The reliance on plentiful and detailed manual annotations for training is a critical limitation of the current state of the art in object localization and detection. This paper introduces self-taught object localization, a novel approach that leverages deep convolutional networks trained for whole-image recognition to localize objects in images without additional human supervision, i.e., without using any ground-truth bounding boxes for training. The key idea is to analyze the change in the recognition scores when artificially masking out different regions of the image. The masking out of a region that contains an object typically causes a significant drop in recognition. This idea is embedded into an agglomerative clustering technique that generates self-taught localization hypotheses. For a small number of hypotheses, our object localization scheme yields a relative gain of more than 22% in both precision and recall with respect to the state of the art (BING and Selective Search) for top-1 subwindow proposal. Our experiments on a challenging dataset of 200 classes indicate that our automatically-generated annotations are accurate enough to train object detectors in a weakly-supervised fashion with recognition results remarkably close to those obtained by training on manually annotated bounding boxes.

1. Introduction

Object recognition is one of the fundamental open challenges of computer vision. The problem statement can take two subtly different forms: 1) whole-image classification [20], where the goal is to categorize a holistic representation of the image, and 2) detection [30], which instead aims at decomposing the image into a set of regions or subwindows individually tested for the presence of the target object. Object detection provides several benefits over holistic classification, including the ability to localize objects in the image, as well as robustness to irrelevant visual elements, such as uninformative background, clutter or the presence of other objects. However, while whole-image classifiers can be trained with image examples labeled merely with class information (e.g., “chair” or “pedestrian”), detectors require richer annotations consisting of manual selections specifying the region or the bounding box containing the target objects in each individual image example. Unfortunately, such detailed annotations are expensive and time-consuming to acquire. This effectively limits the applicability of detectors to scenarios involving only few categories (e.g., [16]). Furthermore, these manual selections are often rather subjective and noisy, and as such they do not provide optimal ground truth regions for training detectors.

In this paper we introduce self-taught object localization, a novel approach to localize objects with minimal human supervision by leveraging the output of a whole-image classifier, trained without object location information. The key idea is to analyze how the recognition score of the classifier varies as we artificially mask-out regions in the image. Intuitively, when the region containing the object is artificially occluded the whole-image classification score will drop significantly. Fig. 1 shows how the partial masking out of the input is propagated through the deep convolutional network and how this affects the recognition score. We embed this idea into a hierarchical clustering technique that merges regions according to their relative drop in classification score in addition to criteria of feature and size similarity, and spatial vicinity. This produces for each image a set of subwindows that are deemed likely to contain the object.

We investigate the use of these subwindow proposals as training data to learn a pool of weakly-supervised ob-
ject detectors, which focus on recognizing the regions most likely to contain the object. By doing so we effectively replace the traditional manually-selected bounding boxes with regions automatically estimated from training images annotated only with image-level labels, which are easy to obtain even for a large number of training images. This framework enables scalable training of object detectors at a much reduced human cost, since no manual annotation of regions is needed. Besides the reduced labeling effort, we demonstrate in our experiments that detectors trained on our subwindow proposals achieve recognition accuracy surprisingly close to that obtained using ground-truth bounding boxes as training data.

An important aspect of our approach is the choice of whole-image classifier used to bootstrap the training of the detector via self-taught localization. We exploit the power of deep networks since, although they perform holistic image recognition, they have been shown to be impressively accurate even in presence of clutter and multiple objects [20]. In addition, deep networks are particularly suited for our strategy because they operate directly on pixels, which we mask out, in contrast to other models (such as those based on bag of visual words) that discard the spatial information. Another advantage is that the intermediate features learned for whole-image classification can be utilized as the representation for the subwindow proposals used to train the object detectors [16].

Our experiments on the ILSVRC2012 dataset [1] show a relative increment of more than 22% in terms of top-1 precision and recall with respect to the state of the art of the subwindow proposal methods. We also show that our self-taught localization model trained on the ILSVRC2012 classes is able to generalize effectively to new categories, by yielding state-of-the-art subwindow proposal results on the unseen PASCAL 2007 dataset [12]. Finally, we demonstrate that the subwindows automatically-generated by our approach can be used as positive training examples to learn object detectors without any additional human supervision. Our results on 200 classes of ILSVRC2012 are close to those obtained with detectors trained on manually-annotated bounding boxes.

2. Related Work

In recent years deep networks have gained great popularity due to their outstanding performance on different tasks, including large-scale image categorization [20, 25], face verification [29], and video recognition [18]. In the context of image recognition, several researchers have attempted to apply deep networks to object localization and detection problems [16, 23, 28, 11]. In [16], a convolutional network [20] is fine-tuned on ground truth bounding boxes and then applied to classify subwindows generated by the region proposal algorithm of Uijlings et al. [30]. In [23, 11, 28] a convolutional network is trained to directly perform regression on the vector-space of all bounding boxes of an image in order to avoid the high computational cost of traditional sliding window or region proposal approaches. These deep networks have shown promising results compared to standard detection schemes relying on hand-crafted features (e.g., [14, 30]). However, all of the aforementioned approaches require manually-annotated ground truth bounding boxes as training data. In contrast, our method populates the images in the training set with automatically-generated bounding boxes, which can be exploited for the automatic learning of detectors.

The subwindow proposal methods presented in [2, 10, 30, 7, 5, 19] focus on generating bounding boxes yielding high recall, i.e., maximizing the probability that each object in the image is covered by at least one subwindow. These methods are typically employed at testing time to replace the classic but computationally expensive sliding window approach. Although they provide the great benefit of speeding up detection at test time, these subwindow proposals cannot be used in lieu of ground truth bounding boxes to train a detector because of their low precision caused by the presence of many false positives. In addition, most of these proposal techniques [2, 7, 19] require ground truth bounding boxes during training, thus effectively increasing the amount of manual annotations needed to train a recognition system. Our algorithm can also be viewed as a subwindow proposal method but it provides precision much superior to that of prior methods: detectors trained on our automatically-generated bounding boxes perform nearly on par with detectors learned from ground-truth annotations.

Most weakly-supervised object detection methods [9, 8, 24, 27, 3] aim at jointly learning and inferring both the class and the position of the objects, which is a very hard learning problem potentially leading to poor local minima. A multiple instance learning framework was proposed in [8, 3], and similarly in [27] using latent SVMs, where the latent variable is the location of the object. We do not attempt to estimate both class and location within a single training stage. Instead, we leverage the powerful deep network trained with only image-level labels to auto-generate training bounding boxes for weakly-supervised detection.

Even though deep networks have shown impressive results, there is still little understanding of what are the critical factors contributing to their outstanding performance. Recently, much work has been devoted to better comprehend deep networks, most often through visualizations of the learned intermediate representations [31, 26] or through semantic interpretation of individual units [21]. The analysis of what a classifier learns have been also investigated in the standard pipeline of bag of feature and SVM [22]. Instead, we study the effects of selectively masking out the input of deep networks giving new insights on what the net-
work has learned during training and how this can be exploited for object localization.

We note that while the approach in [31] also investigates the correlation between occlusion of image regions and classification score, it is exclusively focused on the task of analyzing and visualizing the learned features (no quantitative results are reported on the task of localization or detection). Our idea is complementary to this previous work: we want to exploit the mask-out mechanism not simply as a feature analysis tool but also as an effective procedure to perform object localization. In our experiments we adapted the occlusion-box strategy of [31] to the problem of object localization but we found that this yields much poorer results compared to our approach (for details see Section 3.2). Simonyan et al. [26] proposed an approach that computes an image-specific saliency map by identifying the pixels that are most useful to predict the classification score of a deep network. However, the saliency map is dependent on the class label. Instead, in our experiments we demonstrate that our approach provides state-of-the-art results even when used without class labels to compute sub-window proposals for objects of arbitrary classes.

3. Self-Taught Object Localization

The aim of Self-Taught Localization (in brief STL) is to generate bounding boxes that are very likely to contain objects. The proposed approach relies on the idea of masking out regions of an image provided as input to a deep network (Sec. 3.1). The drop in recognition score caused by masking an image is embedded into an agglomerative clustering method which merges regions for object localization (Sec. 3.2).

3.1. Input Mask-out

Let us assume to have a deep network $f : \mathbb{R}^N \rightarrow \mathbb{R}^C$ that maps an input image $x \in \mathbb{R}^N$ of $N$ pixels to a confidence vector $y \in \mathbb{R}^C$ of $C$ classes. The confidence vector is defined as $y = [y_1, y_2, \ldots, y_C]^T$, where $y_i$ corresponds to the classification score of the $i$-th class.

We propose to mask out the input image $x$ by replacing the pixel values in a given rectangular region of the image $b = [b_x, b_y, w, h] \in \mathbb{N}^4$ with the 3-dimensional vector $g$ (one dimension for each image channel), where $b_x$ and $b_y$ are the $x$ and $y$ coordinates and $w$ and $h$ are the width and height, respectively. The masking vector $g$ is learned from a training set as the mean value of the individual image channels. We denote the function that masks out the image $x$ given the region $b$ using the vector $g$ as $h_g : \mathbb{R}^N \times \mathbb{N}^4 \rightarrow \mathbb{R}^N$. Please note that the output of the function is again an image (see Fig. 1).

We then define the variation in classification score of the image $x$ subject to the masking out of a bounding box $b$ as the output value of function $\delta_f : \mathbb{R}^N \times \mathbb{N}^4 \rightarrow \mathbb{R}^C$ defined as

$$\delta_f(x, b) = \max(f(x) - f(h_g(x, b)), 0) \quad (1)$$

where the max and the difference operators are applied component-wise. This function compares the classification scores of the original image to those of the masked-out image. Intuitively, if the difference for the $c$-th class is large, the masked-out region is very discriminative for that class. Therefore the region $b$ is deemed likely to contain the object of class $c$.

We use the function $\delta_f$ to define two variants of drop in classification score, depending on the availability of class label information for the image. When the ground truth image-level class label $c$ of $x$ is provided, we define the classification drop function $d_{\text{CL}} : \mathbb{R}^N \times \mathbb{N}^4 \rightarrow \mathbb{R}$ as

$$d_{\text{CL}}(x, b) = \delta_f(x, b)^T \mathbb{I}_c, \quad (2)$$

where $\mathbb{I}_c \in \mathbb{R}^C$ is an indicator vector with 1 at the $c$-th position and zeros elsewhere. This drop function enables us to generate class-specific window proposals in order to populate a training set with bounding boxes likely to contain instances of class $c$. We denote the method that uses $d_{\text{CL}}$ as STL-CL.

If the class information is not available, for example when testing a detector, we use the top-$C$ classes predicted by the whole-image classifier $f$ to define $d_U : \mathbb{R}^N \times \mathbb{N}^4 \rightarrow \mathbb{R}$ as

$$d_U(x, b) = \delta_f(x, b)^T \mathbb{I}_{\text{top}-C}, \quad (3)$$

where $\mathbb{I}_{\text{top}-C} \in \mathbb{R}^C$ is an indicator vector with ones at the top-$C$ predictions for the image $x$ and zeros elsewhere. Since the function is not using the class label, the setup is unsupervised and as a consequence class-agnostic. In practice, we used the top-5 predictions of the deep network $f$ applied to the whole image, because it has been shown [20] that the probability of getting the correct image-level class is 18%. The STL method that uses $d_U$ is named STL-U.

As deep convolutional network $f$ we adopt the model introduced in [20] which has been proven to be very effective for image classification. Since the network is applied to mean-centered data, replacing a region of the image with the learned mean RGB value is effectively equivalent to zero-ing out that section of the network input as well as the corresponding units in the hidden convolutional layers (see Fig. 1). We want to point out that our masking-out approach is general and it can be applied to any other classifier that operates on raw pixels.

3.2. Agglomerative Clustering

The initialization of the proposed agglomerative clustering consists of a set of $K$ rectangular regions
\{b_1, b_2, \ldots, b_K\}$ generated for an image $x$ using the segmentation method proposed in [15]. Note that in practice we mask out the rectangular bounding boxes enclosing the segments rather than the segments themselves. We found experimentally that if we mask out the segments, the shape information of the segment is preserved and used by the network to perform recognition, thus causing less substantial drops in classification.

The goal of the agglomerative clustering is to fuse regions (bottom-up) and generate windows that are likely to contain objects (top-down). We propose an iterative method that greedily compares the available regions, and at each iteration merges the two regions that maximize the similarity function discussed below. This procedure terminates when only one region (covering the whole image) is left. The set of generated subwindows are then sorted accordingly to the drop in classification (Eq. 2 for STL$_{CL}$ and Eq. 3 for STL$_U$). We also perform non-maximum suppression of the subwindows with overlap more than 50%.

We define the similarity between regions using four terms capturing the intuitions expressed below. Two bounding boxes are likely to contain parts of the same object if

1. they cause similar large drops in classification score:
   \[
   s_{\text{drop}}(x, b_i, b_j) = 1 - \left| d_m(x, b_i) - d_m(x, b_j) \right|, \quad \max (1 - d_m(x, b_i), 1 - d_m(x, b_j))
   \]

2. they are similar in appearance:
   \[
   s_{\text{app}}(x, b_i, b_j) = z(\phi(x, b_i), \phi(x, b_j))
   \]

3. they cover the image as much as possible, encouraging small windows to merge early:
   \[
   s_{\text{size}}(x, b_i, b_j) = 1 - \frac{\text{size}(b_i) + \text{size}(b_j)}{\text{size}(x)}
   \]

4. they are spatially near each other:
   \[
   s_{\text{fill}}(x, b_i, b_j) = 1 - \frac{\text{size}(b_i \cup b_j) - \text{size}(b_i) - \text{size}(b_j)}{\text{size}(x)}
   \]

where the index $m \in \{\text{CL, U}\}$ in the first term selects STL$_{CL}$ or STL$_U$ presented in Sec. 3.1, $z(\cdot, \cdot)$ is the histogram intersection similarity between the network features extracted by $\phi(\cdot, \cdot)$ (see Sec. 5 for details), $b_i \cup b_j$ is the bounding box that contains $b_i$ and $b_j$. The overall similarity score $s$ is defined as a convex combination of the terms above:

\[
\begin{equation}
   s(b_i, b_j, x) = \sum_{l \in \mathcal{L}} \alpha_l s_l(b_i, b_j, x),
\end{equation}
\]

where $\mathcal{L} = \{\text{drop, app, size, fill}\}$ and the $\alpha$s are set to be uniform weights in our experiments. We empirically found that removing $s_{\text{drop}}$ from Eq. 4 will cause a drop of 8% and 10% in terms of precision and recall, respectively.

Figure 2 illustrates the intuition behind the similarity measure encoded by $s_{\text{drop}}$. This similarity is large if the two regions exhibit similar classification drops when occluded (corresponding to points on the diagonal of the $xy$-plane in the 3D plot) and it is especially large when the drop in score is substantial (points close to $(1, 1)$ in the plot). The term $s_{\text{app}}$ encourages aggregation of regions similar in appearance, while $s_{\text{size}}$ and $s_{\text{fill}}$ borrowed from [30] favor early merging of small regions and regions that are near each other, respectively.

There are many advantages of the proposed similarity with respect to [30]. First, it does not rely on hand-engineered features like SIFT, but instead it leverages the features learned by the deep network. Moreover, our similarity exploits the discriminant power of the deep convolutional network enabling our method to generate class-specific window proposals. Even when used in the class-agnostic regime of Eq. 4 it will tend to generate subwindows that are most informative for recognition (since their occlusion causes large classification drops). Thus, our approach can be viewed as a hybrid scheme combining bottom-up cues (size, appearance) with top-down information (object-class recognition), unlike [30] where the merging of regions is driven by a pure bottom-up procedure.

### 4. Weakly-Supervised Detection using STL

We propose a weakly-supervised detection approach relying on our self-taught localizer. The main idea is to exploit the bounding boxes generated by STL$_{CL}$ (Sec. 3) as positive region examples when training the object detectors, thus eliminating the need for ground truth annotations.

Let us consider the training of an object detector for class $c$. Let us suppose to have a dataset of $M$ training images $\mathcal{D}_c = \{(x_i, B_i, y_i)\}_{i=1}^M$, where $x_i$ is the $i$-th image, $B_i = \{b_{i,j}\}_{j=1}^{K_i}$ is a set of $K_i$ bounding boxes for the $i$-th image and $y_i = \{y_{i,j}\}_{j=1}^{K_i}$ is a set of binary labels for detection, where $y_{i,j} = 1$ denotes a box containing an object of class $c$, and $y_{i,j} = -1$ indicates a negative box.
Each object detector is a function that takes a subwindow \( b_{i,j} \), extracts a feature representation \( \phi(x_i, b_{i,j}) \) from the bounding box of the image, and maps it into a score measuring the confidence that \( b_{i,j} \) contains an object of class \( c \). Inspired by [16], we use as representation \( \phi \) the features extracted by the last fully-connected layer (before the softmax) of the deep convolutional network. We refer the reader to [16] for additional details.

The object detector is iteratively trained using the hard negative mining procedure described in [30]. While this prior work exploited manually-annotated bounding boxes as the positive examples, in our training procedure we replace the ground truth regions with the bounding boxes \( B_i \) produced by the class-specific STL-CL on training images of class \( c \), i.e., we use the class label information for localization of the positive regions.

The negative set is built using the bounding boxes that overlap less than 30% with any STL-CL subwindows \( B_i \) from the positive images, and one randomly-chosen bounding box from each negative image. At each iteration, a linear SVM [13] is trained by automatically choosing the hyper-parameter with a 5-fold cross-validation that maximizes the average precision. The negative set is augmented for the next training iteration by adding for each negative image the bounding box with the highest positive score.

At testing time, each detector is tested on the generated subwindows of a given image, the detection scores are sorted and then pruned via non-maximum suppression: we remove a subwindow if it overlaps for more than 70% with a subwindow that has higher score. In our experiments we tested the proposed STL-U, and Selective Search [30] as methods to generate the bounding boxes for the negative set and for the testing images.

5. Experiments

In this section we present comparative results of our approach with state-of-the-art methods on the task of self-taught localization (Sec. 5.2) and weakly-supervised detection (Sec. 5.3).

5.1. Implementation Details

In our experiments, we used the convolutional network software Caffe [17] for classification \( f \) and feature extraction \( \phi \) with the model trained on ILSVRC-2012 provided by the authors. Note that the training of this network did not use the bounding-box annotations.

5.2. Self-taught Localization

Given a test image, the goal is to generate a set of bounding boxes that enclose the objects of interest with high probability. We consider a bounding box whose intersection over union with the ground truth is at least 50% as true positive, as done in the PASCAL benchmark [12]. The performance is then measured in terms of the mean of the average recall and precision per class, following the evaluation protocol of [29]. We note that the mean of the average recall (or detection rate) per image used in [17] is far less common and prone to issues when the classes in the test set are unbalanced. Thus it is not considered here.

The proposed STL technique introduced in Sec. 3 was tested on two challenging benchmarks: ILSVRC-2012-LOC [11] and PASCAL-VOC-2007 [12]. ILSVRC-2012-LOC is a large-scale benchmark for visual object localization containing 1000 categories. The training set contains 544,546 images with 619,207 annotated bounding boxes. The validation set contains 50,000 images for a total of 76,750 annotated bounding boxes. We indicate with ILSVRC-2012-LOC-200 a subset of 200 randomly-chosen categories that is also used for the weakly-supervised detection task (Sec. 5.3). PASCAL-VOC-2007 contains 20 categories, for a total of 9,963 images divided into training, validation and testing splits. Each image contains multiple objects belonging to different categories at different positions and scales, for a total of 24,640 ground truth bounding boxes.

First of all, we show in Fig. 3 that the difference between the results of our method on ILSVRC-2012-LOC (1000 classes) and the ones on ILSVRC-2012-LOC-200 (200 classes) are negligible. This indicates that the accuracy on the set of 200 classes is representative of the performance on the larger set. For this reason, in all our experiments of weakly-supervised detection (Sec. 5.3) we used the smaller ILSVRC-2012-LOC-200 datasets, as this allowed us to perform faster training and testing, thus enabling a more comprehensive study of the different variants of our method and previously proposed algorithms.

We then compared STL against Selective Search.

---

1To enable future comparisons with our results, we will make publicly available the list of 200 classes.
Figure 4. Comparison of different bounding-box proposal methods. The first row reports the mean recall per class, as a function of the number of proposed subwindows. The second row reports the mean precision per class. The first column shows the evaluation for ILSVRC-2012-LOC-200 (training), which is part of the data used to train the deep network. Note that both our STL-CL and STL-U outperform all the competitors for the first 100 subwindows. The second column contains the evaluations on ILSVRC-2012-LOC (validation), which shows the capability of our methods to generalize to unseen images. Finally, the third column reports the results on never-seen categories of PASCAL-VOC-2007 (test).

(SelSearch) [30] (fast version) and BING [7] (MAXBGR version), which are widely acknowledged as the state-of-the-art in class-agnostic subwindow proposal. We also include as baseline method the conventional sliding window strategy commonly used in detection: a set of rectangles of different sizes is slid over the image and at each position we compute the confidence score as the sum of the classification scores of the top-5 classes predicted by the deep network. The set of subwindows is generated by sliding a square box across the image, using 7 different scales. This produces a number of subwindows that is comparable to the one produced by our method, thus yielding a similar computational cost for detection. This comparison is important as it shows the performance obtained with a subwindow sampling strategy, as opposed to using a system that selects subwindows based on image content.

Figure 4(a,b) show the results in terms of recall and precision on ILSVRC-2012-LOC-200 (training set) and ILSVRC-2012-LOC (validation set) respectively. Note that while ILSVRC-2012-LOC-200 (training) includes images that were used as training examples to learn the deep network, ILSVRC-2012-LOC (validation) does not and thus it is useful to assess the ability of STL to work on new images not seen during the training of the whole-image classifier. In Figure 4(a) one can see that our method significantly outperforms all the other methods for the first 100 proposed subwindows: +126% and +22% relative improvement of recall over SelSearch and BING for top-1 bounding box, respectively. The sliding window approach performs very poorly (−46% in absolute value for top-1 bounding boxes) demonstrating the need for a method that generates bounding boxes of appropriate shape, size and position rather than based on a fixed grid and scale. Note that the performance difference between using the class label of the image (STL-CL) and not using it (STL-U) is negligible. This indicates that our STL approach works equally well even when not given the class label information. Fig. 4(b) shows consistent performance on ILSVRC-2012-LOC (validation), i.e., on images not used for the training of the deep network. Our methods is the best also in this case, indicating that it naturally generalizes to unseen examples.

We have also experimented with a simple baseline that slides occlusion boxes of different scale over a fixed grid of the image and uses the corresponding drops in classification score to localize the object. This is reminiscent of the approach described in [31], which was used for for feature visualization but not for localization. However, we found that even when using a number of occlusion boxes much larger...
than that used by our method (which implies a higher computational cost), the localization results were much poorer, e.g., recall at rank 1 on the ILSVRC-2012-LOC validation dataset is 18% lower than that produced by our method.

The methods SELSEARCH and BING were designed to obtain high recall when using a large number of proposals, which is a desirable property at testing time. However it yields precision not sufficiently high to train a detector. In contrast, STL is by far the best method in term of both precision and recall for a small number of proposals. Although they shows higher recall when using more than 100 bounding boxes, STL remains competitive even at this regime, which is not its intended application domain.

We finally show the capability of our method to generalize to unseen datasets and classes, using the PASCAL-VOC-2007 benchmark [12]. The images of this set have very different statistics than the ones in ILSVRC-2012-LOC, as each image can contain multiple objects belonging to different categories. Moreover, we point out that our classification network was neither trained nor fine-tuned on this dataset. Nevertheless, as shown in Fig. 4(c), our method is able to generalize to this new scenario, outperforming again all the compared methods for small number of proposals. This remarkable result shows that STL performs well on arbitrary classes, as the categories of PASCAL-VOC-2007 do not exactly correspond to classes present in ILSVRC-2012-LOC.

### 5.3. Weakly-supervised Detection

In this section we analyze the effect of training a set of object detectors using the subwindows automatically detected by self-taught localization. To this aim, we trained 200 detectors (one for each class in ILSVRC-2012-LOC-200), using for each a training set of 50 positive images and 4,975 negative images (obtained by sampling 25 examples from each negative class). The test set is composed by 10,000 images of the ILSVRC-2012-LOC validation set.

The proposed approach is trained in the weakly-supervised setup by selecting top-$K$ bounding boxes from each positive image and by mining negative boxes according to the procedure described in Sec. 4. We experimented with different combinations of proposal methods for the positive and the negative boxes. For each combination, at test time on each input image we used the same proposal method that was applied to generate the negative boxes during training. For all combinations we set $K = 3$ to obtain a good recall/precision trade-off based on the results of Figure 4. The first combination that we consider involves $STL_{KL} + STU$, i.e., using our proposal method based on class labels to generate the positive boxes and our unsupervised approach to produce the negative boxes as well as the proposals on the test images. We compare this combination with three other schemes: $BING + BING$, $SELSEARCH + SELSEARCH$ and $GROUNDTRUTH + SELSEARCH$. This latter combination is the fully-supervised approach based on manually-annotated bounding boxes as proposed in [30].

Table 1 shows the results in terms of mean average precision (mAP) across all 200 classes for each method. The first row reports the method used to generate the positive training boxes, the second row indicated the method for the negative and test boxes. From these results we notice that $STL_{KL} + STU$ (second column) outperforms $BING$ (third column) and $SELSEARCH$ (forth column), yielding a relative improvement of 42% and 7%, respectively. This suggests that $STL_{KL}$ generates reliable bounding boxes for the positive set.

An interesting observation that we can draw from Figure 4(a) is that while $STL_{KL}$ produces the highest precision for a small number of subwindows (and therefore it is preferred method to generate the positive boxes), instead $SELSEARCH$ yields higher recall for large numbers of proposals. This suggests that using SELSEARCH for the negative and test images can be advantageous. Based on this observation, we performed an experiment where we tested “the best of the two worlds”, i.e., using $STL_{KL}$ to generate the positive set and $SELSEARCH$ for the negative and test set (Table 1 fifth column). Since $BING$ also produces fairly high precision, we tested $BING + SELSEARCH$ for comparison (Table 1 sixth column). We see that $STL_{KL} + SELSEARCH$ outperforms both $BING + SELSEARCH$ and $SELSEARCH + SELSEARCH$ by a significant margin, yielding a relative improvement of 4.5% and 11.6% over these two combinations, respectively. $STL_{KL} + SELSEARCH$ shows a relative drop in performance of only 19.6% with respect to the fully supervised method (last column). This is a remarkable result given that our method uses only class labels. Finally, we also tested $STL_{KL} + SELSEARCH$ with $K = 1$ obtaining a mAP of 20.93%, which reduces to 17.6% the relative gap with respect to the fully-supervised method.

Table 2 shows the best-10 and worst-5 classes for each method along with the respective APs. It is interesting to notice that 8 out of the 10 best categories are shared be-

| Positive Boxes + Negative/Test Boxes | STL_{KL} (our method) + STU | BING [7] + BING | SELSEARCH [30] + SELSEARCH | STL_{KL} (our method) + SELSEARCH | BING [7] + SELSEARCH | Ground truth + SELSEARCH |
|-------------------------------------|---------------------------|----------------|--------------------------|-------------------------------|-------------------|-------------------------|
| mAP                                | 19.60                     | 13.78          | 20.45                    | 19.55                         | 25.40             |                         |

Table 1. Each column contains the mean Average Precision (%) calculated as the mean across all 200 classes for ILSVRC-2012-LOC-200. It is also reported which method is used to generate the bounding boxes the positive set, the negative set and during testing.
Table 2. Each column contains the best classes (blue) and the worst classes (red) for the object-detectors trained using the annotation method listed at the top. All methods were trained on ILSVRC-2012-LOC-200. Average Precision (%) is listed for each class.

| Positive Boxes + Negative/Test Boxes | GROUNDTRUTH + SELSEARCH | STLCL (our method) + SELSEARCH | SELSEARCH + | BING + |
|-------------------------------------|-------------------------|--------------------------------|------------|--------|
| best classes                        | leopards & 65.28        | leopard & 62.86                 | leopard & 59.29 | leopard & 56.83 |
|                                    | Crock Pot = 62.60       | giant panda & 54.96             | car mirror & 50.86 | giant panda & 50.73 |
|                                    | teapot & 59.12          | car mirror & 51.69              | koala & 49.23   | koala & 49.50 |
|                                    | admiral & 58.55         | Crock Pot & 50.56               | koala & 44.96  | car mirror & 48.25 |
|                                    | car mirror & 58.28      | koala & 50.15                   | giant panda & 44.19 | orangutan & 46.76 |
|                                    | cabbage butterfly & 54.73 | police van & 48.37              | crib & 41.21    | pickup & 45.09 |
|                                    | frilled lizard & 52.10  | admiral & 46.24                 | bullfrog & 41.00 | admiral & 44.69 |
|                                    | police van & 51.86      | necklace & 46.05                | maze & 40.67    | frilled lizard & 44.57 |
|                                    | giant panda & 51.68     | pickup & 45.75                  | orangutan & 46.66 | entertainment center & 43.04 |
| worst classes                      | punching bag & 0.65     | Crock Pot & 0.15                | punching bag & 0.12 | flute & 0.21 |
|                                    | hair spray & 0.54       | basketball & 0.13               | hair spray & 0.03 | punching bag & 0.21 |
|                                    | screwdriver & 0.41      | pole & 0.10                     | basketball & 0.01 | swimming trunks & 0.16 |
|                                    | nail & 0.10             | nail & 0.04                     | pole & 0.01     | pole & 0.03 |
|                                    | pole & 0.05             | nail & 0.04                     | nail & 0.01     | basketball & 0.01 |

Figure 5. Average precision (AP) on the individual 200 classes obtained with the fully supervised approach GROUNDTRUTH+SELSEARCH (x-axis) and our weakly-supervised method STLCL+SELSEARCH (y-axis). Each point represents the AP of these two methods on one particular class.

W between the detectors trained on the ground truth annotations (second column) and our STLCL (third column) as opposed to 5 out of 10 of our competitors.

In Fig. 5 we report the AP on each individual class for the weakly-supervised method STLCL+SELSEARCH (y-axis) and the fully-supervised approach GROUNDTRUTH+SELSEARCH (x-axis). Quite surprisingly, for 41 classes (all points above the diagonal) the proposed method achieves accuracy better than that obtained when using ground truth annotations.

5.4. Analysis of Computational Costs

We analyze the computational cost of the STL method. Let $K$ be the number of segments produced by the method of [15]. During initialization, the similarity of Eq. [4] is evaluated for all segment pairs, for a total of $O(K^2)$ times. However, note that only $K$ evaluations of the convolutional network are needed, one for each masked-out segments.

At the first iteration of the clustering procedure, two of the segments are merged, and there will be $K-1$ remaining segments. Only the similarities involving the newly created segment are updated, which amount to $O(K)$ similarity evaluations, but these can be obtained with a single network evaluation of the image with only the newly merged segment masked-out. In every subsequent iteration, the total number of segments will decrease by one. Thus, in total only $2 \cdot K$ network evaluations are performed over the entire procedure, including those done at initialization.

In practice, the $2 \cdot K$ network evaluations of an image take about 120 seconds for typical values of $K$ used in our experiments. We stress that our code is written in Python and no effort in optimizing it has been made. SELSEARCH [30] and BING [7] are highly optimized and they take 4 and 0.003 seconds per image, respectively. Furthermore, SELSEARCH exploits speedup heuristics, such as merging only adjacent segments (which we could also adopt to make our code much faster). The methods of [4] and [6] take on average 64 and 432 seconds, respectively.

6. Conclusions

This work presents self-taught localization which aims at eliminating the reliance on manually annotated regions to train object detectors. The proposed approach leverages the discriminant power of convolutional networks trained on image-level class labels to automatically determine the subwindows that are most likely to contain the objects of interest. We tested STL on the task of object localization and window proposal and showed that it consistently outperforms the state of the art. We demonstrated that detectors trained on localization hypotheses automatically generated by STL achieve performance nearly comparable to those produced when training on manual bounding boxes. In future work we will investigate the possibility of fine-tuning the network as a localizer on the subwindows generated by STL and how to use them in a multiple instance learning framework in order to have more robust object detectors. We will also make the code of the proposed method publicly available upon acceptance of the paper.
References

[1] Large scale visual recognition challenge, 2012. http://www.image-net.org/challenges/LSVRC/2012/.

[2] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of windows. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34(11):2189–2202, 2012.

[3] K. Ali and K. Saenko. Confidence-rated multiple instance boosting for object detection. In IEEE CVPR, 2014.

[4] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 33(5):898–916, 2011.

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

[6] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In CVPR, pages 3241–3248, 2010.

[7] M. Cheng, Z. Zhang, W. Lin, and P. Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In IEEE CVPR, 2014.

[8] R. G. Cinbis, J. Verbeek, and C. Schmid. Multi-fold MIL Training for Weakly Supervised Object Localization. In IEEE CVPR, 2014.

[9] T. Deselaers, B. Alexe, and V. Ferrari. Weakly supervised localization and learning with generic knowledge. Int. J. Comput. Vision, 100(3):275–293, December 2012.

[10] I. Endres and D. Hoiem. Category-independent object proposals with diverse ranking. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 36(2):222–234, Feb 2014.

[11] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In IEEE CVPR, 2014.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[13] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. Liblinear: A library for large linear classification. JMLR, 9, 2008.

[14] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 32(9):1627–1645, 2010.

[15] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. Int. J. Comput. Vision, 59(2):167–181, September 2004.

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

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093, 2014.

[18] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In CVPR, 2014.

[19] P. Krähenbühl and V. Koltun. Geodesic object proposals. In ECCV 2014, pages 725–739, 2014.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, pages 1097–1105, 2012.

[21] Q. Le, M. Ranzato, R. Monga, K. Chen, G. Corrado, J. Dean, and A. Ng. Building high-level features using large scale unsupervised learning. In ICML, 2012.

[22] L. Liu and L. Wang. What has my classifier learned? visualizing the classification rules of bag-of-feature model by support region detection. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 3586–3593, June 2012.

[23] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In ICLR, 2014.

[24] Z. Shi, T. M Hospedales, and T. Xiang. Bayesian joint topic modelling for weakly supervised object localisation. In International Conference on Computer Vision (ICCV), 2013.

[25] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep fisher networks for large-scale image classification. In NIPS, pages 163–171, 2013.

[26] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. ICLR Workshop, 2014.

[27] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell. One-bit object detection: On learning to localize objects with minimal supervision. In ICML, 2014.

[28] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In NIPS, pages 2553–2561, 2013.

[29] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the gap to human-level performance in face verification. In CVPR, 2014.

[30] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. Int. J. Comput. Vision, 104(2):154–171, 2013.

[31] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. In ECCV, 2014.
A. Visualizing the Mask-out Effect

In this appendix we provide further evidence that supports the quality of the proposed method, including:

- Visualization of the mask-out effect in terms of convolutional feature maps and drop in classification (Figures 6, 7, and 8);
- Visualization of top-1 bounding boxes generated by STL-CL (our method) and SEL_SEARCH [30] (Table 3).

**Mask-out effect.** We show some qualitative examples of the effect of the mask-out operation on images in Figure 6, 7, and 8. Each row reports network input (i.e., the image) and feature maps from each of the 5 convolutional layers of the network (shown as a grid of $F \times F$ feature maps). The first row in each set shows the original images, while the second row shows the effects on the the masked-out image. We also report the value of the drop in classification (Eq. 1) caused by the mask-out operation.

In Fig. 6 we can see some cases where the proposed method succeeds, i.e., where masking out the object region causes a significant drop in classification score. It is interesting to visually note how the mask-out operation propagates through the intermediate convolutional layers of the net until reaching the classification output (as evidenced by the drop). The mask-out operation essentially corresponds to zeroing out the feature map values corresponding to pixels in the masked-out region (e.g., rectangular dark blue box in norm1 feature maps). Fig. 7 shows slightly more complex examples where the drop is not as pronounced as in Fig. 6 but it is still reasonably high.

Fig. 8 shows some hard examples where our localization method is prone to fail because the drop in recognition is not high. In Fig. 8(a), masking out one of the two dogs causes a small drop since there is still one dog that can be recognized by the network. Moreover, a small drop may happen also when the convolutional network uses contextual information (for example the color distribution of the background) that has learned as correlating to some specific category during training, e.g., the eagle and the background landscape in Fig. 8(b). Finally, the basketball example in Fig. 8(c) shows that the network is still able to classify the object (the ball) even when the object of interest is masked out. This is due to the frequent co-occurrence in the training set of basketball and basketball player. The network therefore learned the co-occurrence of the two different objects but not the characteristic of the basketball itself. Fortunately, because STL relies on three other terms it can propose good subwindows also in cases where the mask-out term fails.

**Top-1 bounding box.** We show in Table 3 the top-scoring bounding box on a few sample images of the dataset ILSVRC-2012-LOC-200, using different bounding box proposal methods. In the case of our method (STL-CL), we show the top bounding box selected according to Eq. 1. As already highlighted by the quantitative results, the sub-windows produced by STL-CL are more accurate than those produced by SEL_SEARCH. It is also interesting to notice in the last row of the table that multiple similar instances of the same object are often grouped together because STL-CL yields the maximum drop in classification when all of them are masked out.
Figure 6. Successful examples where masking out the object yields large drops in classification score. Each row reports network input (i.e., the image) and the feature maps from each of the 5 convolutional layers of the network (shown as a grid of $F \times F$ feature maps). The first row in each set shows the original image, the second row the masked-out image.

(a) Drop = 1.000

(b) Drop = 0.915
Figure 7. More complex examples. Because of the presence of some context, the drop in classification score caused by masking out the object (while preserving the context) is not as large as in the previous Figure.
Figure 8. Cases where the masking out of the object fails to significantly drop the classification score due to multiple objects (a) or the presence of context useful to recognize the object, such as in (b) and (c).
Table 3. Top-scoring bounding boxes generated by STL\textsubscript{CL} and SEL\textsc{SEARCH} for a few sample images from the dataset ILSVRC-2012-LOC-200.