Hypercolumns for Object Segmentation and Fine-grained Localization

Bharath Hariharan[1], Pablo Arbeláez[2], Ross Girshick[3], Jitendra Malik[1]
[1]University of California, Berkeley. [2]Universidad de los Andes, Colombia. [3]Microsoft Research, Redmond.

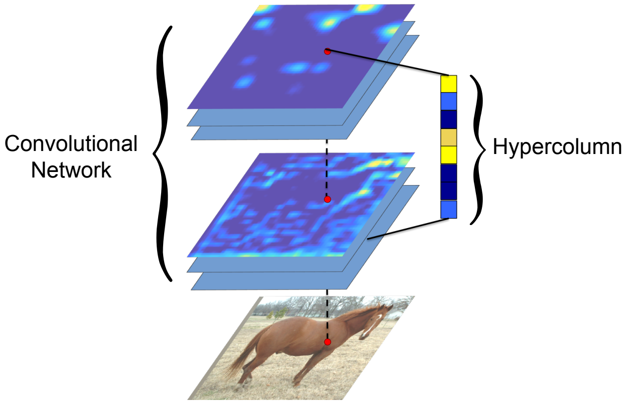

Figure 1: The hypercolumn representation. The bottom image is the input, while the other images represent feature maps of different layers in the CNN. The hypercolumn at a pixel is the vector of activations of all units that lie above that pixel.

Features based on convolutional networks (CNNs) [6] have now led to the best results on a range of recognition tasks [2]. Typically, recognition algorithms use the output of the last layer of the CNN. This makes sense when the task is assigning category labels to images or bounding boxes: the last layer is the most sensitive to category-level semantic information and the most invariant to “nuisance" variables such as pose, illumination, articulation, precise location and so on. However, when the task we are interested in is finer-grained, such as one of segmenting the detected object or estimating its pose, these nuisance variables are precisely what we are interested in. For such applications, the top layer is thus *not* the optimal representation.

The information that is generalized over in the top layer is present in intermediate layers, but intermediate layers are also much less sensitive to semantics. For instance, bar detectors in early layers might localize bars precisely but cannot discriminate between bars that are horse legs and bars that are tree trunks. This observation suggests that reasoning at multiple levels of abstraction and scale is necessary, mirroring other problems in computer vision (such as optical flow) where reasoning across multiple levels has proved beneficial.

In this paper, we think of the layers of the convolutional network as a non-linear counterpart of the image pyramids used in other vision tasks. Our hypothesis is that the information of interest is distributed over *all* levels of the CNN and should be exploited in this way. We define the “hypercolumn" at a given input location as the outputs of all units above that location at all layers of the CNN, stacked into one vector. (Because adjacent layers are correlated, in practice we need not consider all the layers but can simply sample a few.) Figure 1 shows a visualization of the idea. We borrow the term “hypercolumn" from neuroscience, where it is used to describe a set of V1 neurons sensitive to edges at multiple orientations and multiple frequencies arranged in a columnar structure [5]. However, our hypercolumn includes not just edge detectors but also more semantic units and is thus a more general notion.

We show the utility of the hypercolumn representation on two kinds of problems that require precise localization. The first problem is simultaneous detection and segmentation (SDS) [4], where the aim is to both detect and segment every instance of the category in the image. The second problem deals with detecting an object and localizing its parts. We consider two variants of this: one, locating the keypoints [7], and two, segmenting out each part [1].

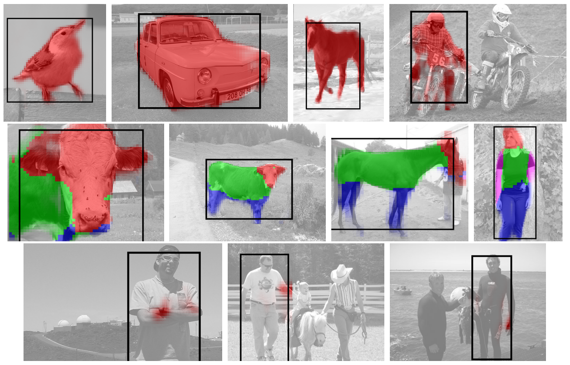

Figure 2: Example results. The first row shows figure-ground segmentations starting from bounding box detections. The second row shows example part labelings. The third row shows example keypoint predictions (left wrist).

We present a general framework for tackling these and other fine-grained localization tasks by framing them as pixel classification. We start from an initial detection of the object (which might come with an initial segmentation). We then classify each pixel in the bounding box as belonging to the object or not (for SDS), as belonging to a part or not (for part labeling) or as lying on a keypoint or not (for keypoint prediction). We use the hypercolumn representation of each pixel as features for this classification task. To incorporate the information provided by the location of the pixel in the bounding box, we use a coarse grid of classifiers, interpolating between them to produce high resolution, precise labelings. Finally, we formulate our entire system as a neural network, allowing end-to-end training for particular tasks simply by changing the target labels.

Our empirical results are:

1. On SDS, the previous state-of-the-art is 49.7 mean AP^r [4]. Substituting hypercolumns into the pipeline of [4] improves this to **52.8**. We also propose a more efficient pipeline that allows us to use a larger network, pushing up the performance to **60.0**.

2. On keypoint prediction, we show that a simple keypoint prediction scheme using hypercolumns achieves a **3.3** point gain in the APK metric [7] over prior approaches working with only the top layer features [3]. While there isn’t much prior work on labeling parts of objects, we show that the hypercolumn framework is significantly better (by **6.6** points on average) than a strong baseline based on the top layer features.

[1] Yihang Bo and Charless C Fowlkes. Shape-based pedestrian parsing. In *CVPR*, 2011.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[3] Georgia Gkioxari, Bharath Hariharan, Ross Girshick, and Jitendra Malik. R-CNNs for pose estimation and action detection. 2014.

[4] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[5] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *The Journal of physiology*, 160 (1), 1962.

[6] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 1989.

[7] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 35(12), 2013.

This is an extended abstract. The full paper is available at the Computer Vision Foundation webpage.