

Pairwise Similarity Knowledge Transfer for Weakly Supervised Object Localization

Amir Rahimi^{*1}, Amirreza Shaban^{*2}, Thalaiyasingam Ajanthan¹, Richard Hartley^{1,3}, and Byron Boots⁴

¹ANU, ACRV , ² Georgia Tech, ³ Google Research, ⁴University of Washington
Corresponding authors: amir.rahimi@anu.edu.au, ashaban@uw.edu.

^{*}Authors contributed equally.

Abstract. Weakly Supervised Object Localization (WSOL) methods only require image level labels as opposed to expensive bounding box annotations required by fully supervised algorithms. We study the problem of learning localization model on target classes with weakly supervised image labels, helped by a fully annotated source dataset. Typically, a WSOL model is first trained to predict class generic objectness scores on an off-the-shelf fully supervised source dataset and then it is progressively adapted to learn the objects in the weakly supervised target dataset. In this work, we argue that learning only an objectness function is a weak form of knowledge transfer and propose to learn a classwise pairwise similarity function that directly compares two input proposals as well. The combined localization model and the estimated object annotations are jointly learned in an alternating optimization paradigm as is typically done in standard WSOL methods. In contrast to the existing work that learns pairwise similarities, our approach optimizes a unified objective with convergence guarantee and it is computationally efficient for large-scale applications. Experiments on the COCO and ILSVRC 2013 detection datasets show that the performance of the localization model improves significantly with the inclusion of pairwise similarity function. For instance, in the ILSVRC dataset, the Correct Localization (CorLoc) performance improves from 72.8% to 78.2% which is a new state-of-the-art for WSOL task in the context of knowledge transfer.

Keywords: Weakly supervised object localization, transfer learning, multiple instance learning, object detection.

1 Introduction

Weakly Supervised Object Localization (WSOL) methods have gained a lot of attention in computer vision [1–9]. Despite their supervised counterparts [10–14] that require the object class and their bounding box annotations, WSOL methods only require the image level labels indicating presence or absence of object classes. In spite of major improvements [1,5] in this area of research, there is still a large performance gap between weakly supervised and fully supervised object localization algorithms. In a successful attempt, WSOL methods are adopted to use an already annotated object detection dataset, called source dataset, to

improve the weakly supervised learning performance in new classes [4,15]. These approaches learn transferable knowledge from the source dataset and use it to speed up learning new categories in the weakly supervised setting.

Multiple Instance Learning (MIL) methods like MI-SVM [16] are the predominant methods in weakly supervised object localization [1,5,6]. Typically, images are decomposed into bags of object proposals and the problem is posed as selecting one proposal from each bag that contains an object class. MIL methods take advantage of alternating optimization to progressively learn a classwise objectness (unary) function and the optimal selection in re-training and re-localization steps, respectively. Typically, the source dataset is used to learn an initial generic objectness function which is used to steer the selection toward objects and away from background proposals [4,15,17–20]. However, solely learning an objectness measure is a sub-optimal form of knowledge transfer as it can only discriminate objects from background proposals, while it is unable to discriminate between different object classes. Deselaers *et al.* [7] propose to additionally learn a pairwise similarity function from the fully annotated dataset and frame WOSL as a graph labeling problem where nodes represent bags and each proposal corresponds to one label for the corresponding node. The edges which reflect the cost of wrong pairwise labeling are derived from the learned pairwise similarities. Additionally, they propose an ad-hoc algorithm to progressively adapt the scoring functions to learn the weakly supervised classes using alternating re-training and re-localization steps. Unlike the alternating optimization in MIL, re-training and re-localization steps in [7] does not optimize a unified objective and therefore the convergence of their method could not be guaranteed. Despite good performance on medium scale problems, this method is less popular especially in large scale problems where computing all the pairwise similarities is intractable.

In this work, we adapt the localization model in MIL to additionally learn a pairwise similarity function and use a two-step alternating optimization to jointly learn the augmented localization model and the optimal selection. In the re-training step, the pairwise and unary functions are learned given the current selected proposals for each class. In the re-localization step, the selected proposals are updated given the current pairwise and unary similarity functions. We show that with a properly chosen localization loss function, the objective in the re-localization step can be equivalently expressed as a graph labeling problem very similar to the model in [7]. We use the computationally effective iterated conditional modes (ICM) graph inference algorithm [21] in the re-localization step which updates the selection of one bag in each iteration. Unfortunately, the ICM algorithm is prone to local minimum and its performance is highly dependent on the quality of its initial conditions. Inspired by the recent work on few-shot object localization [22], we divide the dataset into smaller mini-problems and solve each mini-problem individually using TRWS [23]. We combine the solutions of these mini-problems to initialize the ICM algorithm. Surprisingly, we observe that initializing ICM with the optimal selection from mini-problems of small sizes considerably improves the convergence point of ICM.

Our work addresses the main disadvantages of graph labeling algorithm in [7]. First, we formulate learning pairwise and unary functions and updating the optimal proposal selections with graph labeling within a two-step alternating optimization framework where each step is optimizing a unified objective and the convergence is guaranteed. Second, we propose a computationally efficient graph inference algorithm which uses a novel initialization method combined with ICM updates in the re-localization step. Our experiments show our method significantly improves the performance of MIL methods in large-scale COCO [24] and ILSVRC 2013 detection [25] datasets. Particularly, our method sets a new state-of-the-art performance of 78.2% correct localization [7] for the WSOL task in the ILSVRC 2013 detection dataset¹.

2 Related Work

We review the MIL based algorithms among other branches in WSOL [17, 19]. These approaches exploit alternating optimization to learn a detector and the optimal selection jointly. The algorithm iteratively alternates between re-localizing the objects given the current detector and re-training the detector given the current selection. In the recent years, alternating optimization scheme combined with deep neural networks has been the state-of-the-art in WSOL [1, 2, 26]. However, due to the non-convexity of its objective function, this method is prone to local minimum which typically leads to sub-optimal results [27, 28] e.g. selecting the salient parts instead of the whole object. Addressing this issue has been the main focus of research in WSOL in the recent years [1, 5, 29]. In multi-fold [5], weakly supervised dataset is split into separate training and testing folds to avoid overfitting. Kumar *et al.* [29] propose an iterative self-paced learning algorithm that gradually learns from easy to hard samples to avoid getting stuck in bad local optimum points. Wan *et al.* [1] propose a continuation MIL algorithm to smooth out the non-convex loss function in order to alleviate the local optimum problem in a systematic way.

Transfer learning is another way to improve WSOL performance. These approaches utilize the information in a fully annotated dataset to learn an improved object detector on a weakly supervised dataset [4, 15, 18, 20]. They leverage the common visual information between object classes to improve the localization performance in the target weakly supervised dataset. In a standard knowledge transfer framework, the fully annotated dataset is used to learn a class agnostic objectness measure. This measure is incorporated during the alternating optimization step to steer the detector toward objects and away from the background [4]. Although the objectness measure is a powerful metric in differentiating between background and foreground, it fails to discriminate between different object classes. Several works have utilized pairwise similarity measure for improving WSOL [7, 19, 22]. Shaban *et al.* [22] use a relation network to predict pairwise similarity between pairs of proposals in the context of few-shot object

¹ Source code is available on <https://github.com/AmirooR/Pairwise-Similarity-knowledge-Transfer-WSOL>

co-localization. Deselaers *et al.* [7] frame WSOL as a graph labeling problem with pairwise and unary potentials and progressively adapt the potential functions to learn weakly supervised classes. Tang *et al.* [19] utilize the pairwise similarity between proposals to capture the inter-class diversity for the co-localization task. Hayder *et al.* [30, 31] use pairwise learning for object co-detection.

3 Problem Description and Background

We review the standard dataset definition and optimization method for the weakly supervised object localization problem [1, 4, 5, 7].

Dataset and Notation. Suppose each image is decomposed into a collection of object proposals which form a bag $\mathcal{B} = \{\mathbf{e}_i\}_{i=1}^m$ where an object proposal $\mathbf{e}_i \in \mathbb{R}^d$ is represented by a d -dimensional feature vector. We denote $y(\mathbf{e}) \in \mathcal{C} \cup \{c_\emptyset\}$ the label for object proposal \mathbf{e} . In this definition \mathcal{C} is a set of object classes and c_\emptyset denotes the background class. Given a class $c \in \mathcal{C}$ we can also define the binary label

$$y_c(\mathbf{e}) = \begin{cases} 1 & \text{if } y(\mathbf{e}) = c \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

With this notation a dataset is a set of bags along with the labels. For a weakly supervised dataset, only bag-level labels that denote the presence/absence of objects in a given bag are available. More precisely, the label for bag \mathcal{B} is written as $\mathcal{Y}(\mathcal{B}) = \{c \mid \exists \mathbf{e} \in \mathcal{B} \text{ s.t. } y(\mathbf{e}) = c \in \mathcal{C}\}$. Let $Y_c(\mathcal{B}) \in \{0, 1\}$ denote the binary bag label which indicates the presence/absence of class c in bag \mathcal{B} .

Given a weakly supervised dataset $\mathcal{D}_T = \{\mathcal{T}, \mathcal{Y}_T\}$ called the target dataset, with $\mathcal{T} = \{\mathcal{B}_j\}_{j=1}^N$ and corresponding bag labels $\mathcal{Y}_T = \{\mathcal{Y}(\mathcal{B})\}_{\mathcal{B} \in \mathcal{T}}$, the goal is to estimate the latent proposal unary labeling² \mathbf{y}_c for all object classes $c \in \mathcal{C}_T$ in the target set.

For ease of notation, we also introduce a pairwise labeling function between pairs of proposals. The pairwise labeling function $r : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \{0, 1\}$ is designated to output 1 when two object proposals belong to the same object class and 0 otherwise, *i.e.*,

$$r(\mathbf{e}, \mathbf{e}') = \begin{cases} 1 & \text{if } y(\mathbf{e}) = y(\mathbf{e}') \neq c_\emptyset \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Likewise, given a class c , two proposals are related under the class conditional pairwise labeling function $r_c : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \{0, 1\}$ if they both belong to class c . Similar to the unary labeling, since the pairwise labeling function is also defined over a finite set of variables, it can be seen as a vector. Unless we use the word vector or function, the context will determine whether we use the unary or pairwise labeling as a vector or a function. We use the “hat” notation to refer to the estimated (pseudo) unary or pairwise labeling by the weakly supervised learning algorithm.

² Notice, the labeling is a function defined over a finite set of variables, which can be treated as a vector. Here, \mathbf{y}_c denotes the vector of labels $y_c(\mathbf{e})$ for all proposals \mathbf{e} .

Multiple Instance Learning (MIL). In standard MIL [16], the problem is solved by jointly learning a unary score function $\psi_c^U : \mathbb{R}^d \rightarrow \mathbb{R}$ (typically represented by a neural network) and a feasible (pseudo) labeling $\hat{\mathbf{y}}_c$ that minimize the empirical unary loss

$$\mathcal{L}_c^U(\psi_c^U, \hat{\mathbf{y}}_c | \mathcal{T}) = \sum_{\mathcal{B} \in \mathcal{T}} \sum_{\mathbf{e} \in \mathcal{B}} \ell(\psi_c^U(\mathbf{e}), \hat{y}_c(\mathbf{e})), \quad (3)$$

where the loss function $\ell : \mathbb{R} \times \{0, 1\} \rightarrow \mathbb{R}$ measures the incompatibility between predicted scores $\psi_c^U(\mathbf{e})$ and the pseudo labels $\hat{y}_c(\mathbf{e})$. Here, likewise to the labeling, we denote the class score for all the proposals as a vector ψ_c^U . Note that the unary labeling $\hat{\mathbf{y}}_c$ is feasible if exactly one proposal has label 1 in each positive bag, and every other proposal has label 0 [5]. To this end, the set of feasible labeling \mathcal{F} can be defined as

$$\mathcal{F} = \left\{ \hat{\mathbf{y}}_c \mid \hat{y}_c(\mathbf{e}) \in \{0, 1\}, \sum_{\mathbf{e} \in \mathcal{B}} \hat{y}_c(\mathbf{e}) = Y_c(\mathcal{B}), \forall \mathcal{B} \in \mathcal{T} \right\}. \quad (4)$$

Finally, the problem is framed as minimizing the loss over all possible vectors ψ_c^U (*i.e.*, unary functions represented by the neural network) and the feasible labels $\hat{\mathbf{y}}_c$

$$\begin{aligned} & \min_{\psi_c^U, \hat{\mathbf{y}}_c} \mathcal{L}_c^U(\psi_c^U, \hat{\mathbf{y}}_c | \mathcal{T}), \\ & \text{s.t. } \hat{\mathbf{y}}_c \in \mathcal{F}. \end{aligned} \quad (5)$$

Optimization. This objective is typically minimized in an iterative two-step alternating optimization paradigm [32]. The optimization process starts with some initial value of the parameters and labels, and iteratively alternates between *re-training* and *re-localization* steps until convergence. In the re-training step, the parameters of the unary score function ψ_c^U are optimized while the labels $\hat{\mathbf{y}}_c$ are fixed. In the re-localization step, proposal labels are updated given the current unary scores. The optimization in the re-localization step is equivalent to assigning positive label to the proposal with the highest unary score within each positive bag and label 0 to all other proposals [16]. Formally, label of the proposal $\mathbf{e} \in \mathcal{B}$ in bag \mathcal{B} is updated as

$$\hat{y}_c(\mathbf{e}) = \begin{cases} 1 & \text{if } Y_c(\mathcal{B}) = 1 \text{ and } \mathbf{e} = \operatorname{argmax}_{\mathbf{e}' \in \mathcal{B}} \psi_c^U(\mathbf{e}') \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Knowledge Transfer. In this paper, we also assume having access to an auxiliary fully annotated dataset \mathcal{D}_S (source dataset) with object classes in \mathcal{C}_S which is a disjoint set from the target dataset classes, *i.e.*, $\mathcal{C}_T \cap \mathcal{C}_S = \emptyset$. In the standard practice [4, 18, 20], the source dataset is used to learn a class agnostic unary score $\psi^U : \mathbb{R}^d \rightarrow \mathbb{R}$ which measures how likely the input proposal \mathbf{e} tightly encloses a foreground object. Then, the unary score vector used in Eq. (6) is adapted to

$\psi_c^U \leftarrow \lambda\psi_c^U + (1 - \lambda)\psi^U$ for some $0 \leq \lambda \leq 1$. This steers the labeling toward choosing proposals that contain complete objects. Although the class agnostic unary score function ψ^U is learned on the source classes, since objects share common properties, it transfers to the unseen classes in the target set.

4 Proposed Method

In addition to learning the unary scores, we also learn a classwise pairwise similarity function $\psi_c^P : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ that estimates the pairwise labeling between pairs of proposals. That is for the target class c , pairwise similarity score $\psi_c^P(\mathbf{e}, \mathbf{e}')$ between two input proposals $\mathbf{e}, \mathbf{e}' \in \mathbb{R}^d$ has a high value if two proposals are related, *i.e.*, $\hat{r}_c(\mathbf{e}, \mathbf{e}') = 1$ and a low value otherwise. We define the empirical pairwise similarity loss to measure the incompatibility between pairwise similarity function predictions and the pairwise labeling $\hat{\mathbf{r}}_c$

$$\mathcal{L}_c^P(\psi_c^P, \hat{\mathbf{r}}_c | \mathcal{T}) = \sum_{\substack{\mathcal{B}, \mathcal{B}' \in \mathcal{T} \\ \mathcal{B} \neq \mathcal{B}'}} \sum_{\substack{\mathbf{e} \in \mathcal{B} \\ \mathbf{e}' \in \mathcal{B}'}} \ell(\psi_c^P(\mathbf{e}, \mathbf{e}'), \hat{r}_c(\mathbf{e}, \mathbf{e}')), \quad (7)$$

where ψ_c^P denotes the vector of the pairwise similarities of all pairs of proposals, and $\ell : \mathbb{R} \times \{0, 1\} \rightarrow \mathbb{R}$ is the loss function. We define the overall loss as the weighted sum of the empirical pairwise similarity and the unary loss

$$\mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}) = \alpha \mathcal{L}_c^P(\psi_c^P, \hat{r}_c | \mathcal{T}) + \mathcal{L}_c^U(\psi_c^U, \hat{y}_c | \mathcal{T}), \quad (8)$$

where $\psi_c = [\psi_c^U, \psi_c^P]$ is the vector of unary and pairwise similarity scores combined, and $\hat{\mathbf{z}}_c = [\hat{y}_c, \hat{\mathbf{r}}_c]$ denotes the concatenation of unary and pairwise labeling vectors, and $\alpha > 0$ controls the importance of the pairwise similarity loss.

We employ alternating optimization to jointly optimize the loss over the parameters of the scoring functions ψ_c^U and ψ_c^P (re-training) and labelings $\hat{\mathbf{z}}_c$ (re-localization). In re-training, the objective function is optimized to learn the pairwise similarity and the unary scoring functions from the pseudo labels. In re-localization, we use the current scores to update the labelings.

Training the model with fixed labels, *i.e.* re-training step, is straightforward and can be implemented within any common neural network framework. We use sigmoid cross entropy loss in both empirical unary and pairwise similarity losses

$$\ell(x, y) = -(1 - y) \log(1 - \sigma(x)) - y \log(\sigma(x)), \quad (9)$$

where $x \in \mathbb{R}$ is the predicted logit, $y \in \{0, 1\}$ is the label, and $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ denotes the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$. The choice of the loss function directly affects the objective function in the re-localization step. As we will show later, since sigmoid cross entropy loss is a linear function of label y it leads to a *linear objective function* in the re-localization step. To speed up the re-training step, we train pairwise similarity and unary scoring functions for all the classes together by optimizing the total loss

$$\mathcal{L}(\psi | \hat{\mathbf{z}}, \mathcal{T}) = \sum_{c \in \mathcal{C}_{\mathcal{T}}} \mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}), \quad (10)$$

where $\psi = [\psi_c]_{c \in \mathcal{C}_T}$ and $\hat{\mathbf{z}} = [\hat{\mathbf{z}}_c]_{c \in \mathcal{C}_T}$ are the concatenation of respective vectors for all classes. Note that we learn the parameters of the scoring functions that minimize the loss, while $\hat{\mathbf{z}}$ remains fixed in this step. Since the dataset is large, we employ Stochastic Gradient Descent (SGD) with momentum for optimization. Additionally, we subsample proposals in each bag by sampling 3 proposals with foreground and 7 proposal with background label in each training iteration.

4.1 Re-localization

In this step, we minimize the empirical loss function in Eq. (8) over the feasible labeling $\hat{\mathbf{z}}_c$ for the given model parameters. We first define feasible labeling set \mathcal{A} and simplify the objective function to an equivalent, simple linear form. Then, we discuss algorithms to optimize the objective function in the large scale settings.

For $\hat{\mathbf{z}}_c$ to be feasible, labeling should be feasible, *i.e.*, $\hat{\mathbf{y}}_c \in \mathcal{F}$ and pairwise labeling $\hat{\mathbf{r}}_c$ should also be consistent with the unary labeling. For dataset \mathcal{D}_T and target class c , this constraint set is expressed as

$$\mathcal{A} = \left\{ \hat{\mathbf{z}}_c \left| \begin{array}{l} \sum_{\mathbf{e} \in \mathcal{B}} \hat{y}_c(\mathbf{e}) = Y_c(\mathcal{B}) \quad \mathcal{B} \in \mathcal{T} \\ \sum_{\mathbf{e} \in \mathcal{B}} \hat{r}_c(\mathbf{e}, \mathbf{e}') = \hat{y}_c(\mathbf{e}') \quad \mathcal{B}, \mathcal{B}' \in \mathcal{T}, \mathcal{B}' \neq \mathcal{B}, \mathbf{e}' \in \mathcal{B}' \\ \hat{r}_c(\mathbf{e}, \mathbf{e}'), \hat{y}_c(\mathbf{e}) \in \{0, 1\} \quad c \in \mathcal{C}, \text{ for all } \mathbf{e} \text{ and } \mathbf{e}' \end{array} \right. \right\}. \quad (11)$$

Next, we simplify the loss function in the re-localization step. Let $\mathcal{T}_c = \{\mathcal{B} \mid \mathcal{B} \in \mathcal{T}, c \in \mathcal{Y}(\mathcal{B})\}$ and $\mathcal{T}_{\bar{c}} = \mathcal{T} \setminus \mathcal{T}_c$ denote the set of positive and negative bags with respect to class c . The loss function in Eq. (8) can be decomposed into three parts

$$\begin{aligned} \mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}) &= \mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}_c) + \mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}_{\bar{c}}) + \\ &\quad \sum_{\substack{\mathbf{e} \in \mathcal{B} \in \mathcal{T}_c \\ \mathbf{e}' \in \mathcal{B}' \in \mathcal{T}_{\bar{c}}}} \ell(\psi_c^P(\mathbf{e}, \mathbf{e}'), \hat{r}_c(\mathbf{e}, \mathbf{e}')) + \ell(\psi_c^P(\mathbf{e}', \mathbf{e}), \hat{r}_c(\mathbf{e}', \mathbf{e})), \end{aligned}$$

were the first two terms are the loss function in Eq. (8) defined over the positive set \mathcal{T}_c and negative set $\mathcal{T}_{\bar{c}}$, and last term is the loss defined by the pairwise similarities between these two sets. Since for any feasible labeling all the proposals in negative bags has label 0 and remain fixed, only the value of $\mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}_c)$ changes within \mathcal{A} and other terms are constant. Furthermore, by observing that for sigmoid cross entropy loss in Eq. (9) we have $\ell(x, y) = \ell(x, 0) - yx$, for $y \in [0, 1]^3$, we can further break down $\mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}_c)$ as

$$\begin{aligned} \mathcal{L}_c(\psi_c, \hat{\mathbf{z}}_c | \mathcal{T}_c) &= \mathcal{L}_c(\psi_c, \mathbf{0} | \mathcal{T}_c) \\ &\quad - \alpha \underbrace{\sum_{\substack{\mathcal{B}, \mathcal{B}' \in \mathcal{T}_c \\ \mathcal{B} \neq \mathcal{B}'}} \sum_{\substack{\mathbf{e} \in \mathcal{B} \\ \mathbf{e}' \in \mathcal{B}'}} \psi_c^P(\mathbf{e}, \mathbf{e}') \hat{r}_c(\mathbf{e}, \mathbf{e}') - \sum_{\mathcal{B} \in \mathcal{T}} \sum_{\mathbf{e} \in \mathcal{B}} \psi_c^U(\mathbf{e}) \hat{y}_c(\mathbf{e})}_{\mathcal{L}_{\text{reloc}}(\hat{\mathbf{z}}_c | \psi_c, \mathcal{T}_c)}, \quad (12) \end{aligned}$$

³ See Appendix for the proof.

where $\mathbf{0}$ is zero vector of the same dimension as $\hat{\mathbf{z}}_c$. Since the first term is constant with respect to $\hat{\mathbf{z}}_c = [\hat{\mathbf{y}}_c, \hat{\mathbf{r}}_c]$, re-localization can be equivalently done by optimizing $\mathcal{L}_{\text{reloc}}(\hat{\mathbf{z}}_c | \psi_c, \mathcal{T}_c)$ over the feasible set \mathcal{A}

$$\begin{aligned} & \min_{\hat{\mathbf{z}}_c} -\alpha \hat{\mathbf{r}}_c^\top \psi_c^P - \hat{\mathbf{y}}_c^\top \psi_c^U, \\ & \text{s.t. } \hat{\mathbf{z}}_c \in \mathcal{A}, \end{aligned} \quad (13)$$

where we use the equivalent vector form to represent the re-localization loss in Eq. (12). The re-localization optimization is an Integer Linear Program (ILP) and has been widely studied in literature [33]. The optimization can be equivalently expressed as a graph labeling problem with pairwise and unary potentials [34]. In the equivalent graph labeling problem, each bag is represented by a node in the graph where each proposal of the bag corresponds to a label of that node, and pairwise and unary potentials are equivalent to the negative pairwise similarity and negative unary scores in our problem. We discuss different graph inference methods and their limitations and present a practical method for large-scale settings.

Inference. Finding an optimal solution $\hat{\mathbf{z}}_c^*$ that minimizes the loss function defined in Eq. (13) is NP-hard and thus not feasible to compute exactly, except in small cases. Loopy belief propagation [35], TRWS [23], and AStar [36], are among the many inference algorithms used for approximate graph labeling problem. Unfortunately, finding an approximate labeling quickly becomes impractical as the size of \mathcal{T}_c increases, since the dimension of $\hat{\mathbf{z}}_c$ increases quadratically with the numbers of bags in \mathcal{T}_c due to dense pairwise connectivity. Due to this limitation, we employ an older well-known iterated conditional modes (ICM) algorithm for optimization [21]. In each iteration, ICM only updates one unary label in $\hat{\mathbf{y}}_c$ along with the pairwise labels that are related to this unary label while all the other elements of $\hat{\mathbf{z}}_c$ are fixed. The block that gets updated in each iteration is shown in Fig. 1. ICM generates monotonically non-increasing objective values and is computationally efficient. However, since ICM performs coordinate descent type updates and the problem in Eq. (13) is neither convex nor differentiable as the constraint set is discrete, ICM is prone to get stuck at a local minimum and its solution significantly depends on the quality of the initial labeling.

Algorithm 1: Re-localization

```

Input: Dataset  $\mathcal{D}_{\mathcal{T}}$ , batch size  $K$ , #epochs  $E$ 
Output: Optimal unary labeling  $\hat{\mathbf{y}}^*$ 
for  $c \in \mathcal{C}_{\mathcal{T}}$  do
     $T \leftarrow \text{round}\left(\frac{|\mathcal{T}_c|}{K}\right)$ ,  $\hat{\mathbf{y}}_c \leftarrow \mathbf{0}$ 
    for  $t \leftarrow 1$  to  $T$  do
        // Sample next mini-problem
         $\mathcal{X} \sim \mathcal{T}_c$ 
        // Solve mini-problem with TRWS [23]
         $[\bar{\mathbf{y}}_c^*, \bar{\mathbf{r}}_c^*] \leftarrow \underset{\mathbf{z}_c}{\text{argmin}} -\alpha \bar{\mathbf{r}}_c^\top \bar{\psi}_c^P - \bar{\mathbf{y}}_c^\top \bar{\psi}_c^U \text{ s.t. } \bar{\mathbf{z}}_c \in \bar{\mathcal{A}}$ 
        Update corresponding block of  $\hat{\mathbf{y}}_c$  with  $\bar{\mathbf{y}}^*$ 
        // Finetune for  $E$  epochs
         $\hat{\mathbf{y}}_c^* \leftarrow \text{ICM}(\hat{\mathbf{y}}_c, E)$ 
    return  $\{\hat{\mathbf{y}}_c^*\}_{c \in \mathcal{C}_{\mathcal{T}}}$ 

```



Fig. 1. ICM iteration (left) and initialization (right) graphical models. In both graphs, each node represents a bag (with B proposals) within a dataset with $|\mathcal{T}_c| = 9$ bags. **Left:** ICM updates the unary label of the selected node (shown in green). Edges show all the pairwise labels that gets updated in the process. Since the unary labeling of other nodes are fixed each blue edge represents B elements in vector $\hat{\mathbf{r}}_c$. **Right:** For initialization we divide the dataset into smaller mini-problems (with size $K = 3$ in this example) and solve each of them individually. Each edge represents B^2 pairwise scores that need to be computed.

Recent work [22] has shown that using accurate pairwise and unary functions learned on the source dataset, the re-localization method performs reasonably well by only looking at few bags. Motivated by this, we divide the full size problem into a set of disjoint mini-problems, solve each mini-problem efficiently using TRWS inference algorithm, and use these results to initialize the ICM algorithm.

The initialization algorithm samples a mini-problem $\mathcal{X} \in \mathcal{T}_c$ and optimizes the re-localization problem $\mathcal{L}_{\text{reloc}}(\bar{\mathbf{z}}_c \mid \bar{\psi}_c, \mathcal{X})$ where vectors $\bar{\mathbf{z}}_c$ and $\bar{\psi}_c$ are parts of vectors $\hat{\mathbf{z}}_c$ and $\hat{\psi}_c$ that are within the mini-problem defined by \mathcal{X} (see Fig. 1). This process is repeated until all the bags in the dataset are covered. The complete re-localization step is illustrated in Algorithm 1.

Next, we analysis the time complexity of the re-localization step. We practically observed that computing the pairwise similarity scores is the computation bottleneck, thus we analyze the time complexities in terms of the number of pairwise similarity scores each algorithm computes. Let $M = \max_{c \in \mathcal{C}_T} |\mathcal{T}_c|$ denotes the maximum number of positive bags, and $B = \max_{\mathcal{B} \in \mathcal{T}} |\mathcal{B}|$ be the maximum bag size. To solve the exact optimization in Eq. (13), we need to compute the vector ψ_c with $\mathcal{O}(B^2 M^2)$ elements. On the other hand, each iteration of ICM only computes $\mathcal{O}(BM)$ pairwise similarity scores. We additionally compute a total of $\mathcal{O}(MKB^2)$ pairwise similarity scores for the initialization where K is the size of the mini-problem. Thus, ICM algorithm would be asymptotically more efficient than the exact optimization in terms of total number of pairwise similarity scores it computes, if it is run for $\Omega(MB)$ iterations or $E = \Omega(B)$ epochs. We practically observe that by initializing ICM with the result of the proposed initialization scheme it converges in few epochs.

Even though Eq. (13) is similar to the DenseCRF formulation [37], the pairwise potentials are not amenable to the efficient filtering method [38] which is the backbone of DenseCRF methods [37, 39]. Therefore, it is intractable to use any existing sophisticated optimization algorithm except for ICM and Mean-Field [40]. Nevertheless, our block-wise application of TRWS provides an effective initialization for ICM. We additionally experimented with the block version of ICM [34] but it performs similarly while being slower.

4.2 Knowledge Transfer

To transfer knowledge from the fully annotated source set \mathcal{D}_S , we first learn *class generic* pairwise similarity $\psi^P : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ and unary $\psi^U : \mathbb{R}^d \rightarrow \mathbb{R}$ functions from the source set. Since the labels are available for all the proposals in the source set, learning the pairwise and unary functions is straightforward. We simply use stochastic gradient descent (SGD) to optimize the loss

$$\mathcal{L}^T(\psi^P, \psi^U | \mathcal{S}, \mathbf{r}, \mathbf{o}) = \alpha \sum_{\substack{\mathcal{B}, \mathcal{B}' \in \mathcal{S} \\ \mathcal{B} \neq \mathcal{B}'}} \sum_{\substack{\mathbf{e} \in \mathcal{B} \\ \mathbf{e}' \in \mathcal{B}'}} \ell(\psi^P(\mathbf{e}, \mathbf{e}'), r(\mathbf{e}, \mathbf{e}')) + \sum_{\mathcal{B} \in \mathcal{S}} \sum_{\mathbf{e} \in \mathcal{B}} \ell(\psi^U(\mathbf{e}), o(\mathbf{e})), \quad (14)$$

where $o(\mathbf{e}) \in \{0, 1\}$ is class generic objectness label, *i.e.*,

$$o(\mathbf{e}) = \begin{cases} 1 & \text{if } y(\mathbf{e}) \neq c_\emptyset \\ 0 & \text{otherwise,} \end{cases} \quad (15)$$

and relation function $r : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is defined by Eq. (6). Here we do not use hat notation since groundtruth proposal labels are available for the source dataset \mathcal{D}_S . We skip the details as the loss in Eq. (14) has a similar structure to the re-training loss. Note that in general the class generic functions ψ^U and ψ^P and class specific functions ψ_c^U and ψ_c^P use different feature sets extracted from different networks. Having learned these functions, we adapt both pairwise similarity and score vectors in the re-localization step in Algorithm 1 as

$$\begin{aligned} \psi_c^P &\leftarrow (1 - \lambda_1)\psi_c^P + \lambda_1\psi^P \\ \psi_c^U &\leftarrow (1 - \lambda_2)\psi_c^U + \lambda_2\psi^U, \end{aligned}$$

where $0 \leq \lambda_1, \lambda_2 \leq 1$ controls the weight of transferred and adaptive functions in pairwise similarity and unary functions respectively.

We start the alternating optimization with a *warm-up* re-localization step where only the learned class generic pairwise and unary functions above are used in the re-localization algorithm, *i.e.*, $\lambda_1, \lambda_2 = 1$. The warm-up re-localization step provides high quality pseudo labels to the first re-training step and speeds up the convergence of the alternating optimization algorithm.

4.3 Network Architectures

Proposal and Feature Extraction Following the experiment protocol in [4], we use a Faster-RCNN [41] model trained on the source dataset \mathcal{D}_S to extract region proposals from each image. We keep the box features in the last layer of Faster-RCNN as transferred features to be used in the class generic score functions. Following [4, 15, 42], we extract AlexNet [43] feature vectors from each proposal as input to the class specific scoring functions ψ_c^U and ψ_c^P .

Scoring Functions Let \mathbf{e} and \mathbf{e}' denote features in \mathbb{R}^d extracted from two image proposals. Linear layers are employed to model the class generic unary

function ψ^U and all the classwise unary functions ψ_c^U i.e. $\psi_c^U(\mathbf{e}) = \mathbf{w}_c^\top \mathbf{e} + b_c$ where $\mathbf{w}_c \in \mathbb{R}^d$ is the weight and $b_c \in \mathbb{R}$ is the bias parameter. We borrow the relation network architecture from [22] to model the pairwise similarity functions ψ^P and ψ_c^P . The details of the relation network architecture are discussed in the Appendix.

5 Experiments

We evaluate the main applicability of our technique on different weakly supervised datasets and analyze how each part affects the final results in our method. We report the widely accepted Correct Localization (CorLoc) metric [7] for the object localization task as our evaluation metric.

5.1 COCO 2017 Dataset

We employ a split of COCO 2017 [24] dataset to evaluate the effect of different initialization strategies and our pairwise retraining and re-localization steps. The dataset has 80 classes in total. We take the same split of [22, 44] with 63 source \mathcal{C}_S and 17 target \mathcal{C}_T classes and follow [22] to create the source and target splits to create source and target datasets with 111,085 and 8,245 images, respectively.

Similar to [22], we use Faster-RCNN [41] with ResNet 50 [45] backbone as our proposal generator and feature extractor for knowledge transfer. We keep the top $B = 100$ proposals generated by Faster-RCNN for experiments on the COCO 2017.

We first study different approaches for initializing the ICM method in the re-localization step. Then, we present the result of the full proposed method and compare it with other baselines.

Initialization Scheme Since the ICM algorithm is sensitive to initialization, we devise the following experiment to evaluate different initialization methods. To limit total running time of the experiment, we only do this evaluation in the warm-up re-localization step. We start by training class generic unary and pairwise similarity scoring functions on the source dataset \mathcal{D}_S . Next, we initialize the labeling of the images in \mathcal{D}_T using the following initialization strategies:

- Random: randomly select a proposal from each bag.
- Objectness: select the proposal with the highest unary score from each bag.
- Proposed initialization method: Proposed initialization method discussed in Section 4.1. We conduct the experiment with different mini-problem sizes $K \in \{2, 4, 8, 64\}$. We use TRWS [23] algorithm for inference in each mini-problem.

Finally, we perform ICM with each of the initialization methods. Fig. 2 shows the CorLoc and Energy vs. time plots as well as the computation time for different initialization methods. As expected, $K = 64$ exhibits the best initialization performance. However, ICM converges to similar energy when $4 \leq K \leq 64$ is

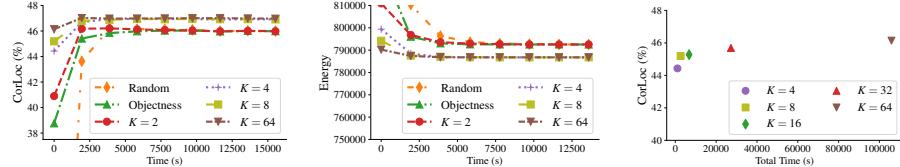


Fig. 2. **Left:** ICM CorLoc@0.5(%) vs. time for different initialization methods. See initialization schemes for definition of each initialization method. Markers indicate start of a new epoch. ICM inference converges in 2 epochs and demonstrates its best performance when initialized with the proposed initialization method. **Middle:** Energy vs. time for different initialization methods. The energies in the plot are computed by summing over energies of all classes. **Right:** Runtime vs. CorLoc(%) comparison of the proposed initialization scheme with various mini-problem sizes. We observe a quadratic time increase with respect to the mini-problem size.

used in the initialization method. In the extreme case with mini-problem of size $K = 2$, ICM converges to a worse local minimum in terms of CorLoc and energy value. Surprisingly, random initialization converges to the same result as objectness and $K = 2$. We also tried initializing ICM with the proposal that covers the complete image as it is the initialization scheme that is commonly used in MIL alternating optimization algorithms [4, 5]. But it performs significantly worse than the other initialization methods.

These results highlight the importance of initialization in ICM inference. Note that increasing K beyond 64 might provide a better initialization to ICM and increase the results further but it quickly becomes impractical as the time plot in Fig. 2 illustrates. As a rule of thumb, one should increase the mini-problem size as far as time and computational resources allow.

Full Pipeline Here, we conduct an experiment to determine the importance of learning pairwise similarities on the COCO dataset. We compare our full method with the unary method which only learns and uses unary scoring functions during, warm-up, re-training and re-localization steps. This method is analogous to [4]. The difference is that it uses cross entropy loss and SGD training instead of Support Vector Machine used in [4]. Also, we do not employ hard-negative mining after each re-training step. For this experiment, we use mini-problems of size $K = 4$ for initializing ICM. We run both methods for 5 iterations of alternating optimization on the target dataset. Our method achieves **48.3%** compared to 39.4% CorLoc@0.5 of the unary method. This clearly shows the effectiveness of our pairwise similarity learning.

5.2 ILSVRC 2013 Detection Dataset

We closely follow the experimental protocol of [4, 15, 42] to create source and target datasets on ILSVRC 2013 [25] detection dataset. The dataset has 200 categories with full bounding box annotations. We use the first 100 alphabetically ordered classes as source categories \mathcal{C}_S and the remaining 100 classes as target categories \mathcal{C}_T . The dataset is divided into source training set \mathcal{D}_S with 63k images,

Table 1. Performance of different methods on ILSVRC 2013. Proposal generators and their backbone models are shown in the second and third column. Total time is shown in “Training+Inference” format. CorLoc is reported on the target set. The last column shows the performance of an object detector trained on the target set and evaluated on the target test set. *The first 3 methods use RCNN detector with AlexNet backbone while other methods utilize Faster-RCNN detector with Inception-Resnet backbone.

Method	Proposal Generator	Backbone	CorLoc@0.5	CorLoc@0.7	Time(hours)	mAP@0.5
LSDA [15]	Selective Search [47]	AlexNet [43]	28.8	-	-	18.1*
Tang <i>et al.</i> [42]	Selective Search [47]	AlexNet [43]	-	-	-	20.0*
Uijlings <i>et al.</i> [4]	SSD [12]	Inception-V3 [48]	70.3	58.8	-	23.3*
Uijlings <i>et al.</i> [4]	Faster-RCNN	Inception-Resnet	74.2	61.7	-	36.9
Warm-up (unary)	Faster-RCNN	Inception-Resnet	68.9	59.5	0	-
Warm-up	Faster-RCNN	Inception-Resnet	73.8	62.3	5+3	-
Unary	Faster-RCNN	Inception-Resnet	72.8	62.0	13+2	38.1
Full (ours)	Faster-RCNN	Inception-Resnet	78.2	65.5	65+13	41.7
Supervised [4]						46.2

target set \mathcal{D}_T with 65k images, and a target test set with 10k images. We report CorLoc of different algorithms on \mathcal{D}_T . Similar to previous works [4, 15, 42], we additionally train a detector from the output of our method on target set \mathcal{D}_T , and evaluate it on the target test set.

For a fair comparison, we use a similar proposal generator and multi-fold strategy as [4]. We use Faster-RCNN [41] with Inception-Resnet [46] backbone trained on source dataset \mathcal{D}_S for object bounding box generation. The experiment on COCO suggests a small mini-problem size K would be sufficient to achieve good performance in the re-localization step. We use $K = 8$ to balance the time and accuracy in this experiment.

Baselines and Results We compare our method with two knowledge transfer techniques [4, 15] for WSOL. In addition, we demonstrate the results of the following baselines that only use unary scoring function:

- Warm-up (unary): To see the importance of learning pairwise similarities in knowledge transfer, we perform the warm-up re-localization with only the transferred unary scores ψ^U . This can be achieved by simply selecting the box with the highest unary score within each bag. We compare this results with the result of the warm-up step which uses both pairwise and unary scores in knowledge transfer.
- Unary: Standard MIL objective in Eq. (5) which only learns labeling and the unary scoring function.

We compare these results with our full pipeline which starts with a warm-up re-localization step followed by alternating re-training and re-localization steps. The results are illustrated in Table 1. Compared to Uijlings *et al.* [4], our method improves the CorLoc@0.5 performance on the target set by 4% and mAP@0.5 on the target test set by 4.8%. Warm-up re-localization improves CorLoc performance of warm-up (unary) by 4.9% with transferring a pairwise similarity



Fig. 3. Success cases on ILSVRC 2013 dataset. Unary method that relies on the objectness function tends to select objects from source classes that have been seen during training. Note that “banana”, “dog”, and “chair” are samples from source classes. Bounding boxes are tagged with method names. “GT” and “WU” stand for groundtruth and warm-up respectively. See Appendix for a larger set of success and failure cases.

measure from the source classes. Note that the result of warm-up step without any re-training performs on par with the Uijlings *et al.* [4] MIL method. The CorLoc performance at the stricter $\text{IoU} > 0.7$ also shows similar results. Some of the success cases are shown in Fig. 3.

Compared to [4], our implementation of the MIL method performs worse with IoU threshold 0.5 but better with stricter threshold 0.7. We believe the reason is having a different loss function and hard-negative mining in [4].

6 Conclusion

We study the problem of learning localization models on target classes from weakly supervised training images, helped by a fully annotated source dataset. We adapt MIL localization model by adding a classwise pairwise similarity module that learns to directly compare two input proposals. Similar to the standard MIL approach, we learn the augmented localization model and annotations jointly by two-step alternating optimization. We represent the re-localization step as a graph labeling problem and propose a computationally efficient inference algorithm for optimization. Compared to the previous work [7] that uses pairwise similarities for this task, the proposed method is represented in alternating optimization framework with convergence guarantee and is computationally efficient in large-scale settings. The experiments show that learning pairwise similarity function improves the performance of WSOL over the standard MIL.

Acknowledgement

We would like to thank Jasper Uijlings for providing helpful instructions and their original models and datasets. We gratefully express our gratitude to Judy Hoffman for her advice on improving the experiments. We also thank the anonymous reviewers for their helpful comments to improve the paper. This research is supported in part by the Australia Research Council Centre of Excellence for Robotics Vision (CE140100016).

References

1. Wan, F., Liu, C., Ke, W., Ji, X., Jiao, J., Ye, Q.: C-mil: Continuation multiple instance learning for weakly supervised object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 2199–2208
2. Gao, J., Wang, J., Dai, S., Li, L.J., Nevatia, R.: Note-rcnn: Noise tolerant ensemble rcnn for semi-supervised object detection. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 9508–9517
3. Arun, A., Jawahar, C., Kumar, M.P.: Dissimilarity coefficient based weakly supervised object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 9432–9441
4. Uijlings, J., Popov, S., Ferrari, V.: Revisiting knowledge transfer for training object class detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 1101–1110
5. Cinbis, R.G., Verbeek, J., Schmid, C.: Weakly supervised object localization with multi-fold multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence* **39**(1) (2016) 189–203
6. Bilen, H., Pedersoli, M., Tuytelaars, T.: Weakly supervised object detection with posterior regularization. In: British Machine Vision Conference. Volume 3. (2014)
7. Deselaers, T., Alexe, B., Ferrari, V.: Localizing objects while learning their appearance. In: Proceedings of the European Conference on Computer Vision, Springer (2010) 452–466
8. Tang, P., Wang, X., Bai, X., Liu, W.: Multiple instance detection network with online instance classifier refinement. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017)
9. Tang, P., Wang, X., Bai, S., Shen, W., Bai, X., Liu, W., Yuille, A.: Pcl: Proposal cluster learning for weakly supervised object detection. *IEEE transactions on pattern analysis and machine intelligence* **42**(1) (2018) 176–191
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 2961–2969
11. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 779–788
12. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proceedings of the European Conference on Computer Vision, Springer (2016) 21–37
13. Singh, B., Najibi, M., Davis, L.S.: Sniper: Efficient multi-scale training. In: Advances in Neural Information Processing Systems. (2018) 9310–9320
14. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. (2017) 2980–2988
15. Hoffman, J., Pathak, D., Tzeng, E., Long, J., Guadarrama, S., Darrell, T., Saenko, K.: Large scale visual recognition through adaptation using joint representation and multiple instance learning. *The Journal of Machine Learning Research* **17**(1) (2016) 4954–4984
16. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Advances in Neural Information Processing Systems. (2003) 577–584
17. Bilen, H., Vedaldi, A.: Weakly supervised deep detection networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2846–2854

18. Rochan, M., Wang, Y.: Weakly supervised localization of novel objects using appearance transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 4315–4324
19. Tang, K., Joulin, A., Li, L.J., Fei-Fei, L.: Co-localization in real-world images. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2014) 1464–1471
20. Guillaumin, M., Ferrari, V.: Large-scale knowledge transfer for object localization in imangenet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2012) 3202–3209
21. Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society: Series B (Methodological) **48**(3) (1986) 259–279
22. Shaban, A., Rahimi, A., Bansal, S., Gould, S., Boots, B., Hartley, R.: Learning to find common objects across few image collections. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 5117–5126
23. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE transactions on pattern analysis and machine intelligence **28**(10) (2006) 1568–1583
24. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Proceedings of the European Conference on Computer Vision. (2014) 740–755
25. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3) (2015) 211–252
26. Zhu, Y., Zhou, Y., Ye, Q., Qiu, Q., Jiao, J.: Soft proposal networks for weakly supervised object localization. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 1841–1850
27. Bilen, H., Pedersoli, M., Tuytelaars, T.: Weakly supervised object detection with convex clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 1081–1089
28. Wan, F., Wei, P., Jiao, J., Han, Z., Ye, Q.: Min-entropy latent model for weakly supervised object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 1297–1306
29. Kumar, M.P., Packer, B., Koller, D.: Self-paced learning for latent variable models. In: Advances in Neural Information Processing Systems. (2010) 1189–1197
30. Hayder, Z., Salzmann, M., He, X.: Object co-detection via efficient inference in a fully-connected crf. In: Proceedings of the European Conference on Computer Vision, Springer (2014) 330–345
31. Hayder, Z., He, X., Salzmann, M.: Structural kernel learning for large scale multiclass object co-detection. In: Proceedings of the IEEE International Conference on Computer Vision, IEEE (2015) 2632–2640
32. Ortega, J.M., Rheinboldt, W.C.: Iterative solution of nonlinear equations in several variables. Volume 30. Siam (1970)
33. Schrijver, A.: Theory of linear and integer programming. John Wiley & Sons (1998)
34. Savchynskyy, B., et al.: Discrete graphical models an optimization perspective. Foundations and Trends® in Computer Graphics and Vision **11**(3-4) (2019) 160–429
35. Weiss, Y., Freeman, W.T.: On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. IEEE Transactions on Information Theory **47**(2) (2001) 736–744

36. Bergtholdt, M., Kappes, J., Schmidt, S., Schnörr, C.: A study of parts-based object class detection using complete graphs. *IJCV* **87**(1-2) (2010) 93
37. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. In: Advances in Neural Information Processing Systems. (2011) 109–117
38. Adams, A., Baek, J., Davis, M.A.: Fast high-dimensional filtering using the permutohedral lattice. In: Computer Graphics Forum. Volume 29., Wiley Online Library (2010) 753–762
39. Ajanthan, T., Desmaison, A., Bunel, R., Salzmann, M., Torr, P.H., Pawan Kumar, M.: Efficient linear programming for dense crfs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017) 3298–3306
40. Blake, A., Kohli, P., Rother, C.: Markov random fields for vision and image processing. Mit Press (2011)
41. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems. (2015) 91–99
42. Tang, Y., Wang, J., Gao, B., Dellandr  a, E., Gaizauskas, R., Chen, L.: Large scale semi-supervised object detection using visual and semantic knowledge transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2119–2128
43. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. (2012) 1097–1105
44. Bansal, A., Sikka, K., Sharma, G., Chellappa, R., Divakaran, A.: Zero-shot object detection. In: Proceedings of the European Conference on Computer Vision. (2018) 384–400
45. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 770–778
46. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-First AAAI Conference on Artificial Intelligence. (2017)
47. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. *International Journal of Computer Vision* **104**(2) (2013) 154–171
48. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2818–2826
49. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 1199–1208
50. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 580–587

Pairwise Similarity Knowledge Transfer for Weakly Supervised Object Localization Supplementary Material

A Missing Proof

Let $\ell : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ be the sigmoid cross-entropy loss function

$$\ell(x, y) = -(1 - y) \log(1 - \sigma(x)) - y \log(\sigma(x)),$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function. Then, $\ell(x, y) = \ell(x, 0) - yx$, for any $x \in \mathbb{R}$ and $y \in [0, 1]$.

Proof.

$$\begin{aligned}\ell(x, y) - \ell(x, 0) &= \left(-(1 - y) \log(1 - \sigma(x)) - y \log(\sigma(x)) \right) + \log(1 - \sigma(x)) \\ &= y \log(1 - \sigma(x)) - y \log(\sigma(x)) \\ &= -yx\end{aligned}$$

Last equality is derived using the fact that $\log(1 - \sigma(x)) - \log(\sigma(x)) = -x$ which can be easily verified by plugging in the sigmoid function.

B Experiment Details

The CorLoc measure used to evaluate the performance of different WSOL methods is the ratio of correctly localized objects in each class and computes the mean over all classes. A localization is correct if it has Intersection-over-Union (IoU) greater than a threshold with the groundtruth object bounding box. We report with 0.5 and 0.7 thresholds in our experiments. All experiments are done on a single Nvidia GTX 1080 GPU and 3.2GHz Intel(R) Xeon(R) CPU with 128 GB of RAM.

B.1 Relation Network Architecture

The relation network [49] $s : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ has two modules. We use the same architecture proposed in [22] in our method. First module maps both input features into a joint feature space using embedding function $\mathcal{E} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ and is defined as

$$\mathcal{E}(\mathbf{e}, \mathbf{e}') = \tanh(W_1 [\mathbf{e}, \mathbf{e}'] + \mathbf{b}_1)\sigma(W_2 [\mathbf{e}, \mathbf{e}'] + \mathbf{b}_2) + \frac{\mathbf{e} + \mathbf{e}'}{2},$$

where $W_1, W_2 \in \mathbb{R}^{d \times 2d}$ and vectors $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^d$ are the parameters of the feature embedding module and \tanh and σ are hyperbolic tangent and sigmoid activation functions respectively. Finally, a linear layer maps these features into similarity score

$$s(\mathbf{e}, \mathbf{e}') = \mathbf{w}^\top \mathcal{E}(\mathbf{e}, \mathbf{e}') + b,$$

where $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$. We share the parameters of the embedding functions in $\psi_c^P(\mathbf{e}, \mathbf{e}')$ for all the classes $c \in \mathcal{C}_{\mathcal{T}}$ to reduce the number of parameters.

B.2 ILSVRC 2013 Detection Experiment

We use an identical dataset and bounding box proposals as were used in [4]⁴. The dataset is created by following the dataset generation protocol of [15, 42]. The val1 split is augmented with images from the training set such that each class has 1000 annotated bounding boxes in total [50]. The dataset has 200 categories with full bounding box annotations. The first 100 alphabetically ordered classes are used as source categories \mathcal{C}_S and the remaining 100 classes are used as target categories \mathcal{C}_T . The source dataset \mathcal{D}_S is formed by all images in the augmented val1 set that have an object in \mathcal{C}_S . As for our target dataset \mathcal{D}_T , all images which have an object in the target categories \mathcal{C}_T are used and all the bounding box annotations are removed and only the bag labels \mathcal{Y}_T are kept. The dataset has 63k and 65k images in source and target sets, respectively.

We also use the same Faster-RCNN [41] model with Inception-Resnet [46] backbone trained on the source dataset \mathcal{D}_S that were used in [4] for object bounding box generation. Since the Faster-RCNN model used in [4] is trained with a class agnostic loss and the feature vectors are not suitable for discriminating between different classes we also trained the same architecture with a multi-class loss and use it to extract the features from the bounding box proposals of [4].

To have a fair comparison to [4], we perform multi-folding strategy [5] to avoid overfitting: the target dataset is split into 10 random folds and then re-training is done on 9 folds while re-localization is performed on the remaining fold. Values of hyper-parameters are obtained using cross validation.

C More Qualitative Results

Qualitative results on ILSVRC 2013 dataset are illustrated in Fig. 4, and Fig. 5. Failure cases on this dataset are also presented in Fig. 6. Refer to Fig. 3 for more information on bounding box tags. Overall, selection of a visually similar object in the image, occlusion and disconnected objects, multi-part objects, and even errors in dataset annotations are the source of most of the failures on this dataset. Fig. 7 shows the qualitative results on the COCO dataset.

⁴ Thanks to Jasper Uijlings for sharing their pre-trained Faster-RCNN model and image splits

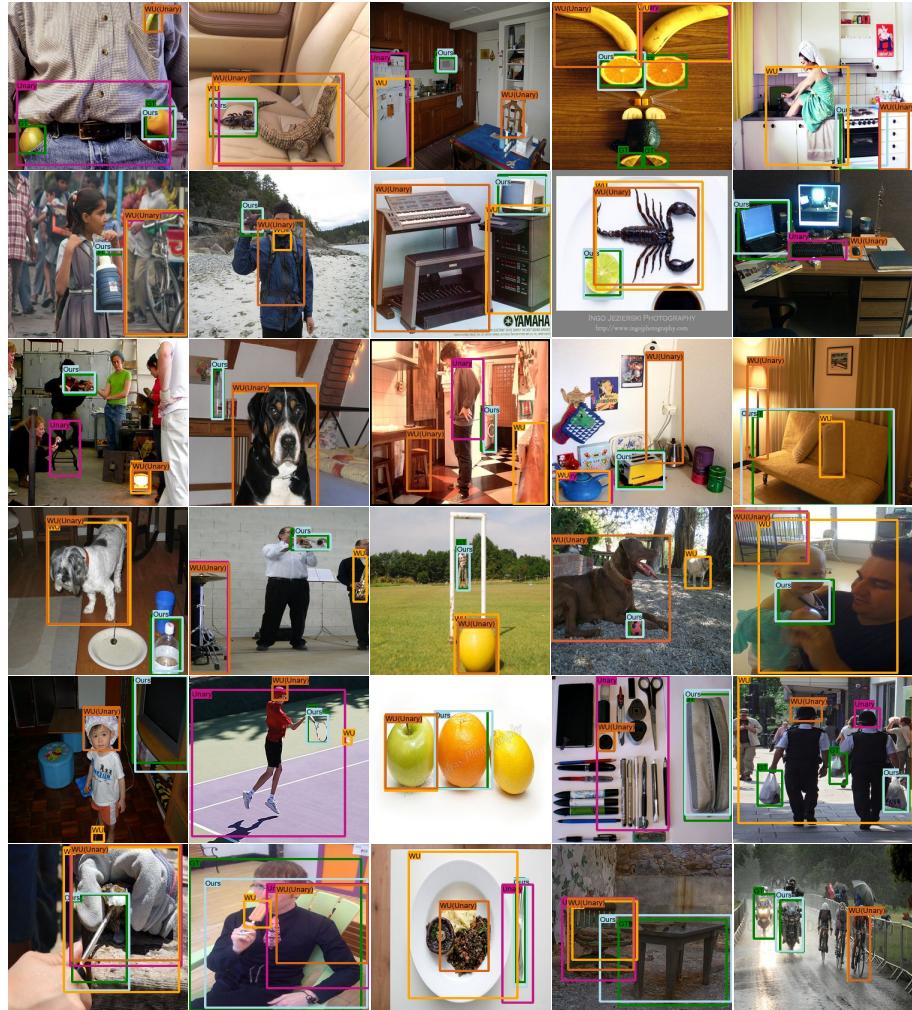


Fig. 4. Extended results of Fig. 3

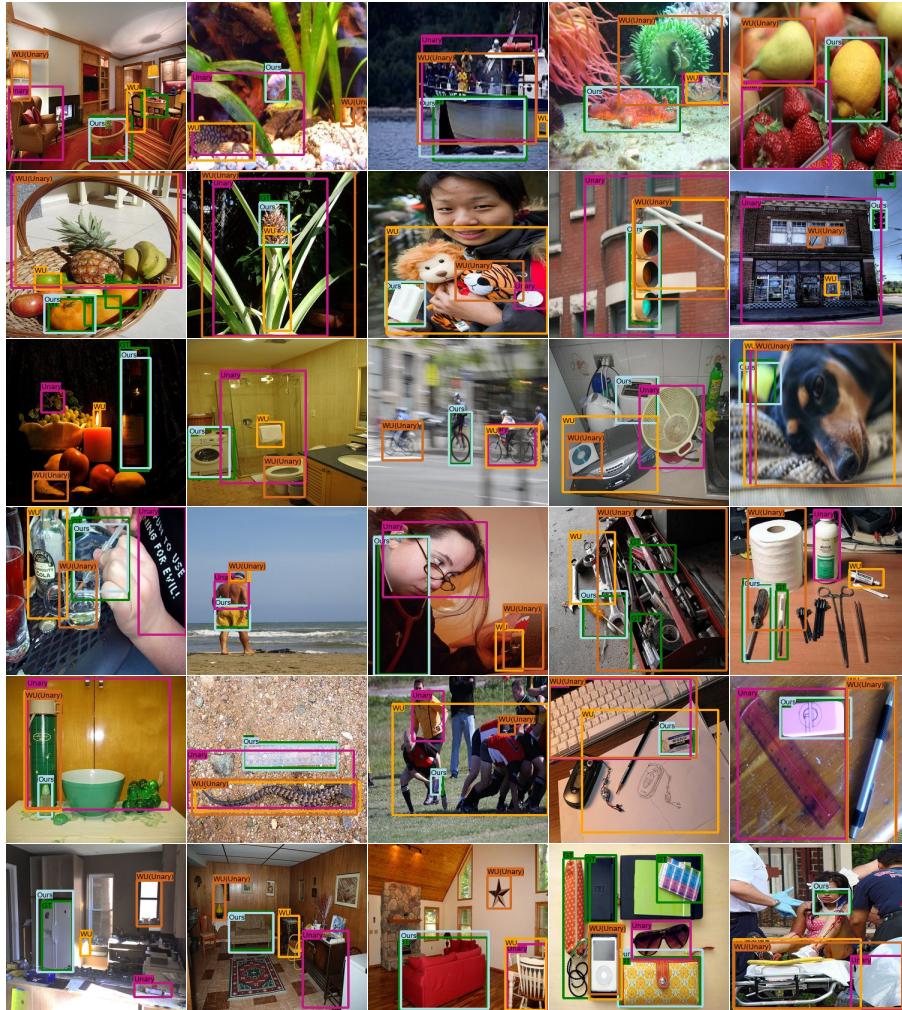


Fig. 5. Extended results of Fig. 3

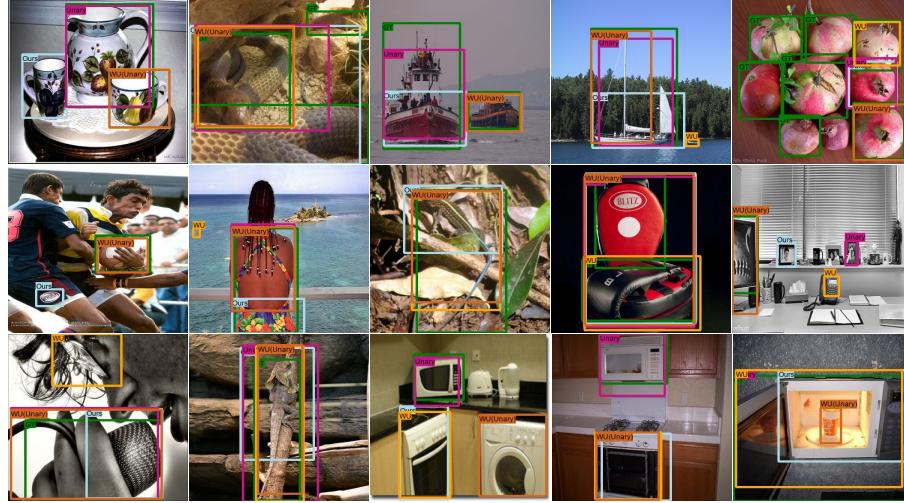


Fig. 6. Failure cases on ILSVRC 2013 dataset.

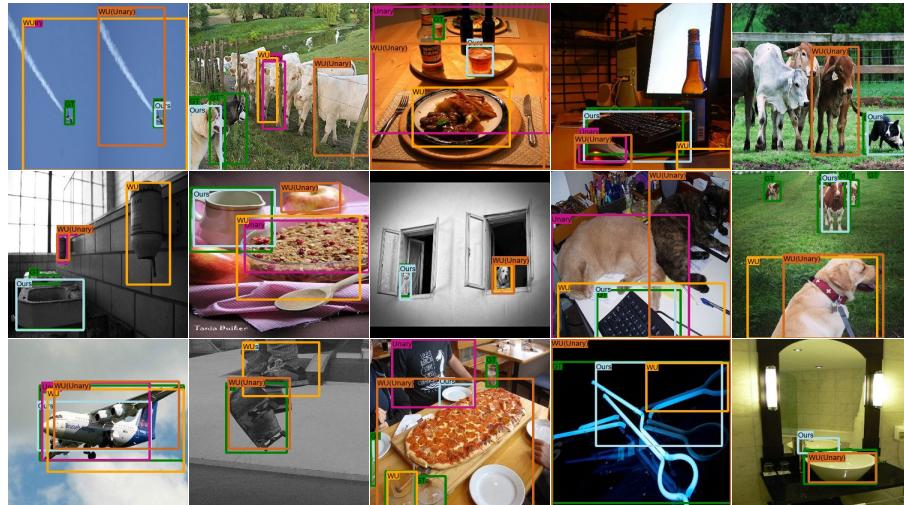


Fig. 7. Success and failure cases on COCO dataset. First two rows show the success cases of our method while the last row shows the failure cases.