Weakly Supervised Object Localization Using Size Estimates

Miaojing Shi(✉) and Vittorio Ferrari

University of Edinburgh, Edinburgh, Scotland, UK
{miaojing.shi,vittorio.ferrari}@ed.ac.uk

Abstract. We present a technique for weakly supervised object localization (WSOL), building on the observation that WSOL algorithms usually work better on images with bigger objects. Instead of training the object detector on the entire training set at the same time, we propose a curriculum learning strategy to feed training images into the WSOL learning loop in an order from images containing bigger objects down to smaller ones. To automatically determine the order, we train a regressor to estimate the size of the object given the whole image as input. Furthermore, we use these size estimates to further improve the re-localization step of WSOL by assigning weights to object proposals according to how close their size matches the estimated object size. We demonstrate the effectiveness of using size order and size weighting on the challenging PASCAL VOC 2007 dataset, where we achieve a significant improvement over existing state-of-the-art WSOL techniques.

1 Introduction

Object class detection has been intensively studied during recent years [1–9]. The goal is to place a bounding box around every instance of a given object class. Given an input image, typically modern object detectors first extract object proposals [7,10,11] and then score them with a classifier to determine their probabilities of containing an instance of certain class [12,13]. Manually annotated bounding boxes are typically required for training the classifier.

Annotating bounding boxes is usually tedious and time consuming. In order to reduce the annotation cost, a commonly used strategy is to learn the detector in a weakly supervised manner: we are given a set of images known to contain instances of a certain object class, but we do not know the object locations in these images. This weakly supervised object localization (WSOL) bypasses the need for bounding box annotation and therefore substantially reduces annotation time. WSOL is typically conducted in two iterative steps [13–20]: (1) re-localizing object instances in the images using the current object detector, and (2) re-training the object detector given the current selection of instances.

WSOL algorithms typically apply both the re-training and re-localization steps on the entire training set at the same time. However, WSOL works better on images with bigger objects. For instance, [16] observed that the performance of several WSOL algorithms consistently decays from easy dataset with many

B. Leibe et al. (Eds.): ECCV 2016, Part V, LNCS 9909, pp. 105–121, 2016.
DOI: 10.1007/978-3-319-46454-1_7

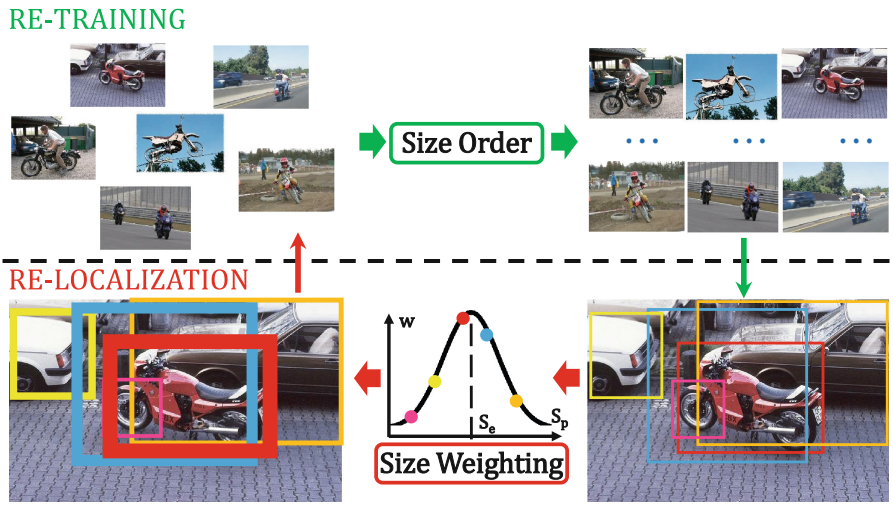


Fig. 1. Overview of our method. We use size estimates to determine the order in which images are fed to a WSOL loop, so that the object detector is re-trained progressively from images with bigger objects down to smaller ones. We also improve the re-localization step, by weighting object proposals according to how close their size (s_p) matches the estimated object size (s_e)

big objects (Caltech4 [21]) to hard dataset with many small objects (PASCAL VOC 07 [2]). In this paper, we propose to feed images into the WSOL learning loop in an order from images containing bigger objects down to smaller ones (Fig. 1, top half). This forms a curriculum learning [22] strategy where the learner progressively sees more and more training samples, starting from easy ones (big objects) and gradually adding harder ones (smaller objects). To understand why this might work better than standard orderless WSOL, let's compare the two. The standard approach re-trains the model from *all* images at each iteration. These include many incorrect localizations which corrupt the model re-training, and result in bad localizations in the next re-localization step, particularly for small objects (Fig. 2). In our approach instead, WSOL learns a decent model from images of big objects in the first few iterations. This initial model then better localizes objects in images of mid-size objects, which in turn leads to an even better model in the next re-training step, as it has now more data, and so on. By the time the process reaches images of small objects, it already has a good detector, which improve the chances of localizing them correctly (Fig. 2).

Our easy-to-hard strategy needs to determine the sequence of images automatically. For this we train a regressor to estimate the size of the object given the whole image as input. In addition to establishing a curriculum, we use these size estimates to improve the re-localization step. We weight object proposals according to how close their size matches the estimated object size (Fig. 1, bottom half). These weights are higher for proposals of size similar to the estimate,

and decrease as their size difference increases. This weighting scheme reduces the uncertainty in the proposal distribution, making the re-localization step more likely to pick a proposals correctly covering the object. Figure 3 shows an example of how size weighting changes the proposal score distribution induced by the current object detector, leading to more accurate localization.

In extensive experiments on the popular PASCAL VOC 2007 dataset, we show that: (1) using our curriculum learning strategy based on object size gives a 7 % improvement in CorLoc compared to the orderless WSOL; (2) by further adding size weighting into the re-localization step, we get another 10 % CorLoc improvement; (3) finally, we employ a deep Neural Network to re-train the model and achieve our best performance, significantly outperforming the state-of-the-art in WSOL [13,15,23].

Compared to standard WSOL, our scheme needs additional data to train the size regressor. This consists of a single scalar value indicating the size of the object, for each image in an external dataset. We do not need bounding-box annotation. Moreover, in Sect. 4.5 we show that we can use a size regressor generic across classes, by training it on different classes than those used during WSOL.

2 Related Work

Weakly-Supervised Object Localization (WSOL). In WSOL the training images are known to contain instances of a certain object class but their locations are unknown. The task is both to localize the objects in the training images and to learn an detector for the class. WSOL is often conceptualised as Multiple Instance Learning (MIL) [12,14,16,18–20,24,25]. Images are treated as bags of object proposals [7,10,11] (instances). A negative image contains only negative instances. A positive image contains at least one positive instance, mixed in with a majority of negative ones. The goal is to find the true positives instances from which to learn a classifier for the object class.

Due to the use of strong CNN features [5,26], recent works on WSOL [12, 14,15,19,20,23] have shown remarkable progress. Moreover, researchers also tried to incorporate various advanced cues into the WSOL process, *e.g.* objectness [13,16,18,27,28], co-occurrence between multiple classes in the same training images [25], and even appearance models from related classes learned from bounding-box annotations [29–31]. In this work, we propose to estimate the size of the object in an image and inject it as a new cue into WSOL. We use it both to determine the sequence of training images in a curriculum learning scheme, and to weight the score function used during the re-localization step.

Curriculum Learning (CL). The curriculum learning paradigm was proposed by Bengio *et al.* [22], in which the model was learnt gradually from easy to hard samples so as to increase the entropy of training. A strong assumption in [22] is that the curriculum is provided by a human teacher. In this sense, determining what constitute an easy sample is subjective and needs to be manually provided. To alleviate this issue, Kumar and Koller [32] formulated CL as a regularization term into the learning objective and proposed a self-paced learning scheme.

The concept of learning in an easy-to-hard order was visited also in computer vision [33–37]. These works focus on a key question: what makes an image easy or hard? The works differ by how they re-interpret "easiness" in different scenarios. Lee and Grauman [33] consider the task of discovering object classes in an unordered image collection. They relate easiness to "objectness" and "context-awareness". Their context-awareness model is initialized with regions of "stuff" categories, and is then used to support discovering "things" categories in unlabelled images. The model is updated by identifying the easy object categories first and progressively expands to harder categories. Sharmanska *et al.* [35] use some privileged information to distinguish between easy and hard examples in an image classification task. The privileged information are additional cues available at training time, but not at test time. They employ several additional cues, such as object bounding boxes, image tags and rationales to define their concept of easiness [36]. Pentina *et al.* [34] consider learning the visual attributes of objects. They let a human decide whether an object is easy or hard to recognize. The human annotator provides a difficulty score for each image, ranging from easy to hard. In this paper, we use CL in a WSOL setting and propose object size as an "easiness" measure. The most related work to ours is the very recent [37], which learns to predict human response times as a measure of difficulty, and shows an example application to WSOL.

3 Method

In this section we first describe a basic MIL framework, which we use as our baseline (Sect. 3.1); then we show how to use object size estimates to improve the basic framework by introducing a sequence during re-training (Sect. 3.2) and a weighting during re-localization (Sect. 3.3). Finally, we explain how to obtain size estimates automatically in Sect. 3.4.

3.1 Basic Multiple Instance Learning Framework

We represent each image in the input set $\mathcal{I}$ as a bag of proposals extracted using the state of the art object proposal method [11]. It returns about 2000 proposals per image, likely to cover all objects. Following [5,14,19,20,23], we describe the proposals by the output of the second-last layer of the CNN model proposed by Krizhevsky *et al.* [26]. The CNN model is pre-trained for whole-image classification on ILSVRC [38], using the Caffe implementation [39]. This produces a 4096-dimensional feature vector for each proposal. Based on this feature representation, we iteratively build an SVM appearance model A (object detector) in two alternating steps: (1) re-localization: in each positive image, we select the highest scoring proposal by the SVM. This produces the set $\mathcal{S}$ which contains the current selection of one instance from each positive image. (2) re-training: we train the SVM using $\mathcal{S}$ as positive training samples, and all proposals from the negative images as negative samples.

As commonly done in [12,13,17,40–42] we initialize the process by training the appearance model using complete images as training samples. Each image in

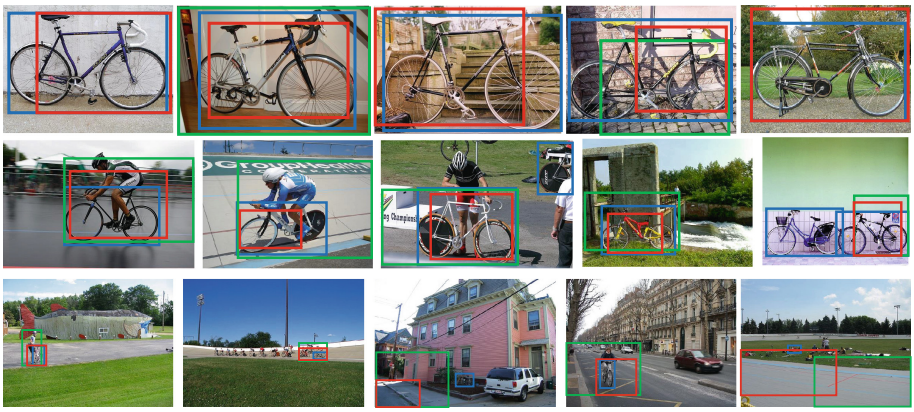


Fig. 2. Illustration of the estimated size order for class *bicycle*, for three batches (one per row). We show the ground-truth object bounding-boxes (blue), objects localized by our WSOL scheme using size order (red), and objects localized by the basic MIL framework (green). In the first, third and last examples of the first row the green and red boxes are identical (Color figure online)

$\mathcal{I}$ provides a training sample. Intuitively, this is a good initialization when the object covers most of the image, which is true only for some images.

3.2 Size Order

Assume we have a way to automatically estimate the size of the object in all input images $\mathcal{I}$ (Sect. 3.4). Based on their object size order, we re-organize MIL on a curriculum, as detailed in Alg. 1.

We split the images into K batches according to their estimated object size (Fig. 2). We start by running MIL on the first batch $\mathcal{I}_1$, containing the largest objects. The whole-image initialization works well on them, leading to a reasonable first appearance model A_1 (though trained from fewer images). We continue running MIL on the first batch $\mathcal{I}_1$ for M iterations to get a solid A_1. The process then moves on to the second batch $\mathcal{I}_2$, which contains mid-size objects, *adding* all its images into the current working set $\mathcal{I}_1 \cup \mathcal{I}_2$, and run the MIL iterations again. Instead of starting from scratch, we use A_1 from the first batch MIL iterations. This model is likely to do a better job at localizing objects in batch $\mathcal{I}_2$ than the whole-image initialization of basic MIL (Fig. 2, second row). Hence, the model trains from better samples in the re-training step. Moreover, the model A_2 output by MIL on $\mathcal{I}_1 \cup \mathcal{I}_2$ will be better than A_1, as it is trained from more samples. Finally, during MIL on $\mathcal{I}_1 \cup \mathcal{I}_2$, the localization of objects in I_1 will also improve (Fig. 2, first row).

The process iteratively moves on to the next batch $k+1$, every time starting from appearance model A_k and running MIL's re-training/re-localization iterations on the image set $\cup_{i=1}^{k+1} \mathcal{I}_i$. As the image set continuously grows, the process does not jump from batch to batch. This helps stabilizing the learning

Alg. 1. Multiple instance learning with size order and size weighting

Initialization:
1) split the input set $\mathcal{I}$ into K batches according to the estimated object size order
2) initialize the positive and negative examples as the entire images in first batch $\mathcal{I}_1$
3) train an appearance model A_1 on the initial training set
for batch $k = 1 : K$ **do**
  for iteration $m = 1 : M$ **do**
    i) **re-localize** the object instances in images $\cup_{i=1}^{k}\mathcal{I}_i$ using current appearance
      model A_k^m and size weighting of object proposals;
    ii) add new negative proposals by hard negative mining;
    iii) **re-train** the appearance model A_k^m given current selection of instances in
      images $\cup_{i=1}^{k}\mathcal{I}_i$;
  end for
end for
Return final detector and selected object instances in $\mathcal{I}$

process and properly training the appearance model from more and more training samples. By the time the process reaches batches with small objects, the appearance model will already be very good and will do a much better job than the whole-image initialization of basic MIL on them (Fig. 2, third row). Figure 2 shows some examples of applying our curriculum learning strategy compared to basic MIL. In all our work, we set $K = 3$ and $M = 3$.

3.3 Size Weighting

In addition to establishing a curriculum, we use the size estimates to refine the re-localization step of MIL. A naive way would be to filter out all proposals with size different from the estimate. However, this is likely to fail as neither the size estimator nor the proposals are perfectly accurate, and therefore even a good proposal covering the object tightly will not exactly match the estimated size.

Instead, we use the size estimate as indicative of the *range* of the real object size. Assuming the error distribution of the estimated size w.r.t the real size is normal, according to the three-sigma rule of thumb [43], the real object size is very likely to lie in this range $[s_e - 3\sigma, s_e + 3\sigma]$ (with 99.7 % probability), where s_e is the estimated size and σ is the standard deviation of the error. We explain in Sect. 3.4 how we obtain σ.

We assign a continuous weight to each proposal p so that it gives a relatively high weight for the size s_p of the proposal falling inside the 3σ interval of the estimated object size s_e, and a very low weight for s_p outside the interval:

$$W(p; s_e, \sigma, \delta) = \min\left(\frac{1}{1 + e^{\delta \cdot (s_e - 3\sigma - s_p)}}\ ,\ \frac{1}{1 + e^{\delta \cdot (s_p - s_e - 3\sigma)}}\right). \tag{1}$$

This function decreases with the difference between s_p and s_e (Fig. 3); δ is a scalar parameter that controls how rapidly the function decreases, particularly outside the three sigma range $[s_l, s_r]$. The model is not sensitive to the exact

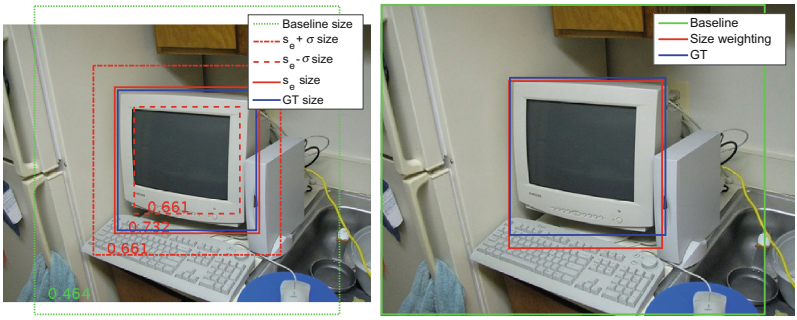


Fig. 3. Illustration of size weighting. Left: behaviour of the size weighting function W. Example sizes are shown by boxes of the appropriate area centered at the ground truth (GT) object; s_e denotes the estimated object size. The size weight W of each box is written in its bottom left corner. Right: detection result using size weighting (red) compared to basic MIL framework (green) (Color figure online)

choice of δ (we set $\delta = 3$ in all experiments). Weights for proposals falling out of the interval $[s_l, s_r]$ quickly go to zero. Thereby this weight W represents the likelihood of proposal p covering the object, according to the size estimate s_e.

We now combine the size weighting W of a proposal with the score given by the SVM appearance model A. First we transform the output of the SVM into a probability using platt-scaling [44]. Assuming that the two score functions are independent, we combine them by multiplication, yielding the final score of a proposal p: $A(p) \cdot W(p; s_e, \sigma, \delta)$. This score is used in the re-localization step of MIL (Sect. 3.1), making it more likely to pick a proposal correctly covering the object. Figure 4 gives some example results of using this size weighting model.

3.4 Size Estimator

In Subsects. 3.2 and 3.3, we assumed the availability of an automatic estimator of the size of objects in images. In this subsection we explain how we do it.

We use Kernel Ridge Regressor (KRR) [45] to estimate the size of the object given the whole image as input. We train it beforehand on an external set $\mathcal{R}$, disjoint from the set $\mathcal{I}$ on which MIL operates (Sect. 3.1). We train a separate size regressor for each object class. For each class, the training set $\mathcal{R}$ contains images annotated with the size s_t of the largest object of that class in it. The training set can be small, as we demonstrate in Sect. 4.4. The input image is represented by a 4096-dimensional CNN feature vector covering the whole image, output of the second-last layer of the AlexNet CNN architecture [26]. The object size is represented by its area normalized by the image area. As area differences grow rapidly, learning to directly regress to area puts more weight on estimation errors on large objects rather than on smaller objects. To alleviate this bias, we apply a r-th root operation on the regression target values $s_t \leftarrow \sqrt[r]{s_t}$. Empirically, we choose $r = 3$, but the regression performance over different r is very close.

We train the KRR by minimizing the squared error on the training set $\mathcal{R}$ and obtain the regressor along with the standard deviation σ of its error by cross-validation on $\mathcal{R}$. We then use this size regressor to automatically estimate the object size on images in the WSOL input set $\mathcal{I}$.

4 Experiments

4.1 Dataset and Settings

Size Estimator Training. We train the size estimator on the trainval set $\mathcal{R}$ of PASCAL VOC 2012 [2] (PASCAL 12 for short). This has 20 classes, a total of 11540 images, and 834 images per class on average.

WSOL. We perform WSOL on the trainval set $\mathcal{I}$ of PASCAL 07 [2], which has different images of the same 20 classes in $\mathcal{R}$ (5011 images in total). While several WSOL works remove images containing only truncated and difficult objects [12, 13, 16, 17], we use the complete set $\mathcal{I}$.

We apply the size estimator on $\mathcal{I}$ and evaluate its performance on it in Sect. 4.2. Then, we use the estimated object sizes to improve the basic MIL approach of Sect. 3.1, as described in Sects. 3.2 and 3.3. Finally, we apply the detectors learned on $\mathcal{I}$ to the test set $\mathcal{X}$ of PASCAL 07, which contains 4952 images in total. We evaluate our method and compare to standard orderless MIL in Sect. 4.3.

CNN. We use AlexNet as CNN architecture [26] to extract features for both size estimation and MIL (Sects. 3.1 and 3.4). As customary [13, 15, 20, 23], we pre-train it for whole-image classification on ILSVRC [38], but we do *not* do any fine-tuning on bounding-boxes.

4.2 Size Estimation

Evaluation Protocol. We train the regressor on set $\mathcal{R}$. We adopt 7-fold cross-validation to obtain the best regressor and the corresponding σ. In order to test the generalization ability of the regressor, we gradually reduce the number of training images from an average of 834 per class to 100, 50, 40, 30 per class.

The regression performance on $\mathcal{I}$ is measured via the mean square error (MSE) between the estimated size and the ground-truth size (both in r^{th} root, see Sect. 3.4), and the Kendall's τ rank correlation coefficient [46] between the estimated size order and the ground-truth size order.

Results. Table 1 presents the results. We tried different r^{th} root of the size value during training. While $r = 3$ gives highest performance, it is not sensitive to exact choice of r, as long as $r > 1$. The table also shows the effect of reducing the number of training images N to 100, 50, 40, and 30 per class. Although performance decreases when training with fewer samples, even using as few as 30 samples per class still delivers good results.

We set $r = 3$ and use all training samples in $\mathcal{R}$ by default in the following experiments. We will also present an in-depth analysis of the impact of varying N on WSOL in Sect. 4.4.

Table 1. Size estimation result on set $\mathcal{I}$ with different r and number N of training images per class. r refers to the r^{th} root on size value applied; 'ALL' indicates using the complete $\mathcal{R}$ set, which has 834 images per class on average

r^{th} root	Kendall's τ	N	Kendall's τ	MSE
	N		$r = 3$	
1	0.604	**ALL**	**0.614**	**0.013**
2	0.612	100	0.561	0.016
3	**0.614**	50	0.542	0.018
4	0.612	40	0.530	0.019
5	0.610	30	0.527	0.020

4.3 Weakly Supervised Object Localization (WSOL)

Evaluation Protocol. In standard MIL, given the training set $\mathcal{I}$ with image-level labels, our goal is to localize the object instances in this set and to train good object detectors for the test set $\mathcal{X}$. We quantify localization performance in the training set with the Correct Localization (CorLoc) measure [12,13,15,16,23,47]. CorLoc is the percentage of images in which the bounding-box returned by the algorithm correctly localizes an object of the target class (intersection-over-union ≥ 0.5 [2]). We quantify object detection performance on the test set $\mathcal{X}$ using mean average precision (mAP), as standard in PASCAL VOC.

As in most previous WSOL methods [12–20,23], our scheme returns exactly one bounding-box per class per training image. This enables clean comparisons to previous work in terms of CorLoc on the training set $\mathcal{I}$. Note that at test time the object detector is capable of localizing multiple objects of the same class in the same image (and this is captured in the mAP measure).

Baseline. We use EdgeBoxes [11] as object proposals and follow the basic MIL framework of Sect. 3.1. For the baseline, we randomly split the training set $\mathcal{I}$ into three batches ($K = 3$), then train an SVM appearance model sequentially batch by batch. We apply three MIL iterations ($M = 3$) within each batch, and use hard negative mining for the SVM [12].

Like in [13,16,18,23,25,27,29,48,49], we combine the SVN score with a general measure of "objectness" [10], which measures how likely it is that a proposal tightly encloses an object of any class (*e.g.* bird, car, sheep), as opposed to background (*e.g.* sky, water, grass). For this we use the objectness measure produced by the proposal generator [11]. Using this additional cue makes the basic MIL start from a higher baseline.

Table 2 shows the result: CorLoc 39.1 on the training set $\mathcal{I}$ and mAP 20.1 on the test set $\mathcal{X}$. Examples are in Fig. 4 first row. In the following, we incorporate our ideas (size order and size weighting) into this baseline (Alg. 1).

Size Order. We use the same settings as the baseline ($K = 3$ and $M = 3$), but now the training set $\mathcal{I}$ is split into batches according to the size estimates.

Table 2. Comparison between the baseline MIL scheme, various versions of our scheme, and the state-of-the-art on PASCAL 07. 'Deep' indicates using additional MIL iterations with Fast R-CNN as detector

Method				CorLoc	mAP
	Size order	Size weight	Deep	-	-
Baseline				39.1	20.1
Our scheme	✓			46.3	24.9
	✓	✓		55.8	28.0
	✓	✓	✓	**60.9**	**36.0**
Baseline			✓	43.2	24.7
Cinbis *et al.* [13]				54.2	28.6
Wang *et al.* [23]				48.5	31.6
Bilen *et al.* [15]				43.7	27.7
Shi *et al.* [47]				38.3	-
Song *et al.* [20]				-	24.6

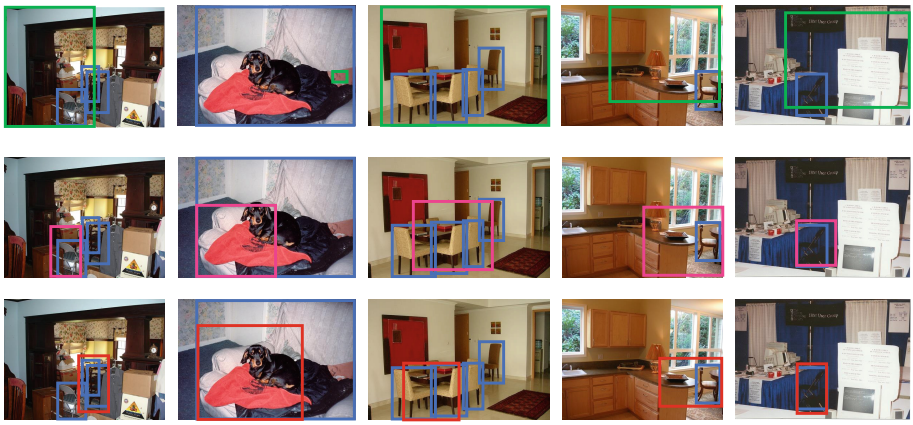


Fig. 4. Example localizations by different WSOL schemes on class *chair*. First row: localizations by the MIL baseline (green, see Sect. 4.3: Baseline setting). Second row: localizations by our method, which adds size order to the baseline (purple, see Sect. 4.3: Size order). Third row: localizations by our method with both size order and weighting (red, see Sect. 4.3: Size weighting). Ground-truth bounding-boxes are shown in blue (Color figure online)

As Table 2 shows, by performing curriculum learning based on size order, we improve CorLoc to 46.3 and mAP to 24.9. Examples are in Fig. 4 second row.

Size Weighting. Significant improvement of CorLoc can be further achieved by adding size weighting on top of size order. Table 2 illustrates this effect: the CorLoc using size order and size weighting goes to 55.8. Compared the baseline

39.1, this is a +16.7 improvement. Furthermore, the mAP improves to 28.0 (+7.9 over the baseline). Examples are in Fig. 4 third row.

Deep Net. So far, we have used an SVM on top of fixed deep features as the appearance model. Now we change the model to a deeper one, which trains all layers during the re-training step of MIL (Sect. 3.1). We take the best detection result we obtained so far (using both size order and size weighting) as an initialization for three additional MIL iterations. During these iterations, we use Fast R-CNN [4] as appearance model. We use the entire set at once (no batches) during the re-training and re-localization steps, and omit bounding-box regression in the re-training step [4], for simplicity. We only carry out three iterations as the system quickly converges after the first iteration.

As Table 2 shows, using this deeper model raises CorLoc to 60.9 and mAP to 36.0, which is a visible improvement. It is interesting to apply these deep MIL iterations also on top of the detections produced by the baseline. This yields a +4.1 higher CorLoc and +4.6 mAP (reaching 43.2 CorLoc and 24.7 mAP). In comparison, the effect of our proposed size order and size weighting is greater (+16.7 CorLoc and +7.9 mAP over the baseline, when both use SVM appearance models). Moreover, size order and weighting have an even greater effect when used in conjunction with the deep appearance model (+17.7 CorLoc and +11.3 mAP, when both the baseline and our method use Fast R-CNN).

Comparison to the State-of-the-Art. Table 2 also compares our method to state-of-the-art WSOL works [13,15,20,23,47]. We compare both the CorLoc on the trainval set $\mathcal{I}$ and mAP on the test set $\mathcal{X}$. We list the best results reported in each paper. Note [13] removes training images with only truncated and difficult object instances, which makes the WSOL problem easier, whereas we train from all images. As the table shows, our method outperforms all these works both in terms of CorLoc and mAP. All methods we compare to, except [47] use AlexNet, pretrained on ILSVRC classification data, as we do.

4.4 Impact of Size of Training Set for Size Regressor

The size estimator we used so far is trained on the complete set $\mathcal{R}$. What if we only have limited training samples with object size annotations? As shown in Sect. 4.2, when we reduce the number of training samples N per class, the accuracy of size estimation decreases moderately. However, we argue that neither Kendall's τ nor MSE are suitable for measuring the impact of the size estimates on MIL, when these are used to establish an order as we do in Sect. 3.2. As $\mathcal{I}$ is split into batches according to the size estimates, only the inter-batch size order matters, the order of images within one batch does not make any difference.

To measure the correlation of inter-batch size order between the ground-truth size sequence Q_{GT} and the estimated size sequence Q_{ES}, we count how many samples in Q^k_{GT} have been successfully retrieved in Q^k_{ES}, where Q^k indicates the set of images in batches 1 through k:

$$\text{recall} = \frac{|Q^k_{GT} \cap Q^k_{ES}|}{|Q^k_{GT}|}, \tag{2}$$

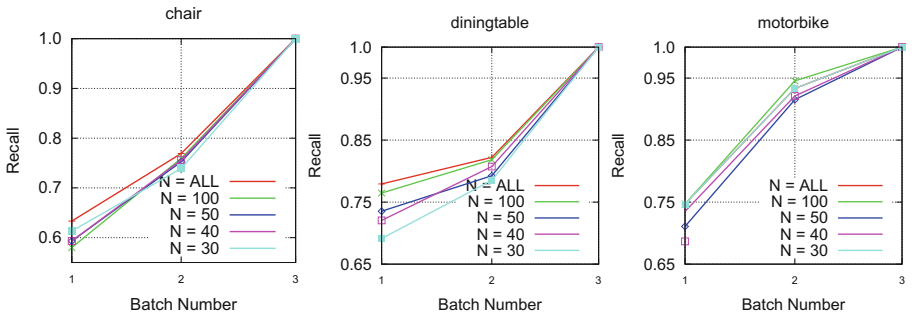


Fig. 5. Correlation between inter-batch size order based on the ground-truth size sequence and the estimated sequence, on class *chair*, *diningtable*, and *motorbike* of $\mathcal{I}$ set; recall is computed as in (2)

$|\cdot|$ denotes number of elements. Figure 5 shows recall curves on set $\mathcal{I}$, with varying N. The curves are quite close to each other, showing that reducing N does not affect the inter-batch order very much.

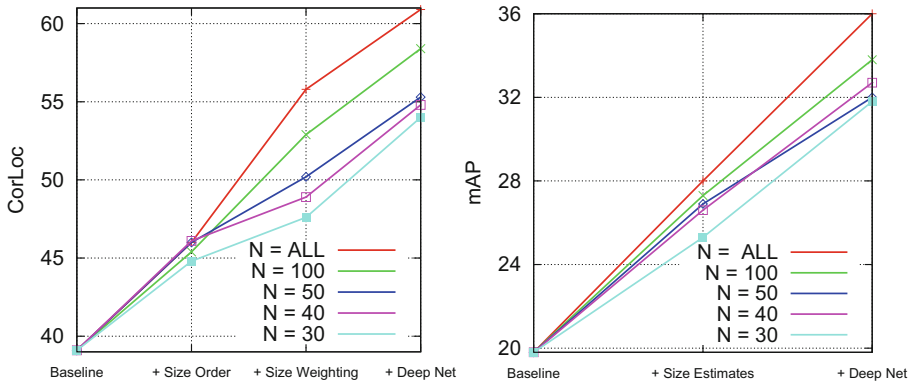
In Fig. 6 we conduct the WSOL experiment of Sect. 4.3, incorporating size order into the basic MIL framework on $\mathcal{I}$, using different size estimators trained with varying N. The 'baseline + size order' result in Fig. 6a shows little variation: even $N = 30$ leads to CorLoc within 2 % of using the full set $N = \text{ALL}$. This is due to the fact shown above, that a less accurate size estimator does not affect the inter-batch size order much.

We also propose to use the size estimate to help MIL with size weighting (Sect. 3.3). Table 1 shows that MSE gets larger when N becomes smaller, which means the estimated object size gets father from the real value. This lower accuracy estimate affects size weighting and, in turn, can affect the performance of MIL. To validate this, we add size weighting on top of size order into MIL in Fig. 6. This time, the CorLoc improvement brought by size weighting varies significantly with N. Nevertheless, even with just $N = 30$ training samples per class, we still get an improvement. We believe this is due to the three-sigma rule we adopted in the weighting function (1). The real object size is very likely to fall into the 3σ range, and so it gets a relatively high weighting compared to the proposals with size outside the range.

Finally, we apply the additional deep MIL iterations presented in Sect. 4.3, 'Deep net' paragraph. Figure 6 shows a consistent trend of improvement across different N and our proposed size order and weighting schemes, on both CorLoc and mAP.

4.5 Further Analysis

Deep *v.s.* Deeper. So far we used AlexNet [26] during deep re-training (Sect. 4.3, 'Deep net' paragraph). Here we use an even deeper CNN architecture,



(a) CorLoc (b) mAP

Fig. 6. WSOL performance on PASCAL 07 when varying N. Size order and weighting are gradually added into the baseline MIL framework, and eventually fed into the deep net. We use 'size estimates' in (b) to denote using both size order and size weighting

Table 3. WSOL results using AlexNet or VGG16 in Fast R-CNN. We report CorLoc on the trainval set $\mathcal{I}$ and mAP on the test set $\mathcal{X}$ of PASCAL 07

CNN architecture	AlexNet [26]	VGG16 [50]
CorLoc (trainval)	60.9	64.7
mAP (test)	36.0	37.2

VGG16 [50]. The result in Table 3 shows the benefits by going deeper, as get to a final CorLoc 64.7 and mAP 37.2.

Class-Specific, Class-Generic and Across-Class. So far we used an object size estimator trained separately for each class. Here we test the class-generalization ability of proposed size order and size weighting ideas. We perform two experiments. In the first, we use the entire $\mathcal{R}$ to train a single size estimator over all 20 classes, and use it on every image in $\mathcal{I}$, regardless of class. We call this estimator *class-generic* as it has to work regardless of the class it is applied to, within the range of classes it has seen during training. In the second experiment, we separate the 20 classes into two groups: (i) bicycle, bottle, car, chair, dining table, dog, horse, motorbike, person, TV monitor; (ii) airplane, bird, boat, bus, cat, cow, potted plant, sheep, sofa, train. We train two size estimators separately, one on each group. When doing WSOL on a class in $\mathcal{I}$, we use the estimator trained on the group not containing that class. We call this estimator *across-class*, as it has to generalize to new classes not seen during training.

Table 4 shows the results of WSOL, in terms of CorLoc on the trainval set $\mathcal{I}$ and the mAP on the test set $\mathcal{X}$ of PASCAL 07. Thanks to our robust batch-by-batch design in curriculum learning, the CorLoc using the size order is about the

same for all size estimators. This shows that it is always beneficial to incorporate our proposed size order into WSOL, even when applied to new classes. When incorporating also size weighting into MIL, the benefits gradually diminish when going from the class-specific to the across-class estimators, as they predict object size less accurately. Nonetheless, we still get about +3 CorLoc when using the class-generic estimator and about +1 when using the across-class one.

The last column of Table 4, reports mAP on the test set, with deep re-training. The class-generic estimator leads to mAP 32.2, and the across-class one to 30.0. They are still substantially better than the baseline (24.7 when using deep re-training, see Table 2). Interestingly, the across-class result is only moderately worse than the class-generic one, which was trained on all 20 classes. This shows our method generalizes well to new classes.

Table 4. WSOL results using different size estimators. The first four columns show CorLoc on the trainval set $\mathcal{I}$; the last row shows mAP on the test set $\mathcal{X}$. The baseline does not use size estimates and is reported for reference

Size estimator	Baseline	+ Size order	+ Size weighting	+ Deep net	mAP on test $\mathcal{X}$
Class-specific	39.1	46.3	55.8	60.9	36.0
Class-generic	39.1	45.6	48.4	54.4	32.2
Across-class	39.1	45.0	45.8	51.1	30.0

5 Conclusions

We proposed to use object size estimates to help weakly supervised object localization (WSOL). We introduced a curriculum learning strategy to feed training images into WSOL in an order from images containing bigger objects down to smaller ones. We also proposed to use the size estimates to help the re-localization step of WSOL, by weighting object proposals according to how close their size matches the estimated object size. We demonstrated the effectiveness of both ideas on top of a standard multiple instance learning WSOL scheme.

Currently we use the output of the MIL framework with size order and size weighting as the starting point for additional iterations that re-train the whole deep net. However, the training set is not batched any more during deep re-training. A promising direction for future work is to embed the size estimates into an MIL loop where the whole deep net is updated. Another interesting direction is to go towards a continuous ordering, *i.e.* where the batch size goes towards 1; efficiently updating the model in that setting is another challenge.

Acknowledgments. Work supported by the ERC Starting Grant VisCul.

References

1. Dalal, N., Triggs, B.: Histogram of oriented gradients for human detection. In: CVPR (2005)
2. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL visual object classes (VOC) challenge. IJCV **88**, 303–338 (2010)
3. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. IEEE Trans. PAMI **32**(9), 1627–1645 (2010)
4. Girshick, R.: Fast R-CNN. In: ICCV (2015)
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
6. Malisiewicz, T., Gupta, A., Efros, A.: Ensemble of exemplar-SVMs for object detection and beyond. In: ICCV (2011)
7. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. IJCV **104**, 154–171 (2013)
8. Viola, P.A., Platt, J., Zhang, C.: Multiple instance boosting for object detection. In: NIPS (2005)
9. Wang, X., Yang, M., Zhu, S., Lin, Y.: Regionlets for generic object detection. In: ICCV, pp. 17–24. IEEE (2013)
10. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? In: CVPR (2010)
11. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 391–405. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10602-1_26
12. Cinbis, R., Verbeek, J., Schmid, C.: Multi-fold mil training for weakly supervised object localization. In: CVPR (2014)
13. Cinbis, R., Verbeek, J., Schmid, C.: Weakly supervised object localization with multi-fold multiple instance learning. IEEE Trans. PAMI (2016)
14. Bilen, H., Pedersoli, M., Tuytelaars, T.: Weakly supervised object detection with posterior regularization. In: BMVC (2014)
15. Bilen, H., Pedersoli, M., Tuytelaars, T.: Weakly supervised object detection with convex clustering. In: CVPR (2015)
16. Deselaers, T., Alexe, B., Ferrari, V.: Localizing objects while learning their appearance. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 452–466. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15561-1_33
17. Russakovsky, O., Lin, Y., Yu, K., Fei-Fei, L.: Object-centric spatial pooling for image classification. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part II. LNCS, vol. 7573, pp. 1–15. Springer, Heidelberg (2012)
18. Siva, P., Xiang, T.: Weakly supervised object detector learning with model drift detection. In: ICCV (2011)
19. Song, H., Girshick, R., Jegelka, S., Mairal, J., Harchaoui, Z., Darell, T.: On learning to localize objects with minimal supervision. In: ICML (2014)
20. Song, H., Lee, Y., Jegelka, S., Darell, T.: Weakly-supervised discovery of visual pattern configurations. In: NIPS (2014)
21. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR (2003)
22. Bengio, J., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: ICML (2009)

23. Wang, C., Ren, W., Zhang, J., Huang, K., Maybank, S.: Large-scale weakly supervised object localization via latent category learning. IEEE Trans. Image Process. **24**(4), 1371–1385 (2015)
24. Dietterich, T.G., Lathrop, R.H., Lozano-Perez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artif. Intell. **89**(1–2), 31–71 (1997)
25. Shi, Z., Siva, P., Xiang, T.: Transfer learning by ranking for weakly supervised object annotation. In: BMVC (2012)
26. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
27. Tang, K., Joulin, A., Li, L.J., Fei-Fei, L.: Co-localization in real-world images. In: CVPR (2014)
28. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. IEEE Trans. PAMI **34**, 2189–2202 (2012)
29. Guillaumin, M., Ferrari, V.: Large-scale knowledge transfer for object localization in imagenet. In: CVPR (2012)
30. Rochan, M., Wang, Y.: Weakly supervised localization of novel objects using appearance transfer. In: CVPR (2015)
31. Hoffman, J., Guadarrama, S., Tzeng, E., Hu, R., Donahue, J.: LSDA: Large scale detection through adaptation. In: NIPS (2014)
32. Kumar, M.P., Packer, B., Koller, D.: Self-paced learning for latent variable models. In: NIPS (2010)
33. Lee, Y.J., Grauman, K.: Learning the easy things first: Self-paced visual category discovery. In: CVPR (2011)
34. Pentina, A., Sharmanska, V., Lampert, C.H.: Curriculum learning of multiple tasks. In: CVPR (2015)
35. Sharmanska, V., Quadrianto, N., Lampert, C.: Learning to rank using privileged information. In: CVPR (2013)
36. Lapin, M., Hein, M., Schiele, B.: Learning using privileged information: Svm+ and weighted svm. Neural Netw. **53**, 95–108 (2014)
37. Ionescu, R.T., Alexe, B., Leordeanu, M., Popescu, M., Papadopoulos, D.P., Ferrari, V.: How hard can it be? Estimating the difficulty of visual search in an image. In: CVPR (2016)
38. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., Fei-Fei, L.: ImageNet large scale visual recognition challenge. IJCV **115**, 211–252 (2015)
39. Jia, Y.: Caffe: an open source convolutional architecture for fast feature embedding (2013). http://caffe.berkeleyvision.org/
40. Pandey, M., Lazebnik, S.: Scene recognition and weakly supervised object localization with deformable part-based models. In: ICCV (2011)
41. Nguyen, M., Torresani, L., de la Torre, F., Rother, C.: Weakly supervised discriminative localization and classification: a joint learning process. In: ICCV (2009)
42. Kim, G., Torralba, A.: Unsupervised detection of regions of interest using iterative link analysis. In: NIPS (2009)
43. Wheeler, D.J., Chambers, D.S., et al.: Understanding Statistical Process Control. SPC Press, Knoxville (1992)
44. Platt, J.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in Large Margin Classifiers (1999)
45. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
46. Kendall, M., Stuart, A.: The Advanced Theory of Statistics. Charles Griffin and Company, London (1983)

47. Shi, Z., Hospedales, T., Xiang, T.: Bayesian joint modelling for object localisation in weakly labelled images. IEEE Trans. PAMI. **37**, 1959–1972 (2015)
48. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: CVPR (2012)
49. Shapovalova, N., Vahdat, A., Cannons, K., Lan, T., Mori, G.: Similarity constrained latent support vector machine: an application to weakly supervised action classification. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7578, pp. 55–68. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33786-4_5
50. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)