Weakly supervised object recognition with convolutional neural networks

Maxime Oquab[*]
INRIA Paris, France
maxime.oquab@inria.fr

Léon Bottou
Microsoft Research, New York, USA
leon@bottou.org

Ivan Laptev[*]
INRIA, Paris, France
ivan.laptev@inria.fr

Josef Sivic[*]
INRIA, Paris, France
josef.sivic@ens.fr

Abstract

Successful visual object recognition methods typically rely on training datasets containing lots of richly annotated images. Annotating object bounding boxes is both expensive and subjective. We describe a weakly supervised convolutional neural network (CNN) for object recognition that does not rely on detailed object annotation and yet returns 86.3% mAP on the Pascal VOC classification task, outperforming previous fully-supervised systems by a sizeable margin. Despite the lack of bounding box supervision, the network produces maps that clearly localize the objects in cluttered scenes. We also show that adding fully supervised object examples to our weakly supervised setup does not increase the classification performance.

1 Introduction

Visual object recognition entails much more than determining whether the image contains instances of certain object categories. For example, each object has a location and a pose; each deformable object has a constellation of parts; and each object can be cropped or partially occluded. A broad definition of object recognition could be the recovery of attributes associated with single objects in the image, as opposed to those describing relations between objects.

Labelling a set of training images with object attributes quickly becomes problematic. The process is expensive and involves a lot of subtle and possibly ambiguous decisions. For instance, consistently annotating locations and scales of objects by bounding boxes works well for some images but fails for partially occluded and cropped objects as illustrated in Figure 1. Annotating object parts becomes even harder since the correspondence of parts among images in the same category is often ill-posed.

Object recognition algorithms of the past decade can roughly be categorized in two styles. The first style extracts local image features (SIFT, HOG), constructs *bag of visual words* representations, and runs statistical classifiers [9, 30, 37, 43]. Although this approach has been shown to yield good performance for image classification, attempts to locate the objects using the position of the visual words have been unfruitful: the classifier often relies on visual words that fall in the background and merely describe the context of the object. The second style of algorithms detects the presence of objects by fitting rich object models such as *deformable part models* [15, 41]. The fitting process can reveal useful attributes of objects such as location, pose and constellations of object parts provided

[*] WILLOW project, Departement d'Informatique de l'École Normale Supérieure, ENS/INRIA/CNRS UMR 8548, Paris, France

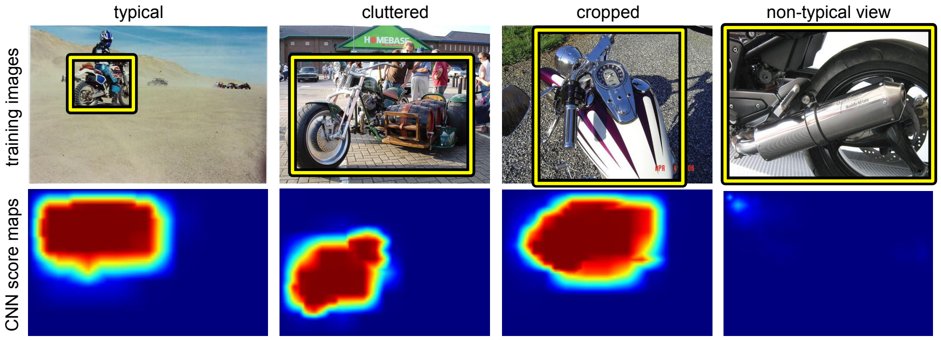


Figure 1: Top row: example images for the motorbike class and corresponding ground truth bounding boxes from Pascal VOC12 training set. Bottom row: corresponding per-pixel score maps produced by our weakly supervised motorbike classifier that has been trained from image class labels only.

locations of objects and possibly their parts in training images. The combination of both styles has shown benefits [19].

A third style of algorithms, *convolutional neural networks* (CNNs) [23, 25] construct successive feature vectors that progressively describe the properties of larger and larger image areas. Building a competitive CNN technology requires serious engineering efforts, possibly rewarded by very good performance [22]. Recent works [5, 17, 27, 32, 33] train convolutional feature extractors on a large supervised image classification task, such as ImageNet, and transfer the trained feature extractors to other object recognition tasks, such as the Pascal VOC tasks.

In this paper, we focus on algorithms that recover classes and locations of objects provided image-level object labels at training only. Figure 1(bottom row) illustrates per-pixel score maps generated by our method for example training images of a motorbike class. Notably, the method correctly recovers locations of objects with common appearance and avoids overfitting to clutter and arbitrary image crops by localizing discriminative object parts, when such are present in the image.

We build on the successful CNN architecture [22] and the follow-up state-of-the-art results for object classification and detection in Pascal VOC [17, 27]. While this previous work has used object bounding boxes for training, we develop a *weakly supervised CNN* that localizes objects while optimizing an image classification criterion only. We modify [22] and treat the last fully connected network layers as convolutions to cope with uncertainty is object localization. We also modify the cost function and introduce the final max-pooling layer that implements weak supervision similar to [24, section 4]. We show our method to outperform all previously published techniques for the Pascal VOC 2012 image classification task and illustrate convincing results of weakly supervised object localization on training and test images.

2 Related work

The fundamental challenge in visual recognition is modeling the intra-class appearance and shape variation of objects. For example, what is the appropriate model of the various appearances and shapes of “chairs”? This challenge is usually addressed by designing some form of a parametric model of the object’s appearance and shape. The parameters of the model are then learnt from a set of instances using statistical machine learning. Learning methods for visual recognition can be characterized based on the required input supervision and the target output.

Unsupervised methods [26, 36] do not require any supervisory signal, just images. While unsupervised learning is appealing, the output is currently often limited only to frequently occurring and visually consistent objects. Fully supervised methods [14] require careful annotation of object location in the form of bounding boxes [14], segmentation [40] or even location of object parts [4], which is costly and can introduce biases. For example, should we annotate the dog’s head or the entire dog? What if a part of the dog’s body is occluded by another object? In this work, we focus on *weakly supervised* learning where only image-level labels indicating the presence or absence of objects are required. This is an important setup for many practical applications as (weak) image-level annotations are often readily available in large amounts, e.g. in the form of text tags [18], full sentences [28] or even geographical meta-data [11].

The target output in visual recognition ranges from image-level labels (object/image classification) [18], locations of objects in the form of bounding boxes (object detection) [14], to object segmentation [4, 40] or even predicting an approximate 3D pose and geometry of objects [20, 34]. In this work, we focus on predicting accurate image-level labels indicating presence/absence of objects. In addition, we provide qualitative evidence (see Section 6 and **additional results on the project webpage [1]**) that the developed system can localize objects and their discriminative parts in both the training and test images.

Initial work [2, 7, 8, 16, 39] on weakly supervised object localization has focused on learning from images containing prominent and centered objects in images with limited background clutter. More recent efforts attempt to learn from images containing multiple objects embedded in complex scenes [3, 10, 29] or from video [31]. These methods typically localize objects with visually consistent appearance in the training data that often contains multiple objects in different spatial configurations and cluttered backgrounds. While these works are promising, their performance is still far from the fully supervised methods.

Our work is related to recent methods that find distinctive mid-level object parts for scene and object recognition in unsupervised [35] or weakly supervised [11, 21] settings. The proposed method can also be seen as a variant of Multiple Instance Learning [38] if we refer to each image as a “bag” and treat each image window as a “sample”.

In contrast to the above methods we develop a weakly supervised learning method based on convolutional neural networks (CNNs) [23, 25]. Convolutional neural networks have recently demonstrated excellent performance on a number of visual recognition tasks that include classification of entire images [12, 22, 42], predicting presence/absence of objects in cluttered scenes [5, 27, 32, 33] or localizing objects by bounding boxes [17, 33]. However, the current CNN architectures assume in training a single prominent object in the image with limited background clutter [12, 22, 33, 42] or require fully annotated object locations in the image [17, 27]. Learning from images containing multiple objects in cluttered scenes with only weak object presence/absence labels has been so far limited to representing entire images without explicitly searching for location of individual objects [5, 32, 42], though some level of robustness to the scale and position of objects is gained by jittering.

In this work, we develop a weakly supervised convolutional neural network pipeline that learns from complex scenes containing multiple objects by explicitly searching over possible object locations and scales in the image. We demonstrate that our weakly supervised approach achieves the best published result on the Pascal VOC 2012 object classification dataset outperforming methods training from entire images [5, 32, 42] as well as performing on par or better than fully supervised methods [27].

3 Network architecture for weakly supervised learning

We build on the fully supervised network architecture of [27] that consists of five convolutional and four fully connected layers and assumes as input a fixed-size image patch containing a single relatively tightly cropped object. To adapt this architecture to weakly supervised learning we introduce the following three modifications. First, we treat the fully connected layers as convolutions, which allows us to deal with nearly arbitrary-sized images as input. Second, we explicitly search for the highest scoring object position in the image by adding a single global max-pooling layer at the output. Third, we use a cost function that can explicitly model multiple objects present in the image. The three modifications are discussed next and the network architecture is illustrated in Figure 2.

Convolutional adaptation layers. The network architecture of [27] assumes a fixed-size image patch of 224×224 RGB pixels as input and outputs a $1 \times 1 \times N$ vector of per-class scores as output, where N is the number of classes. The aim is to apply the network to bigger images in a sliding window manner thus extending its output to $n \times m \times N$ where n and m denote the number of sliding window positions in the x- and y- direction in the image, respectively, computing the N per-class scores at all input window positions. While this type of sliding was performed in [27] by

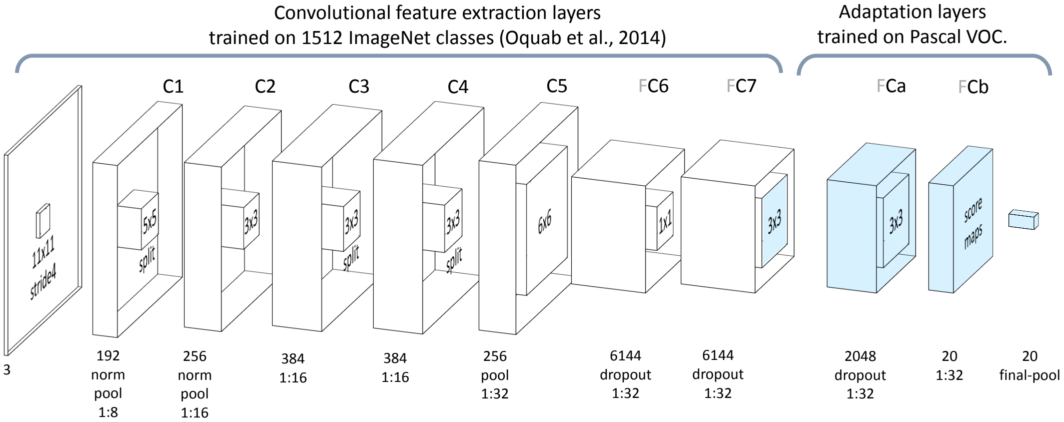


Figure 2: Network architecture. The layer legend indicates the number of maps, whether the layer performs cross-map normalization (norm), pooling (pool), dropout (dropout), and reports its subsampling ratio with respect to the input image. See [22, 27] and Section 3 for full details.

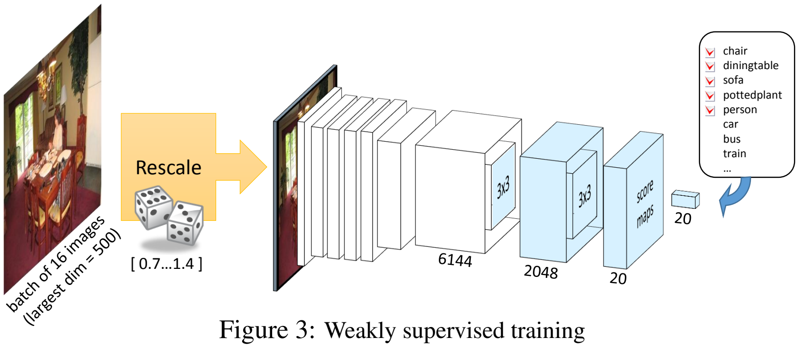


Figure 3: Weakly supervised training

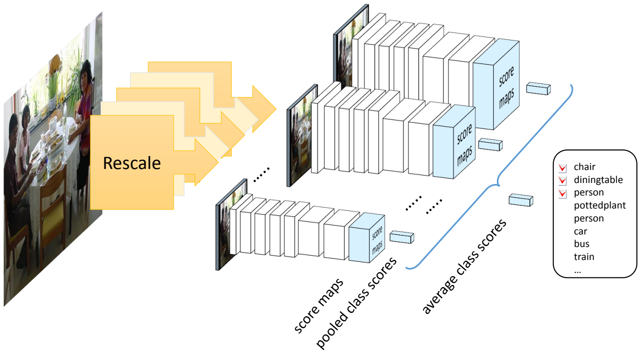


Figure 4: Multiscale object recognition

applying the network to independently extracted image patches, here we achieve the same effect by treating the fully connected adaptation layers as convolutions. For a given input image size, the fully connected layer can be seen as a special case of a convolution layer where the size of the kernel is equal to the size of the layer input. With this procedure the output of the final adaptation layer FC7 becomes a $2 \times 2 \times N$ output score map for a 256×256 RGB input image, as shown in Figure 2. As the global stride of the network is 32 pixels, adding 32 pixels to the image width or height increases the width or height of the output score map by one. Hence, for example, a 2048×1024 pixel input would lead to a 58×26 output score map containing the score of the network for all classes for the different locations of the input 224×224 window with a stride of 32 pixels. While this architecture is typically used for efficient classification at test time, see e.g. [33], here we also use it at training time (as discussed in Section 4) to efficiently examine the entire image for possible locations of the object during weakly supervised training.

Explicit search for object's position via max-pooling. The aim is to output a single image-level score for each of the object classes independently of the input image size. This is achieved by aggregating the $n \times m \times N$ matrix of output scores for $n \times m$ different positions of the input window using a global max-pooling operation into a single $1 \times 1 \times N$ vector, where N is the number of classes. Note that the max-pooling operation effectively searches for the best-scoring candidate object position within the image, which is crucial for weakly supervised learning where the exact position of the object within the image is not given at training. In addition, due to the max-pooling operation the output of the network becomes independent of the size of the input image, which will be used for multi-scale learning in Section 4.

Multi-label classification cost function. The Pascal VOC classification task consists in telling whether at least one instance of a class is present in the image or not. We treat the task as a separate binary classification problem for each class. The loss function is therefore a sum of twenty log-loss functions, one for each of the $\textsc{k} = 20$ classes $k \in \{1 \cdots \textsc{k}\}$,

$$\ell(\, f_k(\mathbf{x})\,,\, y_k\,) \;=\; \sum_k \log(1 + e^{-y_k f_k(\mathbf{x})})\,, \tag{1}$$

where $f_k(\mathbf{x})$ is the output of the network for input image $\mathbf{x}$ and $y_k \in \{-1, 1\}$ is the image label indicating the absence/presence of class k in the input image $\mathbf{x}$. Each class score $f_k(\mathbf{x})$ can be interpreted as a posterior probability indicating the presence of class k in image $\mathbf{x}$ with transformation

$$P(k|\mathbf{x}) \approx \frac{1}{1 + e^{-f_k(\mathbf{x})}}\,. \tag{2}$$

Treating a multi-label classification problem as twenty independent classification problems is often inadequate because it does not model label correlations. This is not a problem here because the twenty classifiers share hidden layers and therefore are not independent. Such a network can model label correlations by tuning the overlap of the hidden state distribution given each label.

4 Weakly supervised learning and classification

In this section we describe details of the training procedure. Similar to [27] we pre-train the convolutional feature extraction layers C1-C7 on images of 1512 classes from the ImageNet dataset and keep their weights fixed. This pre-training procedure is standard and similar to [22]. Next, the goal is to train the adaptation layers Ca and Cb using the Pascal VOC images in a weakly supervised manner, i.e. from image-level labels indicating the presence/absence of the object in the image, but not telling the actual position and scale of the object. This is achieved by stochastic gradient descent training using the network architecture and cost function described in Section 3, which explicitly searches for the best candidate position of the object in the image using the global max-pooling operation. We also search over object scales by training from images of different sizes. The training procedure is illustrated in Figure 3. Details and further discussion are given next.

Stochastic gradient descent with global max-pooling. The global max-pooling operation ensures that the training error backpropagates only to the network weights corresponding to the highest-scoring window in the image. In other words, the max-pooling operation hypothesizes the location of the object in the image at the position with the maximum score. If the image-level label is positive (i.e. the image contains the object) the back-propagated error will adapt the network weights so that the score of this particular window (and hence other similar-looking windows in the dataset) is increased. On the other hand, if the image-level label is negative (i.e. the image does not contain the object) the back-propagated error adapts the network weights so that the score of the highest-scoring window (and hence other similar-looking windows in the dataset) is decreased. For negative images, the max-pooling operation acts in a similar manner to hard-negative mining known to work well in training sliding window object detectors [14]. Note that there is no guarantee the location of the score maxima corresponds to the true location of the object in the image. However, the intuition is that the erroneous weight updates from the incorrectly localized objects will only have limited effect as in general they should not be consistent over the dataset.

Multi-scale sliding-window training. The above procedure assumes that the object scale (the size in pixels) is known and the input image is rescaled so that the object occupies an area that corresponds to the receptive field of the fully connected network layers. In general, however, the actual object size in the image is unknown. In fact, a single image can contain several different objects of different sizes. One possible solution would be to run multiple parallel networks for different image scales that share parameters and max-pool their outputs. We opt for a different less memory demanding solution. Instead, we train from images rescaled to multiple different sizes. The intuition is that if the object appears at the correct scale, the max-pooling operation correctly localizes the object in the image and correctly updates the network weights. When the object appears at the wrong scale the location of the maximum score may be incorrect. As discussed above, the network weight updates from incorrectly localized objects may only have limited negative effect on the results in practice.

In detail, all training images are first rescaled to have a largest side of size 500 pixels, zero-padded to 500×500 pixels and divided to mini-batches of 16 images. Each mini-batch is then resized by a scale factor s uniformly sampled between 0.7 and 1.4. This allows the network to see objects in the image at various scales. In addition, this type of multi-scale training also induces some scale-invariance in the network.

Classification. At test time we apply the same sliding window procedure at multiple finely sampled scales. In detail, the test image is first normalized to have its largest dimension equal to 500 pixels, padded by zeros to 500×500 pixels and then rescaled by a factor s that ranges between 0.5 and 3.7 with a step-size 0.05, which results in 66 different test scales. Scanning the image at large scales allows the network to find even very small objects. For each scale, the per-class scores are computed for all window positions and then max-pooled across the image. These raw per-class scores (before applying the soft-max function (2)) are then aggregated across all scales by averaging them into a single vector of per-class scores. The testing architecture is illustrated in Figure 4.

5 Implementation details

Our training architecture (Figure 3) relies on max-pooling the outputs of the convolutional network operating on a small batch of potentially large images. Several implementation details make this possible.

- In order to accomodate images of various sizes, all network layers are implemented as convolutions. Layers that were described as fully connected layers in [22, 27] are also viewed as convolutions (see Figure 2.)
- The GPU convolution code decomposes each convolution into an intricate sequence of cuBLAS[1] calls on adequately padded copies of the input image and kernel weights. Unlike previous "unfolded" convolution approaches [6], our scheme does not make multiple copies of the same input pixels and therefore consumes an amount of GPU memory comparable to that of the image itself. This implementation runs at least as fast as that of [22] without relying on large mini-batches and without consuming extra memory. This allows for larger images and larger networks.
- The training code performs fast bilinear image scaling using the GPU texture units.
- All the adaptation layers use dropout [22]. However, instead of zeroing the output of single neurons, we zero whole feature maps with probability 50% in order to decorrelate the gradients across different maps and prevent the coadaptation of the learned features.
- We set an independent learning rate for each network parameter using the Adagrad learning rate schedule [13]. Although training the entire CNN with Adagrad may not be straightforward, this procedure works well in our experiments because we only train the adaptation layers.

Our implementation takes the form of additional packages for the Torch7 environment.[2]

[1] http://docs.nvidia.com/cuda/cublas.

[2] http://torch.ch.

6 Experiments

In this section we first describe our experimental setup, evaluate the benefits of localizing objects at training, and compare classification performance of weak vs. strong supervision. Finally, we compare to other state-of-the-art methods, and show qualitative object localization results.

Experimental setup: Pascal VOC 2012 object classification. We apply the proposed method to the Pascal VOC 2012 object classification task. Following [27] the convolutional feature extraction layers are pre-trained on images of 1512 classes from the ImageNet dataset and kept fixed. The adaptation layers are trained on the Pascal VOC 2012 “train+val” set as described in Section 4. Evaluation is performed on the 2012 test set via the Pascal VOC evaluation server. The per-class performance is measured using average precision (the area under the precision-recall curve) and summarized across all classes using mean average precision (mAP). The per-class results are shown in Table 1.

Benefits of object localization during training. First we compare the proposed weakly supervised method (F. Weak supervision in Table 1) with training from full images (D. Full images), where no search for object location during training/testing is performed and images are presented to the network at a single scale. Otherwise the network architectures are identical. The results clearly demonstrate the benefits of sliding window multi-scale training attempting to localize the objects in the training data. Note that this type of full-image training is similar to the setup used by Zeiler and Fergus [42] (A.), Chatfield *et al.* [5] (C.) or Razavian *et al.* [32] (not shown in the table), though their network architectures differ in some details.

Strong vs. weak supervision. Having seen the importance of localizing the objects and their discriminative parts in the training data we next evaluate the importance of strong supervision, i.e. is it beneficial to provide the bounding box supervision during training? To answer this question we augment the weakly supervised setup (F) training data with tightly cropped images around the object bounding boxes. The aim is to help the network localize objects while benefiting from all negative image windows not containing the object class. Perhaps surprisingly, the results of this method (E. Strong+weak) are on par with the weakly supervised only training (F. weak supervision), which indicates there is no additional benefit in providing the detailed bounding box supervision on top of the image-level labels. Furthermore, our weakly supervised setup also significantly outperforms the method of Oquab *et al.* [27] (B.) that uses bounding box supervision and is subject to the biases in the bounding box annotation.

Comparison to other work. Table 1 also shows performance of three other recent competing CNN methods that report results on the Pascal VOC 2012 test data. The results clearly demonstrate the benefits of our method, which yields the best published results on this data, improving the current state of the art from 83.2% mAP (reported by Chatfield et al. [5]) to 86.3%.

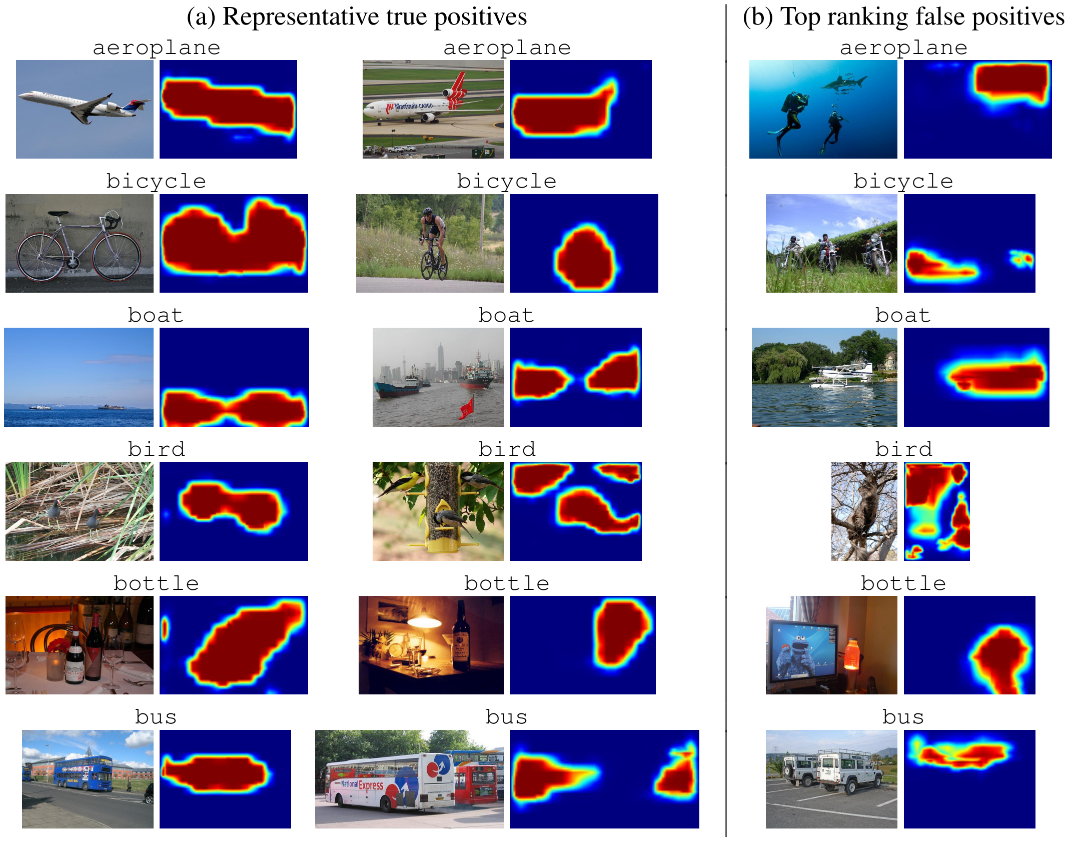
Qualitative localization results. Figure 5 shows examples of images from the Pascal VOC 2012 test set together with output response probability maps for selected object classes. In detail, these maps were obtained by taking the output of the network for scales between 1 and 2.5 with a step of 0.3, resizing them to the size of the image, performing the soft-max transform (2) and choosing the maximum value for each pixel across scales. The **supplementary material** on the project webpage [1] shows similar visualization for a large sample of images for each object class for both test and training data. These qualitative results clearly demonstrate the network can localize objects or at least their discriminative parts (e.g. the head for animals) in both the training and test images.

7 Conclusion

We have described an object recognition CNN trained without taking advantages of the object bounding boxes provided with the Pascal VOC training set. Despite this restriction, this CNN outperforms all previously published results in Pascal VOC classification. The network also provides qualitatively meaningful object localization information. Augmenting the training set with fully labeled examples brings no benefit and instead seems to slightly decrease the performance. Besides

	mAP	plane	bike	bird	boat	btl	bus	car	cat	chair	cow	table	dog	horse	moto	pers	plant	sheep	sofa	train	tv
A. ZEILER AND FERGUS [42]	79.0	96.0	77.1	88.4	85.5	55.8	85.8	78.6	91.2	65.0	74.4	67.7	87.8	86.0	85.1	90.9	52.2	83.6	61.1	91.8	76.1
B. OQUAB ET AL. [27]	82.8	94.6	82.9	88.2	84.1	60.3	89.0	84.4	90.7	72.1	86.8	69.0	92.1	93.4	88.6	96.1	64.3	86.6	62.3	91.1	79.8
C. CHATFIELD ET AL. [5]	83.2	**96.8**	82.5	91.5	**88.1**	62.1	88.3	81.9	**94.8**	70.3	80.2	76.2	92.9	90.3	89.3	95.2	57.4	83.6	66.4	93.5	81.9
D. FULL IMAGES (OUR)	78.7	95.3	77.4	85.6	83.1	49.9	86.7	77.7	87.2	67.1	79.4	73.5	85.3	90.3	85.6	92.7	47.8	81.5	63.4	91.4	74.1
E. STRONG+WEAK (OUR)	86.0	96.5	88.3	91.9	87.7	64.0	90.3	86.8	93.7	74.0	**89.8**	76.3	93.4	94.9	91.2	97.3	66.0	90.9	69.9	93.9	83.2
F. WEAK SUPERVISION (OUR)	**86.3**	96.7	**88.8**	**92.0**	87.4	**64.7**	**91.1**	**87.4**	94.4	**74.9**	89.2	**76.3**	**93.7**	**95.2**	**91.1**	**97.6**	**66.2**	**91.2**	**70.0**	**94.5**	**83.7**

Table 1: Per-class results for object classification on the VOC2012 test set (average precision %). Best results are shown in bold. Our weakly supervised setup outperforms the state-of-the-art on all but three object classes.


(a) Representative true positives

(b) Top ranking false positives

Figure 5: Output probability maps on representative images of several categories from the Pascal VOC 2012 test set. The rightmost column contains the highest-scoring false positive (according to our judgement) for each of these categories. Note that the proposed method provides an approximate localization of the object or its discriminative parts in the image despite being trained only from image-level labels without providing the location of the objects in the training data. **Please see more qualitative localization results for training and test images in the supplementary material on the project webpage [1].**

establishing a new state of the art, this result contributes to the discussion on the subjective nature of bounding box labels.

References

[1] http://www.di.ens.fr/willow/research/weakcnn/, 2014.

[2] H. Arora, N. Loeff, D. Forsyth, and N. Ahuja. Unsupervised segmentation of objects using efficient learning. In *CVPR*, 2007.

[3] M. Blaschko, A. Vedaldi, and A. Zisserman. Simultaneous object detection and ranking with weak supervision. In *NIPS*, 2010.

[4] T. Brox, L. Bourdev, S. Maji, and J. Malik. Object segmentation by alignment of poselet activations to image contours. In *CVPR*, 2011.

[5] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv:1405.3531v2*, 2014.

[6] K. Chellapilla, S. Puri, and P. Simard. High performance convolutional neural networks for document processing. In *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[7] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007.

[8] D. Crandall and D. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *ECCV*, 2006.

[9] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop*, 2004.

[10] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In *ECCV*, 2010.

[11] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A.A. Efros. What makes Paris look like Paris? *ACM Transactions on Graphics (TOG)*, 31(4):101, 2012.

[12] 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. *arXiv:1310.1531*, 2013.

[13] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159, 2011.

[14] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE PAMI*, 32(9):1627–1645, 2010.

[15] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[16] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[18] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *CVPR*, 2009.

[19] Hedi Harzallah, Frédéric Jurie, and Cordelia Schmid. Combining efficient object localization and image classification. In *CVPR*, 2009.

[20] M. Hejrati and D. Ramanan. Analyzing 3d objects in cluttered images. In *NIPS*, 2012.

[21] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, 2013.

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

[23] K.J. Lang and G.E. Hinton. A time delay neural network architecture for speech recognition. Technical Report CMU-CS-88-152, CMU, 1988.

[24] K.J. Lang, A.H. Waibel, and G.E. Hinton. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, 1990.

[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Winter 1989.

[26] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011.

[27] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

[28] V. Ordonez, G. Kulkarni, and T. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011.

[29] M. Pandey and S. Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*, 2011.

[30] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010.

[31] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012.

[32] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *arXiv preprint arXiv:1403.6382*, 2014.

[33] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv:1312.6229*, 2013.

[34] A. Shrivastava and A. Gupta. Building part-based object detectors via 3d geometry. In *ICCV*, 2013.

[35] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012.

[36] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering object categories in image collections. In *ICCV*, 2005.

[37] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[38] P. Viola, J. Platt, C. Zhang, et al. Multiple instance boosting for object detection. In *NIPS*, 2005.

[39] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, 2005.

[40] P. Yadollahpour, D. Batra, and G. Shakhnarovich. Discriminative re-ranking of diverse segmentations. In *CVPR*, 2013.

[41] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011.

[42] M. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *arXiv:1311.2901*, 2013.

[43] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *IJCV*, 73(2):213–238, jun 2007.