeat 10 random splits and average the accuracy. Table 7 summarizes our results.

Caltech101 데이터 세트에는 102개 카테고리(배경 1개)의 9,144개의 이미지가 포함되어 있습니다. 트레이닝에서는 카테고리별로 30매, 테스트에서는 카테고리별로 50매까지 랜덤으로 샘플링을 실시합니다. 랜덤 분할을 10회 반복하여 정확도를 평균화합니다. 표 7은 결과를 정리한 것입니다.

There are some common observations in the Pascal VOC 2007 and Caltech101 results: SPP-net is better than the no-SPP net (Table 7 (b) vs. (a)), and the fullview representation is better than the crop ((c) vs. (b)). But the results in Caltech101 have some differences with Pascal VOC. The fully-connected layers are less accurate, and the SPP layers are better. This is possibly because the object categories in Caltech101 are less related to those in ImageNet, and the deeper layers are more category-specialized. Further, we find that the scale 224 has the best performance among the scales we tested on this dataset. This is mainly because the objects in Caltech101 also occupy large regions of the images, as is the case of ImageNet.

Pascal VOC 2007 및 Caltech101 결과에는 SPP-net이 no-SPP net보다 우수하며(표 7 (b) vs. (a)), 풀뷰 표현은 작물(c) vs. (b)보다 우수하다는 몇 가지 일반적인 관찰이 있다. 그러나 Caltech101의 결과는 Pascal VOC와 몇 가지 차이가 있습니다. 완전히 연결된 레이어는 정확도가 떨어지고 SPP 레이어가 더 우수합니다. 이는 Caltech101의 오브젝트카테고리가 ImageNet의 오브젝트카테고리와 관련성이 적고 레이어가 깊을수록 카테고리에 특화되어 있기 때문일 수 있습니다. 또한 스케일 224는 이 데이터셋에서 테스트한 스케일 중 최고의 성능을 발휘합니다. 이는 주로 ImageNet과 마찬가지로 Caltech101의 오브젝트도 이미지의 큰 영역을 차지하기 때문입니다.

Besides cropping, we also evaluate warping the image to fit the 224×224 size. This solution maintains the complete content, but introduces distortion. On the SPP (ZF-5) model, the accuracy is 89.91% using the SPP layer as features - lower than 91.44% which uses the same model on the undistorted full image.

자르기 외에 224×224 사이즈에 맞추어 이미지를 왜곡하는 것도 평가합니다. 이 솔루션에서는 완전한 내용은 유지되지만 왜곡이 발생합니다. SPP(ZF-5) 모델의 경우 SPP 레이어를 기능으로 사용하는 정확도는 89.91%로, 왜곡되지 않은 전체 이미지에서 동일한 모델을 사용하는 91.44%보다 낮습니다.

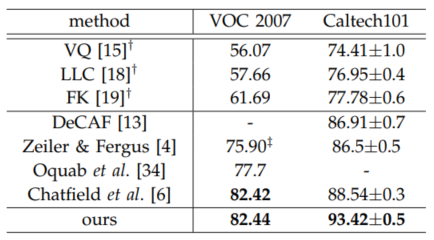


Table 8: Classification results for Pascal VOC 2007 (mAP) and Caltech101 (accuracy). †numbers reported by [27]. ‡our implementation as in Table 6 (a).

표 8: Pascal VOC 2007(mAP) 및 Caltech101(정확도) 분류 결과 [27]에 의해 보고된 §. § 표 6 (a)와 같은 델의 구현.

Table 8 summarizes our results compared with the state-of-the-art methods on Caltech101. Our result (93.42%) exceeds the previous record (88.54%) by a substantial margin (4.88%).

표 8은 Caltech101의 최첨단 방법과 비교한 결과를 요약한 것이다. 우리의 결과(93.42%)는 이전 기록(88.54%)을 크게 웃돌았다(4.88%)

1. **SPP-Net for Object Detection**

Deep networks have been used for object detection. We briefly review the recent state-of-the-art R-CNN method. R-CNN first extracts about 2,000 candidate windows from each image via selective search. Then the image region in each window is warped to a fixed size (227×227). A pre-trained deep network is used to extract the feature of each window. A binary SVM classifier is then trained on these features for detection. R-CNN generates results of compelling quality and substantially outperforms previous methods. However, because R-CNN repeatedly applies the deep convolutional network to about 2,000 windows per image, it is time-consuming. Feature extraction is the major timing bottleneck in testing.

객체 검출에는 딥 네트워크가 사용되고 있습니다. 최신 R-CNN 방법을 간략히 살펴본다. R-CNN은 우선 선택 검색을 통해 각 이미지에서 약 2,000개의 후보 창을 추출합니다. 그런 다음 각 창의 이미지 영역이 고정된 크기(227×227)로 휘어집니다. 각 창의 기능을 추출하기 위해 사전 훈련을 받은 심층 네트워크가 사용됩니다. 다음으로 검출을 위해 이들 기능에 대해 바이너리 SVM 분류기가 훈련됩니다. R-CNN은 매력적인 품질의 결과를 생성하며 이전 방법을 크게 능가한다. 그러나 R-CNN은 이미지당 약 2,000개의 창에 딥 컨볼루션 네트워크를 반복적으로 적용하기 때문에 시간이 걸립니다. 기능 추출은 테스트의 주요 타이밍 병목 현상입니다.

Our SPP-net can also be used for object detection. We extract the feature maps from the entire image only once (possibly at multiple scales). Then we apply the spatial pyramid pooling on each candidate window of the feature maps to pool a fixed-length representation of this window (see Figure 5). Because the time-consuming convolutions are only applied once, our method can run orders of magnitude faster.

SPP-net은 물체 감지에도 사용할 수 있습니다. 전체 이미지에서 기능 맵을 추출하는 것은 1회뿐입니다(여러 축척일 수도 있습니다). 그런 다음 기능 맵의 각 후보 창에 공간 피라미드 풀링을 적용하여 이 창의 고정 길이 표현을 풀링합니다(그림 5 참조). 시간이 많이 걸리는 컨볼루션은 한 번만 적용되기 때문에 우리의 방법은 훨씬 더 빠르게 실행할 수 있습니다.

Our method extracts window-wise features from regions of the feature maps, while R-CNN extracts directly from image regions. In previous works, the Deformable Part Model (DPM) extracts features from windows in HOG feature maps, and the Selective Search (SS) method extracts from windows in encoded SIFT feature maps. The Overfeat detection method also extracts from windows of deep convolutional feature maps, but needs to predefine the window size. On the contrary, our method enables feature extraction in arbitrary windows from the deep convolutional feature maps.

우리의 방법은 피쳐 맵의 영역에서 윈도우별 피쳐를 추출하는 반면, R-CNN은 이미지 영역에서 직접 추출한다. 이전 작업에서는 DPM(Deformable Part Model)이 HOG 피처 맵의 윈도우에서 피처를 추출하고 SS(선택 검색) 메서드가 인코딩된 SIFT 피처 맵의 윈도우에서 피처를 추출했습니다. Overfeat 검출 방법은 딥 컨볼루션피처 맵의 창에서도 추출하지만 창 크기를 미리 정의해야 합니다. 반대로, 우리의 방법은 심층 컨볼루션 피쳐 맵에서 임의 창에서 피쳐 추출을 가능하게 한다.

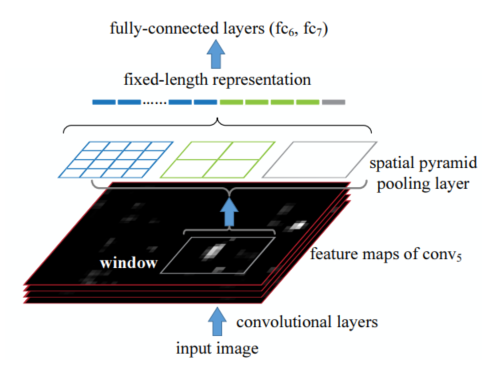


Figure 5: Pooling features from arbitrary windows on feature maps. The feature maps are computed from the entire image. The pooling is performed in candidate windows.

그림 5: 기능 맵 상의 임의의 창에서 기능 풀링 피처 맵은 이미지 전체에서 계산됩니다. 풀링은 후보 창에서 수행됩니다.

* 1. **Detection Algorithm**

We use the “fast” mode of selective search to generate about 2,000 candidate windows per image. Then we resize the image such that min(w, h) = s, and extract the feature maps from the entire image. We use the SPP-net model of ZF-5 (single-size trained) for the time being. In each candidate window, we use a 4-level spatial pyramid (1×1, 2×2, 3×3, 6×6, totally 50 bins) to pool the features. This generates a 12,800-d (256×50) representation for each window. These representations are provided to the fully-connected layers of the network. Then we train a binary linear SVM classifier for each category on these features.

"빠른" 선택 검색 모드를 사용하여 이미지당 약 2,000개의 후보 창을 생성합니다. 그런 다음 min(w, h) = s가 되도록 이미지의 크기를 조정하고 전체 이미지에서 피쳐 맵을 추출합니다. 당분간 ZF-5의 SPP-net 모델을 사용합니다(단일 크기 교육). 각 후보 창에서 4단계 공간 피라미드(1×1, 2×2, 3×3, 6×6, 총 50개의 빈)를 사용하여 특징을 풀링합니다. 그러면 각 창에 대해 12,800-d(256×50)의 표현이 생성됩니다. 이러한 표현은 네트워크의 완전히 연결된 계층에 제공됩니다. 그런 다음 이러한 기능의 각 범주에 대해 이진 선형 SVM 분류기를 교육합니다.

Our implementation of the SVM training follows. We use the ground-truth windows to generate the positive samples. The negative samples are those overlapping a positive window by at most 30% (measured by the intersection-over-union (IoU) ratio). Any negative sample is removed if it overlaps another negative sample by more than 70%. We apply the standard hard negative mining to train the SVM. This step is iterated once. It takes less than 1 hour to train SVMs for all 20 categories. In testing, the classifier is used to score the candidate windows. Then we use non-maximum suppression (threshold of 30%) on the scored windows.

SVM 트레이닝의 실장은 다음과 같습니다. 우리는 양성 샘플을 생성하기 위해 지상 진실 창을 사용합니다. 음의 샘플은 양의 창이 최대 30%까지 겹치는 샘플입니다(교차 결합비(IoU)로 측정). 음의 샘플이 다른 음의 샘플과 70% 이상 겹치면 모든 음의 샘플이 제거됩니다. 표준 하드 네거티브 마이닝을 적용하여 SVM을 훈련시킵니다. 이 스텝은 1회 반복됩니다. 20개 카테고리 모두에 대해 SVM을 훈련하는 데 1시간 미만이 소요됩니다. 테스트에서는 분류자를 사용하여 후보 창에 점수를 매깁니다. 그런 다음 점수가 매겨진 창에서 최대값이 아닌 억제 값(30%)을 사용합니다.

Our method can be improved by multi-scale feature extraction. We resize the image such that min(w, h) = s ∈ S = {480, 576, 688, 864, 1200}, and compute the feature maps of conv5 for each scale. One strategy of combining the features from these scales is to pool them channel-by-channel. But we empirically find that another strategy provides better results. For each candidate window, we choose a single scale s ∈ S such that the scaled candidate window has a number of pixels closest to 224×224. Then we only use the feature maps extracted from this scale to compute the feature of this window. If the pre-defined scales are dense enough and the window is approximately square, our method is roughly equivalent to resizing the window to 224×224 and then extracting features from it. Nevertheless, our method only requires computing the feature maps once (at each scale) from the entire image, regardless of the number of candidate windows.

우리의 방법은 멀티스케일 피쳐 추출을 통해 개선될 수 있다. min(w, h) = s s S = {buff, 576, 688, 864, 1200}이 되도록 이미지의 크기를 조정하고 각 척도에 대한 conv5의 피쳐 맵을 계산합니다. 이러한 스케일로부터 기능을 조합하는 하나의 전략은 채널별로 기능을 풀링하는 것입니다. 그러나 우리는 경험적으로 다른 전략이 더 나은 결과를 제공한다는 것을 알게 되었습니다. 각 후보창에 대해 스케일된 후보창이 224×224에 가장 가까운 픽셀 수를 가지도록 단일 스케일 s δ S를 선택한다. 다음으로 이 스케일에서 추출한 피쳐 맵만 사용하여 이 창의 피쳐를 계산합니다. 미리 정의된 축척이 충분히 조밀하고 창이 거의 정사각형일 경우, 우리의 방법은 윈도우 크기를 224×224로 조정한 후 피쳐를 추출하는 것과 거의 동등합니다. 그러나 이 방법에서는 후보 창의 수에 관계없이 전체 이미지에서 기능 맵을 한 번(각 스케일)만 계산하면 됩니다.

We also fine-tune our pre-trained network, following. Since our features are pooled from the conv5 feature maps from windows of any sizes, for simplicity we only fine-tune the fully-connected layers. In this case, the data layer accepts the fixed-length pooled features after conv5, and the fc6,7 layers and a new 21-way (one extra negative category) fc8 layer follow. The fc8 weights are initialized with a Gaussian distribution of σ=0.01. We fix all the learning rates to 1e-4 and then adjust to 1e-5 for all three layers. During fine-tuning, the positive samples are those overlapping with a ground-truth window by [0.5, 1], and the negative samples by [0.1, 0.5). In each mini-batch, 25% of the samples are positive. We train 250k minibatches using the learning rate 1e-4, and then 50k mini-batches using 1e-5. Because we only fine-tune the fc layers, the training is very fast and takes about 2 hours on the GPU (excluding pre-caching feature maps which takes about 1 hour). Also following [7], we use bounding box regression to post-process the prediction windows. The features used for regression are the pooled features from conv5 (as a counterpart of the pool5 features used in [7]). The windows used for the regression training are those overlapping with a ground-truth window by at least 50%.

또, 사전 트레이닝이 끝난 네트워크의 미세 조정도 실시합니다. NAT 기능은 모든 크기의 창에서 conv5 기능 맵에서 풀링되므로 단순성을 위해 완전히 연결된 계층만 미세 조정합니다. 이 경우 데이터 계층은 conv5 이후 고정 길이 풀링 기능을 수용하고 fc6,7 계층과 새로운 21-way(1개의 추가 음의 카테고리) fc8 계층이 뒤따릅니다. fc8 가중치는 θ=0.01의 가우스 분포로 초기화된다. 모든 학습 속도를 1e-4로 설정한 후 3개 레이어 모두에 대해 1e-5로 조정합니다. 미세 조정 중에 양수 샘플은 [0.5, 1]만큼 지상 진실 창과 겹치고 음수 샘플은 [0.1, 0.5]만큼 겹칩니다. 각 미니 배치에서 표본의 25%가 양성입니다. 학습 속도 1e-4를 사용하여 25만 개의 미니 배치를 교육하고, 1e-5를 사용하여 50만 개의 미니 배치를 교육합니다. fc 레이어만 미세 조정하기 때문에 훈련은 매우 빠르고 GPU에서 약 2시간이 소요됩니다(약 1시간이 걸리는 프리캐싱 기능 맵 제외). 또한 [7]에 이어 경계 상자 회귀 분석을 사용하여 예측 창을 후처리합니다. 회귀 분석에 사용되는 기능은 conv5의 풀링된 피쳐입니다([7]에서 사용되는 pool5 피쳐의 일부로 사용). 회귀 훈련에 사용되는 창은 지상 실측 창과 최소 50% 이상 겹치는 창입니다.

* 1. **Detection Results**

We evaluate our method on the detection task of the Pascal VOC 2007 dataset. Table 9 shows our results on various layers, by using 1-scale (s=688) or 5-scale. Here the R-CNN results are as reported in [7] using the AlexNet with 5 conv layers. Using the pool5 layers (in our case the pooled features), our result (44.9%) is comparable with R-CNN’s result (44.2%). But using the non-fine-tuned fc6 layers, our results are inferior. An explanation is that our fc layers are pretrained using image regions, while in the detection case they are used on the feature map regions. The feature map regions can have strong activations near the window boundaries, while the image regions may not. This difference of usages can be addressed by fine-tuning. Using the fine-tuned fc layers (ftfc6,7), our results are comparable with or slightly better than the fine-tuned results of R-CNN. After bounding box regression, our 5-scale result (59.2%) is 0.7% better than R-CNN (58.5%), and our 1-scale result (58.0%) is 0.5% worse.

Pascal VOC 2007 데이터 세트의 검출 태스크에 대해 우리의 방법을 평가한다. 표 9는 1 척도(s=688) 또는 5 척도를 사용하여 다양한 계층에 대한 결과를 보여준다. 여기서 R-CNN 결과는 [7]에서 보고한 대로 5개의 conv 레이어가 있는 AlexNet을 사용한 결과입니다. pool5 레이어(이 경우 풀링된 기능)를 사용하면 결과(44.9%)는 R-CNN 결과(44.2%)와 유사하다. 그러나 미세 조정되지 않은 fc6 레이어를 사용하면 결과가 열악합니다. fc 레이어는 이미지 영역을 사용하여 사전 테스트되지만 검출의 경우 기능 맵 영역에서 사용됩니다. 피쳐 맵 영역은 창 경계 부근에서 강력한 액티베이션이 발생할 수 있지만 이미지 영역은 그렇지 않을 수 있습니다. 이러한 용도의 차이는 미세 조정을 통해 해결할 수 있습니다. 미세 조정된 fc 레이어(ftfc6,7)를 사용하면 R-CNN의 미세 조정 결과와 비슷하거나 약간 더 우수합니다. 경계 상자 회귀 분석 후 5단계 결과(59.2%)는 R-CNN(58.5%)보다 0.7%, 1단계(58.0%)는 0.5% 더 나쁩니다.

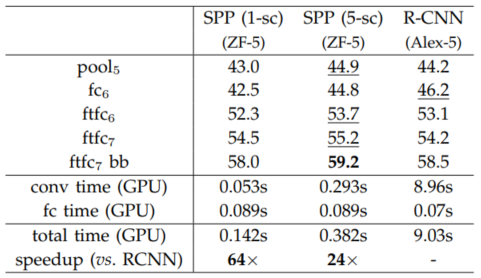


Table 9: Detection results (mAP) on Pascal VOC 2007. “ft” and “bb” denote fine-tuning and bounding box regression.

표 9: Pascal VOC 2007의 검출 결과(mAP) "ft" 및 "bb"는 미세 조정 및 경계 상자 회귀를 나타냅니다.

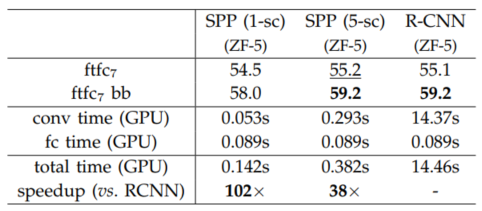


Table 10: Detection results (mAP) on Pascal VOC 2007, using the same pre-trained model of SPP (ZF-5).

표 10: Pascal VOC 2007에서 SPP(ZF-5)의 사전 교육 모델을 사용한 검출 결과(mAP)

In Table 10 we further compare with R-CNN using the same pre-trained model of SPPnet (ZF-5). In this case, our method and R-CNN have comparable averaged scores. The R-CNN result is boosted by this pre-trained model. This is because of the better architecture of ZF-5 than AlexNet, and also because of the multi-level pooling of SPPnet (if using the noSPP ZF-5, the R-CNN result drops). Table 11 shows the results for each category.

표 10에서는 SPPnet의 동일한 사전 교육 모델(ZF-5)을 사용하여 R-CNN과 추가로 비교한다. 이 경우, 우리의 방법과 R-CNN은 비교 가능한 평균 점수를 가진다. R-CNN 결과는 이 사전 교육 모델에 의해 향상됩니다. 이는 AlexNet보다 ZF-5의 아키텍처가 우수하고 SPPnet의 다단계 풀링(noSPP ZF-5를 사용하면 R-CNN 결과가 저하됨) 때문입니다. 표 11은 각 카테고리의 결과를 나타내고 있습니다.

Table 11 also includes additional methods. Selective Search (SS) [20] applies spatial pyramid matching on SIFT feature maps. DPM [23] and Regionlet [39] are based on HOG features [24]. The Regionlet method improves to 46.1% [8] by combining various features including conv5. DetectorNet [40] trains a deep network that outputs pixel-wise object masks. This method only needs to apply the deep network once to the entire image, as is the case for our method. But this method has lower mAP (30.5%).

표 11에는 추가 방법도 포함되어 있습니다. Selective Search(SS) [20]는 SIFT 피쳐 맵에 공간 피라미드 매칭을 적용합니다. DPM [23] 및 Regionlet [39]는 HOG 기능[24]을 기반으로 합니다. Regionlet 방법은 conv5를 포함한 다양한 기능을 결합하여 46.1%[8]로 향상되었습니다. DetectorNet [40]은(는) 픽셀 단위 개체 마스크를 출력하는 심층 네트워크를 훈련합니다. 이 방법에서는, 이 방법의 경우와 같이, 딥 네트워크를 이미지 전체에 1회만 적용하면 됩니다. 단, 이 방법은 mAP(30.5%)가 낮습니다.

**4.3 Complexity and Running Time**

Despite having comparable accuracy, our method is much faster than R-CNN. The complexity of the convolutional feature computation in R-CNN is with the window number n (∼2000). This complexity of our method is at a scale s, where r is the aspect ratio. Assume r is about 4/3. In the single-scale version when s = 688, this complexity is about 1/160 of R-CNN’s; in the 5-scale version, this complexity is about 1/24 of R-CNN’s.

비슷한 정확도를 가지고 있음에도 불구하고 우리의 방법은 R-CNN보다 훨씬 빠릅니다. R-CNN에서 convolutional feature 계산의 복잡성은 O(n,227^2)이고 윈도우 번호는 n(~2000)입니다. 이 방법의 복잡성은 스케일 s에서 O(r∙s^2)이며, 여기서 r은 종횡비입니다. r이 약 4/3라고 가정합니다. s = 688인 단일 규모 버전에서 이 복잡성은 R-CNN의 약 1/160입니다. 5 스케일 버전에서 이 복잡성은 R-CNN의 약 1/24입니다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 11: Comparisons of detection results on Pascal VOC 2007.

표 11: Pascal VOC 2007에 대한 탐지 결과 비교

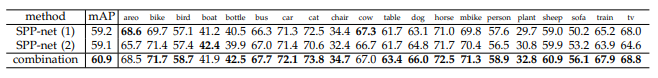


Table 12: Detection results on VOC 2007 using model combination. The results of both models use “ftfc7 bb”.

표 12: 모델 조합을 사용한 VOC 2007 탐지 결과. 두 모델의 결과는 "ftfc7 bb"를 사용합니다.

In Table 10, we provide a fair comparison on the running time of the feature computation using the same SPP (ZF-5) model. The implementation of RCNN is from the code published by the authors implemented in Caffe [35]. We also implement our feature computation in Caffe. In Table 10 we evaluate the average time of 100 random VOC images using GPU. R-CNN takes 14.37s per image for convolutions, while our 1-scale version takes only 0.053s per image. So ours is 270× faster than R-CNN. Our 5-scale version takes 0.293s per image for convolutions, so is 49× faster than R-CNN. Our convolutional feature computation is so fast that the computational time of fc layers takes a considerable portion. Table 10 shows that the GPU time of computing the 4,096-d features is 0.089s per image. Considering both convolutional and fully-connected features, our 1-scale version is 102× faster than R-CNN and is 1.2% inferior; our 5-scale version is 38× faster and has comparable results.

표 10에서는 동일한 SPP(ZF-5) 모델을 사용하여 기능 계산의 실행 시간에 대한 공정한 비교를 제공합니다. RCNN의 구현은 Caffe[35]에 구현된 작성자가 게시한 코드에서 가져옵니다. 또한 Caffe에서 기능 계산을 구현합니다. 표 10에서 GPU를 사용하여 100개의 임의 VOC 이미지의 평균 시간을 평가합니다. R-CNN은 컨볼루션에 대해 이미지당 14.37초가 소요되는 반면 1 스케일 버전은 이미지당 0.053초만 소요됩니다. 따라서 우리는 R-CNN보다 270배 빠릅니다. 우리의 5-스케일 버전은 컨볼루션에 대해 이미지당 0.293초가 걸리므로 R-CNN보다 49배 빠릅니다. 우리의 컨볼루션 피쳐 계산은 너무 빨라서 fc 레이어의 계산 시간이 상당한 부분을 차지합니다. 표 10은 4,096-d fc\_7feature를 계산하는 GPU 시간이 이미지당 0.089초임을 보여줍니다. 컨볼루션 및 완전 연결 기능을 모두 고려할 때 1 스케일 버전은 R-CNN보다 102배 빠르며 1.2% 열등합니다. 5-scale 버전은 38배 더 빠르고 비슷한 결과를 제공합니다.

We also compares the running time in Table 9 where R-CNN uses AlexNet [3] as is in the original paper [7]. Our method is 24× to 64× faster. Note that the AlexNet [3] has the same number of filters as our ZF5 on each conv layer. The AlexNet is faster because it uses splitting on some layers, which was designed for two GPUs in [3].

우리는 또한 R-CNN이 원본 논문[7]에서와 같이 AlexNet[3]을 사용하는 표 9에서 실행 시간을 비교합니다. 우리의 방법은 24배에서 64배 더 빠릅니다. AlexNet[3]은 각 변환 레이어에서 ZF5와 동일한 수의 필터를 가지고 있습니다. AlexNet은 [3]에서 2개의 GPU용으로 설계된 일부 레이어에서 분할을 사용하기 때문에 더 빠릅니다.

We further achieve an efficient full system with the help of the recent window proposal method [25]. The Selective Search (SS) proposal [20] takes about 1-2 seconds per image on a CPU. The method of EdgeBoxes [25] only takes ∼ 0.2s. Note that it is sufficient to use a fast proposal method during testing only. Using the same model trained as above (using SS), we test proposals generated by EdgeBoxes only. The mAP is 52.8 without bounding box regression. This is reasonable considering that EdgeBoxes are not used for training. Then we use both SS and EdgeBox as proposals in the training stage, and adopt only EdgeBoxes in the testing stage. The mAP is 56.3 without bounding box regression, which is better than 55.2 (Table 10) due to additional training samples. In this case, the overall testing time is ∼0.5s per image including all steps (proposal and recognition). This makes our method practical for real-world applications.

우리는 최근 창 제안 방법[25]의 도움으로 효율적인 전체 시스템을 더 달성합니다. SS(Selective Search) 제안[20]은 CPU에서 이미지당 약 1-2초가 걸립니다. EdgeBoxes[25]의 방법은 ~ 0.2초 밖에 걸리지 않습니다. 테스트 중에만 빠른 제안 방법을 사용하는 것으로 충분합니다. 위와 같이 훈련된 동일한 모델(SS 사용)을 사용하여 EdgeBox에서 생성된 제안만 테스트합니다. mAP는 경계 상자 회귀 없이 52.8입니다. 이것은 EdgeBox가 훈련에 사용되지 않는다는 점을 고려하면 합리적입니다. 그런 다음 교육 단계에서는 SS와 EdgeBox를 모두 제안으로 사용하고 테스트 단계에서는 EdgeBox만 채택합니다. mAP는 경계 상자 회귀가 없는 56.3이며 추가 훈련 샘플로 인해 55.2(표 10)보다 우수합니다. 이 경우 전체 테스트 시간은 모든 단계(제안 및 인식)를 포함하여 이미지당 ~0.5초입니다. 이것은 우리의 방법을 실제 적용에 실용적으로 만듭니다.

**4.4 Model Combination for Detection**

Model combination is an important strategy for boosting CNN-based classification accuracy [3]. We propose a simple combination method for detection. We pre-train another network in ImageNet, using the same structure but different random initializations. Then we repeat the above detection algorithm. Table 12 (SPP-net (2)) shows the results of this network. Its mAP is comparable with the first network (59.1% vs. 59.2%), and outperforms the first network in 11 categories.

모델 조합은 CNN 기반 분류 정확도를 높이는 중요한 전략입니다[3]. 우리는 검출을 위한 간단한 조합 방법을 제안합니다. 동일한 구조를 사용하지만 다른 임의 초기화를 사용하여 ImageNet에서 다른 네트워크를 사전 훈련합니다. 그런 다음 위의 감지 알고리즘을 반복합니다. 이 네트워크의 결과는 표 12(SPP-net(2))와 같다. mAP는 첫 번째 네트워크(59.1% 대 59.2%)와 비슷하며 11개 범주에서 첫 번째 네트워크를 능가합니다.

Given the two models, we first use either model to score all candidate windows on the test image. Then we perform non-maximum suppression on the union of the two sets of candidate windows (with their scores). A more confident window given by one method can suppress those less confident given by the other method. After combination, the mAP is boosted to 60.9% (Table 12). In 17 out of all 20 categories the combination performs better than either individual model. This indicates that the two models are complementary.

두 모델이 주어지면 먼저 테스트 이미지의 모든 후보 창에 점수를 매기기 위해 두 모델 중 하나를 사용합니다. 그런 다음 두 후보 창 세트(점수 포함)의 합집합에 대해 최대가 아닌 억제를 수행합니다. 한 방법에 의해 주어진 더 확실한 창은 다른 방법에 의해 주어진 덜 확신하는 것을 억제할 수 있습니다. 조합 후 mAP는 60.9%로 향상됩니다(표 12). 20개 범주 중 17개 범주에서 조합이 개별 모델보다 더 나은 성능을 보입니다. 이는 두 모델이 상호 보완적임을 나타냅니다.

We further find that the complementarity is mainly because of the convolutional layers. We have tried to combine two randomly initialized fine-tuned results of the same convolutional model, and found no gain.

우리는 또한 상보성이 주로 컨볼루션 레이어 때문이라는 것을 발견했습니다. 우리는 동일한 컨볼루션 모델의 무작위로 초기화된 두 개의 미세 조정된 결과를 결합하려고 시도했지만 이득을 찾지 못했습니다.

**4.5 ILSVRC 2014 Detection**

The ILSVRC 2014 detection [26] task involves 200 categories. There are ∼450k/20k/40k images in the training/validation/testing sets. We focus on the task of the provided-data-only track (the 1000-category CLS training data is not allowed to use).

ILSVRC 2014 탐지[26] 작업에는 200개의 범주가 포함됩니다. 훈련/검증/테스트 세트에는 ~450k/20k/40k 이미지가 있습니다. 제공된 데이터 전용 트랙의 작업에 중점을 둡니다(1000개 범주의 CLS 교육 데이터는 사용할 수 없음).

There are three major differences between the detection (DET) and classification (CLS) training datasets, which greatly impacts the pre-training quality. First, the DET training data is merely 1/3 of the CLS training data. This seems to be a fundamental challenge of the provided-data-only DET task. Second, the category number of DET is 1/5 of CLS. To overcome this problem, we harness the provided subcategory labels2 for pre-training. There are totally 499 nonoverlapping subcategories (i.e., the leaf nodes in the provided category hierarchy). So we pre-train a 499- category network on the DET training set. Third, the distributions of object scales are different between DET/CLS training sets. The dominant object scale in CLS is about 0.8 of the image length, but in DET is about 0.5. To address the scale difference, we resize each training image to min(w, h) = 400 (instead of 256), and randomly crop 224×224 views for training. A crop is only used when it overlaps with a ground truth object by at least 50%.

탐지(DET)와 분류(CLS) 훈련 데이터 세트 사이에는 세 가지 주요 차이점이 있으며 이는 사전 훈련 품질에 큰 영향을 미칩니다. 첫째, DET 훈련 데이터는 CLS 훈련 데이터의 1/3에 불과합니다. 이것은 제공된 데이터 전용 DET 작업의 근본적인 문제인 것 같습니다. 둘째, DET의 범주 번호는 CLS의 1/5입니다. 이 문제를 극복하기 위해 사전 훈련을 위해 제공된 하위 범주 레이블2를 활용합니다. 겹치지 않는 하위 범주는 총 499개 있습니다(즉, 제공된 범주 계층 구조의 리프 노드). 그래서 우리는 DET 훈련 세트에서 499개의 범주 네트워크를 사전 훈련합니다. 셋째, DET/CLS 훈련 세트 간에 객체 스케일의 분포가 다릅니다. CLS에서 지배적인 개체 크기는 이미지 길이의 약 0.8이지만 DET에서는 약 0.5입니다. 스케일 차이를 해결하기 위해 각 훈련 이미지의 크기를 min(w, h) = 400(256 대신)으로 조정하고 훈련을 위해 224×224 뷰를 무작위로 자릅니다. 자르기는 실측 객체와 50% 이상 겹치는 경우에만 사용됩니다.

We verify the effect of pre-training on Pascal VOC 2007. For a CLS-pre-training baseline, we consider the pool5 features (mAP 43.0% in Table 9). Replaced with a 200-category network pre-trained on DET, the mAP significantly drops to 32.7%. A 499-category pre-trained network improves the result to 35.9%. Interestingly, even if the amount of training data do not increase, training a network of more categories boosts the feature quality. Finally, training with min(w, h) = 400 instead of 256 further improves the mAP to 37.8%. Even so, we see that there is still a considerable gap to the CLS-pre-training result. This indicates the importance of big data to deep learning

Pascal VOC 2007에 대한 사전 훈련의 효과를 확인합니다. CLS 사전 훈련 기준선의 경우 pool5 기능을 고려합니다(표 9의 mAP 43.0%). DET에 대해 사전 훈련된 200개 범주 네트워크로 교체하면 mAP가 32.7%로 크게 떨어집니다. 499개 범주의 사전 훈련된 네트워크는 결과를 35.9%로 향상시킵니다. 흥미롭게도 훈련 데이터의 양이 증가하지 않더라도 더 많은 범주의 네트워크를 훈련하면 기능 품질이 향상됩니다. 마지막으로, 256 대신 min(w, h) = 400으로 훈련하면 mAP가 37.8%로 추가로 향상됩니다. 그럼에도 불구하고 CLS 사전 훈련 결과에는 여전히 상당한 격차가 있음을 알 수 있습니다. 이는 딥 러닝에 대한 빅 데이터의 중요성을 나타냅니다.

For ILSVRC 2014, we train a 499-category Overfeat7 SPP-net. The remaining steps are similar to the VOC 2007 case. Following [7], we use the validation set to generate the positive/negative samples, with windows proposed by the selective search fast mode. The training set only contributes positive samples using the ground truth windows. We fine-tune the fc layers and then train the SVMs using the samples in both validation and training sets. The bounding box regression is trained on the validation set.

ILSVRC 2014의 경우 499개 범주의 Overfeat7 SPP-net을 훈련합니다. 나머지 단계는 VOC 2007의 경우와 유사합니다. [7]에 따라 검증 세트를 사용하여 선택적 검색 빠른 모드에서 제안한 창과 함께 양/음 샘플을 생성합니다. 훈련 세트는 정답 창을 사용하여 긍정적인 샘플만 제공합니다. fc 레이어를 미세 조정한 다음 검증 및 훈련 세트의 샘플을 사용하여 SVM을 훈련합니다. 경계 상자 회귀는 검증 세트에서 훈련됩니다.

Our single model leads to 31.84% mAP in the ILSVRC 2014 testing set [26]. We combine six similar models using the strategy introduced in this paper. The mAP is 35.11% in the testing set [26]. This result ranks #2 in the provided-data-only track of ILSVRC 2014 (Table 13) [26]. The winning result is 37.21% from NUS, which uses contextual information.

우리의 단일 모델은 ILSVRC 2014 테스트 세트에서 31.84% mAP로 이어집니다[26]. 이 백서에서 소개한 전략을 사용하여 6개의 유사한 모델을 결합합니다. mAP는 테스트 세트에서 35.11%입니다[26]. 이 결과는 ILSVRC 2014(표 13)[26]의 제공된 데이터 전용 트랙에서 2위를 차지했습니다. 콘텍스트 정보를 활용한 NUS가 37.21%로 1위를 차지했다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 13: The competition results of ILSVRC 2014 detection (provided-data-only track) [26]. The best entry of each team is listed.

표 13: ILSVRC 2014 탐지의 경쟁 결과(제공된 데이터 전용 트랙) [26]. 각 팀의 베스트 엔트리가 나열됩니다.

Our system still shows great advantages on speed for this dataset. It takes our single model 0.6 seconds (0.5 for conv, 0.1 for fc, excluding proposals) per testing image on a GPU extracting convolutional features from all 5 scales. Using the same model, it takes 32 seconds per image in the way of RCNN. For the 40k testing images, our method requires 8 GPU·hours to compute convolutional features, while RCNN would require 15 GPU·days.

우리 시스템은 이 데이터 세트의 속도면에서 여전히 큰 이점을 보여줍니다. GPU의 테스트 이미지당 단일 모델이 0.6초(전환의 경우 0.5초, fc의 경우 0.1초, 제안 제외)가 5개 척도 모두에서 컨볼루션 기능을 추출하는 데 걸립니다. 동일한 모델을 사용하면 RCNN 방식으로 이미지당 32초가 소요됩니다. 40k 테스트 이미지의 경우 우리의 방법은 컨볼루션 기능을 계산하는 데 8 GPU·시간이 필요한 반면 RCNN은 15 GPU·일이 필요합니다.

**5. CONCLUSION**

SPP is a flexible solution for handling different scales, sizes, and aspect ratios. These issues are important in visual recognition, but received little consideration in the context of deep networks. We have suggested a solution to train a deep network with a spatial pyramid pooling layer. The resulting SPP-net shows outstanding accuracy in classification/detection tasks and greatly accelerates DNN-based detection. Our studies also show that many time-proven techniques/insights in computer vision can still play important roles in deep-networks-based recognition.

SPP는 다양한 축척, 크기 및 종횡비를 처리할 수 있는 유연한 솔루션입니다. 이러한 문제는 시각적 인식에서 중요하지만 심층 네트워크의 맥락에서 거의 고려되지 않았습니다. 우리는 공간 피라미드 풀링 레이어를 사용하여 심층 네트워크를 훈련하는 솔루션을 제안했습니다. 그 결과 SPP-net은 분류/탐지 작업에서 뛰어난 정확도를 보여주고 DNN 기반 탐지를 크게 가속화합니다. 우리의 연구는 또한 컴퓨터 비전의 많은 시간 입증된 기술/통찰이 딥 네트워크 기반 인식에서 여전히 중요한 역할을 할 수 있음을 보여줍니다.

**APPENDIX A**

APPENDIX A In the appendix, we describe some implementation details:

부록 A 부록에서는 몇 가지 구현 세부 정보를 설명합니다.

Mean Subtraction. The 224×224 cropped training/testing images are often pre-processed by subtracting the per-pixel mean [3]. When input images are in any sizes, the fixedsize mean image is not directly applicable. In the ImageNet dataset, we warp the 224×224 mean image to the desired size and then subtract it. In Pascal VOC 2007 and Caltech101, we use the constant mean (128) in all the experiments.

평균 빼기. 224×224로 자른 훈련/테스트 이미지는 종종 픽셀당 평균을 빼서 사전 처리됩니다[3]. 입력 이미지가 임의의 크기이면 고정 크기 평균 이미지를 직접 적용할 수 없습니다. ImageNet 데이터 세트에서 224×224 평균 이미지를 원하는 크기로 왜곡한 다음 뺍니다. Pascal VOC 2007 및 Caltech101에서는 모든 실험에서 상수 평균(128)을 사용합니다.

Implementation of Pooling Bins. We use the following implementation to handle all bins when applying the network. Denote the width and height of the conv5 feature maps (can be the full image or a window) as w and h. For a pyramid level with n×n bins, the (i, j)-th bin is in the range of . Intuitively, if rounding is needed, we take the floor operation on the left/top boundary and ceiling on the right/bottom boundary.

풀링 빈의 구현. 네트워크를 적용할 때 모든 빈을 처리하기 위해 다음 구현을 사용합니다. conv5 기능 맵(전체 이미지 또는 창일 수 있음)의 너비와 높이를 w 및 h로 표시합니다. n×n 빈이 있는 피라미드 수준의 경우 (i, j) 번째 빈은 . 직관적으로 반올림이 필요한 경우 왼쪽/상단 경계에서 바닥 연산을 수행하고 오른쪽/하단 경계에서 천장 작업을 수행합니다.

선반, 다른, 다양한, 같은이(가) 표시된 사진

자동 생성된 설명

Figure 6: Example detection results of “SPP-net ftfc7 bb” on the Pascal VOC 2007 testing set (59.2% mAP). All windows with scores > 0 are shown. The predicted category/score are marked. The window color is associated with the predicted category. These images are manually selected because we find them impressive. Visit our project website to see all 4,952 detection results in the testing set.

그림 6: Pascal VOC 2007 테스트 세트(59.2% mAP)에서 "SPP-net ftfc7 bb"의 감지 결과 예. 점수가 0보다 큰 모든 창이 표시됩니다. 예상 카테고리/점수가 표시됩니다. 창 색상은 예측 범주와 연결됩니다. 이 이미지는 인상적이기 때문에 수동으로 선택했습니다. 테스트 세트에서 4,952개의 탐지 결과를 모두 보려면 프로젝트 웹 사이트를 방문하십시오.

Mapping a Window to Feature Maps. In the detection algorithm (and multi-view testing on feature maps), a window is given in the image domain, and we use it to crop the convolutional feature maps (e.g., conv5) which have been sub-sampled several times. So we need to align the window on the feature maps.

기능 맵에 창 매핑. 감지 알고리즘(및 기능 맵에 대한 다중 뷰 테스트)에서 이미지 도메인에 창이 주어지고 이를 사용하여 여러 번 서브샘플링된 컨볼루션 기능 맵(예: conv5)을 자릅니다. 따라서 기능 맵에서 창을 정렬해야 합니다.

In our implementation, we project the corner point of a window onto a pixel in the feature maps, such that this corner point in the image domain is closest to the center of the receptive field of that feature map pixel. The mapping is complicated by the padding of all convolutional and pooling layers. To simplify the implementation, during deployment we pad bp/2c pixels for a layer with a filter size of p. As such, for a response centered at ( ) , its effective receptive field in the image domain is centered at (x, y) = (Sx’ , Sy’ ) where S is the product of all previous strides. In our models, S = 16 for ZF-5 on conv5, and S = 12 for Overfeat-5/7 on . Given a window in the image domain, we project the left (top) boundary by: x’ = [x/S] + 1 and the right (bottom) boundary x’ = [x/S] − 1. If the padding is not [p/2], we need to add a proper offset to x.

우리의 구현에서 우리는 이미지 도메인의 이 코너 포인트가 해당 기능 맵 픽셀의 수용 필드의 중심에 가장 가깝도록 창의 모서리 지점을 기능 맵의 픽셀에 투영합니다. 매핑은 모든 컨볼루션 및 풀링 레이어의 패딩으로 인해 복잡합니다. 구현을 단순화하기 위해 배포 중에 필터 크기가 p인 레이어에 대해 bp/2c 픽셀을 채웁니다. 따라서 ( )에 중심을 둔 응답의 경우 이미지 영역에서 유효 수용 필드는 (x, y) = (Sx', Sy') 중심에 있습니다. 여기서 S는 모든 이전 보폭의 곱입니다. . 우리 모델에서 conv5의 ZF-5의 경우 S = 16이고 의 Overfeat-5/7에 대해 S = 12입니다. 이미지 영역의 창이 주어지면 왼쪽(상단) 경계를 x' = [x/S] + 1 및 오른쪽(하단) 경계 x' = [x/S] - 1로 투영합니다. [p/2]가 아니라 x에 적절한 오프셋을 추가해야 합니다.