Learning to Localize Objects with Noisy Labeled Instances

Xiaopeng Zhang,[1] **Yang Yang,**[2] **Jiashi Feng**[1]

[1]National University of Singapore
[2]Center for Future Media and School of Computer Science and Engineering,
University of Electronic Science and Technology of China
{elezxi,elefjia}@nus.edu.sg, dlyyang@gmail.com

Abstract

This paper addresses Weakly Supervised Object Localization (WSOL) with only image-level supervision. We model the missing object locations as latent variables, and contribute a novel self-directed optimization strategy to infer them. With the strategy, our developed Self-Directed Localization Network (SD-LocNet) is able to localize object instance whose initial location is noisy. The self-directed inference hinges on an adaptive sampling method to identify reliable object instance via measuring its localization stability score. In this way, the resulted model is robust to noisy initialized object locations which we find is important in WSOL. Furthermore, we introduce a reliability induced prior propagation strategy to transfer object priors of the reliable instances to those unreliable ones by promoting their feature similarity, which effectively refines the unreliable object instances for better localization. The proposed SD-LocNet achieves 70.9% CorLoc and 51.3% mAP on PASCAL VOC 2007, surpassing the state-of-the-arts by a large margin.

1 Introduction

Weakly Supervised Object Localization (WSOL) refers to localizing objects with only image-level labels that indicate presence of an object, and has been extensively studied (Cinbis, Verbeek, and Schmid 2014), (Song et al. 2014b), (Wang et al. 2014). However, WSOL still remains challenging due to the gap between classification and localization. Successful classification may only need to grasp some discriminative details of an object (*e.g.,* the face of a human instance), while localization is more demanding and requires to predict accurate spatial extent of the whole object. To mitigate this gap, current approaches (Jie et al. 2017), (Li et al. 2016), (Tang et al. 2017) usually model the missing object locations as latent variables, and alternate between updating the latent variables and learning a classifier based on current latent variables. Nevertheless, learning the parameters of a model with latent variables often requires solving a non-convex optimization problem, and is prone to get stuck in local minima if the latent variables were not properly initialized.

In this work, we take the object location as latent variable (whose initial value can be estimated from off-the-shelf WSOL models) and consider how to address the learning difficulties caused by noisy initialized latent variables. Intuitively, different images present different levels of difficulty in object localization due to the variational viewpoints, occlusions and backgrounds. The model is able to successfully localize objects in some easy images, even it is trained from noisy labeled instances. The localizations on these easy images provide valuable information and can be exploited as priors that are easily obtained. Therefore we propose to use such prior knowledge to assist latent variable inference on difficult images. Motivated by such progressive inference strategy, we develop a Self-Directed Localization Network (SD-LocNet), which is able to automatically figure out the easy images during network training and harness the corresponding localizations for latent variable update. Since the latent variables are updated automatically from the easy images to those difficult ones, we refer this process as self directed localization. As illustrated in Fig. 1, SD-LocNet makes use of the localizations from easy images as object priors to direct the discovery of the corresponding category in difficult images, thus improves the localization performance on the whole image set.

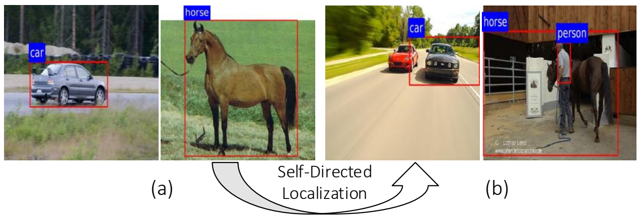

Figure 1: Given a collection of noisy labeled instances, our goal is to automatically discover those well localized instances (a) and propagate such priors to those noisy labeled instances (b) for better localization.

Our proposed SD-LocNet consists of two core modules. The first one is the adaptive sampling module, which targets at mining reliable object instances[1], taken as reliable latent variables, for network training. We propose an effective criterion, named *Localization Stability Score (LSS)*, to measure the reliability of a candidate positive sample being

[1]If an object is easy to localize, we term the object as the *reliable instance* and the corresponding image as the *easy image*. Otherwise, they are defined as *unreliable instance* and *difficult image*.

the ground truth object, and update the latent variables with samples having the highest LSS. Meanwhile, we use LSS to reweight each instance during network training. In this way, we can put more emphasis on the reliable instances and decrease the contributions of those unreliable ones, making the model robust to noisy labeled instances. As the training process proceeds, the model gains increasingly stronger localization ability, and is able to better discover reliable instances, which in turn improves the localization model.

Our adaptive sampling is reminiscent of self-paced learning (Kumar, Packer, and Koller 2010), (Ouyang et al. 2018) and curriculum learning (Bengio et al. 2009) in which easier images are first selected for training. However, self-paced learning usually treats samples with lower loss on current model as easier ones. Comparatively, our adaptive sampling is based on the proposed Localization Stability Score measuring the reliability of candidate positive samples by inspecting *the states among consecutive training epoches*, which is based on the localization stability instead of the classification loss. Besides, the instances are mined from images and updated during each iteration, which differs from self-paced learning where the used images are kept fixed.

The second module of SD-LocNet is the reliability induced prior propagation, which targets at directing the localization on difficult images. We find that emphasizing too much on reliable instances usually slows down the training process because they provide smaller gradients. Furthermore, decreasing weights of unreliable instances would lead to sampling bias, resulting in poor localization on difficult images. To mitigate this issue, we incorporate unreliable instances into model training via propagating object priors of the reliable instances. The intuition is that for instances from the same category, their visual representations should be similar. Instead of penalizing the unreliable instances with smaller decision loss, we enforce them to be similar to the representative clusters dominated by the reliable instances. In this way, the mined instances are encouraged to have large intra-class similarity, which can relieve overfitting and better guide the localization on difficult images.

To sum up, this paper contributes a novel learning strategy for WSOL, which intentionally selects reliable instances for model training, and propagates the object priors of the reliable instances to the unreliable ones for better localization. This strategy is further instantiated to a Self-Directed Localization Network (SD-LocNet) based on fast RCNN (Girshick 2015). Its effectiveness has been validated experimentally. Notably, we achieve 70.9$\%$ localization CorLoc and 51.3$\%$ detection mAP on PASCAL VOC 2007 benchmark, which surpasses the state-of-the-arts by a large margin.

2 Related Work

Most previous works formulate WSOL as a Multiple Instance Learning (MIL) (Dietterich, Lathrop, and Lozano-Pérez 1997) task, in which labels are assigned to bags (a group of instances) instead of an individual instance. These methods typically alternate between learning a discriminative representation of the object and selecting the object samples based on this representation (Cinbis, Verbeek, and Schmid 2014), (Nguyen et al. 2009), (Vijayanarasimhan and Grauman 2008). However, MIL is sensitive to initialization and easy to get stuck in local minima. To solve this issue, many efforts have been devoted to improving the initializations (Li et al. 2016), (Song et al. 2014b), (Wang et al. 2014), or designing robust optimization methods to alleviate the dependency on the initially mined object instances (Bilen, Pedersoli, and Tuytelaars 2015), (Cinbis, Verbeek, and Schmid 2014), (Joulin and Bach 2012), (Song et al. 2014a). However, these methods treat each image equally important and do not consider the intrinsic properties of images.

Another line of research in WSOL develops new CNN architectures and learns to localize objects in an end-to-end manner. These approaches are based on the principle of aggregating pixel-level confidences for image-level loss modeling (Oquab et al. 2015), (Zhou et al. 2016), (Zhu et al. 2017), and gets the final localization from the heatmap derived from the network. However, they tend to emphasize the discriminative details and fail to distinguish multiple objects from the same category. Notably, (Bilen and Vedaldi 2016) proposes a WSDDN model which directly aggregates region-level scores for image-level loss and conveniently enables localization based on the region scores. Based on this WSDDN model, some works exploit context information (Kantorov et al. 2016) and MIL refinement (Tang et al. 2017) to further improve the localization.

3 Self-Directed Localization Network

This section provides details of our proposed Self-Directed Localization Network (Sec. 3.1), as well as the associated training methodology (Sec. 3.2).

3.1 Model Design

A high-level overview of the proposed SD-LocNet model is shown in Fig. 2. The model is built on the fast RCNN (Girshick 2015) architecture. We present the following novel designs to enable the learning of a localization model from noisy labeled object instances.

Latent Variable Initialization Since the object-level annotations are not available in WSOL, we treat the missing annotations as latent variables and formulate the localization model learning as inferring the latent variables. Without loss of generality, we choose the outputs of WSDDN (Bilen and Vedaldi 2016) for latent variable initialization. Given an image x with a set of region proposals $\mathcal{R}$ and image-level label $y \in \{1, -1\}^C$, where $y_c = 1$ indicates the presence of an object from class c, WSDDN explicitly computes image-level classification loss $L_{cls}(\phi_c(x, w_{cls}), y_c)$ via aggregating region proposal classification scores x_c^r, *i.e.*, $\phi_c(x, w_{cls}) = \sum_{r=1}^{|\mathcal{R}|} x_c^r$. Here w_{cls} denotes parameters of the non-linear mapping from input x to classification output $\phi_c(x, w_{cls})$, and x_c^r represents the probability score of region r belonging to category c. For an image x with $y_c = 1$, we choose instance r_c that scores highest among x_c^r as the instance-level bounding box for latent variable initialization. Please refer to (Bilen and Vedaldi 2016) for more details.

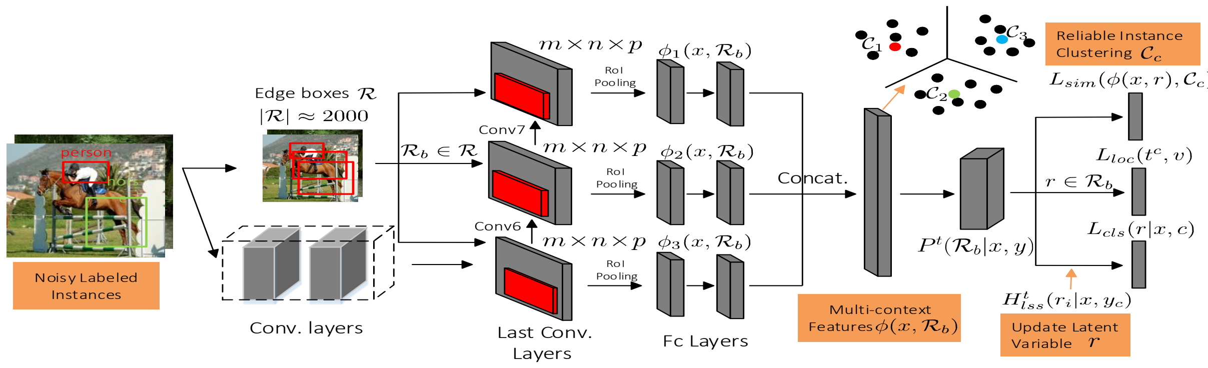

Figure 2: An overview of SD-LocNet. It starts from noisy labeled instances whose object bounding boxes (latent variables) are not accurate, and updates the latent variables with reliable instances discovered by adaptive sampling. These reliable instances are used to direct the localization on difficult images via optimizing similarity loss L_{sim} of instances from the same category.

Adaptive Sampling The initial latent variables inevitably include noisy labeled instances. Our proposed model aims at improving their localizations by latent variable update along with model training. As noisy labeled instances are detrimental for network learning, they should be expelled and emphasis should be placed on the clean and reliably labeled ones for robust model learning. To this end, we define a Localization Stability Score (LSS) to effectively measure the reliability (whether it is well localized) of the mined object instance online along with network training. If a candidate positive sample scores consistently high among consecutive training epoches, it is regarded as containing the object of interest with a high probability.

Specifically, given a training image x with label $y_c = 1$, and the candidate positive samples $\mathcal{R}_p$, where $\mathcal{R}_p \subset \mathcal{R}$. For each region $r \in \mathcal{R}_p$, denote the model probability outputs at current iteration during epoch t as $P^t(r|x, y_c)$. We compute the LSS $H^t_{lss}(r|x, y_c)$ as follows. Instead of relying on static scores at current iteration (Kumar, Packer, and Koller 2010), we utilize the probability scores $\{P^{t-t_i}(r|x, y_c)\}_{t_i=1}^{t_0}$ predicted among continuous t_0 epoches for reliability computation. $H^t_{lss}(r|x, y_c)$ is estimated by the ratio of the mean probabilities $\{P^{t-t_i}(r|x, y_c)\}_{t_i=1}^{t_0}$ and their standard variance, as follows:

$$H^t_{lss}(r|x, y_c) = \frac{\overline{P}^{t-t_i}(r|x, y_c)}{\sqrt{var[P^{t-t_i}(r|x, y_c)] + \epsilon_V}}, \quad (1)$$

where $var[P^{t-t_i}(r|x, y_c)]$ is the variance of predicted probabilities for instance r over the past t_0 consecutive epoches, and ϵ_V is a constant (set as 1) to restrict the scores within range $(0, 1)$. Note that the overhead of computing $H^t_{lss}(r|x, y_c)$ is very small since we need not do any extra forward or backward propagation in the neural network. The probability $P^t(r|x, y_c)$ is computed online by forward propagation and we update the latent variable for the following back propagation at current iteration. Specifically, the sample r_p with the highest H^t_{lss} is chosen as the newly discovered object instance for back propagation:

$$L_{cls}(r_p|x, y_c) = -logP^t(r_p|x, y_c). \quad (2)$$

Reliability Induced Prior Propagation Giving larger weights to reliable instances enables the network robust to noisy labeled instances, but unfavorably discards instances from difficult images and harms sample diversity. We empirically find that simply relying on the classifier model does not work well on transferring the object priors of reliable instances to help discover objects in difficult images. This would make the network overfit reliable instances from easy images and underperform in localization on difficult images.

In order to effectively localize objects in difficult images, we propagate object priors from reliable instances to guide the discovery of the same category objects in the difficult images. Towards this goal, we devise a reliability induced prior propagation module that explicitly imposes similarity constraint among the reliable instances and the mined instances from difficult images. Specifically, for a category c, suppose we have the mined object instances r_{ip} for image x_i with image label $y_{ic} = 1$ and reliability scores $H_{lss}(r_{ip}|x_i, y_{ic})$. The category specific priors are obtained by a weighted k-means clustering performed on features $\phi(x_i, r_{ip})$, weighted by $H_{lss}(r_{ip}|x_i, y_{ic})$. In this way, we obtain K representative clusters $\mathcal{C}_c = \{\mathcal{C}_1, ..., \mathcal{C}_K\}$ for category c. The similarity constraint is represented as the minimal Euclidean distance between the mined new instance $\phi(x, r_p)$ and the representative clusters $\mathcal{C}_c$:

$$L_{sim}(\phi(x, r_p), \mathcal{C}_c) = \min_{k=1,...,K} ||\phi(x, r_p) - \mathcal{C}^k_c||^2. \quad (3)$$

Multi-context Features Context information plays an import role in object localization. In fast RCNN, RoI pooling is performed on a fixed layer (*conv*5 layer), which can be treated as covering a fixed range of context information. However, the optimal context information varies w.r.t. different scenes. To enrich varying context representation, we add two extra convolutional layers (*conv*6 and *conv*7 layer, as shown in Fig. 2) to the end of the last convolutional layer (*conv5* layer) with parameters $3 \times 3 \times p_I \times p_O$, where p_I and p_O denote the number of input and output channels. In this case, each added layer together with *conv*5 layer is followed by an ROI pooling and two fully con-

Algorithm 1 SD-LocNet for WSOL

```
Input: Training set $x_i \in \mathcal{X}$ with image-level labels $y_i \in \mathcal{Y}$, training epoch $T$, latent variable update starting epoch $T_0$, and number of clusters $K$;
  Latent Variable Initialization: For each image $x$ with label $y_c = 1$, and region proposals $\mathcal{R}$, initialize the latent variable with $r_c$, where $r_c \in \mathcal{R}$;
  For epoch $t = 1$ to $T$ do
    Forward Propagation: For each image $x \in \mathcal{X}$, forward $x$ and its sampled region proposals $\mathcal{R}_b$ to produce region probability scores $P(\mathcal{R}_b|x, y)$;
    if $t >= T_0$
      Update Latent Variable:
        1). Compute Localization Stability Score $H^t_{lss}(r|x, y_c)$ for $r \in \mathcal{R}_p$ and $y_c = 1$ as Eq. (1);
        2). Update latent variable with region $r_p$, $r_p = \arg\max_{r_i \in \mathcal{R}_p} H^t_{lss}(r_i|x, y_c)$;
    end if
    Back Propagation: For each proposal $r \in \mathcal{R}_b$, back propagate the loss $L(\mathcal{C}_c, t^c, v|x, c, r)$ defined in Eq. (4) based on current latent variables;
    Update Clusters: For each category c, computing clustering centers $\mathcal{C}_c = \{\mathcal{C}_1, ..., \mathcal{C}_K\}$ based on current network and latent variables;
  end for
Output: Localization model $M$.
```

nected layers independently, producing fixed length vectors $\phi_i(x, r)$, $i = 1, 2, 3$ for each region $r \in \mathcal{R}$. The vectors are concatenated to form the final representation $\phi(x, r) = [\phi_1(x, r)\ \phi_2(x, r)\ \phi_3(x, r]$. In this way, $\phi(x, r)$ covers context with different ranges, which we refers to as multi-context features. At below, we will investigate its effectiveness in improving object localization.

3.2 Training

We develop SD-LocNet by integrating the above novel designs. Training SD-LocNet requires appropriately designing latent variable update and category prior update strategies. The whole training procedure is summarized in Algorithm 1, and is introduced below.

Multi-task Loss Putting the above different modules together gives a multi-task loss objective for model training. Given an image x, each sampled region r is labeled with a class c, a similarity constraint prior $\mathcal{C}_c$, and a bounding box regression target v based on current latent variables (We treat the mined instance r_p in Eq. (2) as ground truth, and follow the positive/negetive sample definitions as in (Girshick 2015)). We use the following multi-task loss L to jointly train classification and cluster similarity, as well as the bounding-box regressor:

$$L(\mathcal{C}_c, t^c, v|x, c, r) = H_{lss}\{L_{cls}(r|x, c) + [c \geq 1]L_{loc}(t^c, v)\} + [c \geq 1]\lambda(1 - H_{lss})L_{sim}(\phi(x, r), \mathcal{C}_c), \tag{4}$$

where L_{cls} and L_{sim} are the classification and similarity losses defined in Eq. (2) and Eq. (3), and H_{lss} is the LSS score defined in Eq. (1). The localization regression term L_{loc} is a robust smooth L_1 loss between the tuple of pseudo ground truth bounding-box regression targets v, and the predicted tuple t^c for class c (Girshick 2015). The Iverson bracket indicator function $[c \geq 1]$ evaluates to 1 when $c \geq 1$ and 0 otherwise. We only impose similarity and regression constraints upon foreground classes with $c \geq 1$, where $c = 0$ means the background class. The parameters λ control the relative contributions of each component. By default, we set $\lambda = 1e-4$, and thus all loss terms are roughly equally weighted. The classification loss L_{cls} (as well as bounding box regression loss L_{loc}) and similarity loss L_{sim} are scaled with weight factor H_{lss} and $1-H_{lss}$, respectively, such that the similarity loss penalizes less on reliable instances and more on unreliable instances (vice versa for the loss L_{cls}). The classification term aims to distinguish the positive class from the others, while the similarity constraint term promotes intra-class similarity. The parameter H_{lss} adapts the contribution of each term in updating the latent variables.

Updating Latent Variables A key component of using H_{lss} as reliability measurement is to avoid network overfitting, since overfitting may result in trivial solution that always chooses the initial regions as the mined objects, and makes the latent variable update ineffective. To solve this issue, we fix the order of training images fed into the network in each epoch, thus ensuring adequate training samples from other images during the latent variable update. Furthermore, we intentionally process images at different scales during the past t_0 epoches, which ensures that the network never sees identical samples during reliability measurement. Meanwhile, it is not necessary to store the prediction history of all previous epoches for reliability measurement. In practice we set t_0 as 4, and the overhead memory is constant. In order to avoid bad latent variable update, we exclude the first few epoches when measuring sample reliability and do not update the latent variables for these epoches.

Updating Clusters and Regression Targets Since the region features vary along with the network training, an ideal clustering strategy should be recomputing the features of the mined instances and running k-means clustering during each iteration. However, it is time-consuming and infeasible for network training. For trade-off, we only update the clusters after each epoch, as well as the bounding-box regression targets. We experimentally find that such approximate updating strategy works well in improving localization performance.

Choosing Candidate Positive Samples Following fast-RCNN, we use mini-batch sampling to generate region proposal features. Our latent variable update is based on current sampled region proposals, without any extra forward propagation, which requires the sampled regions to contain the object of interest as likely as they can be. However, sampling more regions for each mini-batch brings only negligible improvement (Shrivastava, Gupta, and Girshick 2016) but greatly increases training cost. To make a compromise,

Table 1: Effects of various design modules for Localization.

Baseline	WSDDN?	✓	✓	✓	✓	✓	✓	✓
	Multi-context?		✓	✓	✓	✓	✓	✓
	Fast RCNN?			✓	✓	✓	✓	✓
SD-LocNet	Multi-context?				✓	✓	✓	✓
	Adaptive Sampling?					✓		✓
	Prior Propagation?						✓	✓
	CorLoc (%)	55.1	58.6	61.7	62.9	65.7	63.1	**67.4**

we select the candidate positive regions based on the WSDDN (Bilen and Vedaldi 2016) outputs, and only retrain the top $\mathcal{R}_p$ regions based on the probability scores of the corresponding category. These regions are fixed during each mini-batch sampling and forwarded for latent variable update.

4 Experiments

We choose two pretrained models for experiments: 1) VGG-CNN-M (Chatfield et al. 2014), denoted as model **M**, for "medium"; 2) VGG-VD (Simonyan and Zisserman 2014) (the 16-layer model), denoted as model **L**, for "large". Our model contains multi-branch features, which nearly triples the network parameters. For efficiency, we reduce the parameters on the fully connected fc layers via a truncated SVD decomposition (Girshick 2015). Specifically, each *fc* layer with parameters $W \in d \times 4096$ (d is the dimension of input features) is decomposed into two sub-layers *fc-1* and *fc-2*, with weights $W_1 \in d \times 1024$ and $W_2 \in 1024 \times 4096$. We copy the parameters of the reduced *fc* layers to each path for network initialization. This leads to roughly the same number of parameters as the original network. All experiments choose edge boxes (Zitnick and Dollár 2014) to generate $|\mathcal{R}| \approx 2000$ region proposals per image on average, and choose *five-scales* with $s = \{384, 512, 640, 768, 896\}$ for training and testing. We denote the length of its shortest side as the scale s of an image, and cap the longest side at 1,500 pixels to avoid exceeding GPU memory. During network training, each SGD is constructed from $N = 1$ images with mini-batch size $\mathcal{R}_b = 256$, where $\mathcal{R}_p = 100$ proposals are fixed and the others are randomly sampled.

4.1 Experiments on PASCAL VOC 2007 and 2012

We perform experiments on two PASCAL VOC benchmarks: PASCAL VOC 2007 (Everingham et al. 2010) and VOC 2012 (Everingham et al. 2015), which are widely used for WSOL evaluation. PASCAL VOC 2007 contains 9,963 images spanning 20 object classes, with 5,011 images used for *trainval* and the rest 4,952 for *test*. PASCAL VOC 2012 contains 11,540 images for *trainval* and 10,991 for *test*. We use *trainval* split for training and *test* split for test. For performance evaluation, two kinds of measurements are used: 1) localization protocol CorLoc (Deselaers, Alexe, and Ferrari 2012) evaluated on the training set; 2) PASCAL style detection protocol (AP) evaluated on the test set.

Model Analysis We first conduct comparative experiments with different configurations to reveal how each module affects the localization performance. The ablation experiments are performed on PASCAL VOC 2007 with model **M**, and the results are shown in Table 1. For fair comparisons, we set up another baseline by using WSDDN for initialization, and simply training a fast-RCNN model to refine the localization, which is similar with (Tang et al. 2017). From the table we make the following observations:

• *Multi-context features help.* We empirically demonstrate that introducing multi-context features improves localization performance of both WSDDN and our SD-LocNet model consistently. For WSDDN, the performance increases by 3.5% (55.1% $\rightarrow$ 58.6%), while the gain is 1.2% (61.7% $\rightarrow$ 62.9%) when we train a fast-RCNN model using multi-context features. The reason is that multi-context features represent object proposals with mixed context information, and are helpful in improving localization.

• *Adaptive sampling is crucial.* Adding the adaptive sampling module brings another 2.8% (62.9% $\rightarrow$ 65.7%) improvement in localization. Adaptive sampling is able to automatically figure out the well localized instances for latent variable update, and assign each instance a reliable weight for network optimization. Thus the network is robust to those noisy labeled instances.

• *Prior propagation improves localization.* Introducing object prior propagation brings another 1.7% (65.7% $\rightarrow$ 67.4%) improvement. Note, the reliability of object priors is important for performance improvement. For comparison, we simply add similarity constraints by penalizing the differences between mined objects and representative clusters, which are obtained from all instances mined from the previous epoch, without discriminating their reliability. The performance decreases to 63.1%, proving the necessity of selecting trusted instances for prior propagation.

Parameter Analysis We now analyze the influences of three parameters, *i.e.,* the number of clusters K for each category, the start update epoch T_0, and the number of candidate positives $\mathcal{R}_p$. For simplicity, the default settings for three parameters are $K = 3$, $T_0 = 5$, and $R_p = 100$, and the other two parameters are fixed during inspecting the target parameter. The localization results w.r.t. different parameters are shown in Fig. 3. We empirically find that: 1) The performance improves by around 1% when K increases from 1 to 3, and is stable with larger K value. 2) The start update epoch T_0 is robust as the training process goes, as long as we exclude the prediction history near the beginning epoches ($T_0 = 3, 4$). It is intuitive since the network has not learned object priors during the initial few epoches. 3) Our method achieves comparable results when only retain $\mathcal{R}_p = 100$ (retaining over 99% probability contributions for classification) samples on average, which excludes the majority of negative samples for positive instance mining, and accelerates the network training without sacrificing performance. We compute the recall of $\mathcal{R}_p$ with respect to ground truth boxes at different overlapping thresholds, obtained by the proportion of ground truth boxes for which there exists a re-

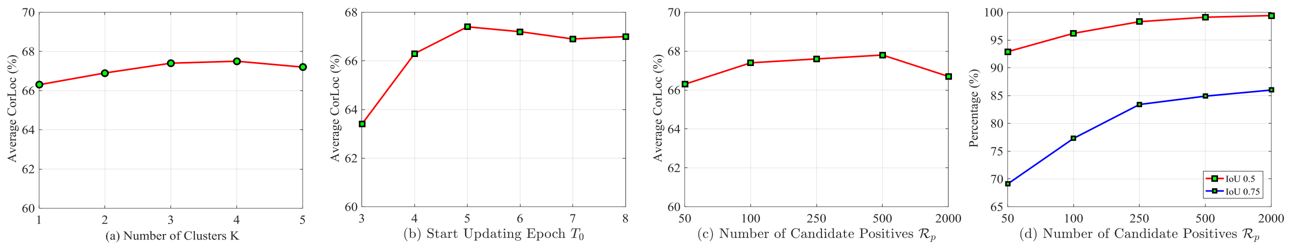

Figure 3: Localization results on PASCAL VOC 2007 versus (a) number of clusters K, (b) start updating epoch T_0, and (c) number of candidate positives $\mathcal{R}_p$. In (d), we compute the recall of $\mathcal{R}_p$ at different overlapping thresholds.

Table 2: Localization precision (%) comparisons on PASCAL VOC 2007 *trainval* split

method	aer	bik	brd	boa	btl	bus	car	cat	cha	cow	tbl	dog	hrs	mbk	prs	plt	shp	sfa	trn	tv	mean
WSDDN(Bilen and Vedaldi 2016)	65.1	58.8	58.5	33.1	39.8	68.3	60.2	59.6	34.8	64.5	30.5	43.0	56.8	82.4	25.5	41.6	61.5	55.9	65.9	63.7	53.5
ConLocNet(Kantorov et al. 2016)	83.3	68.6	54.7	23.4	18.3	73.6	74.1	54.1	8.6	65.1	47.1	59.5	67.0	83.5	35.3	39.9	67.0	49.7	63.5	65.2	55.1
ODGA(Diba et al. 2017b)	85.5	75.0	66.9	47.5	43.6	67.4	83.6	61.7	36.8	75.1	29.8	55.9	70.4	80.6	29.0	52.9	71.0	31.2	66.9	58.1	59.4
OICR(Tang et al. 2017)	81.7	80.4	48.7	**49.5**	32.8	81.7	85.4	40.1	**40.6**	79.5	35.7	33.7	60.5	88.8	21.8	57.9	76.3	59.9	75.3	**81.4**	60.6
OICR **Ens.** (Tang et al. 2017)	**85.8**	82.7	62.8	45.2	43.5	**84.8**	87.0	46.8	15.7	**82.2**	51.0	45.6	83.7	91.2	22.2	59.7	75.3	65.1	76.8	78.1	64.3
SD-LocNet-**M**	76.6	78.0	66.2	47.4	51.8	81.7	86.7	**79.6**	36.7	72.6	60.0	61.6	80.4	88.8	43.4	54.0	73.9	64.3	78.4	65.1	67.4
SD-LocNet-**L**	81.5	**83.5**	**70.0**	46.4	**52.9**	80.1	**89.1**	73.6	37.3	75.2	**73.0**	**62.5**	**85.0**	**91.4**	**51.3**	**63.7**	**79.2**	**72.9**	**80.5**	68.8	**70.9**

Table 3: Detection precision (%) comparisons on PASCAL VOC 2007 *test* split

method	aer	bik	brd	boa	btl	bus	car	cat	cha	cow	tbl	dog	hrs	mbk	prs	plt	shp	sfa	trn	tv	mAP
WSDDN(Bilen and Vedaldi 2016)	39.4	50.1	31.5	16.3	12.6	64.5	42.8	42.6	10.1	35.7	24.9	38.2	34.4	55.6	9.4	14.7	30.2	40.7	54.7	46.9	34.8
ConLocNet(Kantorov et al. 2016)	57.1	52.0	31.5	7.6	11.5	55.0	53.1	34.1	1.7	33.1	49.2	42.0	47.3	56.6	15.3	12.8	24.8	48.9	44.4	47.8	36.3
OICR(Tang et al. 2017)	58.0	62.4	31.1	19.4	13.0	65.1	62.2	28.4	**24.8**	44.7	30.6	25.3	37.8	65.5	15.7	24.1	41.7	46.9	64.3	**62.6**	41.2
ODGA(Diba et al. 2017b)	50.9	61.2	40.5	**31.4**	21.1	**71.6**	58.1	42.9	11.7	46.4	30.7	44.5	48.3	64.9	16.8	24.8	47.1	55.7	61.7	55.8	44.3
OICR **Ens.**(Tang et al. 2017)	**65.5**	67.2	47.2	21.6	22.1	68.0	68.5	35.9	5.7	**63.1**	49.5	30.3	64.7	66.1	13.0	25.6	**50.0**	57.1	60.2	59.0	47.0
SD-LocNet-**M**	59.2	67.2	46.1	29.6	21.6	65.7	68.0	61.4	17.3	53.1	50.6	38.3	67.7	69.1	**26.8**	25.1	49.2	52.2	60.9	39.6	48.4
SD-LocNet-**L**	65.1	**72.5**	**50.3**	31.3	**24.2**	68.3	**70.9**	**65.1**	19.0	51.0	**53.4**	47.5	**70.2**	**69.9**	25.9	**25.6**	45.8	**57.9**	**71.2**	40.9	**51.3**
Fast-RCNN(Girshick 2015)	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8	66.9

Table 4: Comparisons on PASCAL VOC 2012.

Method	CorLoc	mAP
ConLocNet(Kantorov et al. 2016)	54.8	35.3
DSD(Jie et al. 2017)	58.8	38.3
OICR(Tang et al. 2017)	62.1	37.9
ZLDN (Zhang et al. 2018)	61.5	42.9
SD-LocNet	**69.1**	**49.2**
Fast-RCNN (Girshick 2015)	-	65.7

Table 5: Comparisons on MS COCO 2014.

Methods	CorLoc	mAP	
		@.5	@[.5,.95]
WSDDN(Bilen and Vedaldi 2016)	26.1	11.5	4.3
WCCN (Diba et al. 2017a)	-	12.3	-
ODGA (Diba et al. 2017b)	-	12.8	-
SD-LocNet	**40.3**	**21.6**	**9.7**
Fast-RCNN (Girshick 2015)	-	38.6	18.9

gion proposal with overlap at least 0.5 and 0.75, respectively. As shown in Fig. 3 (d), the recall is over 95% at threshold 0.5 when $\mathcal{R}_p = 100$, which demonstrates that the candidate positives contain the object with high probability.

Comparisons with State-of-the-arts We compare our results with state-of-the-arts for both localization and detection. Unless specified, all other results choose model **L**.

• *CorLoc evaluation.* Table 2 shows the localization results on PASCAL VOC 2007 *trainval* split in terms of CorLoc (Deselaers, Alexe, and Ferrari 2012). Our method achieves an accuracy of 67.4% with model **M**, which already outperforms the state-of-the-art (Tang et al. 2017) (60.6%) with a deeper model. Moreover, replacing with model **L**, we achieve a CorLoc of 70.9%, 6.6% better than previous best-performing result (Tang et al. 2017) (64.3%) using model ensemble. We also report localization results on PASCAL VOC 2012, shown in Table 4. SD-LocNet achieves 69.1% CorLoc with model **L**, 7% point higher than previous best result (Tang et al. 2017) (62.1%).

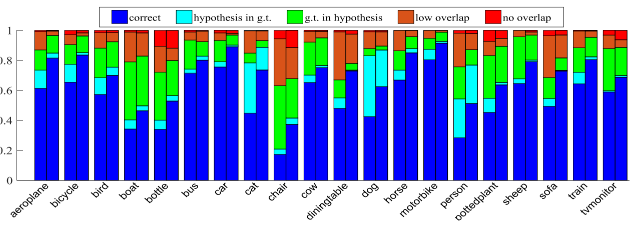

Figure 4: Per-class frequency of error modes on PASCAL VOC 2007 with WSDDN (Bilen and Vedaldi 2016) (left column in each group) and SD-LocNet (right column).

In order to better understand the localization errors, we categorize each of our predicted bounding boxes (object hypotheses) into the following five cases: 1) correct localization, *i.e.*, IoU overlap is greater than 0.5 with the ground truth. 2) hypothesis completely inside the ground truth, 3) ground truth completely inside the hypothesis, 4) no overlap, IoU equals to zero, and 5) low overlap, none of the above. Fig. 4 illustrates the error distribution of WSDDN (Bilen and Vedaldi 2016) and the proposed SD-LocNet across 20 categories on Pascal VOC 2007 trainval set. It can be shown that SD-LocNet improves localization for all classes, but the error modes vary according to different classes. For deformable objects such as *cat* and *person*, the main error comes from localizing object parts (hypotheses in g.t.), while for classes such as *boat* and *chair*, the majority of error is localizing too large region (g.t. in hypotheses), since it is difficult to localize the object or object parts due to the cluttered background or the small scale target. Some example localizations are shown in Fig. 5 (top row). The results of SD-LocNet are shown in green bounding boxes, and the initial localization results returned by WSDDN are in red bounding boxes. It can be seen that SD-LocNet is able to refine the localizations that are noisily labeled.

• *AP evaluation.* Table 3 shows the detection performance on VOC 2007 *test* split. Just using model **M**, our method achieves an accuracy of 48.4%, 4.1% higher than the best-performing method (Diba et al. 2017b) (44.3%) using a single model. When switching to model **L**, the detection accuracy increases to 51.3%, which is about 4.3% better than the best-performing result (Tang et al. 2017) (47.0%). For PASCAL VOC 2012, the result is 49.2% with model **L**, 6.3% more accurate than (Zhang et al. 2018) (42.9%).

• *Comparisons with fully supervised fast-RCNN.* It is interesting to compare our weakly supervised detections with the fast-RCNN (Girshick 2015) which uses ground truth bounding box annotations for training. As shown in Table 3, the performance of SD-LocNet is around 15% lower than fast-RCNN. However, for vehicles like *motorbike* and *train*, the performance approaches (within 5% gap) the fully supervised one (69.9% vs 73.0% for *motorbike*, and 71.2% vs 75.8% for *train*). This implies that it is possible to train corresponding detection models on these classes without requiring object annotations. However, for classes such as *chair* and *person*, the performance gap is still large. It remains a further research orientation to correctly localize these objects for detection model training.

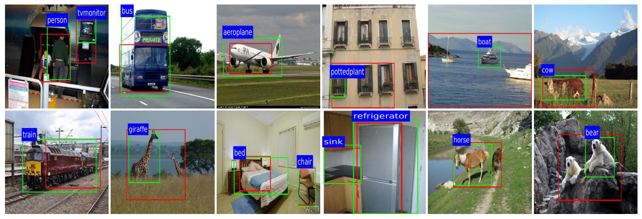

Figure 5: Example localization comparisons of WSDDN (Bilen and Vedaldi 2016) (red bounding boxes) and SD-LocNet (green bounding boxes) on PASCAL VOC 2007 (top tow) and MS COCO 2014 (bottom row). SD-LocNet is able to update the noisy labeled instances for better localization.

4.2 Experiments on MS COCO

To further validate the effectiveness of SD-LocNet, we evaluate it on a much larger dataset MS COCO 2014 (Lin et al. 2014) with over 135k images spanning 80 categories, of which around 80k images are used for *train* and around 40k for *val*. We choose the *train* split for training and the *val* split for test. Compared with PASCAL VOC 2012, MS COCO is more challenging: it includes more images (135k vs 22k), more categories (80 vs 20), more complex scenes (7.7 instances per image vs 2.3 instances per image, averagely), and more images with smaller objects. To our best knowledge, few works have reported results on MS COCO under weakly supervised paradigms.

Table 5 shows localization and detection results on MS COCO 2014. We obtain 40.3% localization accuracy, improving the baseline WSDDN (Bilen and Vedaldi 2016) by 14.2%. For detection, the PASCAL style evaluation (mAP @.5) result is 21.6%, 8.8% better than (Diba et al. 2017b) (12.8%). Some example localization results are shown in Fig. 5 (bottom row). We also report the new COCO-style criterion (mAP@[.5, .95]). Our method achieves 9.7% mAP under the weakly supervised paradigm. For comparisons, the last row shows results of fast-RCNN using ground truth object annotations for training. Our results are promising considering the challenges of this dataset.

5 Conclusions

This paper proposed a robust optimization strategy to improve localization accuracy in WSOL, which first selects reliable instances for model training, and then propagates object priors of the reliable instances to the unreliable ones for better localization. With our effective Localization Stability Score measuring the reliability of candidate positive samples, we emphasize the reliable instances that are well localized for robust learning. Then, we propagate object priors of reliable instances to direct localization on difficult images by explicitly imposing intra-similarity. An SD-LocNet is instantiated from the proposed optimization strategy. Experiments on benchmark datasets well confirm the effectiveness of our proposed method.

Acknowledgements. The work was supported in part to Jiashi Feng by NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112, in part to Yang Yang by NSFC under Project 61572108 and Project 61632007.

References

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *ICML*, 41–48. ACM.

Bilen, H., and Vedaldi, A. 2016. Weakly supervised deep detection networks. In *CVPR*, 2846–2854.

Bilen, H.; Pedersoli, M.; and Tuytelaars, T. 2015. Weakly supervised object detection with convex clustering. In *CVPR*, 1081–1089.

Chatfield, K.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Return of the devil in the details: Delving deep into convolutional nets. *BMVC*.

Cinbis, R. G.; Verbeek, J.; and Schmid, C. 2014. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2409–2416.

Deselaers, T.; Alexe, B.; and Ferrari, V. 2012. Weakly supervised localization and learning with generic knowledge. *IJCV* 100(3):275–293.

Diba, A.; Sharma, V.; Pazandeh, A.; Pirsiavash, H.; and Van Gool, L. 2017a. Weakly supervised cascaded convolutional networks. *CVPR* 914–922.

Diba, A.; Sharma, V.; Stiefelhagen, R.; and Van Gool, L. 2017b. Object discovery by generative adversarial & ranking networks. *arXiv preprint arXiv:1711.08174*.

Dietterich, T. G.; Lathrop, R. H.; and Lozano-Pérez, T. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artifi. Intell.* 89(1):31–71.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *IJCV* 88(2):303–338.

Everingham, M.; Eslami, S. A.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2015. The pascal visual object classes challenge: A retrospective. *IJCV* 111(1):98–136.

Girshick, R. 2015. Fast r-cnn. In *ICCV*, 1440–1448.

Jie, Z.; Wei, Y.; Jin, X.; Feng, J.; and Liu, W. 2017. Deep self-taught learning for weakly supervised object localization. *CVPR*.

Joulin, A., and Bach, F. 2012. A convex relaxation for weakly supervised classifiers. *ICML*.

Kantorov, V.; Oquab, M.; Cho, M.; and Laptev, I. 2016. Contextlocnet: Context-aware deep network models for weakly supervised localization. In *ECCV*, 350–365. Springer.

Kumar, M. P.; Packer, B.; and Koller, D. 2010. Self-paced learning for latent variable models. In *NIPS*, 1189–1197.

Li, D.; Huang, J.-B.; Li, Y.; Wang, S.; and Yang, M.-H. 2016. Weakly supervised object localization with progressive domain adaptation. In *CVPR*, 3512–3520.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *ECCV*, 740–755. Springer.

Nguyen, M. H.; Torresani, L.; de la Torre, F.; and Rother, C. 2009. Weakly supervised discriminative localization and classification: a joint learning process. In *Proc. Int. Conf. Comput. Vis.*, 1925–1932.

Oquab, M.; Bottou, L.; Laptev, I.; and Sivic, J. 2015. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *CVPR*, 685–694.

Ouyang, D.; Shao, J.; Zhang, Y.; Yang, Y.; and Shen, H. T. 2018. Video-based person re-identification via self-paced learning and deep reinforcement learning framework. In *ACM Multimedia*, 1562–1570.

Shrivastava, A.; Gupta, A.; and Girshick, R. 2016. Training region-based object detectors with online hard example mining. In *CVPR*, 761–769.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.

Song, H. O.; Girshick, R. B.; Jegelka, S.; Mairal, J.; Harchaoui, Z.; Darrell, T.; et al. 2014a. On learning to localize objects with minimal supervision. In *ICML*, 1611–1619.

Song, H. O.; Lee, Y. J.; Jegelka, S.; and Darrell, T. 2014b. Weakly-supervised discovery of visual pattern configurations. In *NIPS*, 1637–1645.

Tang, P.; Wang, X.; Bai, X.; and Liu, W. 2017. Multiple instance detection network with online instance classifier refinement. In *CVPR*.

Vijayanarasimhan, S., and Grauman, K. 2008. Keywords to visual categories: Multiple-instance learning forweakly supervised object categorization. In *CVPR*, 1–8. IEEE.

Wang, C.; Ren, W.; Huang, K.; and Tan, T. 2014. Weakly supervised object localization with latent category learning. In *ECCV*. 431–445.

Zhang, X.; Feng, J.; Xiong, H.; and Tian, Q. 2018. Zigzag learning for weakly supervised object detection. In *CVPR*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*, 2921–2929. IEEE.

Zhu, Y.; Zhou, Y.; Ye, Q.; Qiu, Q.; and Jiao, J. 2017. Soft proposal networks for weakly supervised object localization. *arXiv preprint arXiv:1709.01829*.

Zitnick, C. L., and Dollár, P. 2014. Edge boxes: Locating object proposals from edges. In *ECCV*. 391–405.