Instance-Aware, Context-Focused, and Memory-Efficient Weakly Supervised Object Detection

Zhongzheng Ren[1,2*] Zhiding Yu[2] Xiaodong Yang[2*] Ming-Yu Liu[2]
Yong Jae Lee[3] Alexander G. Schwing[1] Jan Kautz[2]
[1]University of Illinois at Urbana-Champaign [2]NVIDIA [3]University of California, Davis

Abstract

Weakly supervised learning has emerged as a compelling tool for object detection by reducing the need for strong supervision during training. However, major challenges remain: (1) differentiation of object instances can be ambiguous; (2) detectors tend to focus on discriminative parts rather than entire objects; (3) without ground truth, object proposals have to be redundant for high recalls, causing significant memory consumption. Addressing these challenges is difficult, as it often requires to eliminate uncertainties and trivial solutions. To target these issues we develop an instance-aware and context-focused unified framework. It employs an instance-aware self-training algorithm and a learnable Concrete DropBlock while devising a memory-efficient sequential batch back-propagation. Our proposed method achieves state-of-the-art results on COCO (12.1% AP, 24.8% AP_{50}), VOC 2007 (54.9% AP), and VOC 2012 (52.1% AP), improving baselines by great margins. In addition, the proposed method is the first to benchmark ResNet based models and weakly supervised video object detection. Refer to our project page for code, models, and more details: https://github.com/NVlabs/wetectron.

1. Introduction

Recent works on object detection [17, 35, 34, 26] have achieved impressive results. However, the training process often requires strong supervision in terms of precise bounding boxes. Obtaining such annotations at a large scale can be costly, time-consuming, or even infeasible. This motivates weakly supervised object detection (WSOD) methods [5, 45, 22] where detectors are trained with weaker forms of supervision such as image-level category labels. These works typically formulate WSOD as a multiple instance learning task, treating the set of object proposals in each image as a bag. The selection of proposals that truly cover objects is modeled using learnable latent variables.

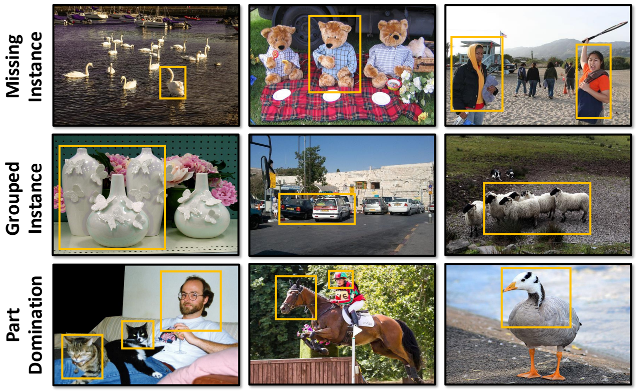


Figure 1: Typical WSOD issues: (1) **Instance Ambiguity:** missing less salient objects (top) or failing to differentiate clustered instances (middle); (2) **Part Domination:** focusing on most discriminative object parts (bottom).

While alleviating the need for precise annotations, existing weakly supervised object detection methods [5, 45, 50, 40, 60] often face three major challenges due to the underdetermined and ill-posed nature, as demonstrated in Fig. 1:

(1) Instance Ambiguity. This arguably the biggest challenge which subsumes two common types of issues: (a) *Missing Instances:* Less salient objects in the background with rare poses and smaller scales are often ignored (top row in Fig. 1). (b) *Grouped Instances:* Multiple instances of the same category are grouped into a single bounding box when spatially adjacent (middle row in Fig. 1). Both issues are caused by bigger or more salient boxes receiving higher scores than smaller or less salient ones.

(2) Part Domination. Predictions tend to be dominated by the most discriminative parts of an object (Fig. 1 bottom). This issue is particularly pronounced for classes with big intra-class difference. For example, on classes such as animals and people, the model often turns into a 'face detector' as faces are the most consistent appearance signal.

(3) Memory Consumption. Existing proposal generation methods [49, 64] often produce dense proposals. Without ground-truth localization, maintaining a large number

[*] Work partially done at NVIDIA.

of proposals is necessary to achieve a reasonable recall rate and good performance. This requires a lot of memory, especially for video object detection. Due to the large number of proposals, most memory is consumed in the intermediate layers after ROI-Pooling.

To address the above three challenges, we propose a unified weakly supervised learning framework that is instance-aware and context-focused. The proposed method tackles **Instance Ambiguity** by introducing an advanced self-training algorithm where instance-level pseudo ground-truth, in forms of category labels and regression targets are computed by considering more instance-associative spatial diversification constraints (Sec. 4.1). The proposed method also addresses **Part Domination** by introducing a parametric spatial dropout termed 'Concrete DropBlock.' This module is learned end-to-end to adversarially maximize the detection objective, thus encouraging the whole framework to consider context rather than focusing on the most discriminative parts (Sec. 4.2). Finally, to alleviate the issue of **Memory Consumption**, our method adopts a sequential batch back-propagation algorithm which processes data in batches at the most memory-heavy stage. This permits the assess to larger deep models such as ResNet [18] in WSOD, as well as the exploration of weakly supervised video object detection (Sec. 4.3).

Tackling the aforementioned three challenges via our proposed framework leads to state-of-the-art performance on several popular datasets, including COCO [29], VOC 2007 and 2012 [11]. The effectiveness and robustness of each proposed module is demonstrated in detailed ablation studies, and further verified through qualitative results. Finally, we conduct additional experiments on videos and give the first benchmark for weakly supervised video object detection on ImageNet VID [8].

2. Related work

Weakly supervised object detection (WSOD). Object detection is one of the most fundamental problems in computer vision. Recent supervised methods [16, 15, 35, 17, 34, 30, 26] have shown great performance in terms of both accuracy and speed. For WSOD, most methods formulate a multiple instance learning problem where input images contain a bag of instances (object proposals). The model is trained with a classification loss to select the most confident positive proposals. Modifications w.r.t. initialization [43, 42], regularization [7, 3, 54], and representations [7, 4, 27] have been shown to improve results. For instance, Bilen and Vedaldi [5] proposed an end-to-end trainable architecture for this task. Follow-up works further improve by leveraging spatial relations [45, 44, 22], better optimization [61, 21, 2, 50], and multitasking with weakly supervised segmentation [13, 37, 12, 40].

Self-training for WSOD. Among the above directions, self-training [66, 65] has been demonstrated to be seminal. Self-training uses instance-level pseudo labels to augment training and can be implemented in an **offline** manner [62, 41, 27, 62]: a WSOD model is first trained using any of the methods discussed above; then the confident predictions are used as pseudo-labels to train a final supervised detector. This iterative knowledge distillation procedure is beneficial since the additional supervised models learn form less noisy data and usually have better architectures for which training is time-consuming. A number of works [45, 44, 50, 12, 60, 46] studied end-to-end implementations of self-training: WSOD models compute and use pseudo labels simultaneously during training, which is commonly referred to as an **online** solution. However, these methods typically only consider the most confident predictions for pseudo-labels. Hence they tend to have overfitting issues with difficult parts and instances ignored.

Spatial dropout. To address the above issue, an effective regularization strategy is to drop parts of spatial feature maps during training. Variants of spatial-dropout have been widely designed for supervised tasks such as classification [14], object detection [53], and human joints localization [48]. Similar approaches have also been applied in weakly supervised tasks for better localization in detection [39] and semantic segmentation [55]. However, these methods are non-parametric and cannot adapt to different datasets in a data-driven manner. As a further improvement, Kingma *et al.* [23] designed variational dropout where the dropout rates are learned during training. Wang *et al.* [53] proposed a parametric but non-differentiable spatial-dropout trained with REINFORCE [57]. In contrast, the proposed 'Concrete DropBlock' module has a parametric and differentiable structured novel form.

Memory efficient back-propagation. Memory has always been a concern since deeper models [18, 38] and larger batch size [32] often tend to yield better results. One way to alleviate this concern is to trade computation time for memory consumption by modifying the back-propagation (BP) algorithm [36]. A suitable technique [24, 33, 6] is to not store some intermediate deep net representations during forward-propagation. One can recover those by injecting small forward passes during back-propagation. Hence, the one-stage back-propagation is divided into several stepwise processes. However, this method cannot be directly applied to our model where a few intermediate layers consume most of the memory. To address it, we suggest a batch operation for the memory-heavy intermediate layers.

3. Background

Bilen and Vedaldi [5] are among the first to develop an end-to-end deep WSOD framework based on the idea of multiple instance learning. Specifically, given an input image I and the corresponding set of pre-computed [49, 64] proposals R, an ImageNet [8] pre-trained neural network is used to produce classification logits $f_w(c,r) \in \mathbb{R}$ and detection logits $g_w(c,r) \in \mathbb{R}$ for every object category $c \in C$ and for every region $r \in R$. The vector w subsumes all trainable parameters. Two score matrices, *i.e.*, $s(c|r)$ of a region r being classified as category c, and $s(r|c)$ of detecting region r for category c are obtained through

$$s_w(c|r) = \frac{\exp f_w(c,r)}{\sum_{c \in C} \exp f_w(c,r)}, \text{ and } s_w(r|c) = \frac{\exp g_w(c,r)}{\sum_{r \in R} \exp g_w(c,r)}. \tag{1}$$

The final score $s_w(c,r)$ for assigning category c to region r is computed via an element-wise product: $s_w(c,r) = s_w(c|r) s_w(r|c) \in [0,1]$. During training, $s_w(c,r)$ is summed for all regions $r \in R$ to obtain the image evidence $\phi_w(c) = \sum_{r \in R} s_w(c,r)$. The loss is then computed via:

$$\mathcal{L}_{\text{img}}(w) = -\sum_{c \in C} y(c) \log \phi_w(c), \tag{2}$$

where $y(c) \in \{0,1\}$ is the ground truth (GT) class label indicating image-level existence of category c. For inference, $s_w(c,r)$ is used for prediction followed by standard non-maximum suppression (NMS) and thresholding.

To integrate online self-training, the region score $s_w(c,r)$ is often used as teacher to generate instance-level pseudo category label $\hat{y}(c,r) \in \{0,1\}$ for every region $r \in R$ [44, 50, 12, 60, 46]. This is done by treating the top-scoring region and its highly-overlapped neighbors as the positive examples for class c. The extra student layer is then trained for region classification via:

$$\mathcal{L}_{\text{roi}}(w) = -\frac{1}{|R|} \sum_{c \in C} \hat{y}(c,r) \log \hat{s}_w(c|r), \tag{3}$$

where $\hat{s}_w(c|r)$ is the output of this layer. During testing, the student prediction $\hat{s}_w(c|r)$ will be used rather than $s_w(c,r)$. We build upon this formulation and develop two additional novel modules as described subsequently.

4. Approach

Image-level labels are an effective form of supervision to mine for common patterns across images. Yet inexact supervision often causes localization ambiguity. To address the mentioned three challenges caused by this ambiguity, we develop the instance-aware and context-focused framework outlined in Fig. 2. It contains a novel online self-training algorithm with ROI regression to reduce instance ambiguity and better leverage the self-training supervision (Sec. 4.1). It also reduces part-domination for classes with large intra-class variance via a novel end-to-end learnable 'Concrete DropBlock' (Sec. 4.2), and it is more memory friendly (Sec. 4.3).

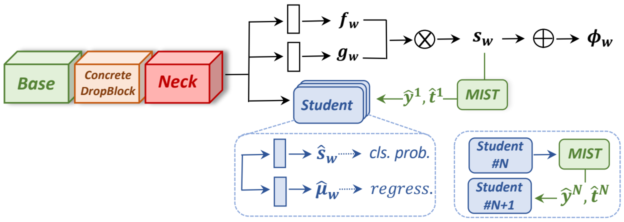


Figure 2: The overall framework. ROI-Pooling and the operations in Eq. (1) are abstracted away for readability.

4.1. Multiple instance self-training (MIST)

With online or offline generated pseudo-labels [44, 41, 62], self-training helps to eliminate localization ambiguities, benefiting mainly from two aspects: (1) Pseudo-labels permit to model proposal-level supervision and inter-proposal relations; (2) Self-training can be broadly regarded as a teacher-student distillation process which has been found helpful to improve the student's representation. We take the following dimensions into account when designing our framework:

Instance-associative: Object detection is often 'instance-associative': highly overlapping proposals should be assigned similar labels. Most self-training methods for WSOD ignore this and instead treat proposals independently. Instead, we impose explicit instance-associative constraints into pseudo box generation.

Representativeness: The score of each proposal in general is a good proxy for its representativeness. It is not perfect, especially in the beginning there is a tendency to focus on object parts. However, the score provides a high recall for being at least located on correct objects.

Spatial-diversity: Imposing spatial diversity to the selected pseudo-labels can be a useful self-training inductive bias. It promotes better coverage on difficult (*e.g.*, rare appearance, poses, or occluded) objects, and higher recall for multiple instances (*e.g.*, diverse scales and sizes).

The above constraints and criteria motivate a novel algorithm to generate diverse yet representative pseudo boxes which are instance-associative. The details are provided in Alg. 1. Specifically, we first sort all the scores across the set R for each class c that appears in the category-label. We then pick the top p percent of the ranked regions to form an initial candidate pool $R'(c)$. Note that the size of the candidate pool $R'(c)$, *i.e.*, $|R'(c)|$ is image-adaptive and content-dependent by being proportional to $|R|$. Intuitively, $|R|$ is a meaningful prior for the overall objectness of an input image. A diverse set of high-scoring non-overlapping regions are then picked from $R'(c)$ as the pseudo boxes $\hat{R}(c)$ using non-maximum suppression. Even though being simple, this effective algorithm leads to significant performance improvements as shown in Sec. 5.

Algorithm 1 Multiple Instance Self-Training

```
Input: Image $I$, class label $y$, proposals $R$, threshold $\tau$, percentage $p$
Output: Pseudo boxes $\hat{R}^1$
Feed $I$ into model; get ROI scores $s$
for ground-truth class $c$ do
    $R(c)_{sorted} \leftarrow$ SORT($s(c,*)$)    //sort ROIs by scores of class $c$
    $R'(c) \leftarrow$ top $p$ percent of $R(c)_{sorted}$
    $\hat{R}(c) \leftarrow r'_0$    // save first region (top-scoring) $r'_0 \in R'$
    for $i$ in $\{2 ... |R'(c)|\}$ do    // start from the second highest
        APPEND($\hat{R}(c), r'_i$) if IoU($r'_i, \hat{r}_j$) $< \tau$, $\forall\, \hat{r}_j \in \hat{R}(c)$
return $\hat{R}(c)$
```

Self-training with regression. Bounding box regression is another module that plays an important role in supervised object detection but is missing in online self-training methods. To close the gap, we encapsulate a classification layer and a regression layer into 'student blocks' as shown via blue boxes in Fig. 2. We jointly optimize them using pseudo-labels $\hat{R}$. The predicted bounding boxes from the regression layer are referred to via $\mu_w(r)$ for all regions $r \in R$. For each region r, if it is highly overlapping with a pseudo-box $\hat{r} \in \hat{R}$ for ground-truth class c, we generate the regression target $\hat{t}(r)$ by using the coordinates of $\hat{r}$ and by marking the classification label $\hat{y}(c,r) = 1$. The complete region-level loss for training the student block is:

$$\begin{aligned}\mathcal{L}_{\text{roi}}(w) = &\frac{1}{|R|}\sum_{r\in R}\lambda_r(\mathcal{L}_{\text{smooth-L1}}(\hat{t}(r),\mu_w(r)) \\ &- \frac{1}{|C|}\sum_{c\in C}\hat{y}(c,r)\log\hat{s}_w(c|r)),\end{aligned} \tag{4}$$

where $\mathcal{L}_{\text{smooth-L1}}$ is the Smooth-L1 objective used in [15] and λ_r is a scalar per-region weight used in [45].

In practice, conflicts happen when we force the $\hat{y}(\cdot,r)$ to be a one-hot vector since the same region can be chosen to be positive for different ground-truth classes, especially in the early stages of training. Our solution is to use that class for pseudo-label $\hat{r}$ which has a higher predicted score $s(c,\hat{r})$. In addition, the obtained pseudo-labels and the proposals are inevitably noisy. Imposing bounding box regression is able to correctly learn from the noisy labels by capturing the most consistent patterns among them, and refining the noisy proposal coordinates accordingly. We empirically verify in Sec. 5.3 that bounding box regression improves both robustness and generalization.

Self-ensembling. We follow [45, 44] to stack multiple student blocks to improve performance. As shown in Fig. 2, the first pseudo-label $\hat{R}^1$ is generated from the teacher branch, and then the student block N generates pseudo-label $\hat{R}^N$ for the next student block $N+1$. This technique is similar to the self-ensembling method [25].

4.2. Concrete DropBlock

Because of the intra-category variation, existing WSOD methods often mistakenly only detect the discriminative parts of an object rather than its full extent. A natural solution for this issue encourages the network to focus on the context which can be achieved by dropping the most discriminative parts. Hence, spatial dropout is an intuitive fit.

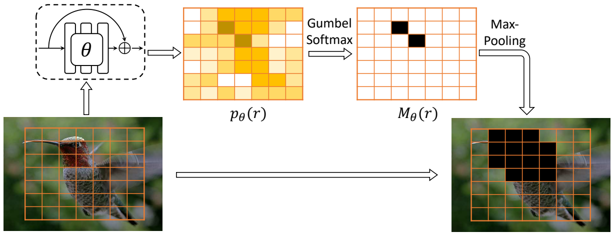
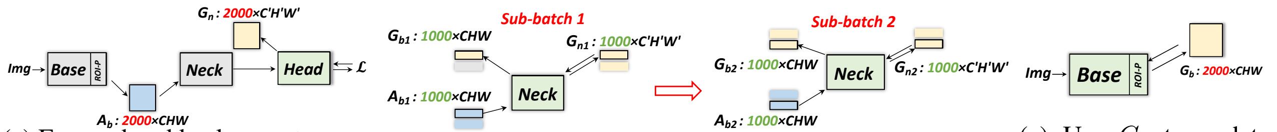


Figure 3: Illustration of the Concrete DropBlock idea. Discriminative parts such as head are zeroed out.

Naïve spatial dropout has limition for detection since the discriminative parts of objects differ in location and size. A more structured DropBlock [14] was proposed where spatial points on ROI feature maps are sampled randomly as blob centers, and the square regions around these centers of size $H \times H$ are then dropped across all channels on the ROI feature map. Finally, the feature values are re-scaled by a factor of the area of the whole ROI over the area of the undropped region so that no normalization has to be applied for inference when no regions are dropped.

DropBlock is a non-parametric regularization technique. While it is able to improve model robustness and alleviate part domination, it basically treats regions equally. We consider dropping more frequently at discriminative parts in an adversarial manner. To this end, we develop the *Concrete DropBlock*: a data-driven and parametric variant of DropBlock which is learned end-to-end to drop the most relevant regions as shown in Fig. 3. Given an input image, the feature maps $\psi_w(r) \in \mathbb{R}^{H\times H}$ are computed for each region $r \in R$ using the layers up until ROI-Pooling. H is the ROI-Pooling output dimension. We then feed $\psi_w(r)$ into a convolutional residual block to generate a probability map $p_\theta(r) \in \mathbb{R}^{H\times H}\ \forall r \in R$ where θ subsumes the trainable parameters of this module. Each element of $p_\theta(r)$ is regarded as an independent Bernoulli variable, and this probability map is transformed via a spatial Gumbel-Softmax [20, 31] into a hard mask $M_\theta(r) \in \{0,1\}^{H\times H}\ \forall r \in R$. This operation is a differentiable approximation of sampling. To avoid trivial solutions (*e.g.*, everything will be dropped or a certain area is dropped consistently), we apply a threshold τ such that $p_\theta(r) = \min(p_\theta(r),\tau)$. This guarantees that the computed mask $M_\theta(r)$ is sparse. We follow DropBlock to finally generate the structured mask and normalize the features. During training, we jointly optimize the original network parameters w and the residual block parameters θ with the following minmax objective:

$$w^*,\theta^* = \arg\min_w \max_\theta \sum_I \mathcal{L}_{\text{img}}(w,\theta) + \mathcal{L}_{\text{roi}}(w,\theta). \tag{5}$$



(a) Forward and back-prop to update 'Head'. A_b, G_n saved.

(b) Split A_b, G_n into sub-batches to update 'Neck'. G_b accumulated.

(c) Use G_b to update 'Base' network.

Figure 7: Seq-BBP: blue, yellow, and green blobs represent activation, gradients, and the module that is being updated.

By maximizing the original loss w.r.t. the Concrete DropBlock parameters, the Concrete DropBlock will learn to drop the most discriminative parts of the objects, as it is the easiest way to increase the training loss. This forces the object detector to also look at the context regions. We found this strategy to improve performance especially for non-rigid object categories, which usually have a large intra-class difference.

Methods	Val-AP	Val-AP$_{50}$	Test-AP	Test-AP$_{50}$
Fast R-CNN	18.9	38.6	19.3	39.3
Faster R-CNN	21.2	41.5	21.5	42.1
WSDDN [5]	-	-	-	11.5
WCCN [9]	-	-	-	12.3
PCL [44]	8.5	19.4		
C-MIDN [12]	9.6	21.4	-	-
WSOD2 [60]	10.8	22.7	-	-
Diba *et al.* [10]+SSD	-	-	-	13.6
OICR [45]+Ens+FRCNN	7.7	17.4	-	-
Ge *et al.* [13]+FRCNN	8.9	19.3	-	-
PCL [44]+Ens.+FRCNN	9.2	19.6	-	-
Ours (single-model)	**11.4**	**24.3**	**12.1**	**24.8**

Table 1: Single model results (VGG16) on COCO.

Methods	Proposal	Backbone	AP	AP$_{50}$
Faster R-CNN	RPN	R101-C4	27.2	48.4
Ours	MCG	VGG16	**11.4**	**24.3**
Ours	MCG	R50-C4	**12.6**	**26.1**
Ours	MCG	R101-C4	**13.0**	**26.3**

Table 2: Single model results (ResNet) on COCO 2014 val.

4.3. Sequential batch back-propagation

In this section, we discuss how we propose to handle memory limitations particularly during training, which turn out to be a major bottleneck preventing previous WSOD methods from using state-of-the-art deep nets. We introduce our memory-efficient *sequential batch forward and backward computation*, tailored for WSOD models.

Vanilla training via back-propagation [36] stores all intermediate activations during the forward pass, which are reused when computing gradients of network parameters. This method is computationally efficient due to memoization, yet memory-demanding for the same reason. More efficient versions [24, 6] have been proposed, where only a subset of the intermediate activations are saved during a forward pass at key layers. The whole model is cut into smaller sub-networks at these key layers. When computing gradients for a sub-network, a forward pass is first applied to obtain the intermediate representations for this sub-network, starting from the stored activation at the input key layer of the sub-network. Combined with the gradients propagated from earlier sub-networks, the gradients of sub-network weights are computed and gradients are also propagated to outputs of earlier sub-networks.

This algorithm is designed for extremely deep networks where the memory cost is roughly evenly distributed along the layers. However, when these deep nets are adapted for detection, the activations (after ROI-Pooling) grow from $1 \times CHW$ (image feature) to $N \times CHW$ (ROI-features) where N is in the thousands for weakly supervised models. Without ground-truth boxes, all these proposals need to be maintained for high recall and thus good performance (see the evidence in Appendix E).

To address this training challenge, we propose a sequential computation in the 'Neck' sub-module as depicted in Fig. 7. During the forward pass, the input image is first passed through the 'Base' and 'Neck,' with only the activation A_b after the 'Base' stored. The output of the 'Neck' then goes into the 'Head' for its first forward and backward pass to update the weights of the 'Head' and the gradients G_n as shown in Fig. 7 (a). To update the parameters of the 'Neck,' we split the ROI-features into 'sub-batches' and run back-propagation on each small sub-batch sequentially. Hence we avoid storing memory-consuming feature maps and their gradients within the 'Neck.' An example of this sequential method is shown in Fig. 7 (b), where we split 2000 proposals into two sub-batches of 1000 proposals each. The gradient G_b is accumulated and used to update the parameters of the 'Base' network via regular back-propagation as illustrated in Fig. 7 (c). For testing, the same strategy can be applied if either the number of ROIs or the size of the 'Neck' is too large.

5. Experiments

We assess our proposed method subsequently after detailing dataset, evaluation metrics and implementation.

Dataset and evaluation metrics. We first conduct experiments on COCO [29], which is the most popular dataset used for supervised object detection but rarely studied in WSOD. We use the COCO 2014 train/val/test split and report standard COCO metrics including AP (averaged over IoU thresholds) and AP$_{50}$ (IoU threshold at 50%).

We then evaluate on both VOC 2007 and 2012 [11], which are commonly used to assess WSOD performance. Average Precision (AP) with IoU threshold at 50% is used to evaluate the accuracy of object detection (Det.) on the testing data. We also evaluate correct localization accuracy (CorLoc.), which measures the percentage of training images of a class for which the most confident predicted box has at least 50% IoU with at least one ground-truth box.

Implementation details. For a fair comparison, all settings of the VGG16 model are kept identical to [45, 44] except those mentioned below. We use 8 GPUs during training with one input image per device. SGD is used for optimization. The default p and IoU in our proposed MIST technique (Alg. 1) are set to 0.15 and 0.2. For the Concrete DropBlock $\tau = 0.3, H = 3$. The ResNet models are identical to [15]. Please check the released code for other details.

5.1. Overall performance

VGG16-COCO. We compare to state-of-the-art WSOD methods on COCO in Tab. 1. Our single model without any post-processing outperforms all previous approaches (w/ bells and whistles) by a great margin. On the private Test-dev benchmark, we increase AP$_{50}$ by 11.2 (+82.3%). For the 2014 validation set, we increase AP and AP$_{50}$ by 0.6 (+5.6%) and 1.6 (+7.1%). Complete results are provided in Appendix A. Note that compared to supervised models shown in the first two rows, the performance gap is still relatively big: ours is 56.9% of Faster R-CNN on average. In addition, our model achieves 12.4 AP and 25.8 AP$_{50}$ on the COCO 2017 split as reported in Tab. 4, which is more commonly adopted in supervised papers.

ResNet-COCO. ResNet models have never been trained and evaluated before for WSOD. Nonetheless, they are the most popular backbone networks for supervised methods. Part of the reason is the larger memory consumption of ResNet. Without the training techniques introduced in Sec. 4.3, it's impossible to train on a standard GPU using all proposals. In Tab. 2 we provide the first benchmark for the COCO dataset using ResNet-50 and ResNet-101. As expected we observe ResNet models to perform better than the VGG16 model. Moreover, we note that the difference between ResNet-50 and ResNet-101 is relatively small.

VGG16-VOC. To fairly compare with most previous WSOD works, we also evaluate our approach on the VOC datasets [11]. The comparison to most recent works is reported in Tab. 3. All entries in this table are single model results. For object detection, our single-model results surpass all previous approaches on the publicly available 2007 test set (+1.3 AP$_{50}$) and on the private 2012 test set (+1.9 AP$_{50}$). In addition, our single model also performs better than all previous methods with bells and whistles (*e.g.*, '+FRCNN': supervised re-training, '+Ens.': model ensemble). Combining the 2007 and 2012 training set, our model achieves 58.1% (+2.1 AP$_{50}$) on the 2007 test set as reported in Tab. 4. CorLoc results on the training set and per-class results are provided in Appendix B. Since VOC is easier than COCO, the performance gap to supervised methods is smaller: ours is 78.1% of Faster R-CNN on average.

Methods	Proposal	07-AP$_{50}$	12-AP$_{50}$
Fast R-CNN	SS	66.9	65.7
Faster R-CNN	RPN	**69.9**	**67.0**
WSDDN [5]	EB	34.8	-
OICR [45]	SS	41.2	37.9
PCL [44]	SS	43.5	40.6
SDCN [28]	SS	50.2	43.5
Yang *et al.* [59]	SS	51.5	45.6
C-MIL [50]	SS	50.5	46.7
WSOD2 [60]	SS	**53.6**	47.2
Pred Net [2]	SS	52.9	48.4
C-MIDN [12]	SS	52.6	**50.2**
C-MIL [50]+FRCNN	SS	53.1	-
SDCN [28]+FRCNN	SS	53.7	46.7
Pred Net [2]+Ens.+FRCNN	SS	53.6	49.5
Yang *et al.* [59]+Ens.+FRCNN	SS	54.5	49.5
C-MIDN [12]+FRCNN	SS	53.6	50.3
Ours (single)	SS	**54.9**	**52.1***

Table 3: Single model (VGG16) detection results on VOC.

Data-Split	07-Trainval	12-Trainval	07-Test
Metrics	CorLoc	CorLoc	Det
Ours-07	68.8	-	54.9
Ours-12	-	70.9	56.3
WSOD2(07+12) [60]	71.4	72.2	56.0
Ours-(07+12)	**71.8**	**72.9**	**58.1**
Metrics	17-Val-AP	17-Val-AP$_{50}$	17-Val-AP$_{75}$
Ours-Train2014	11.4	24.3	9.4
Ours-Train2017	**12.4**	**25.8**	**10.5**

Table 4: Does more data help?

Additional training data. The biggest advantage of WSOD methods is the availability of more data. Therefore, we are interested in studying whether more training data improves results. We train our model on the VOC 2007 trainval (5011 images), 2012 trainval (11540 images), and the combination of both (16555 images) separately, and evaluate on the VOC 2007 test set. As shown in Tab. 4 (top), the performance increase consistently with the amount of training data. We verify this on COCO where 2014-train (82783 images) and 2017-train (128287 images) are used for training, and 2017-val (a.k.a. minival) for testing. Similar results are observed as shown in Tab. 4 (bottom).

5.2. Qualitative results

Qualitatively, we compare our full model with Tang *et al.* [45]. In Fig. 8 we show a set of two pictures side by side, with baselines on the left and our results on the right. Our model is able to address instance ambiguity by: (1) detecting previously ignored instances (Fig. 8 left); (2) predicting tight and precise boxes for multiple instances instead of a big one (Fig. 8 center). Part domination is also alleviated since our model focuses on the full extent of objects (Fig. 8 right). Even though our model can greatly increase the score

*http://host.robots.ox.ac.uk:8080/anonymous/DCJ5GA.html

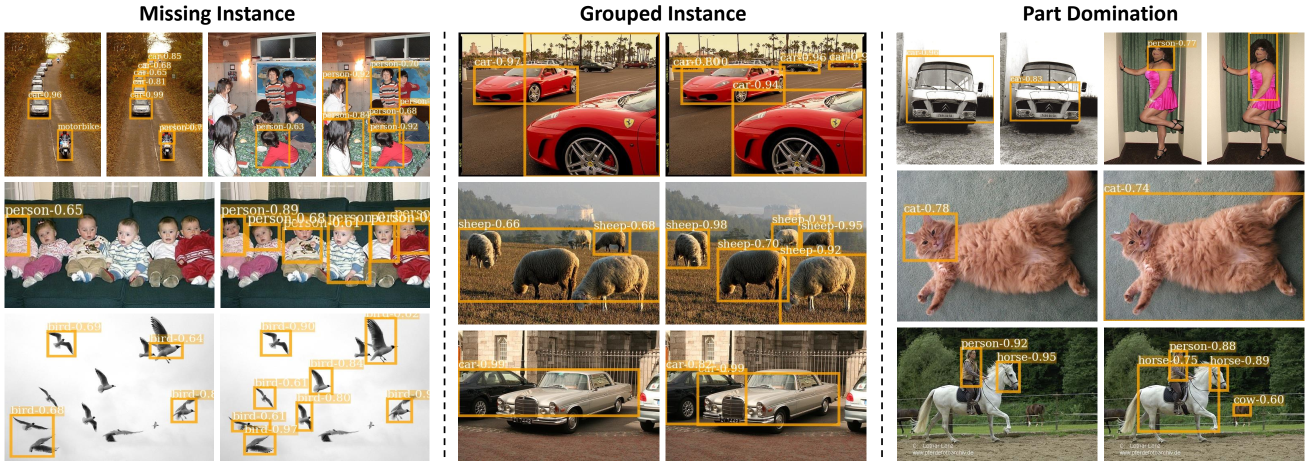


Figure 8: Comparison of our models (right picture in pair) to our baseline (left picture in pair).

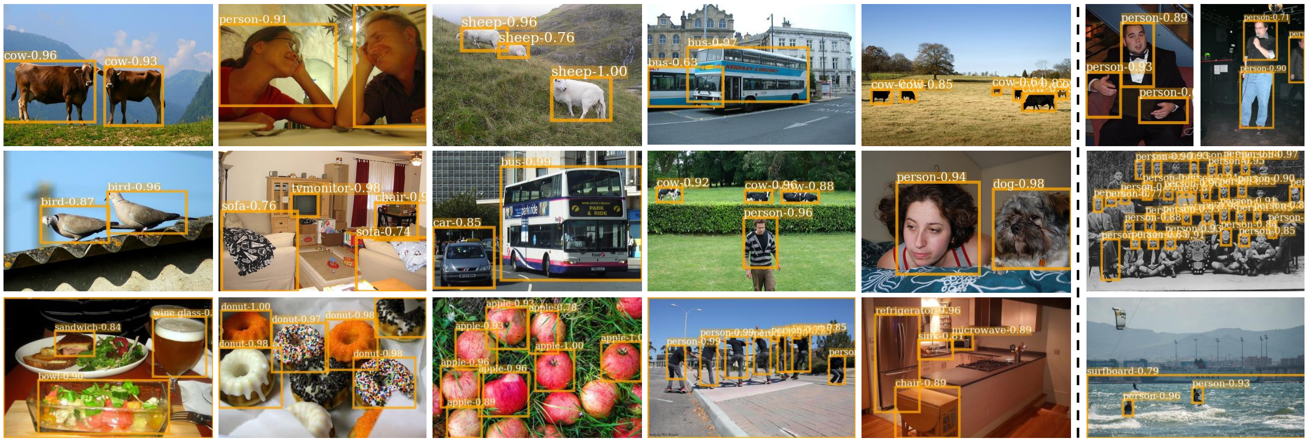


Figure 9: More visualization (top: VOC 2007, middle: VOC 2012, bottom: COCO) and some failure cases (right column).

of larger boxes (see the horse example), the predictions may still be dominated by parts in some difficult cases.

More qualitative results are shown in Fig. 9 for all three datasets we used, as well in Appendix C. Our model is able to detect multiple instances of the same category (cow, sheep, bird, apple, person) and various objects of different classes (food, furniture, animal) in relatively complicated scenes. The COCO dataset is much harder than VOC as the number of objects and classes is bigger. Our model still tells apart objects decently well (Fig. 9 bottom row). We also show some failure cases (Fig. 9 right column) of our model which can be roughly categorized into three types: (1) relevant parts are predicted as instances of objects (hands and legs, bike wheels); (2) in extreme examples, part domination remains (model converges to a face detector); (3) object co-occurrence confuses the detector when it predicts the sea as a surfboard or the baseball court as a bat.

5.3. Analysis

How much does each module help? We study the effectiveness of each module in Tab. 5. We first reproduce the method of Tang *et al*. [45], achieving similar results (first two rows). Applying the developed MIST module improves the results significantly. This aligns with our observation that instance ambiguity is the biggest bottleneck for WSOD. Our conceptually simple solution also outperforms an improved version [44] (PCL), which is based on a computationally expensive and carefully-tuned clustering.

The devised Concrete DropBlock further improves the performance when using MIST as the basis. This module surpasses several variants including: (1) (Img Spa.-Dropout): spatial dropout applied on the image-level features; (2) (ROI-Spa.-Dropout): spatial dropout applied on each ROI where each feature point is treated independently. This setting is similar to [39, 53]; (3) (DropBlock): the best-performing DropBlock setting reported in [14].

Has Instance Ambiguity been addressed? To validate that instance ambiguity is alleviated, we report Average Recall (AR) over multiple IoU values (.50 : .05 : .95), given 1, 10, 100 detections per image (AR^1, AR^{10}, AR^{100}) and for small, medium, annd large objects (AR^s, AR^m, AR^l) on VOC 2007. We compare the model with and without MIST in Tab. 6 where our method increases all recall metrics.

Has Part Domination been addressed? In Fig. 10, we show the 5 categories with the biggest relative performance

Data-Split	07 trainval	07 test	12 trainval	12 test
Metrics	CorLoc	Det.	CorLoc	Det.
Baseline [45]*	60.8	42.5	-	-
+ PCL [44]	62.7	43.5	63.2	40.6
+ MIST w/o Reg.	62.9	48.3	65.1	-
+ MIST	**64.9**	**51.4**	**66.7**	-
+ Img Spa.-Dropout	64.3	51.1	65.9	-
+ ROI Spa.-Dropout	66.8	52.4	67.3	-
+ DropBlock [14]	67.1	52.9	68.4	-
+ Concrete DropBlock	**68.8**	**54.9**	**70.9**	**52.1**

Table 5: Ablation study. (*: our implementation)

Metrics	AR^1	AR^{10}	AR^{100}	AR^s	AR^m	AR^l
w/o MIST	18.6	30.6	32.5	8.8	25.8	38.9
w/ MIST	**20.5**	**37.8**	**43.9**	**15.0**	**34.8**	**51.7**

Table 6: Average Recall (AR) (%) comparison.

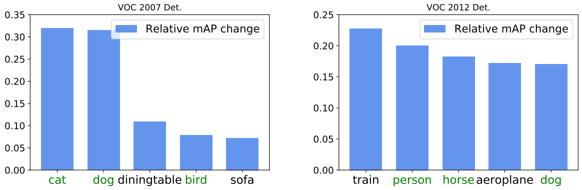


Figure 10: Top-5 classes with biggest performance boost when using Concrete DropBlock. Animal classes are emphasized using green color.

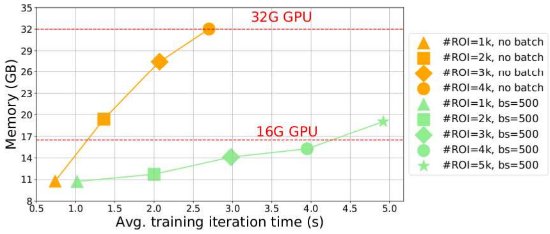


Figure 11: ResNet-101 model memory consumption using different methods and different number of proposals.

improvements on the VOC 2007 and VOC 2012 dataset after applying the Concrete DropBlock. The performance of animal classes including 'person' increases most, which matches our intuition mentioned in Sec. 1: the part domination issue is most prominent for articulated classes with rigid and discriminative parts. Across both datasets, three out of the five top classes are mammals.

Space-time analysis of sequential batch BP? We also study the effect of our sequential batch back-propagation. We fix the input image to be of size 600 × 600, and run two methods (vanilla back-propagation and ours with sub-batch size 500 using ResNet-101 for comparison. We change the number of proposals from 1k to 5k in 1k increments, and report average training iteration time and memory consumption in Fig. 11. We observe: (1) vanilla back-propagation cannot even afford 2k proposals (average number of ROIs widely used in [15, 5, 45]) on a standard 16GB GPU, but ours can easily handle up to 4k boxes; (2) the training process is not greatly slowed down, ours takes ∼1-2× more time than the vanilla version. In practice, input resolution and total number of proposals can be bigger.

Methods	Backbone	Det. (AP)	Backbone	Det. (AP)
Supervised	VGG16	61.7 [58]	R-101	80.5 [58]
[5]	VGG16	24.2	R-101	21.9
[45]	VGG16	34.8	R-101	40.5
Ours (MIST only)	VGG16	35.7	R-101	44.0
Ours	VGG16	**36.6**	R-101	**45.7**
Ours+flow	VGG16	**38.3**	R-101	**46.9**

Table 7: Video Object Detection Results.

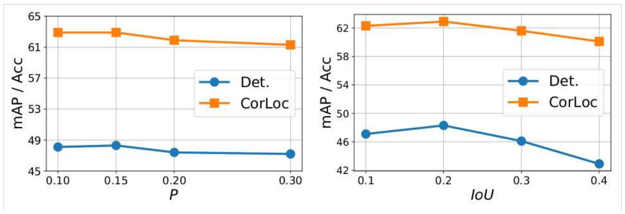


Figure 12: VOC 2007 results for different p and IoU.

Robustness of MIST? To assess robustness we test a baseline model plus this algorithm only using different top-percentage p and rejection IoU on the VOC 2007 dataset. Results are shown in Fig. 12. The best result is achieved with $p = 0.15$ and $IoU = 0.2$, which we use for all the other models and datasets. Importantly, we note that, overall, the sensitivity of the final results on the value of p is small and only slightly larger for IoU.

5.4. Extension: video object detection

We finally generalize our models to video-WSOD, which hasn't been explored in the literature. Following supervised methods, we experiment on the most popular dataset: ImageNet VID [8]. Frame-level category labels are available during training. Uniformly sampled key-frames are used for training following [63] and evaluation settings are also kept identical. Results are reported in Tab. 7. The performance improvement of the proposed MIST and Concrete DropBlock generalize to videos. The memory-efficient sequential batch back-propagation permits to leverage short-term motion patterns (*i.e.*, we use optical-flow following [63]) to further increase the performance. This suggests that videos are a useful domain where we can obtain more data to improve WSOD. Full details are provided in Appendix F.

6. Conclusion

In this paper, we address three major issues of WSOD. For each we have proposed a solution and demonstrated its effectiveness through extensive experiments. We achieve state-of-the-art results on popular datasets (COCO, VOC 07 and 12) and are the first to benchmark ResNet backbones and weakly supervised video object detection.

Acknowledgement: ZR is supported by Yunni & Maxine Pao Memorial Fellowship. This work is supported in part by NSF under Grant No. 1718221 and No. 1751206.

References

[1] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proc. CVPR*, 2014. 12

[2] Aditya Arun, C. V. Jawahar, and M. Pawan Kumar. Dissimilarity coefficient based weakly supervised object detection. In *Proc. CVPR*, 2019. 2, 6, 11, 12, 13

[3] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with posterior regularization. In *Proc. BMVC*, 2014. 2

[4] Hakan Bilen, Marco Pedersoli, and Tinne Tuytelaars. Weakly supervised object detection with convex clustering. In *Proc. CVPR*, 2015. 2, 12, 13

[5] H. Bilen and A. Vedaldi. Weakly supervised deep detection networks. In *Proc. CVPR*, 2016. 1, 2, 5, 6, 8, 11, 12, 13

[6] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *CoRR*, abs/1604.06174, 2016. 2, 5

[7] Ramazan Gokberk Cinbis, Jakob J. Verbeek, and Cordelia Schmid. Multi-fold MIL training for weakly supervised object localization. In *Proc. CVPR*, 2014. 2, 12, 13

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. CVPR*, 2009. 2, 3, 8, 13

[9] Ali Diba, Vivek Sharma, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool. Weakly supervised cascaded convolutional networks. In *Proc. CVPR*, 2017. 5, 12, 13

[10] Ali Diba, Vivek Sharma, Rainer Stiefelhagen, and Luc Van Gool. Object discovery by generative adversarial & ranking networks. 2017. 5

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. In *Proc. IJCV*, 2010. 2, 5, 6

[12] Yan Gao, Boxiao Liu, Nan Guo, Xiaochun Ye, Fang Wan, Haihang You, and Dongrui Fan. C-midn: Coupled multiple instance detection network with segmentation guidance for weakly supervised object detection. In *Proc. ICCV*, 2019. 2, 3, 5, 6, 12

[13] Weifeng Ge, Sibei Yang, and Yizhou Yu. Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning. In *Proc. CVPR*, 2018. 2, 5

[14] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. Dropblock: A regularization method for convolutional networks. In *Proc. NIPS*, 2018. 2, 4, 7, 8

[15] Ross B. Girshick. Fast R-CNN. In *Proc. ICCV*, 2015. 2, 4, 6, 8, 11

[16] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. CVPR*, 2014. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proc. ICCV*, 2017. 1, 2, 11

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 2

[19] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proc. CVPR*, 2017. 13

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proc. ICLR*, 2017. 4

[21] Zequn Jie, Yunchao Wei, Xiaojie Jin, Jiashi Feng, and Wei Liu. Deep self-taught learning for weakly supervised object localization. In *Proc. CVPR*, 2017. 2, 12, 13

[22] Vadim Kantorov, Maxime Oquab, Minsu Cho, and Ivan Laptev. Contextlocnet: Context-aware deep network models for weakly supervised localization. In *Proc. ECCV*, 2016. 1, 2, 12, 13

[23] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Proc. NIPS*. 2015. 2

[24] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proc. CVPR*, 2017. 2, 5

[25] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *Proc. ICLR*, 2017. 4

[26] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proc. ECCV*, 2018. 1, 2

[27] Dong Li, Jia-Bin Huang, Yali Li, Shengjin Wang, and Ming-Hsuan Yang. Weakly supervised object localization with progressive domain adaptation. In *Proc. CVPR*, 2016. 2, 12, 13

[28] Xiaoyan Li, Meina Kan, Shiguang Shan, and Xilin Chen. Weakly supervised object detection with segmentation collaboration. In *Proc. ICCV*, 2019. 6, 12, 13

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollr, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. ECCV*, 2014. 2, 5

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 2

[31] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proc. ICLR*, 2017. 4

[32] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. In *Proc. CVPR*, 2018. 2

[33] Geoff Pleiss, Danlu Chen, Gao Huang, Tongcheng Li, Laurens van der Maaten, and Kilian Q. Weinberger. Memory-efficient implementation of densenets. *CoRR*, abs/1707.06990, 2017. 2

[34] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. CVPR*, 2016. 1, 2

[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *TPAMI*, 2016. 1, 2, 11

[36] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Parallel distributed processing: Explorations in the microstructure of cognition. chapter Learning Internal Representations by Error Propagation. MIT Press, 1986. 2, 5

[37] Yunhang Shen, Rongrong Ji, Yan Wang, Yongjian Wu, and Liujuan Cao. Cyclic guidance for weakly supervised joint detection and segmentation. In *Proc. CVPR*, 2019. 2, 12, 13

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015. 2

[39] Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *Proc. ICCV*, 2017. 2, 7

[40] Krishna Kumar Singh and Yong Jae Lee. You reap what you sow: Using videos to generate high precision object proposals for weakly-supervised object detection. In *Proc. CVPR*, 2019. 1, 2

[41] Krishna Kumar Singh, Fanyi Xiao, and Yong Jae Lee. Track and transfer: Watching videos to simulate strong human supervision for weakly-supervised object detection. In *Proc. CVPR*, 2016. 2, 3

[42] P. Siva and T. Xiang. Weakly supervised object detector learning with model drift detection. In *Proc. ICCV*, 2013. 2

[43] Hyun Oh Song, Yong Jae Lee, Stefanie Jegelka, and Trevor Darrell. Weakly-supervised discovery of visual pattern configurations. In *Proc. NIPS*, 2014. 2

[44] Peng Tang, Xinggang Wang, Song Bai, Wei Shen, Xiang Bai, Wenyu Liu, and Alan Yuille. PCL: Proposal cluster learning for weakly supervised object detection. *TPAMI*, 2018. 2, 3, 4, 5, 6, 7, 8, 12, 13

[45] Peng Tang, Xinggang Wang, Xiang Bai, and Wenyu Liu. Multiple instance detection network with online instance classifier refinement. In *Proc. CVPR*, 2017. 1, 2, 4, 5, 6, 7, 8, 11, 12, 13

[46] Peng Tang, Xinggang Wang, Angtian Wang, Yongluan Yan, Wenyu Liu, Junzhou Huang, and Alan L. Yuille. Weakly supervised region proposal network and object detection. In *Proc. ECCV*, 2018. 2, 3, 12, 13

[47] Eu Wern Teh, Mrigank Rochan, and Yang Wang. Attention networks for weakly supervised object localization. In *Proc. BMVC*, 2016. 12, 13

[48] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proc. CVPR*, 2015. 2

[49] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M.Smeulders. Selective search for object recognition. *IJCV*, 2013. 1, 3, 12

[50] Fang Wan, Chang Liu, Wei Ke, Xiangyang Ji, Jianbin Jiao, and Qixiang Ye. C-MIL: continuation multiple instance learning for weakly supervised object detection. In *Proc. CVPR*, 2019. 1, 2, 3, 6, 11, 12, 13

[51] Fang Wan, Pengxu Wei, Jianbin Jiao, Zhenjun Han, and Qixiang Ye. Min-entropy latent model for weakly supervised object detection. In *Proc. CVPR*, 2018. 12, 13

[52] Chong Wang, Weiqiang Ren, Kaiqi Huang, and Tieniu Tan. Weakly supervised object localization with latent category learning. In *Proc. ECCV*, 2014. 12, 13

[53] Xiaolong Wang, Abhinav Shrivastava, and Abhinav Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *Proc. CVPR*, 2017. 2, 7

[54] Xinggang Wang, Zhuotun Zhu, Cong Yao, and Xiang Bai. Relaxed multiple-instance SVM with application to object discovery. In *Proc. ICCV*, 2015. 2

[55] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proc. CVPR*, 2017. 2

[56] Yunchao Wei, Zhiqiang Shen, Bowen Cheng, Honghui Shi, Jinjun Xiong, Jiashi Feng, and Thomas S. Huang. TS2C: tight box mining with surrounding segmentation context for weakly supervised object detection. In *Proc. ECCV*, 2018. 12, 13

[57] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992. 2

[58] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *Proc. ECCV*, 2018. 8, 13

[59] Ke Yang, Dongsheng Li, and Yong Dou. Towards precise end-to-end weakly supervised object detection network. In *Proc. ICCV*, 2019. 6, 12, 13

[60] Zhaoyang Zeng, Bei Liu, Jianlong Fu, Hongyang Chao, and Lei Zhang. WSOD2: Learning bottom-up and top-down objectness distillation for weakly-supervised object detection. In *Proc. ICCV*, 2019. 1, 2, 3, 5, 6, 11, 12, 13

[61] Xiaopeng Zhang, Jiashi Feng, Hongkai Xiong, and Qi Tian. Zigzag learning for weakly supervised object detection. In *Proc. CVPR*, 2018. 2

[62] Yongqiang Zhang, Yancheng Bai, Mingli Ding, Yongqiang Li, and Bernard Ghanem. W2f: A weakly-supervised to fully-supervised framework for object detection. In *Proc. CVPR*, 2018. 2, 3, 11

[63] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *Proc. ICCV*, 2017. 8, 13

[64] C. Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Proc. ECCV*, 2014. 1, 3

[65] Yang Zou, Zhiding Yu, Xiaofeng Liu, B.V.K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proc. ICCV*, 2019. 2

[66] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proc. ECCV*, 2018. 2