Rich feature hierarchies for accurate object detection and semantic segmentation

Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik
UC Berkeley
{rbg,jdonahue,trevor,malik}@eecs.berkeley.edu

Abstract

Object detection performance, as measured on the canonical PASCAL VOC dataset, has plateaued in the last few years. The best-performing methods are complex ensemble systems that typically combine multiple low-level image features with high-level context. In this paper, we propose a simple and scalable detection algorithm that improves mean average precision (mAP) by more than 30% relative to the previous best result on VOC 2012—achieving a mAP of 53.3%. Our approach combines two key insights: (1) one can apply high-capacity convolutional neural networks (CNNs) to bottom-up region proposals in order to localize and segment objects and (2) when labeled training data is scarce, supervised pre-training for an auxiliary task, followed by domain-specific fine-tuning, yields a significant performance boost. Since we combine region proposals with CNNs, we call our method R-CNN: *Regions with CNN features. Source code for the complete system is available at* http://www.cs.berkeley.edu/~rbg/rcnn.

1. Introduction

Features matter. The last decade of progress on various visual recognition tasks has been based considerably on the use of SIFT [27] and HOG [7]. But if we look at performance on the canonical visual recognition task, PASCAL VOC object detection [13], it is generally acknowledged that progress has been slow during 2010-2012, with small gains obtained by building ensemble systems and employing minor variants of successful methods.

SIFT and HOG are blockwise orientation histograms, a representation we could associate roughly with complex cells in V1, the first cortical area in the primate visual pathway. But we also know that recognition occurs several stages downstream, which suggests that there might be hierarchical, multi-stage processes for computing features that are even more informative for visual recognition.

Fukushima's "neocognitron" [17], a biologically-inspired hierarchical and shift-invariant model for pattern recognition, was an early attempt at just such a process. The neocognitron, however, lacked a supervised training algorithm. Building on Rumelhart *et al*. [30], LeCun *et al*. [24] showed that stochastic gradient descent via backpropagation was effective for training convolutional neural networks (CNNs), a class of models that extend the neocognitron.

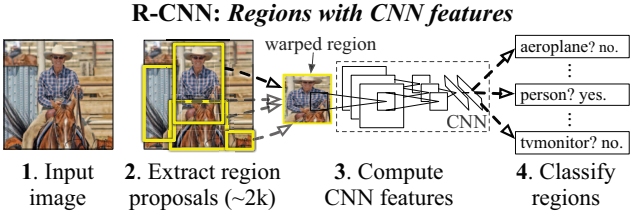

Figure 1: Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large convolutional neural network (CNN), and then (4) classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of **53.7% on PASCAL VOC 2010**. For comparison, [34] reports 35.1% mAP using the same region proposals, but with a spatial pyramid and bag-of-visual-words approach. The popular deformable part models perform at 33.4%.

CNNs saw heavy use in the 1990s (*e.g*., [25]), but then fell out of fashion with the rise of support vector machines. In 2012, Krizhevsky *et al*. [23] rekindled interest in CNNs by showing substantially higher image classification accuracy on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [9, 10]. Their success resulted from training a large CNN on 1.2 million labeled images, together with a few twists on LeCun's CNN (*e.g*., $\max(x, 0)$ rectifying non-linearities and "dropout" regularization).

The significance of the ImageNet result was vigorously debated during the ILSVRC 2012 workshop. The central issue can be distilled to the following: To what extent do the CNN classification results on ImageNet generalize to object detection results on the PASCAL VOC Challenge?

We answer this question by bridging the gap between image classification and object detection. This paper is the first to show that a CNN can lead to dramatically higher object detection performance on PASCAL VOC as compared to systems based on simpler HOG-like features. To achieve this result, we focused on two problems: localizing objects

1063-6919/14 $31.00 © 2014 IEEE
DOI 10.1109/CVPR.2014.81

with a deep network and training a high-capacity model with only a small quantity of annotated detection data.

Unlike image classification, detection requires localizing (likely many) objects within an image. One approach frames localization as a regression problem. However, work from Szegedy *et al*. [33], concurrent with our own, indicates that this strategy may not fare well in practice (they report a mAP of 30.5% on VOC 2007 compared to the 58.5% achieved by our method). An alternative is to build a sliding-window detector. CNNs have been used in this way for at least two decades, typically on constrained object categories, such as faces [29, 35] and pedestrians [31]. In order to maintain high spatial resolution, these CNNs typically only have two convolutional and pooling layers. We also considered adopting a sliding-window approach. However, units high up in our network, which has five convolutional layers, have very large receptive fields (195×195 pixels) and strides (32×32 pixels) in the input image, which makes precise localization within the sliding-window paradigm an open technical challenge.

Instead, we solve the CNN localization problem by operating within the "recognition using regions" paradigm [19], which has been successful for both object detection [34] and semantic segmentation [5]. At test time, our method generates around 2000 category-independent region proposals for the input image, extracts a fixed-length feature vector from each proposal using a CNN, and then classifies each region with category-specific linear SVMs. We use a simple technique (affine image warping) to compute a fixed-size CNN input from each region proposal, regardless of the region's shape. Figure 1 presents an overview of our method and highlights some of our results. Since our system combines region proposals with CNNs, we dub the method R-CNN: Regions with CNN features.

A second challenge faced in detection is that labeled data is scarce and the amount currently available is insufficient for training a large CNN. The conventional solution to this problem is to use *unsupervised* pre-training, followed by supervised fine-tuning (*e.g*., [31]). The second principle contribution of this paper is to show that *supervised* pre-training on a large auxiliary dataset (ILSVRC), followed by domain-specific fine-tuning on a small dataset (PASCAL), is an effective paradigm for learning high-capacity CNNs when data is scarce. In our experiments, fine-tuning for detection improves mAP performance by 8 percentage points. After fine-tuning, our system achieves a mAP of 54% on VOC 2010 compared to 33% for the highly-tuned, HOG-based deformable part model (DPM) [15, 18]. We also point readers to contemporaneous work by Donahue *et al*. [11], who show that Krizhevsky's CNN can be used (without fine-tuning) as a blackbox feature extractor, yielding excellent performance on several recognition tasks including scene classification, fine-grained sub-categorization, and domain adaptation.

Our system is also quite efficient. The only class-specific computations are a reasonably small matrix-vector product and greedy non-maximum suppression. This computational property follows from features that are shared across all categories and that are also two orders of magnitude lower-dimensional than previously used region features (*cf.* [34]).

Understanding the failure modes of our approach is also critical for improving it, and so we report results from the detection analysis tool of Hoiem *et al*. [21]. As an immediate consequence of this analysis, we demonstrate that a simple bounding box regression method significantly reduces mislocalizations, which are the dominant error mode.

Before developing technical details, we note that because R-CNN operates on regions it is natural to extend it to the task of semantic segmentation. With minor modifications, we also achieve competitive results on the PASCAL VOC segmentation task, with an average segmentation accuracy of 47.9% on the VOC 2011 test set.

2. Object detection with R-CNN

Our object detection system consists of three modules. The first generates category-independent region proposals. These proposals define the set of candidate detections available to our detector. The second module is a large convolutional neural network that extracts a fixed-length feature vector from each region. The third module is a set of class-specific linear SVMs. In this section, we present our design decisions for each module, describe their test-time usage, detail how their parameters are learned, and show results on PASCAL VOC 2010-12.

2.1. Module design

Region proposals. A variety of recent papers offer methods for generating category-independent region proposals. Examples include: objectness [1], selective search [34], category-independent object proposals [12], constrained parametric min-cuts (CPMC) [5], multi-scale combinatorial grouping [3], and Cireşan *et al*. [6], who detect mitotic cells by applying a CNN to regularly-spaced square crops, which are a special case of region proposals. While R-CNN is agnostic to the particular region proposal method, we use selective search to enable a controlled comparison with prior detection work (*e.g*., [34, 36]).

Feature extraction. We extract a 4096-dimensional feature vector from each region proposal using the Caffe [22] implementation of the CNN described by Krizhevsky *et al*. [23]. Features are computed by forward propagating a mean-subtracted 227×227 RGB image through five convolutional layers and two fully connected layers. We refer readers to [22, 23] for more network architecture details.

Figure 2: Warped training samples from VOC 2007 train.

In order to compute features for a region proposal, we must first convert the image data in that region into a form that is compatible with the CNN (its architecture requires inputs of a fixed 227×227 pixel size). Of the many possible transformations of our arbitrary-shaped regions, we opt for the simplest. Regardless of the size or aspect ratio of the candidate region, we warp all pixels in a tight bounding box around it to the required size. Prior to warping, we dilate the tight bounding box so that at the warped size there are exactly p pixels of warped image context around the original box (we use $p = 16$). Figure 2 shows a random sampling of warped training regions. The supplementary material discusses alternatives to warping.

2.2. Test-time detection

At test time, we run selective search on the test image to extract around 2000 region proposals (we use selective search's "fast mode" in all experiments). We warp each proposal and forward propagate it through the CNN in order to read off features from the desired layer. Then, for each class, we score each extracted feature vector using the SVM trained for that class. Given all scored regions in an image, we apply a greedy non-maximum suppression (for each class independently) that rejects a region if it has an intersection-over-union (IoU) overlap with a higher scoring selected region larger than a learned threshold.

Run-time analysis. Two properties make detection efficient. First, all CNN parameters are shared across all categories. Second, the feature vectors computed by the CNN are low-dimensional when compared to other common approaches, such as spatial pyramids with bag-of-visual-word encodings. The features used in the UVA detection system [34], for example, are two orders of magnitude larger than ours (360k *vs*. 4k-dimensional).

The result of such sharing is that the time spent computing region proposals and features (13s/image on a GPU or 53s/image on a CPU) is amortized over all classes. The only class-specific computations are dot products between features and SVM weights and non-maximum suppression. In practice, all dot products for an image are batched into a single matrix-matrix product. The feature matrix is typically 2000×4096 and the SVM weight matrix is $4096 \times N$, where N is the number of classes.

This analysis shows that R-CNN can scale to thousands of object classes without resorting to approximate techniques, such as hashing. Even if there were 100k classes, the resulting matrix multiplication takes only 10 seconds on a modern multi-core CPU. This efficiency is not merely the result of using region proposals and shared features. The UVA system, due to its high-dimensional features, would be two orders of magnitude slower while requiring 134GB of memory just to store 100k linear predictors, compared to just 1.5GB for our lower-dimensional features.

It is also interesting to contrast R-CNN with the recent work from Dean *et al*. on scalable detection using DPMs and hashing [8]. They report a mAP of around 16% on VOC 2007 at a run-time of 5 minutes per image when introducing 10k distractor classes. With our approach, 10k detectors can run in about a minute on a CPU, and because no approximations are made mAP would remain at 59% (Section 3.2).

2.3. Training

Supervised pre-training. We discriminatively pre-trained the CNN on a large auxiliary dataset (ILSVRC 2012) with *image-level annotations* (*i.e*., no bounding box labels). Pre-training was performed using the open source Caffe CNN library [22]. In brief, our CNN nearly matches the performance of Krizhevsky *et al*. [23], obtaining a top-1 error rate 2.2 percentage points higher on the ILSVRC 2012 validation set. This discrepancy is due to simplifications in the training process.

Domain-specific fine-tuning. To adapt our CNN to the new task (detection) and the new domain (warped VOC windows), we continue stochastic gradient descent (SGD) training of the CNN parameters using only warped region proposals from VOC. Aside from replacing the CNN's ImageNet-specific 1000-way classification layer with a randomly initialized 21-way classification layer (for the 20 VOC classes plus background), the CNN architecture is unchanged. We treat all region proposals with ≥ 0.5 IoU overlap with a ground-truth box as positives for that box's class and the rest as negatives. We start SGD at a learning rate of 0.001 (1/10th of the initial pre-training rate), which allows fine-tuning to make progress while not clobbering the initialization. In each SGD iteration, we uniformly sample 32 positive windows (over all classes) and 96 background windows to construct a mini-batch of size 128. We bias the sampling towards positive windows because they are extremely rare compared to background.

Object category classifiers. Consider training a binary classifier to detect cars. It's clear that an image region tightly enclosing a car should be a positive example. Similarly, it's clear that a background region, which has nothing to do with cars, should be a negative example. Less clear is how to label a region that partially overlaps a car. We resolve this issue with an IoU overlap threshold, below which regions are defined as negatives. The overlap threshold, 0.3,

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [18]†	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [34]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [36]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [16]†	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	**71.8**	**65.8**	**53.0**	**36.8**	**35.9**	**59.7**	**60.0**	**69.9**	**27.9**	**50.6**	**41.4**	**70.0**	**62.0**	**69.0**	**58.1**	**29.5**	**59.4**	**39.3**	**61.2**	**52.4**	**53.7**

Table 1: Detection average precision (%) on VOC 2010 test. R-CNN is most directly comparable to UVA and Regionlets since all methods use selective search region proposals. Bounding box regression (BB) is described in Section 3.4. At publication time, SegDPM was the top-performer on the PASCAL VOC leaderboard. †DPM and SegDPM use context rescoring not used by the other methods.

was selected by a grid search over $\{0, 0.1, \ldots, 0.5\}$ on a validation set. We found that selecting this threshold carefully is important. Setting it to 0.5, as in [34], decreased mAP by 5 points. Similarly, setting it to 0 decreased mAP by 4 points. Positive examples are defined simply to be the ground-truth bounding boxes for each class.

Once features are extracted and training labels are applied, we optimize one linear SVM per class. Since the training data is too large to fit in memory, we adopt the standard hard negative mining method [15, 32]. Hard negative mining converges quickly and in practice mAP stops increasing after only a single pass over all images.

In supplementary material we discuss why the positive and negative examples are defined differently in fine-tuning versus SVM training. We also discuss why it's necessary to train detection classifiers rather than simply use outputs from the final layer (fc_8) of the fine-tuned CNN.

2.4. Results on PASCAL VOC 2010-12

Following the PASCAL VOC best practices [13], we validated all design decisions and hyperparameters on the VOC 2007 dataset (Section 3.2). For final results on the VOC 2010-12 datasets, we fine-tuned the CNN on VOC 2012 train and optimized our detection SVMs on VOC 2012 trainval. We submitted test results to the evaluation server only once for each of the two major algorithm variants (with and without bounding box regression).

Table 1 shows complete results on VOC 2010. We compare our method against four strong baselines, including SegDPM [16], which combines DPM detectors with the output of a semantic segmentation system [4] and uses additional inter-detector context and image-classifier rescoring. The most germane comparison is to the UVA system from Uijlings *et al*. [34], since our systems use the same region proposal algorithm. To classify regions, their method builds a four-level spatial pyramid and populates it with densely sampled SIFT, Extended OpponentSIFT, and RGB-SIFT descriptors, each vector quantized with 4000-word codebooks. Classification is performed with a histogram intersection kernel SVM. Compared to their multi-feature, non-linear kernel SVM approach, we achieve a large improvement in mAP, from 35.1% to 53.7% mAP, while also being much faster (Section 2.2). Our method achieves similar performance (53.3% mAP) on VOC 2011/12 test.

3. Visualization, ablation, and modes of error

3.1. Visualizing learned features

First-layer filters can be visualized directly and are easy to understand [23]. They capture oriented edges and opponent colors. Understanding the subsequent layers is more challenging. Zeiler and Fergus present a visually attractive deconvolutional approach in [37]. We propose a simple (and complementary) non-parametric method that directly shows what the network learned.

The idea is to single out a particular unit (feature) in the network and use it as if it were an object detector in its own right. That is, we compute the unit's activations on a large set of held-out region proposals (about 10 million), sort the proposals from highest to lowest activation, perform non-maximum suppression, and then display the top-scoring regions. Our method lets the selected unit "speak for itself" by showing exactly which inputs it fires on. We avoid averaging in order to see different visual modes and gain insight into the invariances computed by the unit.

We visualize units from layer pool_5, which is the max-pooled output of the network's fifth and final convolutional layer. The pool_5 feature map is $6 \times 6 \times 256 = 9216$-dimensional. Ignoring boundary effects, each pool_5 unit has a receptive field of 195×195 pixels in the original 227×227 pixel input. A central pool_5 unit has a nearly global view, while one near the edge has a smaller, clipped support.

Each row in Figure 3 displays the top 16 activations for a pool_5 unit from a CNN that we fine-tuned on VOC 2007 trainval. Six of the 256 functionally unique units are visualized (the supplementary material includes more). These units were selected to show a representative sample of what the network learns. In the second row, we see a unit that fires on dog faces and dot arrays. The unit corresponding to the third row is a red blob detector. There are also detectors for human faces and more abstract patterns such as text and triangular structures with windows. The network appears to learn a representation that combines a small number of class-tuned features together with a distributed representa-

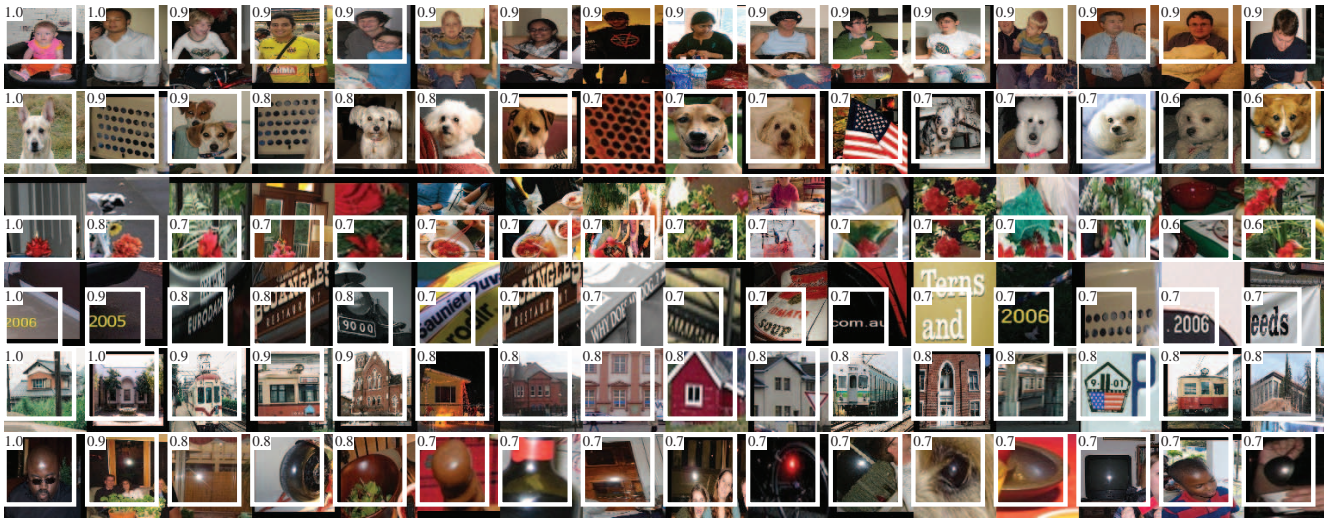

Figure 3: Top regions for six pool$_5$ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN pool$_5$	51.8	60.2	36.4	27.8	23.2	52.8	60.6	49.2	18.3	47.8	44.3	40.8	56.6	58.7	42.4	23.4	46.1	36.7	51.3	55.7	44.2
R-CNN fc$_6$	59.3	61.8	43.1	34.0	25.1	53.1	60.6	52.8	21.7	47.8	42.7	47.8	52.5	58.5	44.6	25.6	48.3	34.0	53.1	58.0	46.2
R-CNN fc$_7$	57.6	57.9	38.5	31.8	23.7	51.2	58.9	51.4	20.0	50.5	40.9	46.0	51.6	55.9	43.3	23.3	48.1	35.3	51.0	57.4	44.7
R-CNN FT pool$_5$	58.2	63.3	37.9	27.6	26.1	54.1	66.9	51.4	26.7	55.5	43.4	43.1	57.7	59.0	45.8	28.1	50.8	40.6	53.1	56.4	47.3
R-CNN FT fc$_6$	63.5	66.0	47.9	37.7	29.9	62.5	70.2	60.2	32.0	57.9	47.0	53.5	60.1	64.2	52.2	31.3	55.0	50.0	57.7	63.0	53.1
R-CNN FT fc$_7$	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN FT fc$_7$ BB	**68.1**	**72.8**	**56.8**	**43.0**	**36.8**	**66.3**	**74.2**	**67.6**	**34.4**	**63.5**	**54.5**	**61.2**	**69.1**	**68.6**	**58.7**	**33.4**	**62.9**	**51.1**	**62.5**	**64.8**	**58.5**
DPM v5 [18]	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	33.7
DPM ST [26]	23.8	58.2	10.5	8.5	27.1	50.4	52.0	7.3	19.2	22.8	18.1	8.0	55.9	44.8	32.4	13.3	15.9	22.8	46.2	44.9	29.1
DPM HSC [28]	32.2	58.3	11.5	16.3	30.6	49.9	54.8	23.5	21.5	27.7	34.0	13.7	58.1	51.6	39.9	12.4	23.5	34.4	47.4	45.2	34.3

Table 2: Detection average precision (%) on VOC 2007 test. Rows 1-3 show R-CNN performance without fine-tuning. Rows 4-6 show results for the CNN pre-trained on ILSVRC 2012 and then fine-tuned (FT) on VOC 2007 trainval. Row 7 includes a simple bounding box regression (BB) stage that reduces localization errors (Section 3.4). Rows 8-10 present DPM methods as a strong baseline. The first uses only HOG, while the next two use different feature learning approaches to augment or replace HOG.

tion of shape, texture, color, and material properties. The subsequent fully connected layer fc$_6$ has the ability to model a large set of compositions of these rich features.

3.2. Ablation studies

Performance layer-by-layer, without fine-tuning. To understand which layers are critical for detection performance, we analyzed results on the VOC 2007 dataset for each of the CNN's last three layers. Layer pool$_5$ was briefly described in Section 3.1. The final two layers are summarized below.

Layer fc$_6$ is fully connected to pool$_5$. To compute features, it multiplies a 4096×9216 weight matrix by the pool$_5$ feature map (reshaped as a 9216-dimensional vector) and then adds a vector of biases. This intermediate vector is component-wise half-wave rectified ($x \leftarrow \max(0, x)$).

Layer fc$_7$ is the final layer of the network. It is implemented by multiplying the features computed by fc$_6$ by a 4096×4096 weight matrix, and similarly adding a vector of biases and applying half-wave rectification.

We start by looking at results from the CNN *without fine-tuning* on PASCAL, *i.e.* all CNN parameters were pre-trained on ILSVRC 2012 only. Analyzing performance layer-by-layer (Table 2 rows 1-3) reveals that features from fc$_7$ generalize worse than features from fc$_6$. This means that 29%, or about 16.8 million, of the CNN's parameters can be removed without degrading mAP. More surprising is that removing *both* fc$_7$ and fc$_6$ produces quite good results even though pool$_5$ features are computed using *only 6%* of the CNN's parameters. Much of the CNN's representational power comes from its convolutional layers, rather than from the much larger densely connected layers. This finding suggests potential utility in computing a dense feature map, in the sense of HOG, of an arbitrary-sized image by using only the convolutional layers of the CNN. This representation would enable experimentation with sliding-window detectors, including DPM, on top of pool$_5$ features.

Performance layer-by-layer, with fine-tuning. We now look at results from our CNN after having fine-tuned its parameters on VOC 2007 trainval. The improvement is striking (Table 2 rows 4-6): fine-tuning increases mAP by 8.0 percentage points to 54.2%. The boost from fine-tuning is much larger for fc_6 and fc_7 than for $pool_5$, which suggests that the $pool_5$ features learned from ImageNet are general and that most of the improvement is gained from learning domain-specific non-linear classifiers on top of them.

Comparison to recent feature learning methods. Relatively few feature learning methods have been tried on PASCAL VOC detection. We look at two recent approaches that build on deformable part models. For reference, we also include results for the standard HOG-based DPM [18].

The first DPM feature learning method, DPM ST [26], augments HOG features with histograms of "sketch token" probabilities. Intuitively, a sketch token is a tight distribution of contours passing through the center of an image patch. Sketch token probabilities are computed at each pixel by a random forest that was trained to classify 35×35 pixel patches into one of 150 sketch tokens or background.

The second method, DPM HSC [28], replaces HOG with histograms of sparse codes (HSC). To compute an HSC, sparse code activations are solved for at each pixel using a learned dictionary of 100 7×7 pixel (grayscale) atoms. The resulting activations are rectified in three ways (full and both half-waves), spatially pooled, unit ℓ_2 normalized, and then power transformed ($x \leftarrow \text{sign}(x)|x|^\alpha$).

All R-CNN variants strongly outperform the three DPM baselines (Table 2 rows 8-10), including the two that use feature learning. Compared to the latest version of DPM, which uses only HOG features, our mAP is more than 20 percentage points higher: 54.2% *vs*. 33.7%—*a 61% relative improvement*. The combination of HOG and sketch tokens yields 2.5 mAP points over HOG alone, while HSC improves over HOG by 4 mAP points (when compared internally to their private DPM baselines—both use non-public implementations of DPM that underperform the open source version [18]). These methods achieve mAPs of 29.1% and 34.3%, respectively.

3.3. Detection error analysis

We applied the excellent detection analysis tool from Hoiem *et al*. [21] in order to reveal our method's error modes, understand how fine-tuning changes them, and to see how our error types compare with DPM. A full summary of the analysis tool is beyond the scope of this paper and we encourage readers to consult [21] to understand some finer details (such as "normalized AP"). Since the analysis is best absorbed in the context of the associated plots, we present the discussion within the captions of Figure 4 and Figure 5.

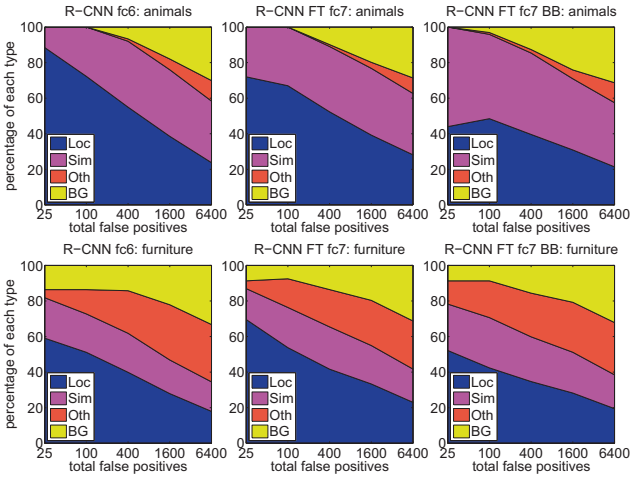

Figure 4: Distribution of top-ranked false positive (FP) types. Each plot shows the evolving distribution of FP types as more FPs are considered in order of decreasing score. Each FP is categorized into 1 of 4 types: Loc—poor localization (a detection with an IoU overlap with the correct class between 0.1 and 0.5, or a duplicate); Sim—confusion with a similar category; Oth—confusion with a dissimilar object category; BG—a FP that fired on background. Compared with DPM (see [21]), significantly more of our errors result from poor localization, rather than confusion with background or other object classes, indicating that the CNN features are much more discriminative than HOG. Loose localization likely results from our use of bottom-up region proposals and the positional invariance learned from pre-training the CNN for whole-image classification. Column three shows how our simple bounding box regression method fixes many localization errors.

3.4. Bounding box regression

Based on the error analysis, we implemented a simple method to reduce localization errors. Inspired by the bounding box regression employed in DPM [15], we train a linear regression model to predict a new detection window given the $pool_5$ features for a selective search region proposal. Full details are given in the supplementary material. Results in Table 1, Table 2, and Figure 4 show that this simple approach fixes a large number of mislocalized detections, boosting mAP by 3 to 4 points.

4. Semantic segmentation

Region classification is a standard technique for semantic segmentation, allowing us to easily apply R-CNN to the PASCAL VOC segmentation challenge. To facilitate a direct comparison with the current leading semantic segmentation system (called O_2P for "second-order pooling") [4], we work within their open source framework. O_2P uses CPMC to generate 150 region proposals per image and then predicts the quality of each region, for each class, using support vector regression (SVR). The high performance of their approach is due to the quality of the CPMC regions

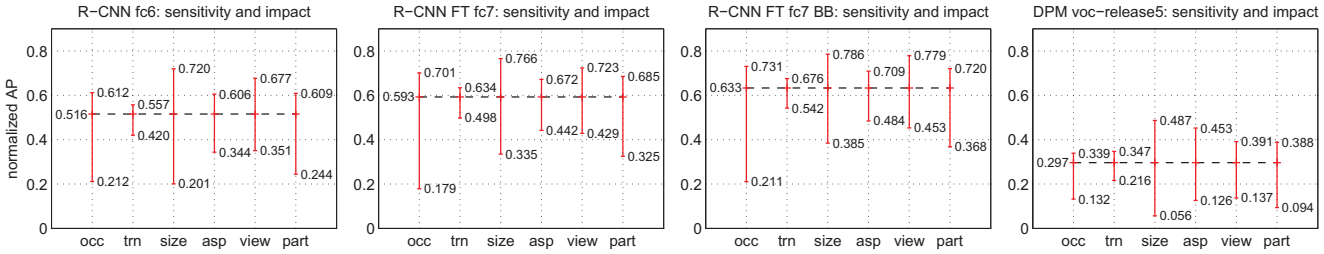

Figure 5: Sensitivity to object characteristics. Each plot shows the mean (over classes) normalized AP (see [21]) for the highest and lowest performing subsets within six different object characteristics (occlusion, truncation, bounding box area, aspect ratio, viewpoint, part visibility). We show plots for our method (R-CNN) with and without fine-tuning (FT) and bounding box regression (BB) as well as for DPM voc-release5. Overall, fine-tuning does not reduce sensitivity (the difference between max and min), but does substantially improve both the highest and lowest performing subsets for nearly all characteristics. This indicates that fine-tuning does more than simply improve the lowest performing subsets for aspect ratio and bounding box area, as one might conjecture based on how we warp network inputs. Instead, fine-tuning improves robustness for all characteristics including occlusion, truncation, viewpoint, and part visibility.

and the powerful second-order pooling of multiple feature types (enriched variants of SIFT and LBP). We also note that Farabet *et al*. [14] recently demonstrated good results on several dense scene labeling datasets (not including PASCAL) using a CNN as a multi-scale per-pixel classifier.

We follow [2, 4] and extend the PASCAL segmentation training set to include the extra annotations made available by Hariharan *et al*. [20]. Design decisions and hyperparameters were cross-validated on the VOC 2011 validation set. Final test results were evaluated only once.

CNN features for segmentation. We evaluate three strategies for computing features on CPMC regions, all of which begin by warping the rectangular window around the region to 227×227. The first strategy (*full*) ignores the region's shape and computes CNN features directly on the warped window, exactly as we did for detection. However, these features ignore the non-rectangular shape of the region. Two regions might have very similar bounding boxes while having very little overlap. Therefore, the second strategy (*fg*) computes CNN features only on a region's foreground mask. We replace the background with the mean input so that background regions are zero after mean subtraction. The third strategy (*full*+*fg*) simply concatenates the *full* and *fg* features; our experiments validate their complementarity.

	full R-CNN		*fg* R-CNN		*full*+*fg* R-CNN	
O_2P [4]	fc_6	fc_7	fc_6	fc_7	fc_6	fc_7
46.4	43.0	42.5	43.7	42.1	**47.9**	45.8

Table 3: Segmentation mean accuracy (%) on VOC 2011 validation. Column 1 presents O_2P; 2-7 use our CNN pre-trained on ILSVRC 2012.

Results on VOC 2011. Table 3 shows a summary of our results on the VOC 2011 validation set compared with O_2P. (See supplementary material for complete per-category results.) Within each feature computation strategy, layer fc_6 always outperforms fc_7 and the following discussion refers to the fc_6 features. The *fg* strategy slightly outperforms *full*, indicating that the masked region shape provides a stronger signal, matching our intuition. However, *full*+*fg* achieves an average accuracy of 47.9%, our best result by a margin of 4.2% (also modestly outperforming O_2P), indicating that the context provided by the *full* features is highly informative even given the *fg* features. Notably, training the 20 SVRs on our *full*+*fg* features takes an hour on a single core, compared to 10+ hours for training on O_2P features.

In Table 4 we present results on the VOC 2011 test set, comparing our best-performing method, fc_6 (*full*+*fg*), against two strong baselines. Our method achieves the highest segmentation accuracy for 11 out of 21 categories, and the highest overall segmentation accuracy of 47.9%, averaged across categories (but likely ties with the O_2P result under any reasonable margin of error). Still better performance could likely be achieved by fine-tuning.

5. Conclusion

In recent years, object detection performance had stagnated. The best performing systems were complex ensembles combining multiple low-level image features with high-level context from object detectors and scene classifiers. This paper presents a simple and scalable object detection algorithm that gives a 30% relative improvement over the best previous results on PASCAL VOC 2012.

We achieved this performance through two insights. The first is to apply high-capacity convolutional neural networks to bottom-up region proposals in order to localize and segment objects. The second is a paradigm for training large CNNs when labeled training data is scarce. We show that it is highly effective to pre-train the network—*with supervision*—for a auxiliary task with abundant data (image classification) and then to fine-tune the network for the target task where data is scarce (detection). We conjecture that the "supervised pre-training/domain-specific fine-tuning" paradigm will be highly effective for a variety of

VOC 2011 test	bg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
R&P [2]	83.4	46.8	18.9	36.6	31.2	42.7	57.3	47.4	44.1	8.1	39.4	**36.1**	36.3	49.5	48.3	50.7	26.3	47.2	22.1	42.0	43.2	40.8
O_2P [4]	**85.4**	**69.7**	22.3	45.2	**44.4**	46.9	66.7	57.8	56.2	**13.5**	**46.1**	32.3	41.2	**59.1**	55.3	51.0	**36.2**	50.4	**27.8**	46.9	**44.6**	47.6
ours (*full+fg* R-CNN fc_6)	84.2	66.9	**23.7**	**58.3**	37.4	**55.4**	**73.3**	**58.7**	**56.5**	9.7	45.5	29.5	**49.3**	40.1	**57.8**	**53.9**	33.8	**60.7**	22.7	**47.1**	41.3	**47.9**

Table 4: Segmentation accuracy (%) on VOC 2011 test. We compare against two strong baselines: the "Regions and Parts" (R&P) method of [2] and the second-order pooling (O_2P) method of [4]. Without any fine-tuning, our CNN achieves top segmentation performance, outperforming R&P and roughly matching O_2P.

data-scarce vision problems.

We conclude by noting that it is significant that we achieved these results by using a combination of classical tools from computer vision *and* deep learning (bottom-up region proposals and convolutional neural networks). Rather than opposing lines of scientific inquiry, the two are natural and inevitable partners.

Acknowledgments. This research was supported in part by DARPA Mind's Eye and MSEE programs, by NSF awards IIS-0905647, IIS-1134072, and IIS-1212798, MURI N000014-10-1-0933, and by support from Toyota. The GPUs used in this research were generously donated by the NVIDIA Corporation.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *TPAMI*, 2012.

[2] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik. Semantic segmentation using regions and parts. In *CVPR*, 2012.

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

[4] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*, 2012.

[5] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *TPAMI*, 2012.

[6] D. Cireşan, A. Giusti, L. Gambardella, and J. Schmidhuber. Mitosis detection in breast cancer histology images with deep neural networks. In *MICCAI*, 2013.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[8] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013.

[9] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Competition 2012 (ILSVRC2012). http://www.image-net.org/challenges/LSVRC/2012/.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[11] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *ICML*, 2014.

[12] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 2010.

[14] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *TPAMI*, 2013.

[15] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 2010.

[16] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun. Bottom-up segmentation for top-down detection. In *CVPR*, 2013.

[17] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[18] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://www.cs.berkeley.edu/~rbg/latent-v5/.

[19] C. Gu, J. J. Lim, P. Arbeláez, and J. Malik. Recognition using regions. In *CVPR*, 2009.

[20] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

[21] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*. 2012.

[22] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/, 2013.

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

[24] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comp.*, 1989.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 1998.

[26] J. J. Lim, C. L. Zitnick, and P. Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *CVPR*, 2013.

[27] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[28] X. Ren and D. Ramanan. Histograms of sparse codes for object detection. In *CVPR*, 2013.

[29] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *TPAMI*, 1998.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Parallel Distributed Processing*, 1:318–362, 1986.

[31] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *CVPR*, 2013.

[32] K. Sung and T. Poggio. Example-based learning for view-based human face detection. Technical Report A.I. Memo No. 1521, Massachussets Institute of Technology, 1994.

[33] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *NIPS*, 2013.

[34] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *IJCV*, 2013.

[35] R. Vaillant, C. Monrocq, and Y. LeCun. Original approach for the localisation of objects in images. *IEE Proc on Vision, Image, and Signal Processing*, 1994.

[36] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013.

[37] M. Zeiler, G. Taylor, and R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *CVPR*, 2011.