Weakly Supervised Object Localization Using Things and Stuff Transfer

Miaojing Shi[1,2] Holger Caesar[1] Vittorio Ferrari[1]
[1]University of Edinburgh [2]Tencent Youtu Lab
name.surname@ed.ac.uk

Abstract

*We propose to help weakly supervised object localization for classes where location annotations are not available, by transferring things and stuff knowledge from a source set with available annotations. The source and target classes might share similar appearance (*e.g. *bear fur is similar to cat fur) or appear against similar background (*e.g. *horse and sheep appear against grass). To exploit this, we acquire three types of knowledge from the source set: a segmentation model trained on both thing and stuff classes; similarity relations between target and source classes; and co-occurrence relations between thing and stuff classes in the source. The segmentation model is used to generate thing and stuff segmentation maps on a target image, while the class similarity and co-occurrence knowledge help refining them. We then incorporate these maps as new cues into a multiple instance learning framework (MIL), propagating the transferred knowledge from the pixel level to the object proposal level. In extensive experiments, we conduct our transfer from the PASCAL Context dataset (source) to the ILSVRC, COCO and PASCAL VOC 2007 datasets (targets). We evaluate our transfer across widely different thing classes, including some that are not similar in appearance, but appear against similar background. The results demonstrate significant improvement over standard MIL, and we outperform the state-of-the-art in the transfer setting.*

1. Introduction

The goal of object class detection is to place a tight bounding box on every instance of an object class. Given an input image, recent object detectors [1, 2, 3, 4] first extract object proposals [5, 6, 7] and then score them with a classifier to determine their probabilities of containing an instance of the class. Manually annotated bounding boxes are typically required for training (full supervision).

Annotating bounding boxes is tedious and time-consuming. In order to reduce the annotation cost, many previous works learn the detector in a weakly supervised setting [8, 9, 4, 10, 11, 12, 13, 14, 15], *i.e.* given a set of images known to contain instances of a certain object class, but without their locations. This weakly supervised object localization (WSOL) bypasses the need for bounding box annotation and substantially reduces annotation time.

Despite the low annotation cost, the performance of WSOL is considerably lower than that of full supervision. To improve WSOL, various advanced cues can be added, *e.g.* objectness [10, 16, 4, 12, 17, 15], which gives an estimation of how likely a proposal contains an object; co-occurrence among multiple classes in the same training images [18]; object size estimates based on an auxiliary dataset with size annotations [15]; and appearance models transferred from object classes with bounding box annotations to new object classes [19, 20, 21].

There are two types of classes that can be transferred from a source set with manually annotated locations: things (objects) and stuff (materials and backgrounds). Things have a specific spatial extent and shape (*e.g.* helicopter, cow, car), while stuff does not (*e.g.* sky, grass, road). Current transfer works mostly focus on transferring appearance models among similar thing classes [19, 21, 20] (*things-to-things*). In contrast, using stuff to find things [22, 23] is largely unexplored, particularly in the WSOL setting (*stuff-to-things*).

In this paper, we transfer a fully supervised segmentation model from the source set to help WSOL on the target set. We introduce several schemes to conduct the transfer of both things and stuff knowledge, guided by the similarity between classes. Particularly, we transfer the co-occurrence knowledge between thing and stuff classes in the source via a second order scheme to thing classes in the target. We propagate the transferred knowledge from the pixel level to the object proposal level and inject it as a new cue into a multiple instance learning framework (MIL).

In extensive experiments, we show that our method: (1) improves over a standard MIL baseline on three datasets: ILSVRC [24], COCO [25], PASCAL VOC 2007 [26]; (2) outperforms the things-to-things transfer method [21] and the state-of-the-art WSOL methods [27, 4, 28] on VOC 2007; (3) outperforms another things-to-things transfer method (LSDA [20]) on ILSVRC.

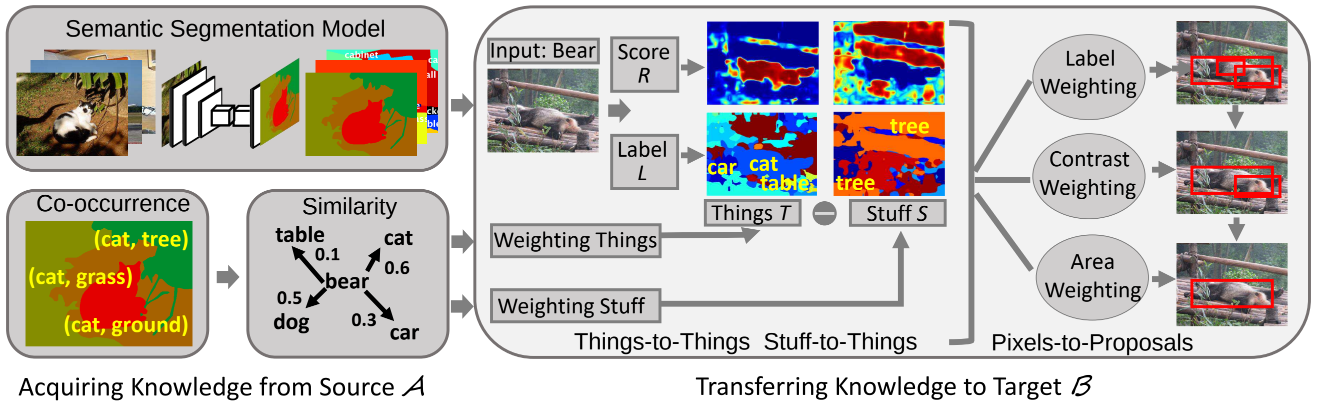

Figure 1: An overview of our things and stuff transfer (TST) method. We acquire the 1) segmentation model, 2) co-occurrence relation and 3) similarity relation from the source $\mathcal{A}$ and transfer them to the target $\mathcal{B}$. We use the segmentation model to generate two maps: thing (T) and stuff (S) maps; each of them contains one score (R) map and one label (L) map. The knowledge of class similarity and co-occurrence is specifically transferred as weighting functions to the thing and stuff label maps. Based on the transferred knowledge, we propose three scoring schemes (label weighting, contrast weighting, and area weighting) to propagate the information from pixels to proposals. The rightmost image column illustrates some highly ranked proposals in the image by gradually adopting the three schemes.

2. Related Work

Weakly supervised object localization. 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 a detector for the class.

Due to the use of strong CNN features [29, 30], recent works on WSOL [9, 4, 13, 28, 27, 15] have shown remarkable progress. Moreover, researchers also tried to incorporate various advanced cues into the WSOL process, *e.g.* objectness [4, 10, 12, 17, 16], object size [15], co-occurrence [18] among classes, and transferring appearance models of the source thing classes to help localize similar target thing classes [19, 21, 20]. This paper introduces a new cue called things and stuff transfer (TST), which learns a semantic segmentation model from the source on both things and stuff annotations and transfers its knowledge to help localize the target thing class.

Transfer learning. The goal of transfer learning is to improve the learning of a target task by leveraging knowledge from a source task [31]. It is intensively studied in image classification, segmentation and object detection [32, 33, 34, 35, 36, 37, 38]. Many methods use the parameters of the source classifiers as priors for the target model [32, 33, 34]. Other works [35, 36] transfer knowledge through an intermediate attribute layer, which captures visual qualities shared by many object classes (*e.g.* "striped", "yellow"). A third family of works transfer object parts between classes [33, 37, 38], *e.g.* wheels between cars and bicycles.

In this work we are interested in the task where we have the location annotations in the source and transfer them to help learn the classes in the target [18, 39, 21, 19, 23, 22]. We categorize the transfer into two types: 1) *Things-to-things.* Guillaumin *et al.* [19] transferred spatial location, appearance, and context information from the source thing classes to localize the things in the target; Shi *et al.* [18] and Rochan *et al.* [21] follow a similar spirit to [19]; while Kuettel *et al.* [39] instead transferred segmentation masks. 2) *Stuff-to-things.* Heitz *et al.* [22] proposed a context model to utilize stuff regions to find things, in a fully supervised setting for the target objects; Lee *et al.* [23] also made use of stuff annotations in the source to discover things in the target, in an unsupervised setting.

Our work offers several new elements over these: (1) we encode the transfer as a combination of both *things-to-things* and *stuff-to-things*; (2) we propose a model to propagate the transferred knowledge from the pixel level to the proposal level; (3) we introduce a second order transfer, *i.e.* *stuff-to-things-to-things*.

3. Overview of our method

In this section we define the notations and introduce our method on a high level, providing some details for each part.

Notations. We have a source set $\mathcal{A}$ and a target set $\mathcal{B}$. We have every image pixelwise annotated for both stuff and things in $\mathcal{A}$; whereas we have only image level labels for images in $\mathcal{B}$. We denote by $\mathcal{A}^T$ the set of thing classes in $\mathcal{A}$, and a^t an individual thing class; analogue we have $\mathcal{A}^S$ and a^s for stuff classes in $\mathcal{A}$ and $\mathcal{B}^T$ and b^t for thing classes in $\mathcal{B}$. Note that there are no stuff classes in $\mathcal{B}$, as datasets labeled only by thing classes are more common in practice (*e.g.* PASCAL VOC [40], ImageNet [24], COCO [25]).

Method overview. Our goal is to conduct WSOL on $\mathcal{B}$, where the training images are known to contain instances of a certain object class but their locations are unknown. A standard WSOL approach, *e.g.* MIL, treats images as bags of object proposals [5, 6, 7] (instances). The task is both to localize the objects (select the best proposal) in the training images and to learn a detector for the target class. To improve MIL, we transfer knowledge from $\mathcal{A}$ to $\mathcal{B}$, incorporating new cues into it.

Fig. 1 illustrates our transfer. We first acquire three types of knowledge in the source $\mathcal{A}$ (Sec. 4): 1) a semantic segmentation model (Sec. 4.1), 2) the thing class similarities between $\mathcal{A}$ and $\mathcal{B}$ (Sec. 4.2) and 3) the co-occurrence frequencies between thing and stuff classes in $\mathcal{A}$ (Sec. 4.3). Afterwards, we transfer the knowledge to $\mathcal{B}$ (Sec. 5). Given an image in $\mathcal{B}$, we first use the segmentation model to generate the thing (T) and stuff (S) maps of it (Sec. 5.1). T contains one score map (R) and one label (L) map, so does S. The segmentation model transfers knowledge generically to every image in $\mathcal{B}$. Building upon its result, we propose three proposal scoring schemes: label weighting (LW, Sec. 5.2), contrast weighting (CW, Sec. 5.3), and area weighting (AW, Sec. 5.4). These link the pixel level segmentation to the proposal level score. In each scheme, two scoring functions are proposed separately on thing and stuff maps. We combine the three schemes to provide an even better proposal score to help MIL (Sec 5.5).

Scoring schemes. LW transfers the similarity and co-occurrence relations as weighting functions to the thing and stuff label maps, respectively. Since we do not have stuff annotations on $\mathcal{B}$, we conduct the co-occurrence knowledge transfer as a second-order transfer by finding the target class' most similar thing class in $\mathcal{A}$. We believe that the target class should appear against a similar background with its most similar class. For example, in Fig. 1 target class bear's most similar class in $\mathcal{A}$ is cat, LW up-weights the cat score on T and its frequently co-occurring tree score on S.

LW favours small proposals with high weighted scores. To counter this effect, we introduce the CW score. It measures the dissimilarity of a proposal to its surroundings, measured on the thing/stuff score maps (Fig. 3). CW up-weights proposals that are more likely to contain an entire object in T or an entire stuff region in S.

Finally, the AW score encourages proposals to incorporate as much as possible of the connected components of pixels on a target's K most similar classes in $\mathcal{A}$ (*e.g.* Fig. 1: the cat area in the T map). While CW favors objects in general, AW focuses on objects of the target class in particular.

4. Acquiring knowledge from the source $\mathcal{A}$

4.1. Segmentation model

We employ the popular fully convolutional network (FCN-16s) [41] to train an end-to-end semantic segmentation model on both thing and stuff classes of $\mathcal{A}$. Given a new image, the FCN model is able to predict a likelihood distribution over all classes at each pixel. Notice that the FCN model is first pretrained for image classification on ILSVRC 2012 [24], then fine-tuned for semantic segmentation on $\mathcal{A}$. While it is possible that some of the target classes are seen during pretraining, only image-level labels are used. Therefore the weakly supervised setting still holds for the target classes.

4.2. Similarity relations

We compute the thing class similarities $V(a^t, b^t)$ between any thing class pair (a^t, b^t). We propose two similarity measures to compute V as follows:

Appearance similarity. Every image in $\mathcal{A}$ or $\mathcal{B}$ is represented by a 4096-dimensional CNN feature vector covering the whole image, using the output of the fc7 layer of the AlexNet CNN architecture [30]. The similarity of two images is the inner product of their feature vectors. The similarity $V_{\text{APP}}(a^t, b^t)$ is therefore the average similarity between images in a^t and images in b^t.

Semantic similarity. We compute the commonly used Lin [42] similarity $V_{\text{SEM}}(a^t, b^t)$ between two nouns b^t and a^t in the WordNet hierarchy [43].

4.3. Co-occurrence relation

We denote by $U(a^s, a^t)$ the co-occurrence frequency of any stuff and thing class pair (a^s, a^t) in $\mathcal{A}$. This frequency is computed and normalized over all the images in $\mathcal{A}$.

5. Transferring knowledge to the target $\mathcal{B}$

This section transfers the source knowledge to the target set $\mathcal{B}$. In this set, we have access only to image level labels, but no location annotations. We call the classes that are listed on the image level label list *target classes*. Given a new image of class b^t, we first use the FCN model trained on $\mathcal{A}$ to generate the thing (T) and stuff (S) segmentations separately (Sec. 5.1). Then we introduce three proposal scoring schemes to propagate the information from pixel level to proposal level (Sec. 5.2 - 5.4). Finally we combine the three scoring schemes into a single window score (Sec. 5.5). The scoring scheme parameters are learned in Sec. 5.6.

5.1. Generating thing and stuff segmentations

We apply the trained FCN model (Sec. 4.1) to a target image in $\mathcal{B}$. Usually, the output semantic segmentation is obtained by maximizing over all the class scores at each pixel [41, 44, 45, 46, 47, 48]. In this paper, we instead generate two output segmentations, one for things T and one for stuff S. We denote i as the i-th pixel in the image. We use $R^T = \{r_i^T\}$ and $L^T = \{l_i^T\}$ to denote the score (R) and label (L) maps for T. They are generated by keeping the maximum score and the corresponding label over all the thing classes $\mathcal{A}^T$ at each pixel i. Similar to R^T and L^T, $R^S = \{r_i^S\}$ and $L^S = \{l_i^S\}$ are generated by keeping the maximum score over all the stuff classes $\mathcal{A}^S$ at each pixel.

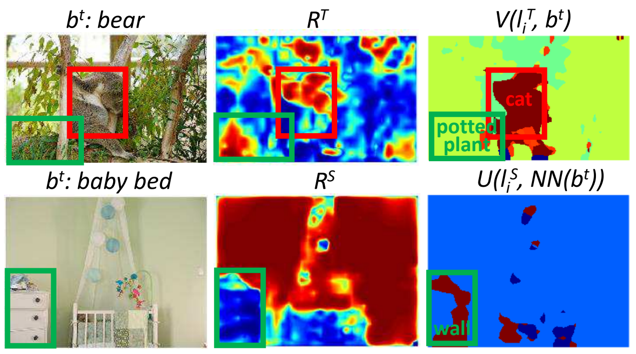

Figure 2: Label weighting example. Top: thing label weighting (class bear); bottom: stuff label weighting (class baby bed). R^T and R^S denote the thing and stuff score heatmaps, respectively; while $V(l_i^T, b^t)$ and $U(l_i^S, \mathrm{NN}(b^t))$ denote the thing and stuff label weighting heatmaps. We illustrate some proposals in each image. We print the dominantly predicted labels in the proposals to show how label weighting favours b^t's NN class in thing maps and its frequently co-occurring stuff class in stuff maps.

Fig. 1 shows an example of a bear image (target). The thing and stuff maps are produced by the semantic segmentation model. The R heatmaps indicate the probability of assigning a certain thing or stuff label to each pixel. Building upon these heatmaps, we propose three proposal scoring schemes to link the pixel level result to the proposal level score (Sec. 5.2 - 5.4). These try to give high scores to proposals containing the target class.

5.2. Label weighting (LW)

Because bear is more similar to cat than to table, we want to up-weight the proposal area in the thing map if it is predicted as cat. Meanwhile, because bear frequently appears against tree, we also want to up-weight the proposal area in the stuff map if it is predicted as tree. To do this, we transfer the knowledge of similarity and co-occurrence relations acquired in the source to the target class (bear), and use both relations to modulate the segmentation scores in T and S. Both relations and segmentation scores play a role in the label weighting proposal scoring scheme.

Thing label weighting. We can generate a thing label weighting map depending on how close the predicted class l_i^T at pixel i in L^T is to the target class b^t. The thing label (l_i^T) weight is given by the class similarity score $V(l_i^T, b^t)$ (Sec. 4.2). In Fig. 1 the target class bear is more similar to cat than to table. If a pixel is predicted as cat, then we assign a high label weight, otherwise we assign a low one.

Stuff label weighting. We do not have stuff annotations in $\mathcal{B}$. To conduct the stuff label weighting, we first find b^t's most similar thing class in $\mathcal{A}^T$ according to a similarity relation V (we denote it by $\mathrm{NN}(b^t)$). We believe that b^t should appear against a similar background (stuff) as its most similar thing class $\mathrm{NN}(b^t)$. We employ the co-occurrence frequency $U(l_i^S, \mathrm{NN}(b^t))$ of $\mathrm{NN}(b^t)$ as the corresponding stuff label weight for l_i^S at pixel i as stuff label weighting L^S.

In Fig. 1, cat frequently co-occurs with trees, and so does bear. So, if a certain pixel is predicted as tree, it gets assigned a high stuff label weight.

Proposal scoring. To score the proposals in an image, we multiply the label weights $V(l_i^T, b^t)$ and $U(l_i^S, \mathrm{NN}(b^t))$ with the segmentation scores r_i^T and r_i^S at each pixel. The weighting scheme is conducted separately on T and S. Given a window proposal w, we average the weighted scores inside w:

$$\begin{aligned} \mathrm{LW}^t(w, \alpha^t) &= f\left(\tfrac{1}{|w|} \textstyle\sum_{i \in w} r_i^T V(l_i^T, b^t), \alpha^t\right) \\ \mathrm{LW}^s(w, \alpha^s) &= f\left(\tfrac{1}{|w|} \textstyle\sum_{i \in w} r_i^S U(l_i^S, \mathrm{NN}(b^t)), \alpha^s\right) \end{aligned} \quad (1)$$

where $|w|$ denotes the size of w (area in pixels). We apply an exponential function $f(x) = \exp(\alpha \cdot x)$ to both thing and stuff LWs, α^t and α^s are the parameters.

Fig. 2 offers two examples (bear and baby bed) for our thing and stuff label weighting schemes. The red proposal in the top row is mostly classified as a cat and the green proposal as a potted plant. Both proposals have high scores in the thing score map R^T, but the red proposal has a higher thing label weight $V(l_i^T, b^t)$, because cat is more similar to bear than to potted plant. In contrast, the green proposal in the bottom row has low scores in R^S but a high label weight $U(l_i^T, \mathrm{NN}(b^t))$, as baby bed co-occurs more frequently with wall.

Notice that the thing label weighting can be viewed as a first-order transfer where the information goes directly from the source thing classes to the target thing classes. Instead, the stuff label weighting can be viewed as second-order transfer where the information first goes from the source stuff classes to the source thing classes, and then to the target thing classes. To the best of our knowledge, such second-order transfer has not been proposed before.

5.3. Contrast weighting (CW)

The LW scheme favours small proposals with high label weights, which typically cover only part of an object (top right image in Fig. 1). To counter this effect, contrast weighting (CW) measures the dissimilarity of a proposal to its immediate surrounding area on the thing/stuff score maps. It up-weights proposals that are more likely to contain an entire object or an entire stuff region.

The surrounding $Surr(w, \theta)$ of a proposal w is a rectangular ring obtained by enlarging it by a factor θ in all directions [5] (Fig. 3, the yellow ring). The CW between a window and its surrounding ring is computed as the Chi-square distance between their score map (R) histograms $h(\cdot)$

$$\mathrm{CW}(w, \theta) = \chi^2(h(w), h(Surr(w, \theta))) \quad (2)$$

We apply the CW scheme on both R^T and R^S and obtain $\mathrm{CW}^t(w, \theta^t)$ and $\mathrm{CW}^s(w, \theta^s)$. In Fig. 3 the red proposal has a higher CW^t score compared to the green one.

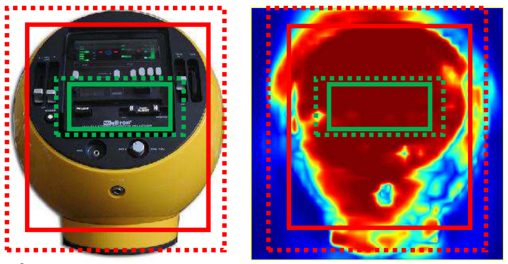
Figure 3: Contrast weighting example. An image of a tape player and some of its window proposals (left). CW^t is computed on the thing score map R^t (right). The red proposal has a higher contrast CW^t (with its surrounding dashed ring) than the green one.

5.4. Area weighting (AW)

Thing area weighting. Fig. 4 gives an example of an electric fan and its semantic segmentation map. Its 3-NN classes in terms of appearance similarity (Sec. 4.2) are table, chair and people. Between the white and yellow proposals, the CW scheme gives a bigger score to the white one, because its contrast is high. Instead, the yellow proposal incorporates most of the fan area, but is unfortunately predicted as table and chair. The thing area weighting scheme helps here boosting the yellow proposal's score. We find the K-NN classes of b^t in $\mathcal{A}^T$ by using one of the similarity measures in Sec. 4.2. Given a window w, we denote by $Area(w, b^t)$ the segment areas of any K-NN(b^t) inside w; while $Area(O(w), b^t)$ is the area that expands the current segments to their connected components inside and outside w. We measure the area ratio between the segments and their corresponding connected components:

$$Ratio^t(w) = \frac{Area(w, b^t)}{Area(O(w), b^t)} \tag{3}$$

If none of the K-NN classes occurs in w, we simply set $Ratio^t$ to zero. Throughout this paper, K is set to 3.

Stuff area weighting. In Fig. 4 among the three proposals, the green one is the best detection of the fan. However, its score is not the highest according to LW^t, CW^t and AW^t, as it contains some stuff area (wall) surrounding the fan. A bounding box usually has to incorporate some stuff area to fit an object tightly, as objects are rarely perfectly rectangle-shaped. We propose to up-weight a window w if stuff occupies a small but non-zero fraction of the window. We denote with $Ratio^s(w)$ the percentage of stuff pixels in window w.

For thing and stuff area weighting we apply a cumulative distribution function (CDF) of the normal distribution

$$\begin{aligned} \mathrm{AW}^t(w, \mu^t, \sigma^t) &= \mathrm{CDF}(Ratio^t(w) | \, \mu^t, \sigma^t) \\ \mathrm{AW}^s(w, \mu^s, \sigma^s) &= \mathrm{CDF}(Ratio^s(w) | \, \mu^s, \sigma^s) \end{aligned} \tag{4}$$

where μ^t and σ^t are the mean and standard deviation. We choose $\mu^t = \mu^s = 0$ and σ^t, σ^s are free parameters (Sec. 5.6).

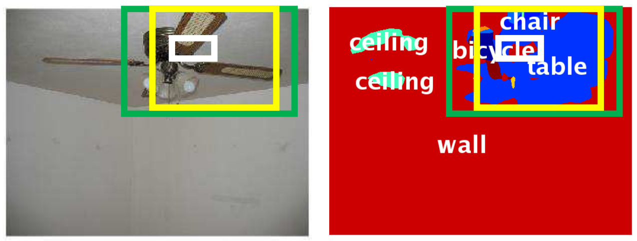

Figure 4: Area weighting example. An image of an electric fan (left) and its semantic segmentation (right). Thing area weighting favours the yellow proposal compared to the white one, as it incorporates most of the the connected component area of table and chair. Stuff area weighting further favours the green proposal as it allows certain stuff area in a proposal as the surrounding area of electric fan.

5.5. Combining the scoring schemes

For each proposal in an image, the above scoring schemes can be independently computed, each on the thing and stuff map. The scoring schemes tackle different problems, and are complementary to each other. This sections combines them to give our final TST (things and stuff transfer) window score W.

All the scoring functions on the thing map are multiplied together as a thing score $W^t = \mathrm{LW}^t * \mathrm{CW}^t * \mathrm{AW}^t$. This gives a higher score if a proposal mostly contains a target thing labeled as present in that image. Similarly, we have the stuff score $W^s = \mathrm{LW}^s * \mathrm{CW}^s * \mathrm{AW}^s$, which gives a higher score if a proposal mostly contains stuff. To combine the thing and stuff scores, we simply subtract W^s from W^t

$$W = W^t - W^s \tag{5}$$

5.6. Parameter learning

In the WSOL setting, we do not have the ground truth bounding box annotations in the target set $\mathcal{B}$. Thus we learn the score parameters α^t, α^s, θ^t, θ^s, σ^t and σ^s on the source set $\mathcal{A}$, where we have ground truth. We train the semantic segmentation model on the *train* set of $\mathcal{A}$, and then apply it to the *val* set of $\mathcal{A}$. For each image in the *val* set, we rank all its proposals using (5). We jointly learn the score parameters by maximizing the performance over the entire validation set.

6. Overall system

In WSOL, given the target training set in $\mathcal{B}$ with image level labels, the goal is to localize the object instances in it and to train good object detectors for the target test set. We explain here how we build a complete WSOL system by building on a MIL framework and incorporating our transfer cues into it.

Basic MIL. We build a Basic MIL pipeline as follows. We represent each image in the target set $\mathcal{B}$ as a bag of object proposals extracted using Edge Boxes [7]. They return

Method	APP	SEM
Basic MIL	39.7	
DT ≈ [21] (transfer only)	15.0	-
DT + MIL ≈ [21] (full)	39.5	-
TST	46.7	46.0
Basic MIL + Objectness [7]	47.6	
DT ≈ [21] (transfer only)	34.5	-
DT + MIL ≈ [21] (full)	49.1	-
TST	52.7	52.5
Deep MIL + Objectness [7]	48.4	
TST	54.0	53.8
TST + ILSVRC-dets	-	**55.1**

Table 1: CorLoc on ILSVRC-20; DT: direct transfer; DT+MIL: direct transfer plus MIL. TST is our method; ILSVRC-dets: Sec. 7.2, last paragraph. The transfers are guided by either the semantic (SEM) or the appearance (APP) class similarity.

about 5,000 proposals per image, likely to cover all objects. Following [29, 8, 13, 14, 28], we describe the proposals by the output of the fc7 layer of the AlexNet CNN architecture [30]. The CNN model is pre-trained for whole-image classification on ILSVRC [24], using the Caffe implementation [49]. This produces a 4,096-dimensional feature vector for each proposal. Based on this feature representation for each target class, we iteratively build an SVM appearance model (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 positive set which contains the current selection of one instance from each positive image. (2) Re-training: we train the SVM using the current selection of positive samples, and all proposals from the negative images as negative samples. As in [10, 12, 4, 19, 17], we also linearly combine the SVM score with a general measure of objectness [5, 7]. This leads to a higher MIL baseline.

Incorporating things and stuff transfer (TST). We incorporate our things and stuff transfer (TST) into Basic MIL by linearly combining the SVM score with our proposal scoring function (5). Note how the behavior of (5) depends on the class similarity measure used within it (either appearance or semantic similarity, Sec. 4.2).

Deep MIL. Basic MIL uses an SVM on top of fixed deep features as the appearance model. Now we change the model to fine-tune all layers of the deep network during the re-training step of MIL. We take the output of Basic MIL as an initialization for two additional MIL iterations. During these iterations, we use Fast R-CNN [1].

7. Experiments

7.1. Datasets and evaluation protocol

We use one source set $\mathcal{A}$ (PASCAL Context) and several different target sets $\mathcal{B}$ in turn (ILSVRC-20, COCO-07 and PASCAL VOC 2007). Each target set contains a training set and a test set. We perform WSOL on the target training set to localize objects within it. Then we train a Fast R-CNN [2] detector from it and apply it on the target test set.

Evaluation protocol. We quantify localization performance in the target training set with the CorLoc measure [9, 4, 10, 50, 28, 27]. We quantify object detection performance on the target test set using mean average precision (mAP). As in most previous WSOL methods [8, 9, 3, 4, 10, 11, 12, 13, 14, 28], our scheme returns exactly one bounding-box per class per training image. 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).

Source set: PASCAL Context. PASCAL Context [51] augments PASCAL VOC 2010 [26] with class labels at every pixel. As in [51], we select the 59 most frequent classes. We categorize them into *things* and *stuff*. There are 40 thing classes, including the original 20 PASCAL classes and new classes such as book, cup and window. There are 19 stuff classes, such as sky, water and grass. We train the semantic segmentation model (Sec. 4.1) on the *train* set of $\mathcal{A}$ and set the score parameters (Sec. 5.6) on the *val* set, using the 20 PASCAL classes from $\mathcal{A}$ as targets.

Target set: ILSVRC-20. The ILSVRC [24] dataset originates from the ImageNet dataset [52], but is much harder [24]. As the target training set we use the *train60k* subset [29] of ILSVRC 2014. As the target test set we use the 20k images of the validation set. To conduct WSOL on *train60k*, we carefully select 20 target classes: ant, babybed, basketball, bear, burrito, butterfly, cello, coffee-maker, electric-fan, elephant, goldfish, golfcart, monkey, pizza, rabbit, strainer, tape-player, turtle, waffle-iron and whale. ILSVRC-20 contains 3,843 target training set images and 877 target test set images. This selection is good because: (1) they are visually considerably different from any source class; (2) they appear against similar background classes as the source classes, so we can show the benefits of stuff transfer; (3) they are diverse, covering a broad range of object types.

Target set: COCO-07. The COCO 2014 [25] dataset has fewer object classes (80) than ILSVRC (200), but more instances. COCO is generally more difficult than ILSVRC for detection, as objects are smaller [25]. There are also more instances per image: 7.7 in COCO compared to 3.0 in ILSVRC [25]. We select 7 target classes to carry out WSOL: apple, giraffe, kite, microwave, snowboard, tennis racket and toilet. COCO-07 contains 11,489 target training set images and 5,443 target test set images.

Target set: PASCAL VOC 2007. The PASCAL VOC 2007 [40] dataset is one of the most important object detection datasets. It includes 5,011 training (*trainval*) images and 4,952 test images, which we directly use as our target

Class	ant	bbed	bask	bear	burr	butt	cell	cmak	efan	elep	gfis	gcar	monk	pizz	rabb	stra	tpla	turt	wiro	whal	Avg.	Avg.(8)
LSDA [20]	-	-	-	-	-	-	-	-	-	-	-	-	22.9	27.6	40.2	6.8	19.1	31.9	8.6	**20.3**	-	22.2
Deep MIL+Obj.	39.2	24.2	0.2	13.0	16.5	28.9	29.7	8.9	**39.1**	34.4	9.1	40.3	18.0	29.7	32.8	19.6	27.0	27.0	5.9	2.9	22.3	20.4
+TST (APP)	**39.9**	**31.0**	0.6	16.8	11.3	32.2	32.0	6.0	34.9	38.4	**13.6**	**65.1**	23.8	32.5	40.7	**24.8**	28.6	25.1	**9.9**	5.1	25.6	23.8
+TST (SEM)	34.1	26.8	0.6	19.7	16.8	31.7	**32.6**	8.6	31.2	37.2	11.5	57.8	22.9	31.2	**45.2**	18.7	**30.3**	28.2	8.1	6.2	25.0	23.9
+ ILSVRC-dets	34.1	24.7	**3.3**	**21.5**	**18.6**	**35.1**	**32.6**	**9.1**	32.9	**38.8**	11.1	58.5	**24.5**	**33.9**	44.5	18.4	28.4	**32.1**	9.9	5.7	**25.9**	**24.7**

Table 2: mAP Performance on the test set of ILSVRC-20. All our methods start from DeepMIL with objectness. For comparison we also show the performance on the 8 classes common to our target set and that of LSDA [20].

training set and target test set, respectively. For our experiments we use all 20 thing classes in VOC 2007. Since the thing classes in our source set (PASCAL Context) overlap with those of VOC 2007, when doing our TST transfer to a target class we remove it from the sources. For example, when we transfer to "dog" in VOC 2007, we remove "dog" from the FCN model trained on PASCAL Context.

7.2. ILSVRC-20

Table 1 presents results for our method (TST) and several alternative methods on ILSVRC-20.

Our transfer (TST). Our results (TST) vary depending on the underlying class similarity measure used, either appearance (APP) or semantic (SEM) (Sec. 4.2). TST (APP) leads to slightly better results than TST (SEM). We achieve a $+7\%$ improvement in CorLoc (46.7) compared to Basic MIL without objectness, and $+5\%$ improvement (52.7) over Basic MIL with objectness. Hence, our transfer method is effective, and is complementary to objectness. Fig. 5 shows example localizations by Basic MIL with objectness and TST (APP).

Comparison to direct transfer (DT). We compare here to a simpler way to transfer knowledge. We train a fully supervised object detector for each source thing class. Then, for every target class we find the most similar source class from the 40 PASCAL Context thing classes, and use it to directly detect the target objects. For the appearance similarity measure (APP) all NN classes of ILSVRC-20 are part of PASCAL VOC and PASCAL Context. Therefore we have bounding box annotations for these classes. However, for the semantic similarity measure (SEM) not all NN classes of ILSVRC-20 are part of PASCAL VOC. Therefore we do not have bounding box annotations for these classes and cannot apply DT. DT is similar to the 'transfer only' method in [21] (see Sec. 4.2 and Table 2 in [21]).

As Table 1 shows, the results are quite poor as the source and target classes are visually quite different, *e.g.* the most similar class to ant according to APP is bird; while for waffle-iron, it is table; for golfcart, it is person. This shows that the transfer task we address (from PASCAL Context to ILSVRC-20) is challenging and cannot be solved by simply using object detectors pre-trained on the source classes.

Comparison to direct transfer with MIL (DT+MIL). We improve the direct transfer method by using the DT detector to score all proposals in a target image, and then combining this score with the standard SVM score for the target class during the MIL re-localization step. This is very similar to the full method of [21] and is also close to [19]. The main difference from [21] is that we train the target class' SVM model in an MIL framework (Sec. 6), whereas [21] simply trains it by using proposals with high objectness as positive samples.

As Table 1 shows, DT+MIL performs substantially better than DT alone, but it only slightly exceeds MIL without transfer, again due to the source and target classes being visually different ($+1.5\%$ over Basic MIL with objectness). Importantly, our method (TST) achieves higher results, demonstrating that it is a better way to transfer knowledge ($+5\%$ over Basic MIL with objectness).

Deep MIL. As Table 1 shows, Deep MIL improves slightly over Basic MIL (from 47.6 to 48.4, both with objectness). When built on Deep MIL, our TST transfer raises CorLoc to 54.0 (APP) and 53.8 (SEM), a $+5\%$ improvement over Deep MIL (confirming what we observed when building on Basic MIL). Table 2 shows the mAP of Deep MIL and our method (TST) on the test set. The observed improvements in CorLoc on the training set nicely translate to better mAP on the test set ($+3.3\%$ over Deep MIL).

Comparison to LSDA [20]. We compare to LSDA [20], which trains fully supervised detectors for 100 classes of the ILSVRC 2013 dataset (sources) and transfers to the other 100 classes (targets). We report in Table 2 the mAP on the 8 classes common to both their target set and ours. On these 8 classes, we improve on [20] by $+1.7\%$ mAP while using a substantially smaller source set (5K images in PASCAL Context, compared to 105K images in their 100 source classes from ILSVRC 2013).

Furthermore, we can also incorporate detectors for their 100 source classes in our method, in a similar manner as for the DT+MIL method. For each target class we use the detector of the 3 most similar source classes as a proposal scoring function during MIL's re-localization step. We choose the SEM measure to guide the transfer as it is fast to compute. This new scoring function is referred to as ILSVRC-dets in Table 1 and 2. When using the ILSVRC-dets score, our mAP improves further, to a final value $+2.5\%$ better than LSDA [20].

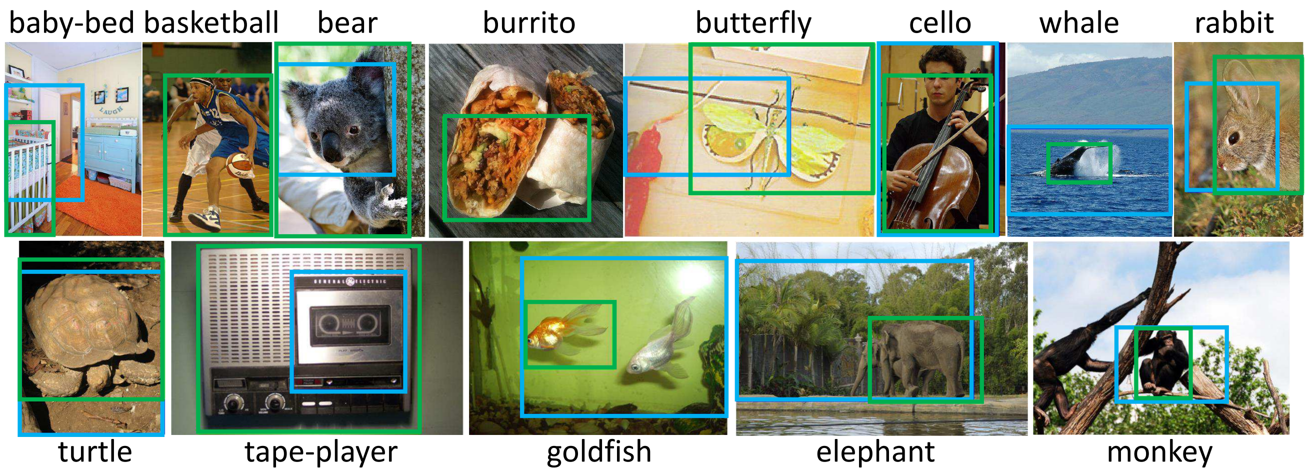

Figure 5: We show localizations on ILSVRC-20 of Basic MIL with objectness (blue) and our TST (APP) method (green).

Method	training (CorLoc)	test (mAP)
Deep MIL + Obj.	15.8	9.1
+TST (SEM)	18.0	11.0
+TST (APP)	**18.8**	**11.3**

Table 3: CorLoc and mAP on COCO-07. Objectness [7] is added on top of the baseline. TST (SEM) and TST (APP) are separately added to the baseline with objectness.

7.3. COCO-07

Table 3 presents results on COCO-07, which is a harder dataset. Compared to Deep MIL with objectness, our transfer method improves CorLoc by +3.0% and mAP by +2.2% (APP).

7.4. PASCAL VOC 2007

Table 4 presents results on PASCAL VOC 2007. As our baseline system, we use both objectness and multifolding [4] in Deep MIL. This performs at 50.7 CorLoc and 28.1 mAP. Our transfer method TST strongly improves CorLoc to 59.9 (+9.2%) and mAP to 33.8 (+5.7%).

Comparison to [21]. They present results on this dataset in a transfer setting, by using detectors trained in a fully supervised setting for all 200 classes of ILSVRC (excluding the target class). Adopting their protocol, we also use those detectors in our method (analog to the LSDA comparison above). This leads to our highest CorLoc of 60.8, which outperforms [21], as well as state-of-the-art WSOL works [28, 27, 4] (which do not use such transfer). For completeness, we also report the corresponding mAPs. Our mAP 34.5 matches the result of [27] based on their 'S' neural network, which corresponds to the AlexNet we use. They propose an advanced WSOL technique that integrates both recognition and detection tasks to jointly train a weakly supervised deep network, whilst we build on a weaker MIL system. We believe our contributions are complementary: we could incorporate our TST transfer cues into their WSOL technique and get even better results.

Method	ILSVRC-dets	CorLoc	mAP
Wang et al. [28]		48.5	31.6
Bilen and Vedaldi [27] (S)		54.2	**34.5**
Cinbis et al. [4]		54.2	28.6
Rochan and Wang [21]	✓	58.8	-
Deep MIL + Obj. + MF		50.7	28.1
+TST (SEM)		59.9	33.8
+TST (SEM)	✓	**60.8**	**34.5**

Table 4: Performance on PASCAL VOC 2007. We start from Deep MIL with objectness [7] and multifolding [4] as a baseline. Then we add our method TST (SEM) to it. Rochan and Wang [21] do not report mAP. (S) denotes the S model (roughly AlexNet) in [27], which corresponds to the network architecture we use in all experiments. ILSVRC-dets indicates using detectors trained from ILSVRC during transfer.

Finally, we note that our experimental protocol guarantees no overlap in either images nor classes between source and target sets (Sec. 7.1). However, in general VOC 2007 and PASCAL Context (VOC 2010) share similar attributes, which makes this transfer task easier in our setting.

8. Conclusion

We present weakly supervised object localization using things and stuff transfer. We transfer knowledge by training a semantic segmentation model on the source set and using it to generate thing and stuff maps on a target image. Class similarity and co-occurrence relations are also transferred and used as weighting functions. We devise three proposal scoring schemes on both thing and stuff maps and combine them to produce our final TST score. We plug the score into an MIL pipeline and show significant improvements on the ILSVRC-20, VOC 2007 and COCO-07 datasets. We compare favourably to two previous transfer works [21, 20].

Acknowledgements. Work supported by the ERC Starting Grant VisCul.

References

[1] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *CVPR*, pages 437–446, 2015. 1, 6

[2] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 1, 6

[3] R.G. Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014. 1, 6

[4] R.G. Cinbis, J. Verbeek, and C. Schmid. Weakly supervised object localization with multi-fold multiple instance learning. *IEEE Trans. on PAMI*, 2016. 1, 2, 6, 8

[5] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, 2010. 1, 3, 4, 6

[6] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 2013. 1, 3

[7] P. Dollar and C. Zitnick. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 1, 3, 5, 6, 8

[8] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with posterior regularization. In *BMVC*, 2014. 1, 6

[9] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with convex clustering. In *CVPR*, 2015. 1, 2, 6

[10] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In *ECCV*, 2010. 1, 2, 6

[11] O. Russakovsky, Y. Lin, K. Yu, and L. Fei-Fei. Object-centric spatial pooling for image classification. In *ECCV*, 2012. 1, 6

[12] P. Siva and T. Xiang. Weakly supervised object detector learning with model drift detection. In *ICCV*, 2011. 1, 2, 6

[13] H.O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darell. On learning to localize objects with minimal supervision. In *ICML*, 2014. 1, 2, 6

[14] H.O. Song, Y.J. Lee, S. Jegelka, and T. Darell. Weakly-supervised discovery of visual pattern configurations. In *NIPS*, 2014. 1, 6

[15] M. Shi and V. Ferrari. Weakly supervised object localization using size estimates. In *ECCV*, 2016. 1, 2

[16] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Trans. on PAMI*, 2012. 1, 2

[17] K. Tang, A. Joulin, L-J. Li, and L. Fei-Fei. Co-localization in real-world images. In *CVPR*, 2014. 1, 2, 6

[18] Z. Shi, P. Siva, and T. Xiang. Transfer learning by ranking for weakly supervised object annotation. In *BMVC*, 2012. 1, 2

[19] M. Guillaumin and V. Ferrari. Large-scale knowledge transfer for object localization in imagenet. In *CVPR*, 2012. 1, 2, 6, 7

[20] J. Hoffman, S. Guadarrama, E. Tzeng, R. Hu, and J. Donahue. LSDA: Large scale detection through adaptation. In *NIPS*, 2014. 1, 2, 7, 8

[21] M. Rochan and Y. Wang. Weakly supervised localization of novel objects using appearance transfer. In *CVPR*, 2015. 1, 2, 6, 7, 8

[22] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008. 1, 2

[23] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011. 1, 2

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 2015. 1, 2, 3, 6

[25] T-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C.L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 1, 2, 6

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 2010. 1, 6

[27] H. Bilen and A. Vedaldi. Weakly supervised deep detection networks. In *CVPR*, 2016. 1, 2, 6, 8

[28] C. Wang, W. Ren, J. Zhang, K. Huang, and S. Maybank. Large-scale weakly supervised object localization via latent category learning. *IEEE Transactions on Image Processing*, 24(4):1371–1385, 2015. 1, 2, 6, 8

[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2, 6

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 3, 6

[31] S. K. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. on KDE*, 2010. 2

[32] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011. 2

[33] Y. Aytar and A. Zisserman. Enhancing exemplar svms using part level transfer regularization. In *BMVC*, 2012. 2

[34] T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *CVPR*. IEEE, 2010. 2

[35] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 2

[36] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What helps where - and why? semantic relatedness for knowledge transfer. In *CVPR*, 2010. 2

[37] P. Ott and M. Everingham. Shared parts for deformable part-based models. In *CVPR*, 2011. 2

[38] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009. 2

[39] D. Kuettel, M. Guillaumin, and V. Ferrari. Segmentation Propagation in ImageNet. In *ECCV*, 2012. 2

[40] M. Everingham, S. Eslami, L. van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes challenge: A retrospective. *IJCV*, 2015. 2, 6

[41] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 3

[42] D. Lin. An information-theoretic definition of similarity. In *ICML*, 1998. 3

[43] C. Fellbaum. Wordnet: An on-line lexical database, 1998. 3

[44] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015. 3

[45] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 3

[46] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Trans. on PAMI*, 35(8):1915–1929, 2013. 3

[47] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015. 3

[48] P. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene parsing. In *ICML*, 2014. 3

[49] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. `http://caffe.berkeleyvision.org/`, 2013. 6

[50] Z. Shi, T.M. Hospedales, and T. Xiang. Bayesian joint modelling for object localisation in weakly labelled images. *IEEE Trans. on PAMI*, 2015. 6

[51] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014. 6

[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 6