Region-Based Convolutional Networks for Accurate Object Detection and Segmentation

Ross Girshick, Jeff Donahue, *Student Member, IEEE*,
Trevor Darrell, *Member, IEEE*, and Jitendra Malik, *Fellow, IEEE*

Abstract—Object detection performance, as measured on the canonical PASCAL VOC Challenge datasets, plateaued in the final years of the competition. The best-performing methods were complex ensemble systems that typically combined 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 50 percent relative to the previous best result on VOC 2012—achieving a mAP of 62.4 percent. Our approach combines two ideas: (1) one can apply high-capacity convolutional networks (CNNs) to bottom-up region proposals in order to localize and segment objects and (2) when labeled training data are scarce, supervised pre-training for an auxiliary task, followed by domain-specific fine-tuning, boosts performance significantly. Since we combine region proposals with CNNs, we call the resulting model an *R-CNN* or *Region-based Convolutional Network*. Source code for the complete system is available at http://www.cs.berkeley.edu/~rbg/rcnn.

Index Terms—Object recognition, detection, semantic segmentation, convolutional networks, deep learning, transfer learning

1 Introduction

Recognizing objects and localizing them in images is one of the most fundamental and challenging problems in computer vision. There has been significant progress on this problem over the last decade due largely to the use of low-level image features, such as SIFT [1] and HOG [2], in sophisticated machine learning frameworks. But if we look at performance on the canonical visual recognition task, PASCAL VOC object detection [3], it is generally acknowledged that progress slowed from 2010 onward, with small gains obtained by building ensemble systems and employing minor variants of successful methods.

SIFT and HOG are semi-local 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.

In this paper, we describe an object detection and segmentation system that uses multi-layer convolutional networks to compute highly discriminative, yet invariant, features. We use these features to classify image regions, which can then be output as detected bounding boxes or pixel-level segmentation masks. On the PASCAL detection benchmark, our system achieves a relative improvement of more than 50 percent mean average precision (mAP) compared to the best methods based on low-level image features. Our approach also scales well with the number of object categories, which is a long-standing challenge for existing methods.

We trace the roots of our approach to Fukushima's "neocognitron" [4], a hierarchical and shift-invariant model for pattern recognition. While the basic architecture of the neocognitron is used widely today, Fukushima's method had limited empirical success in part because it lacked a supervised training algorithm. Rumelhart et al. [5] showed that a similar architecture could be trained with supervised error backpropagation to classify synthetic renderings of the characters 'T' and 'C'. Building on this work, LeCun et al. demonstrated in an influential sequence of papers (from [6] to [7]) that stochastic gradient descent (SGD) via backpropagation was effective for training deeper networks for challenging real-world handwritten character recognition problems. These models are now known as convolutional (neural) networks, CNNs, or ConvNets.

CNNs saw heavy use in the 1990s, but then fell out of fashion with the rise of support vector machines. In 2012, Krizhevsky et al. [8] rekindled interest in CNNs by showing a substantial improvement in 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 CNNs from the 1990s (e.g., $\max(x, 0)$ "ReLU" non-linearities, "dropout" regularization, and a fast GPU implementation).

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?

- *R. Girshick is with Microsoft Research, Redmond, WA. E-mail: rbg@eecs.berkeley.edu.*
- *J. Donahue, T. Darrell, and J. Malik are with the Department of Electrical Engineering and Computer Science, UC Berkeley, Berkeley, CA. E-mail: {jdonahue, malik}@eecs.berkeley.edu.*

Manuscript received 25 Nov. 2014; revised 5 May 2015; accepted 18 May 2015. Date of publication 24 May 2015; date of current version 9 Dec. 2015. Recommended for acceptance by A. Vedaldi.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2437384

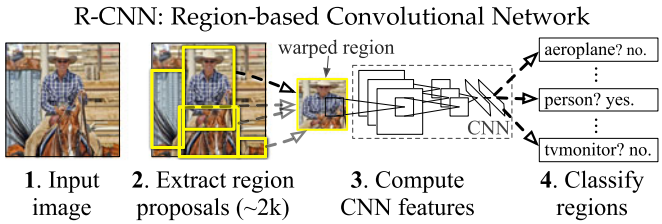

Fig. 1. Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large CNN, and then (4) classifies each region using class-specific linear SVMs. We trained an R-CNN that achieves a mean average precision of 62.9 percent on PASCAL VOC 2010. For comparison, [21] reports 35.1 percent 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 percent. On the 200-class ILSVRC2013 detection dataset, we trained an R-CNN with a mAP of 31.4 percent, a large improvement over OverFeat [19], which had the previous best result at 24.3 percent mAP.

We answered this question in a conference version of this paper [11] by showing 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 bridged the gap between image classification and object detection by developing solutions to two problems: (1) How can we localize objects with a deep network? and (2) How can we train 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 is to frame detection as a regression problem. This formulation can work well for localizing a single object, but detecting multiple objects requires complex workarounds [12] or an ad hoc assumption about the number of objects per image [13]. 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 [14], [15], hands [16], and pedestrians [17]. This approach is attractive in terms of computational efficiency, however its straightforward application requires all objects to share a common aspect ratio. The aspect ratio problem can be addressed with mixture models (e.g., [18]), where each component specializes in a narrow band of aspect ratios, or with bounding-box regression (e.g., [18], [19]).

Instead, we solve the localization problem by operating within the "recognition using regions" paradigm [20], which has been successful for both object detection [21] and semantic segmentation [22]. At test time, our method generates around 2,000 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 warping technique (anisotropic image scaling) to compute a fixed-size CNN input from each region proposal, regardless of the region's shape. Fig. 1 shows an overview of a region-based convolutional network (R-CNN) and highlights some of our results.

A second challenge faced in detection is that labeled data are scarce and the amount currently available is insufficient for training large CNNs from random initializations. The conventional solution to this problem is to use *unsupervised* pre-training, followed by supervised fine-tuning (FT) (e.g., [17]). 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 are scarce. In our experiments, fine-tuning for detection can improve mAP by as much as 8 percentage points. After fine-tuning, our system achieves a mAP of 63 percent on VOC 2010 compared to 33 percent for the highly-tuned, HOG-based deformable part model (DPM) [18], [23].

Our original motivation for using regions was born out of a pragmatic research methodology: move from image classification to object detection as simply as possible. Since then, this design choice has proved valuable because R-CNNs are straightforward to implement and train (compared to sliding-window CNNs) and it provides a unified solution to object detection and segmentation.

This journal paper extends our earlier work [11] in a number of ways. First, we provide more implementation details, rationales for design decisions, and ablation studies. Second, we present new results on PASCAL detection using deeper networks. Our approach is agnostic to the particular choice of network architecture used and we show that recent work on deeper networks (e.g., [24]) translates into large improvements in object detection. Finally, we give a head-to-head comparison of R-CNNs with the recently proposed OverFeat [19] detection system. OverFeat uses a sliding-window CNN for detection and was a top-performing method on the ILSVRC 2013 detection challenge. We train an R-CNN that significantly outperforms OverFeat, with a mAP of 31.4 percent versus 24.3 percent on the 200-class ILSVRC2013 detection dataset.

2 Related Work

Deep CNNs for object detection. There were several efforts [12], [13], [19] to use convolutional networks for PASCAL-style object detection concurrent with the development of R-CNNs. Szegedy et al. [12] model object detection as a regression problem. Given an image window, they use a CNN to predict foreground pixels over a coarse grid for the whole object as well as the object's top, bottom, left and right halves. A grouping process then converts the predicted masks into detected bounding boxes. Szegedy et al. train their model from a random initialization on VOC 2012 trainval and get a mAP of 30.5 percent on VOC 2007 test. In comparison, an R-CNN using the same network architecture gets a mAP of 58.5 percent, but uses supervised ImageNet pre-training. One hypothesis is that [12] performs worse because it does not use ImageNet pre-training. Recent work from Agrawal et al. [25] shows that this is not the case; they find that an R-CNN trained from a random initialization on VOC 2007 trainval (using the same network architecture as [12]) achieves a mAP of 40.7 percent on VOC 2007 test despite using half the amount of training data as [12].

Scalability and speed. In addition to being accurate, it's important for object detection systems to scale well as the number of object categories increases. Significant effort has gone into making methods like DPM [18] scale to thousands of object categories. For example, Dean et al. [26] replace

exact filter convolutions in DPM with hashtable lookups. They show that with this technique it's possible to run 10k DPM detectors in about 5 minutes per image on a desktop workstation. However, there is an unfortunate tradeoff. When a large number of DPM detectors compete, the approximate hashing approach causes a substantial loss in detection accuracy. R-CNNs, in contrast, scale very well with the number of object classes to detect because nearly all computation is shared between all object categories. The only class-specific computations are a reasonably small matrix-vector product and greedy non-maximum suppression. Although these computations scale linearly with the number of categories, the scale factor is small. Measured empirically, it takes only 30 ms longer to detect 200 classes than 20 classes on a CPU, without any approximations. This makes it feasible to rapidly detect tens of thousands of object categories without any modifications to the core algorithm.

Despite this graceful scaling behavior, an R-CNN can take 10 to 45 seconds per image on a GPU, depending on the network used, since each region is passed through the network independently. Recent work from He et al. [27] ("SPPnet") improves R-CNN efficiency by sharing computation through a feature pyramid, allowing for detection at a few frames per second. Building on SPPnet, Girshick [28] shows that it's possible to further reduce training and testing times, while improving detection accuracy and simplifying the training process, using an approach called "Fast R-CNN." Fast R-CNN reduces detection times (excluding region proposal computation) to 50 to 300 ms per image, depending on network architecture.

Localization methods. The dominant approach to object detection has been based on sliding-window detectors. This approach goes back (at least) to early face detectors [15], and continued with HOG-based pedestrian detection [2], and part-based generic object detection [18]. An alternative is to first compute a pool of (likely overlapping) image regions, each one serving as a candidate object, and then to filter these candidates in a way that aims to retain only the true objects. Multiple segmentation hypotheses were used by Hoiem et al. [29] to estimate the rough geometric scene structure and by Russell et al. [30] to automatically discover object classes in a set of images. The "selective search" algorithm of van de Sande et al. [21] popularized the multiple segmentation approach for object detection by showing strong results on PASCAL object detection. Our approach was inspired by the success of selective search.

Object proposal generation is now an active research area. EdgeBoxes [31] outputs high-quality rectangular (box) proposals quickly ($\sim$0.3 s per image). BING [32] generates box proposals at $\sim$3 ms per image, however it has subsequently been shown that the proposal quality is too poor to be useful in R-CNNs [33]. Other methods focus on pixel-wise segmentation, producing regions instead of boxes. These approaches include RIGOR [34] and MCG [35], which take 10 to 30 s per image and GOP [36], a faster methods that takes $\sim$1 s per image. For a more in-depth survey of proposal algorithms, Hosang et al. [33] provide an insightful meta-evaluation of recent methods.

Transfer learning. R-CNN training is based on *inductive transfer learning*, using the taxonomy of Pan and Yang [37]. To train an R-CNN, we typically start with ImageNet classification as a source task and dataset, train a network using supervision, and then transfer that network to the target task and dataset using supervised fine-tuning. This method is related to traditional multi-task learning [38], [39], except that we train for the tasks sequentially and are ultimately only interested in performing well on the target task.

This strategy is different from the dominant paradigm in recent neural network literature of *unsupervised tranfer learning* (see [40] for a survey covering unsupervised pre-training and represetation learning more generally). Supervised transfer learning using CNNs, but without fine-funing, was also investigated in concurrent work by Donahue et al. [41]. They show that Krizhevsky et al.'s CNN, once trained on ImageNet, can be used as a blackbox feature extractor, yielding excellent performance on several recognition tasks including scene classification, fine-grained sub-categorization, and domain adaptation. Hoffman et al. [42] show how transfer learning can be used to train R-CNNs for classes that have image-level labels, but no bounding-box training data. Their approach is based on modeling the task shift from image classification to object detection and then transfering that knowledge to classes that have no detection training data.

R-CNN extensions. Since their introduction, R-CNNs have been extended to a variety of new tasks and datasets. Karpathy et al. [43] learn a model for bi-directional image and sentence retrieval. Their image representation is derived from an R-CNN trained to detect 200 classes on the ILSVRC2013 detection dataset. Gkioxari et al. [44] use multi-task learning to train R-CNNs for person detection, 2D pose estimation, and action recognition. Hariharan et al. [45] propose a unification of the object detection and semantic segmentation tasks, termed "simultaneous detection and segmentation" (SDS), and train a two-column R-CNN for this task. They show that a single region proposal algorithm (MCG [35]) can be used effectively for traditional bounding-box detection as well as semantic segmentation. Their PASCAL segmentation results improve significantly on the ones reported in this paper. Gupta et al. [46] extend R-CNNs to object detection in depth images. They show that a well-designed input signal, where the depth map is augmented with height above ground and local surface orientation with respect to gravity, allows training an R-CNN that outperforms existing RGB-D object detection baselines. Song et al. [47] train an R-CNN using weak, image-level supervision by mining for positive training examples using a submodular cover algorithm and then training a latent SVM.

Many systems based on, or implementing, R-CNNs were used in the recent ILSVRC2014 object detection challenge [48], resulting in substantial improvements in detection accuracy. In particular, the winning method, GoogLeNet [49], [50], uses an innovative network design in an R-CNN. With a single network (and a slightly simpler pipeline that excludes SVM training and bounding-box regression), they improve R-CNN performance to 38.0 percent mAP from a basline of 34.5 percent. They also show that an ensemble of six networks improves their result to 43.9 percent mAP.

3 Object Detection with An 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 convolutional 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 detection results on PASCAL VOC 2010-12 and ILSVRC2013.

3.1 Module Design

3.1.1 Region Proposals

A variety of recent papers offer methods for generating category-independent region proposals. Examples include: objectness [51], selective search [21], category-independent object proposals [52], constrained parametric min-cuts (CPMC) [22], multi-scale combinatorial grouping [35], and Cireşan et al. [53], 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., [21], [54]).

3.1.2 Feature Extraction

We extract a fixed-length feature vector from each region proposal using a CNN. The particular CNN architecture used is a system hyperparameter. Most of our experiments use the Caffe [55] implementation of the CNN described by Krizhevsky et al. [8] (TorontoNet), however we have also experimented with the 16-layer deep network from Simonyan and Zisserman [24] (OxfordNet). In both cases, the feature vectors are 4,096-dimensional. Features are computed by forward propagating a mean-subtracted $S \times S$ RGB image through the network and reading off the values output by the penultimate layer (the layer just before the softmax classifier). For TorontoNet, $S = 227$ and for OxfordNet $S = 224$. We refer readers to [8], [24], [55] for more network architecture details.

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 $S \times S$ pixel size).[1] 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 $p = 16$). Fig. 2 shows a random sampling of warped training regions. Alternatives to warping are discussed in Section 7.1.

1. Of course the entire network can be run convolutionally, which enables handling arbitrary input sizes, however then the output size is no longer a fixed-length vector. The output can be converted to a fixed-length through another transformation, such as in [27].

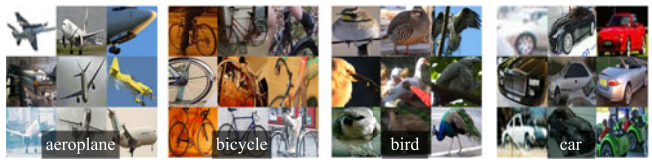

Fig. 2. Warped training samples from VOC 2007 train.

3.2 Test-Time Detection

At test time, we run selective search on the test image to extract around 2,000 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 compute features. 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.

3.2.1 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 [21], for example, are two orders of magnitude larger than ours (360 k versus 4 k-dimensional).

The result of such sharing is that the time spent computing region proposals and features (10 s/image on an NVIDIA Titan Black GPU or 53 s/image on a CPU, using TorontoNet) 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 $2{,}000 \times 4{,}096$ and the SVM weight matrix is $4{,}096 \times N$, where N is the number of classes.

This analysis shows that R-CNNs can scale to thousands of object classes without resorting to approximate techniques, such as hashing. Even if there were 100 k 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 134 GB of memory just to store 100 k linear predictors, compared to just 1.5 GB for our lower-dimensional features.

It is also interesting to contrast R-CNNs with the recent work from Dean et al. on scalable detection using DPMs and hashing [56]. They report a mAP of around 16 percent on VOC 2007 at a run-time of 5 minutes per image when introducing 10 k distractor classes. With our approach, 10 k detectors can run in about a minute on a CPU, and because no approximations are made mAP would remain at 59 percent with TorontoNet and 66 percent with OxfordNet (Section 4.2).

TABLE 1
Detection Average Precision (Percent) on VOC 2010 Test

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 [23]	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 [21]	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 [54]	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 [57]	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 T-Net	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 T-Net 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
R-CNN O-Net	76.5	70.4	58.0	40.2	39.6	61.8	63.7	81.0	36.2	64.5	45.7	80.5	71.9	74.3	60.6	31.5	64.7	52.5	64.6	57.2	59.8
R-CNN O-Net BB	**79.3**	**72.4**	**63.1**	**44.0**	**44.4**	**64.6**	**66.3**	**84.9**	**38.8**	**67.3**	**48.4**	**82.3**	**75.0**	**76.7**	**65.7**	**35.8**	**66.2**	**54.8**	**69.1**	**58.8**	**62.9**

T-Net stands for TorontoNet and O-Net for OxfordNet (Section 3.1.2). R-CNNs are most directly comparable to UVA and Regionlets since all methods use selective search region proposals. Bounding-box regression is described in Section 7.3. 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. SegDPM and all R-CNNs use additional training data.

3.3 Training

3.3.1 Supervised Pre-Training

We discriminatively pre-trained the CNN on a large auxiliary dataset (ILSVRC2012 classification) using *image-level annotations* only (bounding-box labels are not available for this data). Pre-training was performed using the open source Caffe CNN library [55].

3.3.2 Domain-Specific Fine-Tuning

To adapt the CNN to the new task (detection) and the new domain (warped proposal windows), we continue stochastic gradient descent training of the CNN parameters using only warped region proposals. Aside from replacing the CNN's ImageNet-specific 1000-way classification layer with a randomly initialized $(N+1)$-way classification layer (where N is the number of object classes, plus 1 for background), the CNN architecture is unchanged. For VOC, $N = 20$ and for ILSVRC2013, $N = 200$. 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 minibatch of size 128. We bias the sampling towards positive windows because they are extremely rare compared to background. OxfordNet requires more memory than TorontoNet making it necessary to decrease the minibatch size in order to fit on a single GPU. We decreased the batch size from 128 to just 24 while maintaining the same biased sampling scheme.

3.3.3 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, 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 [21], decreased mAP by 5 points. Similarly, setting it to 0 decreased mAP by four 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 are too large to fit in memory, we adopt the standard hard negative mining method [18], [58]. Hard negative mining converges quickly and in practice mAP stops increasing after only a single pass over all images.

In Section 7.2 we discuss why the positive and negative examples are defined differently in fine-tuning versus SVM training. We also discuss the trade-offs involved in training detection SVMs rather than simply using the outputs from the final softmax layer of the fine-tuned CNN.

3.4 Results on PASCAL VOC 2010-12

Following the PASCAL VOC best practices [3], we validated all design decisions and hyperparameters on the VOC 2007 dataset (Section 4.2). 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.[2] We compare our method against four strong baselines, including SegDPM [57], which combines DPM detectors with the output of a semantic segmentation system [59] and uses additional inter-detector context and image-classifier rescoring. The most germane comparison is to the UVA system from Uijlings et al. [21], 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 4,000-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 percent to 53.7 percent mAP with TorontoNet and 62.9 percent with OxfordNet, while also being much faster. R-CNNs achieve similar performance (53.3 percent / 62.4 percent mAP) on VOC 2012 test.

2. We use VOC 2010 because there are more published results compared to 2012. Additionally, VOC 2010, 2011, 2012 are very similar datasets, with 2011 and 2012 being identical (for the detection task).

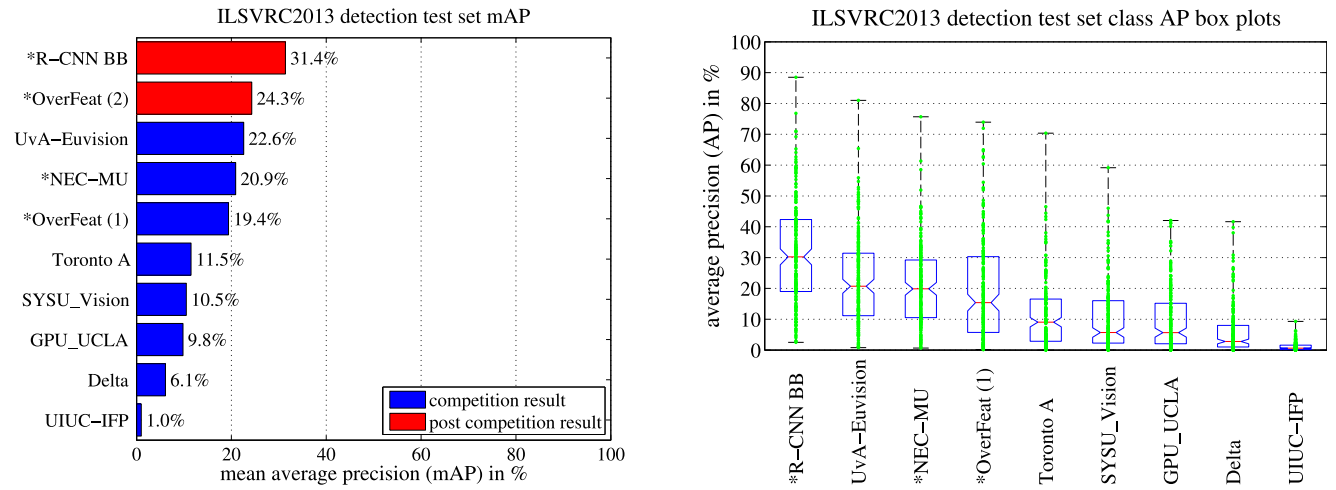

Fig. 3. (Left) Mean average precision on the ILSVRC2013 detection test set. Methods preceeded by * use outside training data (images and labels from the ILSVRC classification dataset in all cases). (Right) Box plots for the 200 average precision values per method. A box plot for the post-competition OverFeat result is not shown because per-class APs are not yet available. The red line marks the median AP, the box bottom and top are the 25th and 75th percentiles. The whiskers extend to the min and max AP of each method. Each AP is plotted as a green dot over the whiskers (best viewed digitally with zoom).

3.5 Results on ILSVRC2013 Detection

We ran an R-CNN on the 200-class ILSVRC2013 detection dataset using the same system hyperparameters that we used for PASCAL VOC. We followed the same protocol of submitting test results to the ILSVRC2013 evaluation server only twice, once with and once without bounding-box regression.

Fig. 3 compares our R-CNN to the entries in the ILSVRC 2013 competition and to the post-competition OverFeat result [19]. Using TorontoNet, our R-CNN achieves a mAP of 31.4 percent, which is significantly ahead of the second-best result of 24.3 percent from OverFeat. To give a sense of the AP distribution over classes, box plots are also presented. Most of the competing submissions (OverFeat, NEC-MU, Toronto A, and UIUC-IFP) used convolutional networks, indicating that there is significant nuance in how CNNs can be applied to object detection, leading to greatly varying outcomes. Notably, UvA-Euvision's entry did not use CNNs and was based on a fast VLAD encoding [60].

In Section 5, we give an overview of the ILSVRC2013 detection dataset and provide details about choices that we made when training R-CNNs on it.

4 Analysis

4.1 Visualizing Learned Features

First-layer filters can be visualized directly and are easy to understand [8]. They capture oriented edges and opponent colors. Understanding the subsequent layers is more challenging. Zeiler and Fergus present a visually attractive deconvolutional approach in [63]. 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 of a TorontoNet, 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 Fig. 4 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. 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 representation 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. Agrawal et al. [25] provide a more in-depth analysis of the learned features.

4.2 Ablation Studies

4.2.1 *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 TorontoNet's last three layers. Layer pool_5 was briefly described in Section 4.1. The final two layers are summarized below.

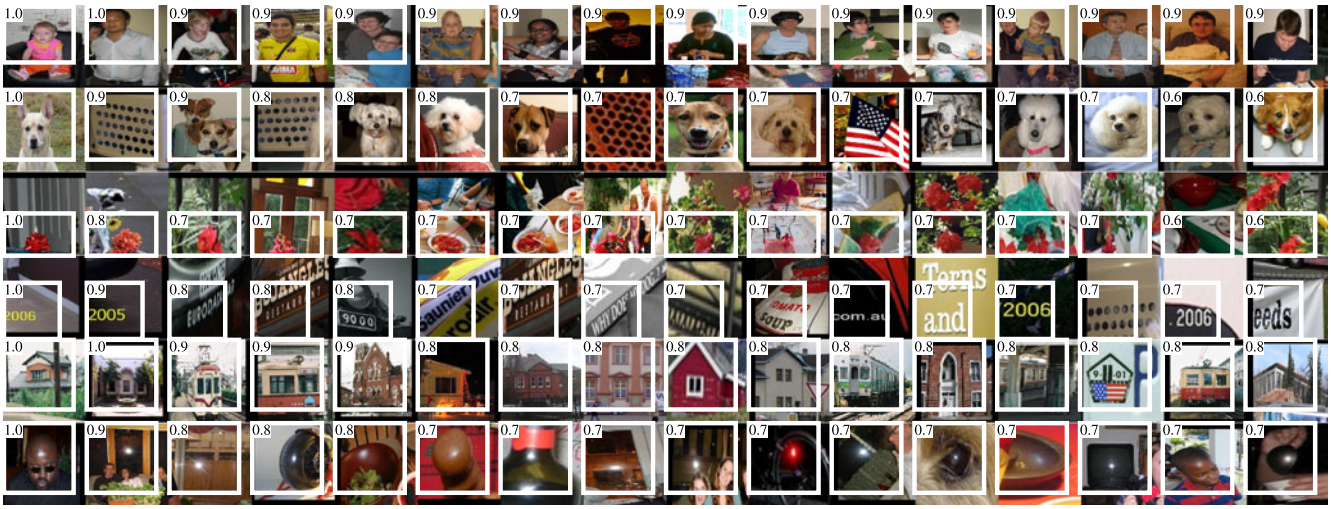

Fig. 4. Top regions for six pool_5 units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

Layer fc_6 is fully connected to pool_5. To compute features, it multiplies a $4{,}096 \times 9{,}216$ weight matrix by the pool_5 feature map (reshaped as a 9,216-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 $4{,}096 \times 4{,}096$ 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 percent, 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 percent* 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.

4.2.2 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 percent. 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.

4.2.3 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 [23].

The first DPM feature learning method, DPM ST [61], 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

TABLE 2
Detection Average Precision (Percent) on VOC 2007 Test

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 [23]	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 [61]	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 [62]	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

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 stage that reduces localization errors (Section 7.3). 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. All R-CNN results use TorontoNet.

TABLE 3
Detection Average Precision (Percent) on VOC 2007 Test for Two Different CNN Architectures

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 T-Net	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 T-Net 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
R-CNN O-Net	71.6	73.5	58.1	42.2	39.4	70.7	76.0	74.5	38.7	71.0	56.9	74.5	67.9	69.6	59.3	**35.7**	62.1	64.0	66.5	**71.2**	62.2
R-CNN O-Net BB	**73.4**	**77.0**	**63.4**	**45.4**	**44.6**	**75.1**	**78.1**	**79.8**	**40.5**	**73.7**	**62.2**	**79.4**	**78.1**	**73.1**	**64.2**	35.6	**66.8**	**67.2**	**70.4**	71.1	**66.0**

The first two rows are results from Table 2 using Krizhevsky et al.'s TorontoNet architecture (T-Net). Rows three and four use the recently proposed 16-layer OxfordNet architecture (O-Net) from Simonyan and Zisserman [24].

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 [62], 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 percent vs. 33.7 percent—*a 61 percent relative improvement*. The combination of HOG and sketch tokens yields 2.5 mAP points over HOG alone, while HSC improves over HOG by four mAP points (when compared internally to their private DPM baselines—both use non-public implementations of DPM that underperform the open source version [23]). These methods achieve mAPs of 29.1 percent and 34.3 percent, respectively.

4.3 Network Architectures

Most results in this paper use the TorontoNet network architecture from Krizhevsky et al. [8]. However, we have found that the choice of architecture has a large effect on R-CNN detection performance. In Table 3, we show results on VOC 2007 test using the 16-layer deep OxfordNet recently proposed by Simonyan and Zisserman [24]. This network was one of the top performers in the recent ILSVRC 2014 classification challenge. The network has a homogeneous structure consisting of 13 layers of 3×3 convolution kernels, with five max pooling layers interspersed, and topped with three fully-connected layers.

To use OxfordNet in an R-CNN, we downloaded the publicly available pre-trained network weights for the `VGG_ILSVRC_16_layers` model from the Caffe Model Zoo.[3] We then fine-tuned the network using the same protocol as we used for TorontoNet. The only difference was to use smaller minibatches (24 examples) as required in order to fit within GPU memory. The results in Table 3 show that an R-CNN with OxfordNet substantially outperforms an R-CNN with TorontoNet, increasing mAP from 58.5 percent to 66.0 percent. However there is a considerable drawback in terms of compute time, with the forward pass of OxfordNet taking roughly seven times longer than TorontoNet. From a transfer learning point of view, it is very encouraging that large improvements in image classification translate directly into large improvements in object detection.

3. https://github.com/BVLC/caffe/wiki/Model-Zoo

4.4 Detection Error Analysis

We applied the excellent detection analysis tool from Hoiem et al. [64] 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 [64] 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 Figs. 5 and 6.

4.5 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 [18], we train a linear regression model to predict a new detection window

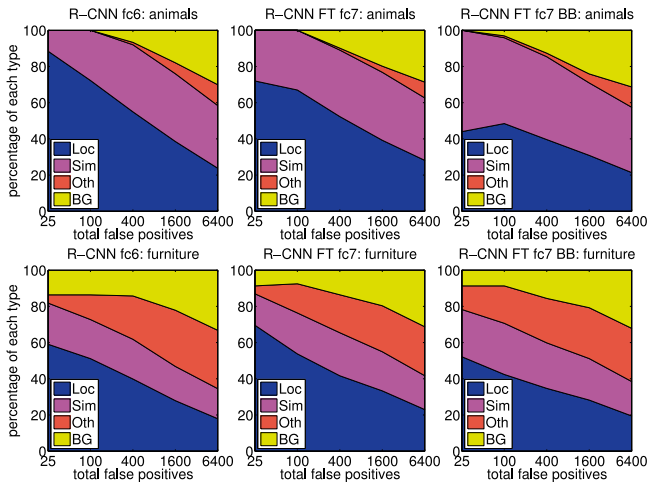

Fig. 5. Distribution of top-ranked false positive (FP) types for R-CNNs with TorontoNet. Each plot shows the evolving distribution of FP types as more FPs are considered in order of decreasing score. Each FP is categorized into 1 of 4 types: Loc—poor localization (a detection with an IoU overlap with 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 [64]), 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.

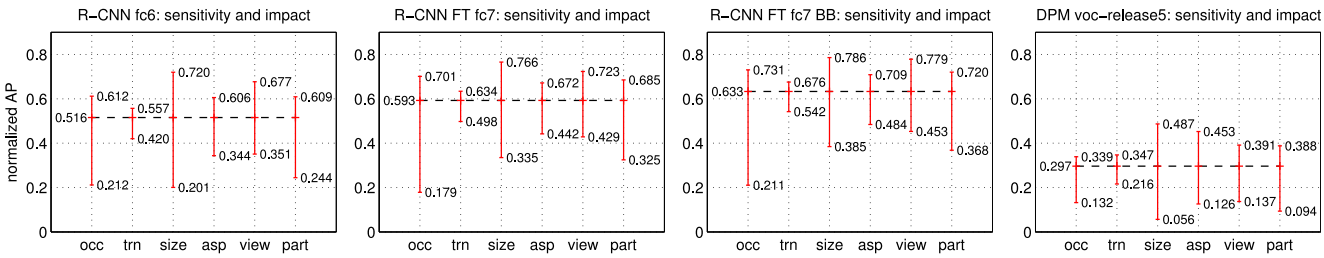

Fig. 6. Sensitivity to object characteristics. Each plot shows the mean (over classes) normalized AP (see [64]) for the highest and lowest performing *subsets* within six different object characteristics (occlusion, truncation, bounding-box area, aspect ratio, viewpoint, part visibility). For example, bounding-box area comprises the subsets extra-small, small ,..., extra-large. We show plots for our method (R-CNN) with and without fine-tuning and bounding-box regression 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.

given the pool_5 features for a selective search region proposal. Full details are given in Section 7.3. Results in Tables 1, 2, and Fig. 5 show that this simple approach fixes a large number of mislocalized detections, boosting mAP by three to four points.

4.6 Qualitative Results

Qualitative detection results on ILSVRC2013 are presented in Fig. 8. Each image was sampled randomly from the val_2 set and all detections from all detectors with a precision greater than 0.5 are shown. Note that these are not curated and give a realistic impression of the detectors in action.

5 The ILSVRC2013 Detection Dataset

In Section 3 we presented results on the ILSVRC2013 detection dataset. This dataset is less homogeneous than PASCAL VOC, requiring choices about how to use it. Since these decisions are non-trivial, we cover them in this section. The methodology and "val_1" and "val_2" data splits introduced in this section were used widely by participants in the ILSVRC2014 detection challenge.

5.1 Dataset Overview

The ILSVRC2013 detection dataset is split into three sets: train (395,918), val (20,121), and test (40,152), where the number of images in each set is in parentheses. The val and test splits are drawn from the same image distribution. These images are scene-like and similar in complexity (number of objects, amount of clutter, pose variability, etc.) to PASCAL VOC images. The val and test splits are exhaustively annotated, meaning that in each image all instances from all 200 classes are labeled with bounding boxes. The train set, in contrast, is drawn from the ILSVRC2013 *classification* image distribution. These images have more variable complexity with a skew towards images of a single centered object. Unlike val and test, the train images (due to their large number) are not exhaustively annotated. In any given train image, instances from the 200 classes may or may not be labeled. In addition to these image sets, each class has an extra set of negative images. Negative images are manually checked to validate that they do not contain any instances of their associated class. The negative image sets were not used in this work. More information on how ILSVRC was collected and annotated can be found in [65], [66].

The nature of these splits presents a number of choices for training an R-CNN. The train images cannot be used for hard negative mining, because annotations are not exhaustive. Where should negative examples come from? Also, the train images have different statistics than val and test. Should the train images be used at all, and if so, to what extent? While we have not thoroughly evaluated a large number of choices, we present what seemed like the most obvious path based on previous experience.

Our general strategy is to rely heavily on the val set and use some of the train images as an auxiliary source of positive examples. To use val for both training and validation, we split it into roughly equally sized "val_1" and "val_2" sets. Since some classes have very few examples in val (the smallest has only 31 and half have fewer than 110), it is important to produce an approximately class-balanced partition. To do this, a large number of candidate splits were generated and the one with the smallest maximum relative class imbalance was selected.[4] Each candidate split was generated by clustering val images using their class counts as features, followed by a randomized local search that may improve the split balance. The particular split used here has a maximum relative imbalance of about 11 percent and a median relative imbalance of 4 percent. The $\text{val}_1/\text{val}_2$ split and code used to produce them are publicly available in the R-CNN code repository, allowing other researchers to compare their methods on the val splits used in this report.

5.2 Region Proposals

We followed the same region proposal approach that was used for detection on PASCAL. Selective search [21] was run in "fast mode" on each image in val_1, val_2, and test (but not on images in train). One minor modification was required to deal with the fact that selective search is not scale invariant and so the number of regions produced depends on the image resolution. ILSVRC image sizes range from very small to a few that are several mega-pixels, and so we resized each image to a fixed width (500 pixels) before running selective search. On val, selective search resulted in an average of 2,403 region proposals per image with a 91.6 percent recall of all ground-truth bounding boxes (at 0.5 IoU threshold). This recall is notably lower

4. Relative imbalance is measured as $|a-b|/(a+b)$ where a and b are class counts in each half of the split.

TABLE 4
ILSVRC2013 Ablation Study of Data Usage Choices, Fine-Tuning, and Bounding-Box Regression

test set	val_2	val_2	val_2	val_2	val_2	val_2	test	test
SVM training set	val_1	$val_1 + train_{.5k}$	$val_1 + train_{1k}$	$val_1 + train_{1k}$	$val_1 + train_{1k}$	$val_1 + train_{1k}$	$val + train_{1k}$	$val + train_{1k}$
CNN fine-tuning set	n/a	n/a	n/a	val_1	$val_1 + train_{1k}$	$val_1 + train_{1k}$	$val_1 + train_{1k}$	$val_1 + train_{1k}$
bbox reg set	n/a	n/a	n/a	n/a	n/a	val_1	n/a	val
CNN feature layer	fc_6	fc_6	fc_6	fc_7	fc_7	fc_7	fc_7	fc_7
mAP	20.9	24.1	24.1	26.5	29.7	**31.0**	30.2	**31.4**
median AP	17.7	21.0	21.4	24.8	29.2	**29.6**	29.0	**30.3**

All experiments use TorontoNet.

than in PASCAL, where it is approximately 98 percent, indicating significant room for improvement in the region proposal stage.

5.3 Training Data

For training data, we formed a set of images and boxes that includes all selective search and ground-truth boxes from val_1 together with up to N ground-truth boxes per class from train (if a class has fewer than N ground-truth boxes in train, then we take all of them). We'll call this dataset of images and boxes $val_1 + train_N$. In an ablation study, we show mAP on val_2 for $N \in \{0, 500, 1000\}$ (Section 5.5).

Training data are required for three procedures in R-CNN: (1) CNN fine-tuning, (2) detector SVM training, and (3) bounding-box regressor training. CNN fine-tuning was run for 50k SGD iteration on $val_1 + train_N$ using the exact same settings as were used for PASCAL. Fine-tuning on a single NVIDIA Tesla K20 took 13 hours using Caffe. For SVM training, all ground-truth boxes from $val_1 + train_N$ were used as positive examples for their respective classes. Hard negative mining was performed on a randomly selected subset of 5,000 images from val_1. An initial experiment indicated that mining negatives from all of val_1, versus a 5,000 image subset (roughly half of it), resulted in only a 0.5 percentage point drop in mAP, while cutting SVM training time in half. No negative examples were taken from train because the annotations are not exhaustive. The extra sets of verified negative images were not used. The bounding-box regressors were trained on val_1.

5.4 Validation and Evaluation

Before submitting results to the evaluation server, we validated data usage choices and the effect of fine-tuning and bounding-box regression on the val_2 set using the training data described above. All system hyperparameters (e.g., SVM C hyperparameters, padding used in region warping, NMS thresholds, bounding-box regression hyperparameters) were fixed at the same values used for PASCAL. Undoubtedly some of these hyperparameter choices are slightly suboptimal for ILSVRC, however the goal of this work was to produce a preliminary R-CNN result on ILSVRC without extensive dataset tuning. After selecting the best choices on val_2, we submitted exactly two result files to the ILSVRC2013 evaluation server. The first submission was without bounding-box regression and the second submission was with bounding-box regression. For these submissions, we expanded the SVM and bounding-box regressor training sets to use $val+train_{1k}$ and val, respectively. We used the CNN that was fine-tuned on $val_1 + train_{1k}$ to avoid re-running fine-tuning and feature computation.

5.5 Ablation Study

Table 4 shows an ablation study of the effects of different amounts of training data, fine-tuning, and bounding-box regression. A first observation is that mAP on val_2 matches mAP on test very closely. This gives us confidence that mAP on val_2 is a good indicator of test set performance. The first result, 20.9 percent, is what R-CNN achieves using a CNN pre-trained on the ILSVRC2012 classification dataset (no fine-tuning) and given access to the small amount of training data in val_1 (recall that half of the classes in val_1 have between 15 and 55 examples). Expanding the training set to $val_1 + train_N$ improves performance to 24.1 percent, with essentially no difference between $N = 500$ and $N = 1000$. Fine-tuning the CNN using examples from just val_1 gives a modest improvement to 26.5 percent, however there is likely significant overfitting due to the small number of positive training examples. Expanding the fine-tuning set to $val_1 + train_{1k}$, which adds up to 1000 positive examples per class from the train set, helps significantly, boosting mAP to 29.7 percent. Bounding-box regression improves results to 31.0 percent, which is a smaller relative gain that what was observed in PASCAL.

5.6 Relationship to OverFeat

There is an interesting relationship between R-CNN and OverFeat: OverFeat can be seen (roughly) as a special case of an R-CNN. If one were to replace selective search region proposals with a multi-scale pyramid of regular square regions and change the per-class bounding-box regressors to a single bounding-box regressor, then the systems would be very similar (modulo some potentially significant differences in how they are trained: CNN detection fine-tuning, using SVMs, etc.). It is worth noting that OverFeat has a significant speed advantage over R-CNN: it is about 9× faster, based on a figure of 2 seconds per image quoted from [19]. This speed comes from the fact that OverFeat's sliding windows (i.e., region proposals) are not warped at the image level and therefore computation can be easily shared between overlapping windows. Sharing is implemented by running the entire network in a convolutional fashion over arbitrary-sized inputs. OverFeat is slower than the pyramid-based version of R-CNN from He et al. [27].

TABLE 5
Segmentation Mean Accuracy (Percent) on VOC 2011 Validation

	full R-CNN		*fg* R-CNN		*full + fg* R-CNN	
O_2P [59]	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

Column 1 presents O_2P; 2-7 use our CNN pre-trained on ILSVRC 2012.

6 Semantic Segmentation

Region classification is a standard technique for semantic segmentation, allowing us to easily apply R-CNNs 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") [59], 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 and the powerful second-order pooling of multiple feature types (enriched variants of SIFT and LBP). We also note that Farabet et al. [67] 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 [59], [68] and extend the PASCAL segmentation training set to include the extra annotations made available by Hariharan et al. [69]. Design decisions and hyperparameters were cross-validated on the VOC 2011 validation set. Final test results were evaluated only once.

6.1 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 (we use TorontoNet for these experiments). 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.

6.2 Results on VOC 2011

Table 5 shows a summary of our results on the VOC 2011 validation set compared with O_2P. 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 percent, our best result by a margin of 4.2 percent (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.

Table 6 shows the per-category segmentation accuracy on VOC 2011 val for each of our six segmentation methods in addition to the O_2P method [59]. These results show which methods are strongest across each of the 20 PASCAL classes, plus the background class.

In Table 7 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 percent, 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.

More recently, a number of semantic segmentation approaches based on deep CNNs have lead to dramatic improvements, pushing segmentation mean accuracy over 70 percent [70], [71], [72], [73]. The highest performing of these methods combine fully-convolution networks (fine-tuned for segmentation) with efficient fully-connected Gaussian CRFs [74].

7 Implementation and Design Details

7.1 Object Proposal Transformations

The convolutional networks used in this work require a fixed-size input (e.g., 227×227 pixels) in order to produce a fixed-size output. For detection, we consider object proposals that are arbitrary image rectangles. We evaluated two approaches for transforming object proposals into valid CNN inputs.

TABLE 6
Per-Category Segmentation Accuracy (Percent) on the VOC 2011 Validation Set

VOC 2011 val	bg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
O_2P [59]	**84.0**	**69.0**	21.7	47.7	42.2	42.4	**64.7**	**65.8**	57.4	**12.9**	37.4	20.5	43.7	35.7	52.7	51.0	**35.8**	**51.0**	28.4	59.8	49.7	46.4
full R-CNN fc_6	81.3	56.2	23.9	42.9	40.7	38.8	59.2	56.5	53.2	11.4	34.6	16.7	48.1	37.0	51.4	46.0	31.5	44.0	24.3	53.7	51.1	43.0
full R-CNN fc_7	81.0	52.8	**25.1**	43.8	40.5	42.7	55.4	57.7	51.3	8.7	32.5	11.5	48.1	37.0	50.5	46.4	30.2	42.1	21.2	57.7	**56.0**	42.5
fg R-CNN fc_6	81.4	54.1	21.1	40.6	38.7	**53.6**	59.9	57.2	52.5	9.1	36.5	**23.6**	46.4	38.1	53.2	51.3	32.2	38.7	**29.0**	53.0	47.5	43.7
fg R-CNN fc_7	80.9	50.1	20.0	40.2	34.1	40.9	59.7	59.8	52.7	7.3	32.1	14.3	48.8	42.9	54.0	48.6	28.9	42.6	24.9	52.2	48.8	42.1
full+fg R-CNN fc_6	83.1	60.4	23.2	48.4	**47.3**	52.6	61.6	60.6	**59.1**	10.8	**45.8**	20.9	**57.7**	43.3	**57.4**	**52.9**	34.7	48.7	28.1	60.0	48.6	**47.9**
full+fg R-CNN fc_7	82.3	56.7	20.6	**49.9**	44.2	43.6	59.3	61.3	57.8	7.7	38.4	15.1	53.4	**43.7**	50.8	52.0	34.1	47.8	24.7	**60.1**	55.2	45.7

These experiments use TorontoNet without fine-tuning.

TABLE 7
Segmentation Accuracy (Percent) on VOC 2011 Test

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 [68]	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 [59]	**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**

We compare against two strong baselines: the "Regions and Parts" (R&P) method of [68] and the second-order pooling (O_2P) method of [59]. Without any fine-tuning, our CNN achieves top segmentation performance, outperforming R&P and roughly matching O_2P. These experiments use TorontoNet without fine-tuning.

The first method ("tightest square with context") encloses each object proposal inside the tightest square and then scales (isotropically) the image contained in that square to the CNN input size. Fig. 7 column (B) shows this transformation. A variant on this method ("tightest square without context") excludes the image content that surrounds the original object proposal. Fig. 7 column (C) shows this transformation. The second method ("warp") anisotropically scales each object proposal to the CNN input size. Fig. 7 column (D) shows the warp transformation.

For each of these transformations, we also consider including additional image context around the original object proposal. The amount of context padding (p) is defined as a border size around the original object proposal in the transformed input coordinate frame. Fig. 7 shows $p = 0$ pixels in the top row of each example and $p = 16$ pixels in the bottom row. In all methods, if the source rectangle extends beyond the image, the missing data are replaced with the image mean (which is then subtracted before inputing the image into the CNN). A pilot set of experiments showed that warping with context padding ($p = 16$ pixels) outperformed the alternatives by a large margin (3-5 mAP points). Obviously more alternatives are possible, including using replication instead of mean padding. Exhaustive evaluation of these alternatives is left as future work.

7.2 Positive Versus Negative Examples and Softmax

Two design choices warrant further discussion. The first is: Why are positive and negative examples defined differently for fine-tuning the CNN versus training the object detection SVMs? To review the definitions briefly, for fine-tuning we map each object proposal to the ground-truth instance with which it has maximum IoU overlap (if any) and label it as a positive for the matched ground-truth class if the IoU is at least 0.5. All other proposals are labeled "background" (i.e., negative examples for all classes). For training SVMs, in contrast, we take only the ground-truth boxes as positive examples for their respective classes and label proposals with less than 0.3 IoU overlap with all instances of a class as a negative for that class. Proposals that fall into the grey zone (more than 0.3 IoU overlap, but are not ground truth) are ignored.

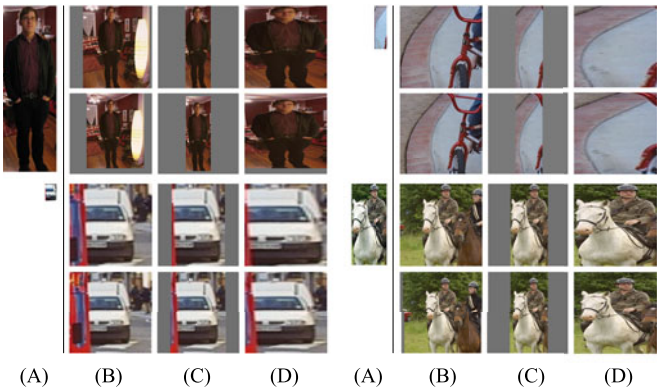

Fig. 7. Different object proposal transformations. (A) the original object proposal at its actual scale relative to the transformed CNN inputs; (B) tightest square with context; (C) tightest square without context; (D) warp. Within each column and example proposal, the top row corresponds to $p = 0$ pixels of context padding while the bottom row has $p = 16$ pixels of context padding.

Historically speaking, we arrived at these definitions because we started by training SVMs on features computed by the ImageNet pre-trained CNN, and so fine-tuning was not a consideration at that point in time. In that setup, we found that our particular label definition for training SVMs was optimal within the set of options we evaluated (which included the setting we now use for fine-tuning). When we started using fine-tuning, we initially used the same positive and negative example definition as we were using for SVM training. However, we found that results were much worse than those obtained using our current definition of positives and negatives.

Our hypothesis is that this difference in how positives and negatives are defined is not fundamentally important and arises from the fact that fine-tuning data are limited. Our current scheme introduces many "jittered" examples (those proposals with overlap between 0.5 and 1, but not ground truth), which expands the number of positive examples by approximately 30x. We conjecture that this large set is needed when fine-tuning the *entire* network to avoid overfitting. However, we also note that using these jittered examples is likely suboptimal because the network is not being fine-tuned for precise localization.

This leads to the second issue: Why, after fine-tuning, train SVMs at all? It would be cleaner to simply apply the last layer of the fine-tuned network, which is a 21-way softmax regression classifier, as the object detector. We tried this and found that performance on VOC 2007 dropped from 54.2 to 50.9 percent mAP. This performance drop likely arises from a combination of several factors including that the definition of positive examples used in fine-tuning does not emphasize precise localization and the softmax classifier was trained on randomly sampled negative examples rather than on the subset of "hard negatives" used for SVM training.

This result shows that it's possible to obtain close to the same level of performance without training SVMs after

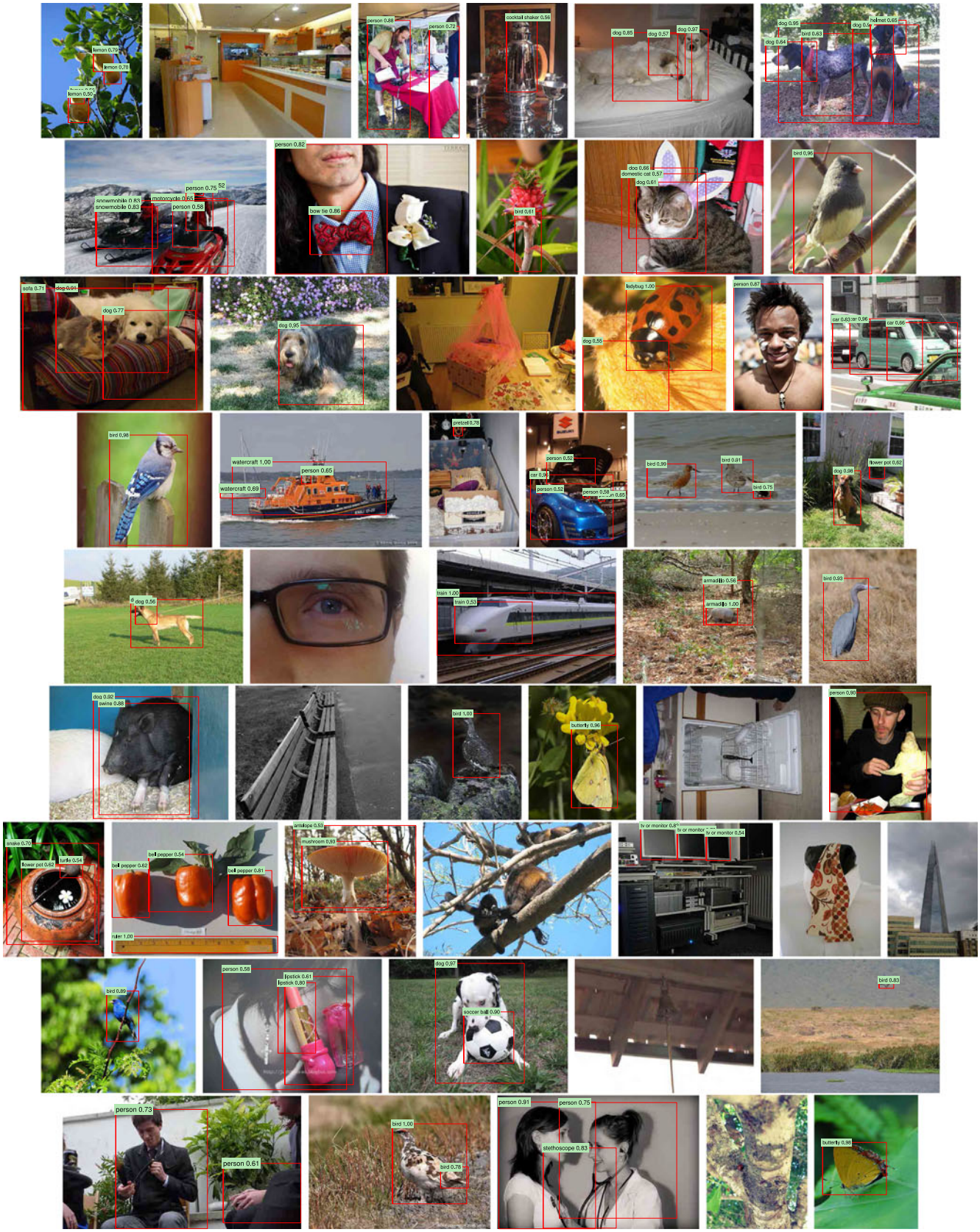

Fig. 8. Example detections on the val_2 set from the configuration that achieved 31.0 percent mAP on val_2. Each image was sampled randomly (these are *not* curated). All detections at precision greater than 0.5 are shown. Each detection is labeled with the predicted class and the precision value of that detection from the detector's precision-recall curve. Viewing digitally with zoom is recommended.

fine-tuning. We conjecture that with some additional tweaks to fine-tuning the remaining performance gap may be closed. If true, this would simplify and speed up R-CNN training with no loss in detection performance.

7.3 Bounding-Box Regression

We use a simple bounding-box regression stage to improve localization performance. After scoring each selective search proposal with a class-specific detection SVM, we predict a new bounding box for the detection using a class-specific bounding-box regressor. This is similar in spirit to the bounding-box regression used in deformable part models [18]. The primary difference between the two approaches is that here we regress from features computed by the CNN, rather than from geometric features computed on the inferred DPM part locations.

The input to our training algorithm is a set of N training pairs $\{(P^i, G^i)\}_{i=1,\ldots,N}$, where $P^i = (P^i_x, P^i_y, P^i_w, P^i_h)$ specifies the pixel coordinates of the center of proposal P^i's bounding box together with P^i's width and height in pixels. Hence forth, we drop the superscript i unless it is needed. Each ground-truth bounding box G is specified in the same way: $G = (G_x, G_y, G_w, G_h)$. Our goal is to learn a transformation that maps a proposed box P to a ground-truth box G.

We parameterize the transformation in terms of four functions $d_x(P)$, $d_y(P)$, $d_w(P)$, and $d_h(P)$. The first two specify a scale-invariant translation of the center of P's bounding box, while the second two specify log-space translations of the width and height of P's bounding box. After learning these functions, we can transform an input proposal P into a predicted ground-truth box $\hat{G}$ by applying the transformation

$$\hat{G}_x = P_w d_x(P) + P_x \tag{1}$$

$$\hat{G}_y = P_h d_y(P) + P_y \tag{2}$$

$$\hat{G}_w = P_w \exp(d_w(P)) \tag{3}$$

$$\hat{G}_h = P_h \exp(d_h(P)). \tag{4}$$

Each function $d_\star(P)$ (where $\star$ is one of x, y, h, w) is modeled as a linear function of the pool_5 features of proposal P, denoted by $\phi_5(P)$. (The dependence of $\phi_5(P)$ on the image data is implicitly assumed.) Thus we have $d_\star(P) = \mathbf{w}_\star^{\mathrm{T}} \phi_5(P)$, where $\mathbf{w}_\star$ is a vector of learnable model parameters. We learn $\mathbf{w}_\star$ by optimizing the regularized least squares objective (ridge regression):

$$\mathbf{w}_\star = \underset{\hat{\mathbf{w}}_\star}{\arg\min} \sum_i^N (t^i_\star - \hat{\mathbf{w}}_\star^{\mathrm{T}} \phi_5(P^i))^2 + \lambda ||\hat{\mathbf{w}}_\star||^2. \tag{5}$$

The regression targets $t_\star$ for the training pair (P, G) are defined as

$$t_x = (G_x - P_x)/P_w \tag{6}$$

$$t_y = (G_y - P_y)/P_h \tag{7}$$

$$t_w = \log(G_w/P_w) \tag{8}$$

$$t_h = \log(G_h/P_h). \tag{9}$$

As a standard regularized least squares problem, this can be solved efficiently in closed form.

We found two subtle issues while implementing bounding-box regression. The first is that regularization is important: we set $\lambda = 1000$ based on a validation set. The second issue is that care must be taken when selecting which training pairs (P, G) to use. Intuitively, if P is far from all ground-truth boxes, then the task of transforming P to a ground-truth box G does not make sense. Using examples like P would lead to a hopeless learning problem. Therefore, we only learn from a proposal P if it is *nearby* at least one ground-truth box. We implement "nearness" by assigning P to the ground-truth box G with which it has maximum IoU overlap (in case it overlaps more than one) if and only if the overlap is greater than a threshold (which we set to 0.6 using a validation set). All unassigned proposals are discarded. We do this once for each object class in order to learn a set of class-specific bounding-box regressors.

At test time, we score each proposal and predict its new detection window only once. In principle, we could iterate this procedure (i.e., re-score the newly predicted bounding box, and then predict a new bounding box from it, and so on). However, we found that iterating does not improve results.

7.4 Analysis of Cross-Dataset Redundancy

One concern when training on an auxiliary dataset is that there might be redundancy between it and the test set. Even though the tasks of object detection and whole-image classification are substantially different, making such cross-set redundancy much less worrisome, we still conducted a thorough investigation that quantifies the extent to which PASCAL test images are contained within the ILSVRC 2012 training and validation sets. Our findings may be useful to researchers who are interested in using ILSVRC 2012 as training data for the PASCAL image classification task.

We performed two checks for duplicate (and near-duplicate) images. The first test is based on exact matches of flickr image IDs, which are included in the VOC 2007 test annotations (these IDs are intentionally kept secret for subsequent PASCAL test sets). All PASCAL images, and about half of ILSVRC, were collected from flickr.com. This check turned up 31 matches out of 4,952 (0.63 percent).

The second check uses GIST [75] descriptor matching, which was shown in [76] to have excellent performance at near-duplicate image detection in large (> 1 million) image collections. Following [76], we computed GIST descriptors on warped 32×32 pixel versions of all ILSVRC 2012 trainval and PASCAL 2007 test images.

Euclidean distance nearest-neighbor matching of GIST descriptors revealed 38 near-duplicate images (including all 31 found by flickr ID matching). The matches tend to vary slightly in JPEG compression level and resolution, and to a lesser extent cropping. These findings show that the overlap is small, less than 1 percent. For VOC 2012, because flickr

IDs are not available, we used the GIST matching method only. Based on GIST matches, 1.5 percent of VOC 2012 test images are in ILSVRC 2012 trainval. The slightly higher rate for VOC 2012 is likely due to the fact that the two datasets were collected closer together in time than VOC 2007 and ILSVRC 2012 were.

8 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 more than a 50 percent 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 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 are 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 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 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 US National Science Foundation 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. R. Girshick is with Microsoft Research and was with the Department of Electrical Engineering and Computer Science, UC Berkeley during the majority of this work.

References

[1] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.
[3] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 80, no. 2, pp. 303–338, 2010.
[4] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, 1980.
[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distrib. Process.*, vol. 1, pp. 318–362, 1986.
[6] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[8] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[9] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei. Imagenet large scale visual recognition competition 2012 (ILSVRC2012) [Online]. Available: http://www.image-net.org/challenges/LSVRC/2012/, 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 248–255.
[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 580–587.
[12] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2553–2561.
[13] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 2155–2162.
[14] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 23–28, Jan. 1998.
[15] R. Vaillant, C. Monrocq, and Y. LeCun, "Original approach for the localisation of objects in images," *IEE Proc. Vis., Image, Signal Process.*, vol. 141, no. 4, pp. 245–250, Aug. 1994.
[16] J. Platt and S. Nowlan, "A convolutional neural network hand tracker," in *Proc. Adv. Neural Inf. Process. Syst.*, 1995, pp. 901–908.
[17] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Pedestrian detection with unsupervised multi-stage feature learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3626–3633.
[18] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
[19] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2014, p. 16.
[20] C. Gu, J. J. Lim, P. Arbeláez, and J. Malik, "Recognition using regions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 1030–1037.
[21] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 3, pp. 154–171, 2013.
[22] J. Carreira and C. Sminchisescu, "CPMC: Automatic object segmentation using constrained parametric min-cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1312–1328, Jul. 2012.
[23] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5 [Online]. Available: http://www.cs.berkeley.edu/~rbg/latent-v5/, 2012.
[24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.
[25] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 329–344.
[26] T. Dean, J. Yagnik, M. Ruzon, M. Segal, J. Shlens, and S. Vijayanarasimhan, "Fast, accurate detection of 100,000 object classes on a single machine," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1814–1821.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 346–361.
[28] R. Girshick, "Fast R-CNN," *arXiv e-prints*, vol. arXiv:1504.08083v1 [cs.CV], 2015.
[29] D. Hoiem, A. Efros, and M. Hebert, "Geometric context from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 654–661.
[30] B. C. Russell, W. T. Freeman, A. A. Efros, J. Sivic, and A. Zisserman, "Using multiple segmentations to discover objects and their extent in image collections," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2006, pp. 1605–1614.

[31] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 391–405.
[32] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. H. S. Torr, "BING: Binarized normed gradients for objectness estimation at 300fps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 3286–3293.
[33] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *arXiv e-prints*, vol. arXiv:1502.05082v1 [cs.CV], 2015.
[34] A. Humayun, F. Li, and J. M. Rehg, "RIGOR: Reusing inference in graph cuts for generating object regions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 336–343.
[35] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 328–335.
[36] P. Krähenbühl, and V. Koltun, "Geodesic object proposals," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 725–739.
[37] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[38] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," in *Proc. 10th Int. Conf. Mach. Learn.*, 1993, pp. 41–48.
[39] S. Thrun, "Is learning the n-th thing any easier than learning the first?" in *Proc. Adv. Neural Inf. Process. Syst.*, 1996, pp. 640–646.
[40] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[41] 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 *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 647–655.
[42] J. Hoffman, S. Guadarrama, E. Tzeng, J. Donahue, R. Girshick, T. Darrell, and K. Saenko, "From large-scale object classifiers to large-scale object detectors: An adaptation approach," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3536–3544.
[43] A. Karpathy, A. Joulin, and L. Fei-Fei, "Deep fragment embeddings for bidirectional image sentence mapping," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1889–1897.
[44] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik, "R-CNNs for pose estimation and action detection," *arXiv e-prints*, vol. arXiv:1406.5212v1 [cs.CV], 2014.
[45] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 297–312.
[46] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 345–360.
[47] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell, "On learning to localize objects with minimal supervision," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1611–1619.
[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *arXiv e-prints*, vol. arXiv:1409.0575v1 [cs.CV], 2014.
[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *arXiv e-prints*, vol. arXiv:1409.4842v1 [cs.CV], 2014.
[50] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov, "Scalable, high-quality object detection," *arXiv e-prints*, vol. arXiv:1412.1441v2 [cs.CV], 2015.
[51] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2189–2202, Nov. 2012.
[52] I. Endres and D. Hoiem, "Category independent object proposals," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 575–588.
[53] D. Cireşan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Proc. Medical Image Comput. Comput.-Assisted Intervention*, 2013, pp. 411–418.
[54] X. Wang, M. Yang, S. Zhu, and Y. Lin, "Regionlets for generic object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 17–24.
[55] Y. Jia. (2013). Caffe: An open source convolutional architecture for fast feature embedding [Online]. Available: http://caffe.berkeleyvision.org/
[56] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1814–1821.
[57] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun, "Bottom-up segmentation for top-down detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3294–3301.
[58] K. Sung and T. Poggio, "Example-based learning for view-based human face detection," Artif. Intell. Lab., Massachussets Inst. Technol., Cambridge, MA, USA, Tech. Rep. A.I. Memo No. 1521, 1994.
[59] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Semantic segmentation with second-order pooling," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 430–443.
[60] K. E. van de Sande, C. G. Snoek, and A. W. Smeulders, "Fisher and vlad with flair," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 2377–2384.
[61] J. J. Lim, C. L. Zitnick, and P. Dollár, "Sketch tokens: A learned mid-level representation for contour and object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3158–3165.
[62] X. Ren and D. Ramanan, "Histograms of sparse codes for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3246–3253.
[63] M. Zeiler, G. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 2018–2025.
[64] D. Hoiem, Y. Chodpathumwan, and Q. Dai, "Diagnosing error in object detectors," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 340–353.
[65] J. Deng, O. Russakovsky, J. Krause, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Scalable multi-label annotation," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2014, pp. 3099–3102.
[66] H. Su, J. Deng, and L. Fei-Fei, "Crowdsourcing annotations for visual object detection," in *Proc. AAAI 4th Human Comput. Workshop*, 2012.
[67] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1915–1929, Aug. 2013.
[68] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik, "Semantic segmentation using regions and parts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 3378–3385.
[69] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 991–998.
[70] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 447–456.
[71] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *Proc. Int. Conf. Learn. Representations*, 2015.
[72] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr, "Conditional random fields as recurrent neural networks," *arXiv e-print*, vol. arXiv:1502.03240v2 [cs.CV], 2015.
[73] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.
[74] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 109–117.
[75] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, pp. 145–175, 2001.
[76] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid, "Evaluation of gist descriptors for web-scale image search," in *Proc. ACM Int. Conf. Image Video Retrieval*, 2009, p. 19.

Ross Girshick received the MS and PhD degrees in computer science, both from the University of Chicago where he studied under the supervision of Pedro Felzenszwalb. He is a researcher at Microsoft Research, Redmond. Prior to joining Microsoft Research, he was a Postdoctorial fellow at the University of California, Berkeley where he collaborated with Jitendra Malik and Trevor Darrell. During the course of PASCAL VOC object detection challenge, he participated in multiple winning object detection entries and was awarded a "lifetime achievement" prize for his work on the widely used Deformable Part Models.

Jeff Donahue received the BS degree in computer science from UT Austin in 2011, completing an honors thesis with Kristen Grauman. He is a fourth year PhD student at UC Berkeley, supervised by Trevor Darrell. His research interests are in visual recognition and machine learning, most recently focusing on the application of deep learning to visual localization and sequencing problems. He is a student member of the IEEE.

Trevor Darrell received the BSE degree from the University of Pennsylvania in 1988, having started his career in computer vision as an undergraduate researcher in Ruzena Bajcsy's GRASP lab. He received the SM and PhD degrees from MIT in 1992 and 1996, respectively. His group is co-located at the University of California, Berkeley, and the UCB-affiliated International Computer Science Institute (ICSI), also located in Berkeley, CA. He is on the faculty of the CS Division of the EECS Department at UCB and is the vision group lead at ICSI. His group develops algorithms for large-scale perceptual learning, including object and activity recognition and detection, for a variety of applications including multimodal interaction with robots and mobile devices. His interests include computer vision, machine learning, computer graphics, and perception-based human computer interfaces. He was previously on the faculty of the MIT EECS department from 1999-2008, where he directed the Vision Interface Group. He was a member of the research staff at Interval Research Corporation from 1996-1999. He is a member of the IEEE.

Jitendra Malik received the BTech degree in EE from the Indian Institute of Technology, Kanpur, in 1980 and the PhD degree in CS from Stanford University in 1985. In January 1986, he joined UC Berkeley, where he is currently the Arthur J. Chick professor in the Department of EECS. He is also on the faculty of the Department of Bioengineering, and the Cognitive Science and Vision Science groups. During 2002-2004, he served as the chair of the Computer Science Division and during 2004-2006 as the Department Chair of EECS. His research group has worked on computer vision, computational modeling of human vision, computer graphics and the analysis of biological images. Several well-known concepts and algorithms arose in this research, such as anisotropic diffusion, normalized cuts, high dynamic range imaging, and shape contexts. He is one of the most highly cited researchers in computer vision, with 10 of his papers having received more than a thousand citations each. He has graduated 33 PhD students, many of whom are now prominent researchers in academia and industry. He received the Gold Medal for the best graduating student in electrical engineering from IIT Kanpur in 1980 and a Presidential Young Investigator Award in 1989. He was awarded the Longuet-Higgins Prize for a contribution that has stood the test of time twice, in 2007 and in 2008. He is a member of the National Academy of Engineering and a fellow of the American Academy of Arts and Sciences. In 2013, he received the IEEE PAMI-TC Distinguished Researcher in Computer Vision Award, and in 2014 the K.S.Fu Prize from the International Association of Pattern Recognition. He is a fellow of both the ACM and IEEE.