Evaluating Weakly Supervised Object Localization Methods Right

Junsuk Choe[1,3*] Seong Joon Oh[2*] Seungho Lee[1]
Sanghyuk Chun[3] Zeynep Akata[4] Hyunjung Shim[1†]

[1]School of Integrated Technology, Yonsei University
[2]Clova AI Research, LINE Plus Corp.
[3]Clova AI Research, NAVER Corp.
[4]University of Tuebingen

Abstract

Weakly-supervised object localization (WSOL) has gained popularity over the last years for its promise to train localization models with only image-level labels. Since the seminal WSOL work of class activation mapping (CAM), the field has focused on how to expand the attention regions to cover objects more broadly and localize them better. However, these strategies rely on full localization supervision to validate hyperparameters and for model selection, which is in principle prohibited under the WSOL setup. In this paper, we argue that WSOL task is ill-posed with only image-level labels, and propose a new evaluation protocol where full supervision is limited to only a small held-out set not overlapping with the test set. We observe that, under our protocol, the five most recent WSOL methods have not made a major improvement over the CAM baseline. Moreover, we report that existing WSOL methods have not reached the few-shot learning baseline, where the full-supervision at validation time is used for model training instead. Based on our findings, we discuss some future directions for WSOL. Source code and dataset are available at https://github.com/clovaai/wsolevaluation.

1. Introduction

As human labeling for every object is too costly and weakly-supervised object localization (WSOL) requires *only* image-level labels, the WSOL research has gained significant momentum [60, 58, 59, 6, 26, 56] recently.

Among these, class activation mapping (CAM) [60] uses the intermediate classifier activations focusing on the most discriminative parts of the objects to localize the objects of the target class. As the aim in object localization is to cover the full extent of the object, focusing only on the most discriminative parts of the objects is a limitation. WSOL techniques since CAM have focused on this limitation and have proposed different architectural [58, 59, 6] and data-augmentation [26, 56] solutions. The reported state-of-the-art WSOL performances have made a significant improvement over the CAM baseline, from 49.4% to 62.3% [6] and 43.6% to 48.7% [6] top-1 localization performances on Caltech-UCSD Birds-200-2011 [53] and ImageNet [42], respectively. However, these techniques have introduced a set of hyperparameters for suppressing the discriminative cues of CAM and different ways for selecting these hyperparameters. One of such hyperparameters is the operating threshold τ for generating object bounding boxes from the score maps. Among others, the mixed policies for selecting τ has contributed to the illusory improvement of WSOL performances over the years; see Figure 1.

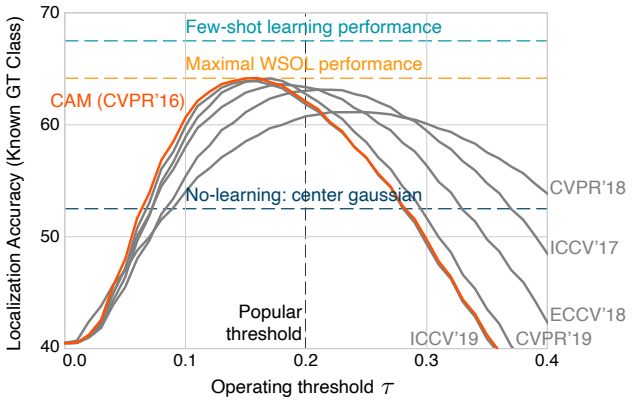


Figure 1. **WSOL 2016-2019.** Recent improvements in WSOL are illusory due to (1) different amount of implicit full supervision through validation and (2) a fixed score-map threshold (usually $\tau = 0.2$) to generate object boxes. Under our evaluation protocol with the same validation set sizes and oracle τ for each method, CAM is still the best. In fact, our few-shot learning baseline, *i.e.* using the validation supervision (10 samples/class) at training time, outperforms existing WSOL methods. Results on ImageNet.

Due to the lack of a unified definition of the WSOL task, we revisit the problem formulation of WSOL and show that WSOL problem is ill-posed in general without any localiza-

*Equal contribution. Work done at Clova AI Research.
†Hyunjung Shim is a corresponding author.

tion supervision. Towards a well-posed setup, we propose a new WSOL setting where a small held-out set with full supervision is available to the learners.

Our contributions are as follows. (1) Propose new experimental protocol that uses a fixed amount of full supervision for hyperparameter search and carefully analyze six WSOL methods on three architectures and three datasets. (2) Propose new evaluation metrics as well as data, annotations, and benchmarks for the WSOL task at https://github.com/clovaai/wsolevaluation. (3) Show that WSOL has not progressed significantly since CAM, when the calibration dependency and the different amounts of full supervision are factored out. Moreover, searching hyperparameters on a held-out set consisting of 5 to 10 full localization supervision per class often leads to significantly lower performance compared to the few-shot learning (FSL) baselines that use the full supervision directly for model training. Finally, we suggest a shift of focus in future WSOL research: consideration of learning paradigms utilizing both weak and full supervisions, and other options for resolving the ill-posedness of WSOL (*e.g.* background-class images).

2. Related Work

By model output. Given an input image, *semantic segmentation* models generate pixel-wise class predictions [11, 32], *object detection* models [11, 13] output a set of bounding boxes with class predictions, and *instance segmentation* models [28, 7, 18] predict a set of disjoint masks with class *and* instance labels. *Object localization* [42], on the other hand, assumes that the image contains an object of single class and produces a binary mask or a bounding box around that object coming from the class of interest.

By type of supervision. Since bounding box and mask labels cost significantly more than image-level labels, *e.g.* categories [2], researchers have considered different types of localization supervision: image-level labels [36], gaze [35], points [2], scribbles [27], boxes [8], or a mixture of multiple types [21]. Our work is concerned with the object localization task with only image-level category labels [34, 60].

By amount of supervision. Learning from a small amount of labeled samples per class is referred to as few-shot learning (FSL) [54]. We recognize the relationship between our new WSOL setup and the FSL paradigm; we consider FSL methods as baselines for future WSOL methods.

WSOL works. Class activation mapping (CAM) [60] turns a fully-convolutional classifier into a score map predictor by considering the activations before the global average pooling layer. Vanilla CAM has been criticized for its focus on the small discriminative part of the object. Researchers have considered dropping regions in inputs at random [26, 56] to diversify the cues used for recognition. Adversarial erasing techniques [58, 6] drop the most discriminative part at the current iteration. Self-produced guidance (SPG) [59] is trained with auxiliary foreground-background masks generated by its own activations. Other than object classification in static images, there exists work on localizing informative video frames for action recognition [38, 29, 55], but they are beyond the scope of our analysis.

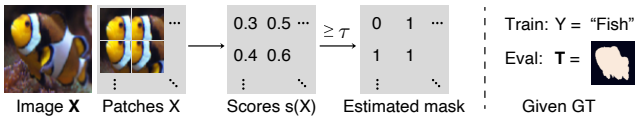


Figure 2. **WSOL as MIL.** WSOL is interpreted as a patch classification task trained with multiple-instance learning (MIL). The score map $s(\mathbf{X})$ is thresholded at τ to estimate the mask $\mathbf{T}$.

Relation to explainability. WSOL methods share similarities with the model explainability [17], specifically the *input attribution* task: analyzing which pixels have led to the image classification results [16]. There are largely two streams of work on visual input attribution: variants of input gradients [51, 46, 43, 49, 44, 50, 25, 37] and counterfactual reasoning [40, 12, 61, 41, 15, 20]. While they can be viewed as WSOL methods, we have not included them in our studies because they are seldom evaluated in WSOL benchmarks. Analyzing their possibility as WSOL methods is an interesting future study.

Our scope. We study the WSOL task, rather than weakly-supervised detection, segmentation, or instance segmentation. The terminologies tend to be mixed in the earlier works of weakly-supervised learning [45, 14, 9, 48]. Extending our analysis to other weakly-supervised learning tasks is valid and will be a good contribution to the respective communities.

3. Problem Formulation of WSOL

We define and formulate the weakly-supervised object localization (WSOL) task as an image patch classification and show the ill-posedness of the problem. We will discuss possible modifications to resolve the ill-posedness in theory.

3.1. WSOL Task as Multiple Instance Learning

Given an image $\mathbf{X} \in \mathbb{R}^{H \times W}$, **object localization** is the task to identify whether or not the pixel belongs to the object of interest, represented via dense binary mask $\mathbf{T} = (T_{11}, \cdots, T_{HW})$ where $T_{ij} \in \{0, 1\}$ and (i, j) indicate the pixel indices. When the training set consists of precise image-mask pairs $(\mathbf{X}, \mathbf{T})$, we refer to the task as **fully-supervised object localization (FSOL)**. In this paper, we consider the case when only an image-level label $Y \in \{0, 1\}$ for global presence of the object of interest is provided per training image $\mathbf{X}$. This task is referred to as the **weakly-supervised object localization (WSOL)**.

One can treat an input image $\mathbf{X}$ as a bag of stride-1 sliding window patches of suitable side lengths, h and w:

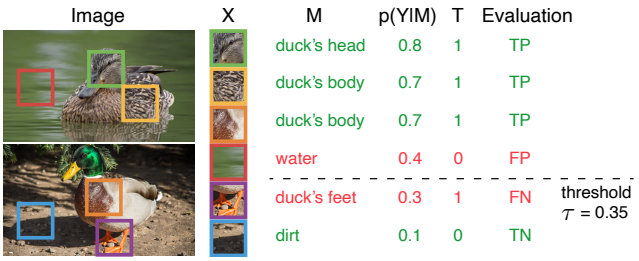


Figure 3. **Ill-posed WSOL: An example.** Even the true posterior $s(M) = p(Y|M)$ may not lead to the correct prediction of T if background cues are more associated with the class than the foreground cues (*e.g.* $p(\text{duck}|\text{water}) > p(\text{duck}|\text{feet})$).

$(X_{11}, \cdots, X_{HW})$ with $X_{ij} \in \mathbb{R}^{h \times w}$. The object localization task is then the problem of predicting the object presence T_{ij} at the image patch X_{ij}. The weak supervision imposes the requirement that each training image $\mathbf{X}$, represented as $(X_{11}, \cdots, X_{HW})$, is only collectively labeled with a single label $Y \in \{0, 1\}$ indicating whether at least one of the patches represents the object. This formulation is an example of the multiple-instance learning (MIL) [22], as observed by many traditional WSOL works [36, 45, 14, 48].

Following the patch classification point of view, we formulate WSOL task as a mapping from patches X to the binary labels T (indices dropped). We assume that the patches X, image-level labels Y, and the pixel-wise labeling T in our data arise in an i.i.d. fashion from the joint distribution $p(X, Y, T)$. See Figure 2 for an overview. The aim of WSOL is to produce a scoring function $s(X)$ such that thresholding it at τ closely approximates binary label T. Many existing approaches for WSOL, including CAM [60], use the scoring rules based on the posterior $s(X) = p(Y|X)$. See Appendix §A.1 for the interpretation of CAM as pixel-wise posterior approximation.

3.2. When is WSOL ill-posed?

We show that if background cues are more strongly associated with the target labels T than some foreground cues, the localization task cannot be solved, even when we know the exact posterior $p(Y|X)$ for the image-level label Y. We will make some strong assumptions in favor of the learner, and then show that WSOL still cannot be perfectly solved.

We assume that there exists a finite set of **cue** labels $\mathcal{M}$ containing all patch-level concepts in natural images. For example, patches from a duck image are one of {duck's head, duck's feet, sky, water, $\cdots$} (see Figure 3). We further assume that every patch X is equivalently represented by its cue label $M(X) \in \mathcal{M}$. Therefore, from now on, we write M instead of X in equations and examine the association arising in the joint distribution $p(M, Y, T)$. We write $M^{\text{fg}}, M^{\text{bg}} \in \mathcal{M}$ for foreground and background cues.

We argue that, even with access to the joint distribution $p(Y, M)$, it may not be possible to make perfect predictions for the patch-wise labels $T(M)$ (proof in Appendix §A.2).

Lemma 3.1. *Assume that the true posterior $p(Y|M)$ with a continuous pdf is used as the scoring rule $s(M) = p(Y|M)$. Then, there exists a scalar $\tau \in \mathbb{R}$ such that $s(M) \geq \tau$ is identical to T if and only if the foreground-background posterior ratio $\frac{p(Y=1|M^{fg})}{p(Y=1|M^{bf})} \geq 1$ almost surely, conditionally on the event $\{T(M^{fg}) = 1$ and $T(M^{bf}) = 0\}$.*

In other words, if the posterior likelihood for the image-level label Y given a foreground cue M^{fg} is less than the posterior likelihood given background M^{bg} for some foreground and background cues, no WSOL method can make a correct prediction. This pathological scenario is described in Figure 3: Duck's feet are less seen in duck images than the water background. Such cases are abundant in user-collected data (Appendix Figure 9).

This observation implies a data-centric solution towards well-posed WSOL: we can augment (1) positive samples ($Y = 1$) with more less-represented foreground cues (*e.g.* duck images with feet) and (2) negative samples ($Y = 0$) with more target-correlated background cues (*e.g.* non-duck images with water background). Such data-centric approaches are promising future directions for WSOL.

How have WSOL methods addressed the ill-posedness? Previous solutions to the WSOL problem have sought architectural modifications [58, 59, 6] and data augmentation [26, 56] schemes that typically require heavy hyperparameter search and model selection, which are a form of implicit localization supervision. For example, [26] has found the operating threshold τ via "observing a few qualitative results", while others have evaluated their models over the test set to select reasonable hyperparameter values (Table 1 of [26], Table 6 of [58], and Table 1 of [6]). [59] has performed a "grid search" over possible values. We argue that certain level of localization labels are inevitable for WSOL. In the next section, we propose to allow a fixed number of fully labeled samples for hyperparameter search and model selection for a more realistic evaluation.

4. Evaluation Protocol for WSOL

We reformulate the WSOL evaluation based on our observation of the ill-posedness. We define performance metrics, benchmarks, and the hyperparameter search procedure.

4.1. Evaluation metrics

The aim of WSOL is to produce score maps, where their pixel value s_{ij} is higher on foreground $T_{ij} = 1$ and lower on background $T_{ij} = 0$ (§3.1). We discuss how to quantify the above conditions and how prior evaluation metrics have failed to clearly measure the relevant performance. We then propose the `MaxBoxAcc` and `PxAP` metrics for bounding box and mask ground truths, respectively.

The *localization accuracy* [42] metric entangles classification and localization performances by counting the num-

ber of images where both tasks are performed correctly. We advocate the measurement of localization performance alone, as the goal of WSOL is to localize objects (§3.1) and not to classify images correctly. To this end, we only consider the score maps s_{ij} corresponding to the ground-truth classes in our analysis. Metrics based on such are commonly referred to as the *GT-known* metrics [26, 58, 59, 6].

A common practice in WSOL is to normalize the score maps per image because the score statistics differ vastly across images. Either max normalization (divide through by $\max_{ij} s_{ij}$) or min-max normalization (additionally map $\min_{ij} s_{ij}$ to zero) has been used; see Appendix §B.1 for the full summary. We always use the min-max normalization.

After normalization, WSOL methods threshold the score map at τ to generate a tight box around the binary mask $\{(i,j) \mid s_{ij} \geq \tau\}$. WSOL metrics then measure the quality of the boxes. τ is typically treated as a fixed value [60, 58, 56] or a hyperparameter to be tuned [26, 59, 6]. We argue that the former is misleading because the ideal threshold τ depends heavily on the data and model architecture and fixing its value may be disadvantageous for certain methods. To fix the issue, we propose new evaluation metrics that are independent of the threshold τ.

Masks: `PxAP`. When masks are available for evaluation, we measure the pixel-wise precision and recall [1]. Unlike single-number measures like mask-wise IoU, those metrics allow users to choose the preferred operating threshold τ that provides the best precision-recall trade-off for their downstream applications. We define the **pixel precision and recall at threshold** τ as:

$$\texttt{PxPrec}(\tau) = \frac{|\{s_{ij}^{(n)} \geq \tau\} \cap \{T_{ij}^{(n)} = 1\}|}{|\{s_{ij}^{(n)} \geq \tau\}|} \quad (1)$$

$$\texttt{PxRec}(\tau) = \frac{|\{s_{ij}^{(n)} \geq \tau\} \cap \{T_{ij}^{(n)} = 1\}|}{|\{T_{ij}^{(n)} = 1\}|} \quad (2)$$

For threshold independence, we define and use the **pixel average precision**, $\texttt{PxAP} := \sum_l \texttt{PxPrec}(\tau_l)(\texttt{PxRec}(\tau_l) - \texttt{PxRec}(\tau_{l-1}))$, the area under curve of the pixel precision-recall curve. We use `PxAP` as the final metric in this paper.

Bounding boxes: `MaxBoxAcc`. Pixel-wise masks are expensive to collect; many datasets only provide box annotations. Since it is not possible to measure exact pixel-wise precision and recall with bounding boxes, we suggest a surrogate in this case. Given the ground truth box B, we define the **box accuracy at score map threshold** τ **and IoU threshold** δ, $\texttt{BoxAcc}\ (\tau, \delta)$ [60, 42], as:

$$\texttt{BoxAcc}(\tau, \delta) = \frac{1}{N} \sum_n 1_{\text{IoU}\left(\text{box}(s(\mathbf{X}^{(n)}), \tau), B^{(n)}\right) \geq \delta} \quad (3)$$

where $\text{box}(s(\mathbf{X}^{(n)}), \tau)$ is the tightest box around the largest-area connected component of the mask $\{(i,j) \mid s(X_{ij}^{(n)}) \geq \tau\}$. In datasets where more than one bounding box are provided (*e.g.* ImageNet), we count the number of images where the box prediction overlaps with *at least one* of the ground truth boxes with IoU $\geq \delta$. When δ is 0.5, the metric is identical to the commonly-called *GT-known localization accuracy* [26] or *CorLoc* [10], but we suggest a new naming to more precisely represent what is being measured. For score map threshold independence, we report the box accuracy at the optimal threshold τ, the **maximal box accuracy** $\texttt{MaxBoxAcc}(\delta) := \max_\tau \texttt{BoxAcc}(\tau, \delta)$, as the final performance metric. In this paper, we set δ to 0.5, following the prior works [60, 26, 58, 59, 6, 56].

Statistics	ImageNet	CUB	OpenImages
#Classes	1000	200	100
#images/class			
`train-weaksup`	∼1.2K	∼30	∼300
`train-fullsup`	10	∼5	25
`test`	10	∼29	50

Table 1. **Dataset statistics.** "∼" indicates that the number of images per class varies across classes and the average value is shown.

Better box evaluation: `MaxBoxAccV2`. After the acceptance at CVPR 2020, we have developed an improved version of `MaxBoxAcc`. It is better in two aspects. (1) `MaxBoxAcc` measures the performance at a fixed IoU threshold ($\delta = 0.5$), only considering a specific level of fineness of localization outputs. We suggest averaging the performance across $\delta \in \{0.3, 0.5, 0.7\}$ to address diverse demands for localization fineness. (2) `MaxBoxAcc` takes the *largest* connected component for estimating the box, assuming that the object of interest is usually large. We remove this assumption by considering the best match between the set of all estimated boxes and the set of all ground truth boxes. We call this new metric as `MaxBoxAccV2`. For future WSOL researches, we encourage using the `MaxBoxAccV2` metric. The code is already available in our repository. We show the evaluation results under the new metric in the in Appendix §C.7.

4.2. Data splits and hyperparameter search

For a fair comparison of the WSOL methods, we fix the amount of full supervision for hyperparameter search. As shown in Table 1 we propose three disjoint splits for every dataset: `train-weaksup`, `train-fullsup`, and `test`. The `train-weaksup` contains images with weak supervision (the image-level labels). The `train-fullsup` contains images with full supervision (either bounding box or binary mask). It is left as freedom for the user to utilize it for hyperparameter search, model selection, ablative studies, or even model fitting. The `test` split contains images with full supervision; it must be used only for the final performance report. For example, checking the `test` results multiple times with different model configurations violates the protocol as the learner implicitly uses more full supervision than allowed.

As WSOL benchmark datasets, ImageNet [42] and

Caltech-UCSD Birds-200-2011 (CUB) [53] have been extensively used. For ImageNet, the 1.2M "train" and 10K "validation" images for 1 000 classes are treated as our `train-weaksup` and `test`, respectively. For `train-fullsup`, we use the ImageNetV2 [39]. We have annotated bounding boxes on those images. CUB has 5 994 "train" and 5 794 "test" images for 200 classes. We treat them as our `train-weaksup` and `test`, respectively. For `train-fullsup`, we have collected 1 000 extra images (~ 5 images per class) from Flickr, on which we have annotated bounding boxes. For ImageNet and CUB we use the oracle box accuracy `BoxAcc`.

We contribute a new WSOL benchmark based on the OpenImages instance segmentation subset [3]. It provides a fresh WSOL benchmark to which the models have not yet overfitted. To balance the original OpenImages dataset, we have sub-sampled 100 classes and have randomly selected 29 819, 2 500, and 5 000 images from the original "train", "validation", and "test" splits as our `train-weaksup`, `train-fullsup`, and `test` splits, respectively. We use the pixel average precision `PxAP`. A summary of dataset statistics is in Table 1. Details on data collection and preparation are in Appendix §B.2.

Hyperparameter search. To make sure that the same amount of localization supervision is provided for each WSOL method, we refrain from employing any source of human prior outside the `train-fullsup` split. If the optimal hyperparameter for an arbitrary dataset and architecture is not available by default, we subject it to the hyperparameter search algorithm. For each hyperparameter, its *feasible range*, as opposed to *sensible range*, is used as the search space, to minimize the impact of human bias.

We employ the random search hyperparameter optimization [4]; it is simple, effective, and parallelizable. For each WSOL method, we sample 30 hyperparameters to train models on `train-weaksup` and validate on `train-fullsup`. The best hyperparameter combination is then selected. Since running 30 training sessions is costly for ImageNet (1.2M training images), we use 10% of images in each class for fitting models during the search. We verify in Appendix §B.4 that the ranking of hyperparameters is preserved even if the training set is sub-sampled.

5. Experiments

5.1. Evaluated Methods

We evaluate six widely used WSOL methods published in peer-reviewed venues. We describe each method in chronological order and discuss the set of hyperparameters. The full list of hyperparameters is in Appendix §C.1.

Class activation mapping (CAM) [60] trains a classifier of fully-convolutional backbone with the global average pooling (GAP) structure. At test time, CAM uses the logit outputs before GAP as the score map s_{ij}. CAM has the learning rate and the score-map resolution as hyperparameters and all five methods below use CAM in the background.

Hide-and-seek (HaS) [26] is a data augmentation technique that randomly selects grid patches to be dropped. The hyperparameters are the drop rate and grid size.

Adversarial complementary learning (ACoL) [58] proposes an architectural solution: a two-head architecture where one adversarially erases the high-scoring activations in the other. The erasing threshold is a hyperparameter.

Self-produced guidance (SPG) [59] is another architectural solution where internal pseudo-pixel-wise supervision is synthesized on the fly. Three tertiary pixel-wise masks (foreground, unsure, background) are generated from three different layers using two thresholding hyperparameters for each mask and are used as auxiliary supervisions.

Attention-based dropout layer (ADL) [6] has proposed a module that, like ACoL, adversarially produces drop masks at high-scoring regions, while not requiring an additional head. Drop rate and threshold are the hyperparameters.

CutMix [56] is a data augmentation technique, where patches in training images are cut and pasted to other images during training. The target labels are also mixed. The hyperparameters are the size prior α and the mix rate r.

Few-shot learning (FSL) baseline. The full supervision in `train-fullsup` used for validating WSOL hyperparameters can be used for training a model itself. Since only a few fully labeled samples per class are available, we refer to this setting as the few-shot learning (FSL) baseline.

As a simple baseline, we consider a foreground saliency mask predictor [30]. We alter the last layer of a fully convolutional network (FCN) into a 1×1 convolutional layer with $H \times W$ score map output. Each pixel is trained with the binary cross-entropy loss against the target mask, as done in [5, 32, 33]. For OpenImages, the pixel-wise masks are used as targets; for ImageNet and CUB, we build the mask targets by labeling pixels inside the ground truth boxes as foreground [23]. At inference phase, the $H \times W$ score maps are evaluated with the box or mask metrics.

Center-gaussian baseline. The Center-gaussian baseline generates isotropic Gaussian score maps centered at the images. We set the standard deviation to 1, but note that it does not affect the `MaxBoxAcc` and `PxAP` measures. This provides a no-learning baseline for every localization method.

5.2. Comparison of WSOL methods

We evaluate the six WSOL methods over three backbone architectures, *i.e.* VGG-GAP [47, 60], InceptionV3 [52], and ResNet50 [19], and three datasets, *i.e.* CUB, ImageNet and OpenImages. For each (method, backbone, dataset) tuple, we have randomly searched the optimal hyperparameters over the `train-fullsup` with 30 trials, totalling about 9 000 GPU hours. Since the sessions are paralleliz-

Methods	ImageNet (MaxBoxAcc) VGG	Inception	ResNet	Mean	CUB (MaxBoxAcc) VGG	Inception	ResNet	Mean	OpenImages (PxAP) VGG	Inception	ResNet	Mean	Total Mean
CAM [60]	61.1	65.3	64.2	63.5	71.1	62.1	73.2	68.8	58.1	61.4	58.0	59.1	63.8
HaS [26]	+0.7	+0.1	-1.0	-0.1	+5.2	-4.4	+4.9	+1.9	-1.2	-2.9	+0.2	-1.3	+0.2
ACoL [58]	-0.8	-0.7	-2.5	-1.4	+1.2	-2.5	-0.5	-0.6	-3.4	+1.6	-0.2	-0.7	-0.9
SPG [59]	+0.5	+0.1	-0.7	+0.0	-7.4	+0.7	-1.8	-2.8	-2.2	+1.0	-0.3	-0.5	-1.1
ADL [6]	-0.3	-3.8	+0.0	-1.4	+4.6	+1.3	+0.3	+2.0	+0.2	+0.7	-3.7	-0.9	-0.1
CutMix [56]	+1.0	+0.1	-0.3	+0.3	+0.8	+3.4	-5.4	-0.4	+0.1	+0.3	+0.7	+0.4	+0.1
Best WSOL	62.2	65.5	64.2	63.8	76.2	65.5	78.1	70.8	58.3	63.0	58.6	59.5	64.0
FSL baseline	62.8	68.7	67.5	66.3	86.3	94.0	95.8	92.0	61.5	70.3	74.4	68.7	75.7
Center baseline	52.5	52.5	52.5	52.5	59.7	59.7	59.7	59.7	45.8	45.8	45.8	45.8	52.3

Table 2. **Re-evaluating WSOL.** How much have WSOL methods improved upon the vanilla CAM model? `test` split results are shown, relative to the vanilla CAM performance (increase or decrease). Hyperparameters have been optimized over the identical `train-fullsup` split for all WSOL methods and the FSL baseline: (10,5,5) full supervision/class for (ImageNet,CUB,OpenImages). Reported results are in the Appendix Table 5; classification accuracies are in Appendix Table 4.

able, it has taken only about 200 hours over 50 P40 GPUs to obtain the results. The results are shown in Table 2. We use the same batch sizes and training epochs to enforce the same computational budget. The checkpoints that achieves the best localization performance on `train-fullsup` are used for evaluation.

Contrary to the improvements reported in prior work (Appendix Table 4), recent WSOL methods have not led to major improvements compared to CAM, when validated in the same data splits and same evaluation metrics. On ImageNet, methods after CAM are generally struggling: only CutMix has seen a boost of +0.3pp on average. On CUB, ADL has attained a +2.0pp gain on average, but ADL fails to work well on other benchmarks. On the new WSOL benchmark, OpenImages, no method has improved over CAM, except for CutMix (+0.4pp on average). The best overall improvements over CAM (63.8% total mean) is a mere +0.2pp boost by HaS. In general, we observe a random mixture of increases and decreases in performance over the baseline CAM, depending on the architecture and dataset. An important result in the table to be discussed later is the comparison against the few-shot learning baseline (§5.5).

Some reasons for the discrepancy between our results and the reported results include (1) the confounding of the actual score map improvement and the calibration scheme, (2) different types and amounts of full supervision employed under the hood, and (3) the use of different training settings (*e.g.* batch size, learning rates, epochs). More details about the training settings are in Appendix §C.3.

Which checkpoint is suitable for evaluation? After the acceptance by CVPR 2020, we believe that it is inappropriate to use the best checkpoint for WSOL evaluation. This is because the best localization performances are achieved before convergence in many cases (Appendix §C.2. At early epochs, the localization performance fluctuates a lot, so the peak performance is noise rather than the real performance.

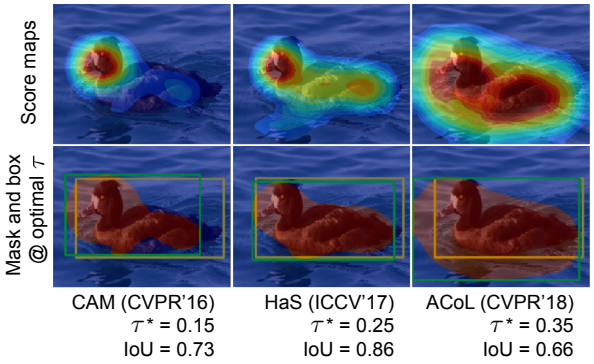


Figure 4. **Selecting τ.** Measuring performance at a fixed threshold τ can lead to a false sense of improvement. Compared to CAM, HaS and ACoL expand the score maps, but they do not necessarily improve the box qualities (IoU) at the optimal $\tau^\star$. Predicted and ground-truth boxes are shown as green and yellow boxes.

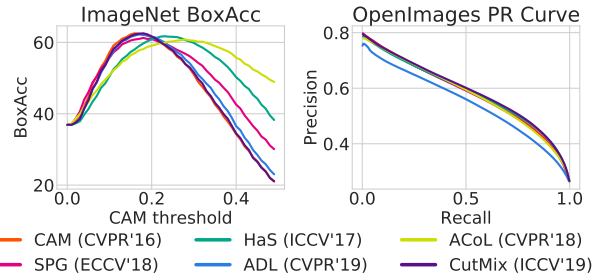
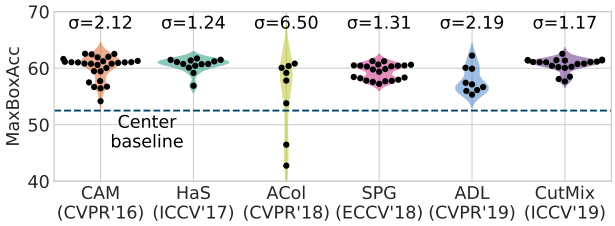


Figure 5. **Performance at varying operating thresholds.** ImageNet: `BoxAcc`(τ) versus τ. OpenImages: `PxPrec`(τ) versus `PxRec`(τ). Both use ResNet.

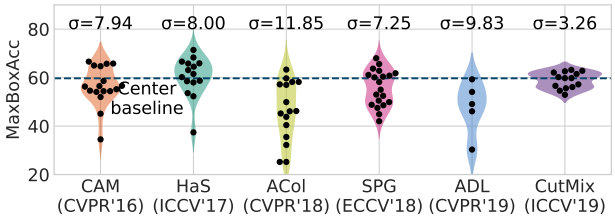
Hence, we recommend using the final checkpoint for future WSOL researchers. The evaluation results are shown in Appendix Table 6.

5.3. Score calibration and thresholding

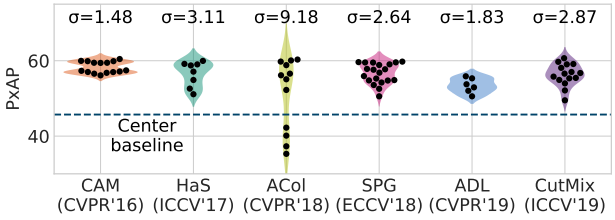
WSOL evaluation must focus more on score map evaluation, independent of the calibration. As shown in Fig-



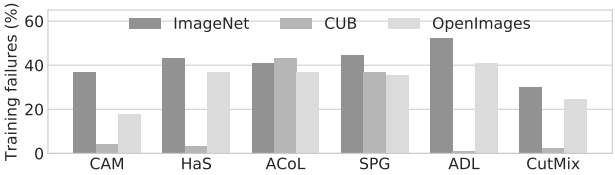
(a) ResNet50 architecture, ImageNet dataset.



(b) ResNet50 architecture, CUB dataset.



(c) ResNet50 architecture, OpenImages dataset.



(d) Ratio of training failures in different WSOL methods.

Figure 6. **Results of the 30 hyperparameter trials.** ImageNet performances of all 30 randomly chosen hyperparameter combinations for each method, with ResNet50 backbone. The violin plots show the estimated distributions (kernel density estimation) of performances. σ are the sample standard deviations.

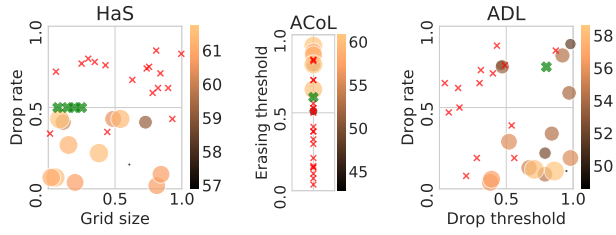


Figure 7. **Impact of hyperparameters for feature erasing.** Color and size of the circles indicate the performance at the corresponding hyperparameters. ✕: non-convergent training sessions. ✖: hyperparameters suggested by the original papers.

ure 4 the min-max normalized score map for CAM predicts a peaky foreground score on the duck face, While HaS and ACoL score maps show more distributed scores in body areas, demonstrating the effects of adversarial erasing during training. However, the maximal IoU performances do not differ as much. This is because WSOL methods exhibit different score distributions (Figure 5 and Appendix §C.4). Fixing the operating threshold τ at a pre-defined value, therefore, can lead to an apparent increase in performance without improving the score maps.

Under our threshold-independent performance measures (`MaxBoxAcc` and `PxAP`) shown in Figure 5, we observe that (1) the methods have different optimal $\tau^\star$ on ImageNet and (2) the methods do not exhibit significantly different `MaxBoxAcc` or `PxAP` performances. This provides an explanation of the lack of improvement observed in Table 2. We advise future WSOL researchers to report the threshold-independent metrics.

5.4. Hyperparameter analysis

Different types and amounts of full supervision used in WSOL methods manifest in the form of model hyperparameter selection (§3). Here, we measure the impact of the validation on `train-fullsup` by observing the performance distribution among 30 trials of random hyperparameters. We then study the effects of feature-erasing hyperparameters, a common hyperparameter type in WSOL methods.

Performance with 30 hyperparameter trials. To measure the sensitivity of each method to hyperparameter choices, we plot the performance distribution of the intermediate models in the 30 random search trials. We say that a training session is *non-convergent* if the training loss is larger than 2.0 at the last epoch. We show the performance distributions of the converged sessions, and report the ratio of non-convergent sessions separately.

Our results in Figure 6 indicate the diverse range of performances depending on the hyperparameter choice. Vanilla CAM is among the less sensitive, with the smallest standard deviation $\sigma = 1.5$ on OpenImages. This is the natural consequence of its minimal use of hyperparameters. We thus suggest to use the vanilla CAM when absolutely no full supervision is available. ACoL and ADL tend to have greater variances across benchmarks ($\sigma = 11.9$ and 9.8 on CUB). We conjecture that the drop threshold for adversarial erasing is a sensitive hyperparameter.

WSOL on CUB are generally struggling: random hyperparameters often show worse performance than the center baseline (66% cases). We conjecture that CUB is a disadvantageous setup for WSOL: as all images contain birds, the models only attend on bird parts for making predictions. We believe adding more non-bird images can improve the overall performances (§3.2).

We show the non-convergence statistics in Figure 6d. Vanilla CAM exhibit a stable training: non-convergence rates are lowest on OpenImages and second lowest on Ima-

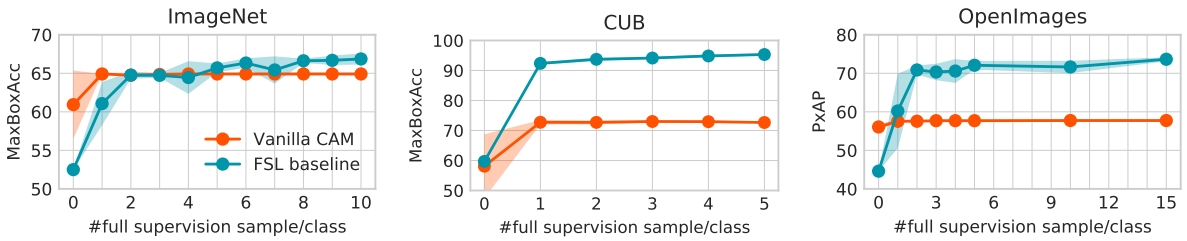


Figure 8. **WSOL versus few-shot learning.** The mean and standard error of models trained on three samples of full-supervision subsets are reported. ResNet50 is used throughout. At 0 full supervision, Vanilla CAM=random-hyperparameter and FSL=center-gaussian baseline.

geNet. ACoL and SPG suffer from many training failures, especially on CUB (43% and 37%, respectively).

In conclusion, vanilla CAM is stable and robust to hyperparameters. Complicated design choices introduced by later methods only seem to lower the overall performances rather than providing new avenues for performance boost.

Effects of erasing hyperparameters. Many WSOL methods since CAM have introduced different forms of erasing to encourage models to extract cues from broader regions (§5.1). We study the contribution of such hyperparameters in ADL, HaS, and ACoL in Figure 7. We observe that the performance improves with higher erasing thresholds (ADL drop threshold and ACoL erasing threshold). We also observe that lower drop rates leads to better performances (ADL and HaS). The erasing hyperparameters introduced since CAM only negatively impact the performance.

5.5. Few-shot learning baselines

Given that WSOL methods inevitably utilize some form of full localization supervision (§3), it is important to compare them against the few-shot learning (FSL) baselines that use it for model tuning itself.

Performances of the FSL baselines (§4.2) are presented in Table 2. Our simple FSL method performs better than the vanilla CAM at 10, 5, and 5 fully labeled samples per class for ImageNet, CUB, and OpenImages, respectively. The mean FSL accuracy on CUB is 92.0%, which is far better than that of the maximal WSOL performance of 70.8%.

We compare FSL against CAM at different sizes of `train-fullsup` in Figure 8. We simulate the zero-fully-labeled WSOL performance with a set of randomly chosen hyperparameters (§5.4); for FSL, we simulate the no-learning performance through the center-gaussian baseline.

FSL baselines surpass the CAM results already at 1-2 full supervision per class for CUB and OpenImages (92.4 and 70.9% `MaxBoxAcc` and `PxAP`). We attribute the high FSL performance on CUB to the fact that all images are birds; with 1 sample/class, there are effectively 200 birds as training samples. For OpenImages, the high FSL performance is due to the rich supervision provided by pixel-wise masks. On ImageNet, FSL results are not as great: they surpass the CAM result at 8-10 samples per class. Overall, however, FSL performances are strikingly good, even at a low data regime. Thus, given a few fully labeled samples, it is perhaps better to train a model with it than to search hyperparameters. Only when there is absolutely no full supervision (0 fully labeled sample), CAM is meaningful (better than the no-learning center-gaussian baseline).

6. Discussion and Conclusion

After years of weakly-supervised object localization (WSOL) research, we look back on the common practice and make a critical appraisal. Based on a precise definition of the task, we have argued that WSOL is ill-posed and have discussed how previous methods have used different types of implicit full supervision (*e.g.* tuning hyperparameters with pixel-level annotations) to bypass this issue (§3). We have then proposed an improved evaluation protocol that allows the hyperparameter search over a few labeled samples (§4). Our empirical studies lead to some striking conclusions: CAM is still not worse than the follow-up methods (§5.2) and it is perhaps better to use the full supervision directly for model fitting, rather than for hyperparameter search (§5.5).

We propose the following future research directions for the field. (1) Resolve the ill-posedness via *e.g.* adding more background-class images (§3.2). (2) Define the new task, *semi-weakly-supervised object localization*, where methods incorporating both weak and full supervision are studied.

Our work has implications in other tasks where learners are not supposed to be given full supervision, but are supervised implicitly via model selection and hyperparameter fitting. Examples include weakly-supervised vision tasks (*e.g.* detection and segmentation), zero-shot learning, and unsupervised tasks (*e.g.* disentanglement [31]).

Acknowledgements. We thank Dongyoon Han, Hyojin Park, Jaejun Yoo, Jung-Woo Ha, Junho Cho, Kyungjune Baek, Muhammad Ferjad Naeem, Rodrigo Benenson, Youngjoon Yoo, and Youngjung Uh for the feedback. NAVER Smart Machine Learning (NSML) [24] has been used. Graphic design by Kay Choi. The work is supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the MSIP (NRF-2019R1A2C2006123) and ICT R&D pro-

gram of MSIP/IITP [R7124-16-0004, Development of Intelligent Interaction Technology Based on Context Awareness and Human Intention Understanding]. This work was also funded by DFG-EXC-Nummer 2064/1-Projektnummer 390727645 and the ERC under the Horizon 2020 program (grant agreement No. 853489).

References

[1] Radhakrishna Achanta, Sheila Hemami, Francisco Estrada, and Sabine Süsstrunk. Frequency-tuned salient region detection. In *CVPR*, 2009. 4

[2] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. Whats the point: Semantic segmentation with point supervision. In *ECCV*, 2016. 2

[3] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *CVPR*, 2019. 5, 13

[4] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012. 5

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 5

[6] Junsuk Choe and Hyunjung Shim. Attention-based dropout layer for weakly supervised object localization. In *CVPR*, 2019. 1, 2, 3, 4, 5, 6, 12, 15, 16, 17

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 2

[8] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *ICCV*, 2015. 2

[9] Thomas Deselaers, Bogdan Alexe, and Vittorio Ferrari. Localizing objects while learning their appearance. In *ECCV*. Springer, 2010. 2

[10] Thomas Deselaers, Bogdan Alexe, and Vittorio Ferrari. Weakly supervised localization and learning with generic knowledge. *International journal of computer vision*, 100(3):275–293, 2012. 4

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 2

[12] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, 2017. 2

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

[14] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014. 2, 3

[15] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *ICML*, 2019. 2

[16] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93, 2019. 2

[17] David Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2, 2017. 2

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

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 13, 16

[20] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *ECCV*, 2018. 2

[21] Seunghoon Hong, Hyeonwoo Noh, and Bohyung Han. Decoupled deep neural network for semi-supervised semantic segmentation. In *NIPS*, 2015. 2

[22] James D Keeler, David E Rumelhart, and Wee Kheng Leow. Integrated segmentation and recognition of hand-printed numerals. In *NIPS*, 1991. 3

[23] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, 2017. 5

[24] Hanjoo Kim, Minkyu Kim, Dongjoo Seo, Jinwoong Kim, Heungseok Park, Soeun Park, Hyunwoo Jo, KyungHyun Kim, Youngil Yang, Youngkwan Kim, et al. Nsml: Meet the mlaas platform with a real-world case study. *arXiv preprint arXiv:1810.09957*, 2018. 8

[25] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self driving vehicles. In *ECCV*, 2018. 2

[26] Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *ICCV*, 2017. 1, 2, 3, 4, 5, 6, 12, 14, 15, 16, 17

[27] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *CVPR*, 2016. 2

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

[29] Daochang Liu, Tingting Jiang, and Yizhou Wang. Completeness modeling and context separation for weakly supervised temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[30] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on Pattern analysis and machine intelligence*, 33(2):353–367, 2010. 5

[31] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier

Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *ICML*, 2019. 8

[32] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2, 5

[33] Seong Joon Oh, Rodrigo Benenson, Anna Khoreva, Zeynep Akata, Mario Fritz, and Bernt Schiele. Exploiting saliency for object segmentation from image level labels. In *CVPR*, 2017. 5

[34] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *CVPR*, 2015. 2

[35] Dim P Papadopoulos, Alasdair DF Clarke, Frank Keller, and Vittorio Ferrari. Training object class detectors from eye tracking data. In *ECCV*, 2014. 2

[36] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*, 2015. 2, 3

[37] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *CVPR*, 2018. 2

[38] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[39] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019. 5, 12

[40] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *SIGKDD*, 2016. 2

[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, 2018. 2

[42] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1, 2, 3, 4, 12

[43] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2016. 2

[44] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 2

[45] Miaojing Shi and Vittorio Ferrari. Weakly supervised object localization using size estimates. In *ECCV*. Springer, 2016. 2, 3

[46] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLRW*, 2014. 2

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5, 16

[48] Hyun Oh Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, and Trevor Darrell. On learning to localize objects with minimal supervision. In *ICML*, 2014. 2, 3

[49] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. In *ICLRW*, 2015. 2

[50] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017. 2

[51] AH Sung. Ranking importance of input parameters of neural networks. *Expert Systems with Applications*, 15(3-4):405–411, 1998. 2

[52] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 5, 16

[53] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 1, 5

[54] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero- and few-label semantic segmentation. In *CVPR*, 2019. 2

[55] Haolan Xue, Chang Liu, Fang Wan, Jianbin Jiao, Xiangyang Ji, and Qixiang Ye. Danet: Divergent activation for weakly supervised object localization. In *ICCV*, 2019. 2

[56] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 1, 2, 3, 4, 5, 6, 12, 15, 16, 17

[57] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 15

[58] Xiaolin Zhang, Yunchao Wei, Jiashi Feng, Yi Yang, and Thomas S Huang. Adversarial complementary learning for weakly supervised object localization. In *CVPR*, 2018. 1, 2, 3, 4, 5, 6, 12, 14, 15, 16, 17

[59] Xiaolin Zhang, Yunchao Wei, Guoliang Kang, Yi Yang, and Thomas Huang. Self-produced guidance for weakly-supervised object localization. In *ECCV*, 2018. 1, 2, 3, 4, 5, 6, 12, 15, 16, 17

[60] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 1, 2, 3, 4, 5, 6, 11, 12, 14, 15, 16, 17

[61] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *ICLR*, 2017. 2

Appendix

We include additional materials in this document. Each section matches with the main paper sections: §A to main paper §3, §B to main paper §4, and §C to main paper §5.

A. Problem Formulation of WSOL

A.1. CAM as patch-wise posterior approximation

In the main paper §3.1, we have described class activation mapping (CAM) [60] as a patch-wise posterior approximator trained with image-level labels. We describe in detail why this is so.

Equivalent re-formulation of CAM. Originally, CAM is a technique applied on a convolutional neural network classifier $h : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^C$, where C is the number of classes, of the following form:

$$h_c(X) = \sum_d W_{cd} \left(\frac{1}{HW} \sum_{ij} g_{dij}(\mathbf{X}) \right) \tag{4}$$

where c, d are the channel-dimension indices and i, j are spatial-dimension indices. In other words, h is a fully convolutional neural network, followed by a global average pooling (GAP) and a linear (fully-connected) layer into a C-dimensional vector. We may swap the GAP and linear layers without changing the representation:

$$h_c(X) = \frac{1}{HW} \sum_{ij} \left(\sum_d W_{cd} g_{dij}(\mathbf{X}) \right) \tag{5}$$

$$=: \frac{1}{HW} \sum_{ij} f_{cij}(\mathbf{X}) \tag{6}$$

where f is now a fully-convolutional network. Each pixel (i, j) in the feature map, $(f_{1ij}(\mathbf{X}), \cdots, f_{Cij}(\mathbf{X}))$, corresponds to the classification result of the corresponding field of view in the input $\mathbf{X}$, written as X_{ij}. Thus, we equivalently write

$$h_c(X) = \frac{1}{HW} \sum_{ij} f_c(X_{ij}) \tag{7}$$

where f is now re-defined as a image patch classifier with 1-dimensional feature output (not fully convolutional).

CAM as patch-wise posterior approximator. h is now the average-pooled value of the patch-wise classification scores $f_c(X_{ij})$. CAM trains f by maximizing the image-wide posterior of the ground truth class, where the posterior is defined as the softmax over $f(X_{ij})$:

$$\log p(Y|\mathbf{X}) := \log \text{softmax}^Y \left(\frac{1}{HW} \sum_{ij} f(X_{ij}) \right). \tag{8}$$

In other words, CAM trains the network for patch-wise scores $f_c(X_{ij})$ to estimate the image-wide posterior $p(Y|\mathbf{X})$.

At inference time, CAM estimates the pixel-wise posterior $p(Y|X_{ij})$ approximately by performing $p(Y|X_{ij}) \approx f_Y(X_{ij}) / \max_{\alpha\beta} f_Y(X_{\alpha\beta})$ (modulo some calibration to make sure $p(Y|X_{ij}) \in [0, 1]$).

A.2. Proof for the ill-posedness lemma

We first define an evaluation metric for our score map for an easier argumentation.

Definition A.1. *For a scoring rule s and a threshold τ, we define the **pixel-wise localization accuracy** $\texttt{PxAcc}(s, \tau)$ as the probability of correctly predicting the pixel-wise labels:*

$$\begin{aligned}\texttt{PxAcc}(s, \tau) = P_{X,T}(s(X) \geq \tau \mid T = 1) \cdot P_{X,T}(T = 1) \\ + P_{X,T}(s(X) < \tau \mid T = 0) \cdot P_{X,T}(T = 0)\end{aligned}$$

We prove the following lemma.

Lemma A.2. *Assume that the true posterior $p(Y|M)$ with a continuous pdf is used as the scoring rule $s(M) = p(Y|M)$. Then, there exists a scalar $\tau \in \mathbb{R}$ such that $\texttt{PxAcc}(p(Y|M), \tau) = 1$ if and only if the foreground-background posterior ratio $\frac{p(Y=1|M^{fg})}{p(Y=1|M^{bf})} \geq 1$ almost surely, conditionally on the event $\{T(M^{fg}) = 1$ and $T(M^{bf}) = 0\}$.*

Proof. We write $E := \{T(M^{\text{fg}}) = 1 \text{ and } T(M^{\text{bf}}) = 0\}$.
(**Proof for "if"**) Assume $\alpha \geq 1$ almost surely, given E. Let

$$\tau := \min_{G: P(G \Delta \{T(m)=0\})=0} \max_{m \in G} p(Y = 1|M = m) \tag{9}$$

where Δ is the set XOR operation: $A \Delta B := (A \cup B) \setminus (A \cap B)$. Then, for almost all $M^{\text{fg}}, M^{\text{bg}}$ following E,

$$p(Y = 1|M^{\text{fg}}) \geq \tau \geq p(Y = 1|M^{\text{bg}}). \tag{10}$$

Therefore,

$$\begin{aligned}&P(p(Y = 1|M^{\text{fg}}) \geq \tau \mid T(M^{\text{fg}}) = 1) \\ &= P(p(Y = 1|M^{\text{bg}}) \leq \tau \mid T(M^{\text{bg}}) = 0) = 1\end{aligned} \tag{11}$$

and so $\texttt{PxAcc}(p(Y|M), \tau) = 1$.
(**Proof for "only if"**) Assume $\texttt{PxAcc}(p(Y|M), \tau) = 1$ for some τ. W.L.O.G., we assume that $P(T(M) = 1) \neq 0$ and $P(T(M) = 0) \neq 0$ (otherwise, $P(E) = 0$ and the statement is vacuously true). Then, Equation 11 must hold to ensure $\texttt{PxAcc}(p(Y|M), \tau) = 1$. Equation 10 then also holds almost surely, implying $\alpha \geq 1$ almost surely. ■

A.3. Foreground-background posterior ratio

We have described the the pathological scenario for WSOL as when the foreground-background posterior ratio

Figure 9. **Ducks.** Random duck images on Flickr. They contain more lake than feet pixels: $p(\text{water}|\text{duck}) \gg p(\text{feet}|\text{duck})$.

α is small (§3.2). We discuss in greater detail what it means and whether there are data-centric approaches to resolve the issue. For quick understanding, assume the task is the localization of duck pixels in images. The foreground cue of interest M^{fg} is "feet" of a duck and background cue of interest M^{bg} is "water". Then, we can write the posterior ratio as

$$\alpha := \frac{p(\text{duck}|\text{feet})}{p(\text{duck}|\text{water})} = \frac{p(\text{feet}|\text{duck})}{p(\text{water}|\text{duck})} \cdot \left(\frac{p(\text{feet})}{p(\text{water})}\right)^{-1}$$

$\alpha < 1$ implies that lake patches are more abundant in duck images than are duck's feet (see Figure 9) for an illustration.

To increase α, two approaches can be taken. (1) Increase the likelihood ratio $\frac{p(\text{feet}|\text{duck})}{p(\text{water}|\text{duck})}$. This can be done by collecting more images where duck's feet have more pixels than lake does. (2) Decrease the prior ratio $\frac{p(\text{feet})}{p(\text{water})}$. Note that the prior ratio can be written

$$\frac{p(\text{feet})}{p(\text{water})} = \frac{p(\text{feet}|\text{duck})p(\text{duck}) + p(\text{feet}|\text{duck}^c)p(\text{duck}^c)}{p(\text{water}|\text{duck})p(\text{duck}) + p(\text{water}|\text{duck}^c)p(\text{duck}^c)}$$

With fixed likelihoods $p(\text{feet}|\text{duck})$ and $p(\text{water}|\text{duck})$, one can decrease the prior ratio by increasing the likelihood of lake cues in non-duck images $p(\text{water}|\text{duck}^c)$. We can alter WSOL into a more well-posed task also by including many background images containing confusing background cues.

Such data-centric approaches are promising future research directions for turning WSOL into a well-posed task.

B. Evaluation Protocol for WSOL

B.1. Score map normalization

A common practice in WSOL is to normalize the score maps per image because the maximal (and minimal) scores differ vastly across images. Prior WSOL papers have introduced either max normalization (dividing through by $\max_{ij} s_{ij}$) or min-max normalization (additionally mapping $\min_{ij} s_{ij}$ to zero). After normalization, WSOL methods threshold the score map at τ to generate a tight box around the binary mask $\{(i,j) \mid s_{ij} \geq \tau\}$. τ is typically treated as a fixed value [60, 58, 56] or a hyperparameter to be tuned [26, 59, 6]. We summarize that how prior works calibrate and threshold score maps in Table 3. As discussed in §4.1 of main paper, we use the min-max normalization.

Method	Paper	Code
CAM [60]	$\bar{s}_{ij} \geq 0.2$	$\bar{s}_{ij} \geq 0.2$
HaS [26]	Follow CAM†	Follow CAM
ACoL [58]	Follow CAM	$\hat{s}_{ij} \geq$ unknown
SPG [59]	Grid search threshold	$\hat{s}_{ij} \geq$ unknown
ADL [6]	Not discussed	$\hat{s}_{ij} \geq 0.2^\dagger$
CutMix [56]	$\bar{s}_{ij} \geq 0.15$	$\hat{s}_{ij} \geq 0.15$
Our protocol	$\hat{s}_{ij} \geq \tau^\star$	$\hat{s}_{ij} \geq \tau^\star$

$$\bar{s}_{ij} := \frac{s_{ij}}{\max_{kl} s_{kl}} \qquad \hat{s}_{ij} := \frac{s_{ij} - \min_{kl} s_{kl}}{\max_{kl} s_{kl} - \min_{kl} s_{kl}}$$

Table 3. **Calibration and thresholding in WSOL.** Score calibration is done per image: max ($\bar{s}_{ij}$) or min-max ($\hat{s}_{ij}$) normalization. Thresholding is required only for the box evaluation. $\tau^\star$ is the optimal threshold (§4.1 in main paper). Daggers (†) imply that the threshold depends on the backbone architecture.

B.2. Data preparation

We present the following data-wise contributions in this paper (contributions **bolded**):

- CUB: **New data** (5 images per class) with **bounding box annotations**.
- ImageNet: ImageNetV2 [39] with new **bounding box annotations**.
- OpenImages: **Processed** a split for the WSOL training, hyperparameter search, and evaluation with ground truth object masks.

B.2.1 ImageNet

The *test set* of ImageNet-1k dataset [42] is not available. Therefore, many researchers report the accuracies on the *validation set* for their final results [56]. Since this practice may let models overfit to the evaluation split over time, ImageNetV2 [39] has been proposed as the new test sets for ImageNet-1k trained models. ImageNetV2 includes three subsets according to the sampling strategies, `MatchedFrequency`, `Threshold0.7`, and `TopImages`, each with 10 000 images (10 images per class). We use the `Threshold0.7` split as our `train-fullsup`. Since ImageNetV2 does not contain localization supervision, we have annotated bounding boxes on those images, following the annotation protocol of ImageNet. The total number of annotated bounding boxes are 18 532.

B.2.2 CUB

We have collected 5 images for each of the 200 CUB fine-grained bird classes from Flickr. The overall procedure is summarized as:

1. Crawl images from Flickr.
2. De-duplicate images.
3. Manually prune irrelevant images (three people).
4. Prune with model classification scores.
5. Resize images.
6. Annotate bounding boxes.

Crawl images from Flickr. Since original CUB itself is collected from Flickr, we use Flickr as the source of bird images. We have used the `Flickr API`[1] to crawl up to 400 images per class with class name as search terms. We have only crawled images under the following licenses:

- `Attribution`
- `Attribution-NonCommercial`
- `Public Domain Dedication`
- `Public Domain Mark`

De-duplicate images. We de-duplicate the images using the `ImageHash` library[2] first among the crawled images themselves and then against the training and test splits of CUB.

Manually prune irrelevant images (three people). Since crawled images of each class contain negative-class images (non birds and birds of wrong categories), three humans have participated in the prune-out process. We have only kept the images that all three have voted for "positive".

Prune with model classification scores. To match with the original CUB data distribution, we have trained a fine-grained bird classifier using ResNet50 [19] on the CUB training split. Among crawled images, those with ground truth class confidence scores lower than 0.5 have been pruned out.

Achieving 5 images per class. At this point, most classes have more than 5 images per class and a few have less than 5. In the former case, we have randomly sampled 5 images per class.

Resize images. Photographic technologies have made significant advances over the decade since the time original CUB was collected. As a result, image size statistics differ. To fix this, we have resized the images appropriately.

[1] https://www.flickr.com/services/api/
[2] https://pypi.org/project/ImageHash/

Figure 10. **CUB version 2.** Sample images.

Annotate bounding boxes. The evaluation on CUB is based on bounding boxes. Thus, the held-out set images are supplied with tight bounding boxes around birds (one per image).

B.2.3 OpenImages

There are three significant differences between OpenImagesV5 [3] and CUB or ImageNet that make the OpenImages not suitable as a WSOL benchmark in its original form. (1) Images are multi-labeled; it is not sensible to train classifiers with the standard softmax cross-entropy loss assuming single label per image. (2) OpenImages has less balanced label distributions. (3) There are nice instance segmentation masks, but they have many missing instances.

We have therefore processed a subset of OpenImages into a WSOL-friendly dataset where the above three issues are resolved. The procedure is as follows:

1. Prune multi-labeled samples.
2. Exclude classes with not enough samples.
3. Randomly sample images for each class.
4. Prepare binary masks.
5. Introduce ignore regions.

Prune multi-labeled samples. We prune the multi-labeled samples in the segmentation subset of OpenImages. This process rejects 34.5% samples in OpenImages.

Exclude classes with not enough samples. After pruning multi-label samples, some classes are not available in validation and test sets. We have first defined the minimum number of samples per classes as (300, 25, 50) for (train, validation, test), and have excluded classes that do not meet the minimum requirement, resulting in 100 classes (out of 350 original classes).

Randomly sample images for each class. We have randomly sampled (300, 25, 50) samples per class for (train, validation, test). This eventually results in 29 819

Figure 11. **OpenImages V5 sample.** Example of "group-of" box annotations. There is only one "Book" mask in the left bottom corner.

`train-weaksup` (train), 2 500 `train-fullsup` (validation), and 5 000 `test` (test) samples.

Prepare binary masks. WSOL methods are evaluated against binary masks of the foreground category in each image. Since OpenImages comes with instance-level masks of various categories, we have taken the union of instance masks of the class of interest in every image to build the binary masks.

Introduce ignore regions. OpenImages has instance-wise masks, but not all of them are annotated as masks. When the objects are difficult to be annotated instance-wise or there are simply too many, the "group-of" box annotations are provided over the region containing multiple instances (*e.g.* hundreds of books on bookshelves in an image taken at a library – See Figure 11). We have indicated the regions with the "group-of" boxes as "ignore" regions, and have excluded them from the evaluation during the computation of `PxPrec`, `PxRec`, and `PxAP`.

B.3. Transferability of rankings between `train-fullsup` and `test`

As detailed in main paper §4.2, we search hyperparameters on `train-fullsup` and test them on `test`. To validate if the found hyperparameter rankings do transfer well between the splits, we show the preservation of ranking statistics in Table 4 and visualize the actual rankings in Figure 14. We observe that the rankings are relatively well-preserved (with Kendall's tau values > 0.7).

B.4. Hyperparameter search with proxy `train-weaksup`

We have reduced the training set size (`train-weaksup`) to 10% for ImageNet hyperparameter search for the interest of computational efficiency (§4.2). We examine how much this reduction affects the rankings of hyperparameters. We show a similar analysis as in §B.3: see Figure 12. We observe again that with Kendall's tau value 0.743, the two rankings are largely preserved.

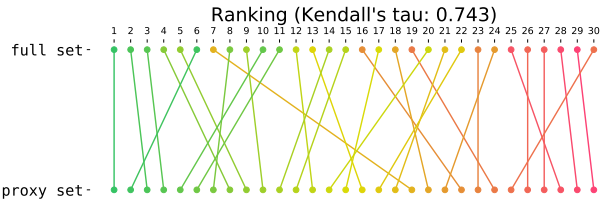


Figure 12. **Proxy ImageNet ranking**. Ranking of hyperparameters is largely preserved between the models trained on the full `train-weaksup` and its 10% proxy. Kendall's tau is 0.743.

C. Experiments

C.1. Prior WSOL methods and hyperparameters

We describe each method in greater detail here. The list of hyperparameters for each method is in Table 5.

CAM, CVPR'16 [60]. Class Activation Mapping trains a classifier of fully-convolutional backbone with global average pooling structure, and at test time uses the logit outputs before the global average pooling for the scoring rule $s(X_{ij})$. See Appendix §A.1 for the interpretation of CAM as a posterior approximator. CAM has learning rate (LR) and the score-map resolution (SR) as hyperparameters. LR is sampled log-uniformly from $[10^{-5}, 10^{0}]$, where end points correspond roughly to "no training" and "training always diverges" cases. SR is sampled from Categorical$\{14, 28\}$, two widely used resolutions in prior WSOL methods. All five methods below use CAM technique in the background, and have LR and HR as design choices.

HaS, ICCV'17 [26]. Hide-and-Seek (HaS) is a data augmentation technique that divides an input image into grid-like patches, and then randomly select patches to be dropped. The hyperparameters of HaS are drop rate (DR) and drop area (DA). Specifically, the size of each patch is decided by DA, and the probability of each patch to be selected for erasing is decided by DR. DA is sampled from a uniform distribution $U[0, 1]$, where 0 corresponds to "no grid" and 1 indicates "full image as one patch".

ACoL, CVPR'18 [58]. Adversarial Complementary Learning (ACoL) adds one more classification head to backbone networks. From one head, ACoL finds the high-score region using CAM using erasing threshold (ET) and erases it from an internal feature map. The other head

Methods	ImageNet				CUB				OpenImages				Total
	VGG	Inception	ResNet	Mean	VGG	Inception	ResNet	Mean	VGG	Inception	ResNet	Mean	Mean
CAM [60]	0.887	0.795	0.933	0.872	0.731	0.706	0.758	0.732	0.869	0.714	0.934	0.839	0.814
HaS [26]	0.855	0.795	0.867	0.839	0.422	0.714	0.867	0.668	0.786	1.000	0.667	0.818	0.775
ACoL [58]	0.981	1.000	0.619	0.867	0.895	0.850	0.850	0.854	0.983	0.970	0.956	0.970	0.897
SPG [59]	0.944	0.766	0.900	0.870	0.697	0.779	0.889	0.788	0.758	0.874	0.929	0.854	0.837
ADL [6]	0.917	0.944	0.857	0.906	0.891	1.000	0.810	0.900	0.891	1.000	1.000	0.964	0.923
CutMix [56]	0.897	0.571	1.000	0.823	0.544	0.407	0.879	0.610	0.838	0.867	0.714	0.806	0.746

Table 4. **In-distribution ranking preservation.** Kendall's tau values for the hyperparameter ranking between `train-fullsup` and `test` are shown.

Methods	Hyperparameter	Distribution
Common	Learning rate	LogUniform$[10^{-5}, 10^{0}]$
	Score-map resolution	Categorical$\{14, 28\}$
HaS [26]	Drop rate	Uniform$[0, 1]$
	Drop area	Uniform$[0, 1]$
ACoL [58]	Erasing threshold	Uniform$[0, 1]$
SPG [59]	Threshold δ_l^{B1}	Uniform$[0, 1]$
	Threshold δ_h^{B1}	Uniform$[\delta_l^{B1}, 1]$
	Threshold δ_l^{B2}	Uniform$[0, 1]$
	Threshold δ_h^{B2}	Uniform$[\delta_l^{B2}, 1]$
	Threshold δ_l^{C}	Uniform$[0, 1]$
	Threshold δ_h^{C}	Uniform$[\delta_l^{C}, 1]$
ADL [6]	Drop rate	Uniform$[0, 1]$
	Erasing threshold	Uniform$[0, 1]$
CutMix [56]	Size prior	$\frac{1}{\text{Unif}(0,2]} - \frac{1}{2}$
	Mix rate	Uniform$[0, 1]$

Table 5. **Hyperparameter search spaces.**

learns remaining regions using the erased feature map. We sample ET from a uniform distribution $U[0, 1]$, where 0 means "erasing whole feature map" and 1 means "do not erase".

SPG, ECCV'18 [59]. Self-produced Guidance (SPG) utilizes spatial information about fore- and background using three additional branches (`SPG-B1,B2,C`). To divide foreground and background from score-map, they introduce two hyperparameters, δ_l and δ_h, per each branch. When the score is lower than δ_l, the pixel is considered as background, and the pixel is considered as foreground when the score is higher than δ_h. The remaining region (higher than δ_l, lower than δ_h) is ignored. We first sample δ_l from $U[0, 1]$, and then δ_h is sampled from $U[\delta_l, 1]$.

ADL, CVPR'19 [6]. Attention-based Dropout Layer (ADL) is a block applied on an internal feature map during training. ADL produces a drop mask by finding the high-score region to be dropped using another scoring rule [57]. Also, ADL produces an importance map by normalizing the score map and uses it to increase classification power of the backbone. At each iteration, only one component is applied between the drop mask and importance map. The hyper-parameters of ADL are drop rate (DR) that indicates how frequently the drop mask is selected and erasing threshold (ET) that means how large regions are dropped. We sample DR and ET from uniform distributions $U[0, 1]$.

CutMix, ICCV'19 [56]. CutMix is a data augmentation technique, where patches in training images are cut and paste to other images and target labels are mixed likewise. Its hyperparameters consist of the size prior α (used for sampling sizes according to $\sim$ Beta(α, α)) and the mix rate r (Bernoulli decision for "CutMix or not"). The size prior is sampled from the positive range $\frac{1}{\text{Unif}(0,2]} - \frac{1}{2}$; then, Var(Beta$(\alpha, \alpha)$) follows the uniform distribution between 0 and 0.25 (maximal variance; two Dirac deltas at 0 and 1).

C.2. Classification results of WSOL methods

The widely-used "top-1 localization accuracy" for WSOL represents both the classification *and* localization performances. The metric can be misleading as a localization metric, as the increase in numbers can also be attributed to the improved classification accuracies. Thus, in the main paper, we have suggested using on the "GT-known" type of metrics like `MaxBoxAcc` and `PxAP` that measures the localization performances given perfect classification.

Here, we measure the classification accuracies of the models in Table 2 of the main paper, to complete the analysis. The performances are reported in Table 6. There are in general great fluctuations in the classification results (26.8% to 74.5% on CUB). This is because we have selected the hyperparameters for each model that maximize the localization performances on `train-fullsup` split. In many cases, the best localization performances are achieved at early epochs, before the classifiers are sufficiently trained (see also §C.6 and Figure 20). The result signifies that localization and classification performances may not necessarily correlate and, therefore, localization-only metrics like `MaxBoxAcc` and `PxAP` must be used for model selection and evaluation of WSOL methods.

Methods	ImageNet VGG	ImageNet Inception	ImageNet ResNet	ImageNet Mean	CUB VGG	CUB Inception	CUB ResNet	CUB Mean	OpenImages VGG	OpenImages Inception	OpenImages ResNet	OpenImages Mean	Total Mean
CAM [60]	63.8	68.7	75.9	69.5	26.8	61.8	58.4	49.0	67.3	36.6	72.6	58.8	59.1
HaS [26]	61.9	65.5	63.1	63.5	70.9	69.9	74.5	71.8	60.0	68.4	74.0	67.5	67.6
ACoL [58]	60.3	64.6	61.6	62.2	56.1	71.6	64.0	63.9	68.2	40.7	70.7	59.9	62.0
SPG [59]	61.6	65.5	63.4	63.5	63.1	58.8	37.8	53.2	71.7	43.5	65.4	60.2	59.0
ADL [6]	60.8	61.6	64.1	62.2	31.1	45.5	32.7	36.4	66.1	46.6	56.1	56.3	51.6
CutMix [56]	62.2	65.5	63.9	63.8	29.2	70.2	55.9	51.8	68.1	53.1	73.7	65.0	60.2

Table 6. **Classification performance of WSOL methods.** Classification accuracies of the models in Table 2 of the main paper are shown. Hyperparameters for each model are optimally chosen for the localization performances on `train-fullsup` split. Classification performances may be sub-optimal when the best localization performance is achieved at early epochs.

	Methods	Top-1 localization accuracy: ImageNet V	ImageNet I	ImageNet R	CUB V	CUB I	CUB R	GT-known localization; MaxBoxAcc and PxAP: ImageNet V	ImageNet I	ImageNet R	CUB V	CUB I	CUB R	OpenImages V	OpenImages I	OpenImages R
Reported	CAM [60]	42.8	-	46.3	37.1	43.7	49.4	-	62.7	-	-	-	-	-	-	-
	HaS [26]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	ACoL [58]	45.8	-	-	45.9	-	-	-	-	-	-	-	-	-	-	-
	SPG [59]	-	48.6	-	-	46.6	-	-	64.7	-	-	-	-	-	-	-
	ADL [6]	44.9	48.7	-	52.4	53.0	-	-	-	-	75.4	-	-	-	-	-
	CutMix [56]	43.5	-	47.3	-	52.5	54.8	-	-	-	-	-	-	-	-	-
Reproduced	CAM [60]	45.5	48.8	51.8	45.8	40.4	56.1	61.1	65.3	64.2	71.1	62.1	73.2	58.1	61.4	58.0
	HaS [26]	46.3	49.7	49.9	55.6	41.1	60.7	61.9	65.5	63.1	76.2	57.7	78.1	56.9	58.5	58.2
	ACoL [58]	45.5	49.9	47.4	44.8	46.8	57.8	60.3	64.6	61.6	72.3	59.5	72.7	54.6	63.0	57.8
	SPG [59]	44.6	48.6	48.5	42.9	44.9	51.5	61.6	65.5	63.4	63.7	62.7	71.4	55.9	62.4	57.7
	ADL [6]	44.4	45.0	51.5	39.2	35.2	41.1	60.8	61.6	64.1	75.6	63.3	73.5	58.3	62.1	54.3
	CutMix [56]	46.1	49.2	51.5	47.0	48.3	54.5	63.9	62.2	65.4	71.9	65.5	67.8	58.2	61.7	58.6

	Architecture
V	VGG-GAP [47]
I	InceptionV3 [52]
R	ResNet50 [19]

Table 7. **Previously reported vs our results.** The first six rows are reported results in prior WSOL papers. When there are different performance reports for the same method in different papers, we choose the greater performance. The last six rows are our re-implemented results under the new evaluation method (§4).

Methods	ImageNet (MaxBoxAccV2) VGG	ImageNet (MaxBoxAccV2) Inception	ImageNet (MaxBoxAccV2) ResNet	ImageNet (MaxBoxAccV2) Mean	CUB (MaxBoxAccV2) VGG	CUB (MaxBoxAccV2) Inception	CUB (MaxBoxAccV2) ResNet	CUB (MaxBoxAccV2) Mean	OpenImages (PxAP) VGG	OpenImages (PxAP) Inception	OpenImages (PxAP) ResNet	OpenImages (PxAP) Mean	Total Mean
CAM [60]	60.0	63.4	63.7	62.4	63.7	56.7	63.0	61.1	58.3	63.2	58.5	60.0	61.2
HaS [26]	+0.6	+0.3	-0.3	+0.2	+0.0	-3.3	+1.7	-0.5	-0.2	-5.1	-2.6	-2.6	-1.0
ACoL [58]	-2.6	+0.3	-1.4	-1.2	-6.3	-0.5	3.5	-1.1	-4.0	-6.0	-1.2	-3.7	-2.0
SPG [59]	-0.1	-0.1	-0.4	-0.2	-7.4	-0.8	-2.6	-3.6	+0.0	-0.9	-1.8	-0.9	-1.6
ADL [6]	-0.2	-2.0	+0.0	-0.7	+2.6	+2.1	-4.6	+0.0	+0.4	-6.4	-3.3	-3.1	-1.3
CutMix [56]	-0.6	+0.5	-0.4	-0.2	-1.4	+0.8	-0.2	-0.3	-0.2	-0.7	-0.8	-0.6	-0.3
Best WSOL	60.6	63.9	63.7	62.6	66.3	58.8	66.4	61.1	58.7	63.2	58.5	60.0	61.2
FSL baseline	60.3	65.3	66.3	64.0	71.6	86.6	82.4	80.2	65.9	74.1	74.4	71.5	71.9
Center baseline	52.5	52.5	52.5	52.5	59.7	59.7	59.7	59.7	45.8	45.8	45.8	45.8	52.3

Table 8. **Re-evaluating WSOL with `MaxBoxAccV2`.** We re-evaluate six recently proposed WSOL methods with `MaxBoxAccV2`. The experimental setting is the same as that of Table 2 in the main paper.

C.3. Reproducing prior WSOL results

We also summarize reported results in prior WSOL papers [60, 26, 58, 59, 6, 56] along with our reproduced results in Table 7. While our main evaluation metrics are the classification-disentangled GT-known measures (§4.1 in main paper), we also report the *top-1 localization* metrics that also measure the classification scores to match the reported numbers. Note that we use oracle τ for Top-1 localization accuracy.

We experimentally observe that training epochs and batch sizes influence the localization accuracies ($\sim$ 10pp in `MaxBoxAcc`). However, the training details for each method are not given by the corresponding papers. All

methods should share the same training budget for fair comparisons. Therefore, we fix our training epochs to (10, 50, 10) for (ImageNet, CUB, OpenImages) experiments, and fix the batch size to 32 for all datasets.

Following the prior methods [60, 26, 58, 59, 6, 56], we train all six methods by fine-tuning ImageNet pre-trained model. Specifically, we apply $10\times$ scale of learning rate to higher-level layers of backbone networks during training. We consider the last two layers for VGG, last three layers for InceptionV3, and last three residual blocks, and fully connected layers for ResNet as the higher-level layers.

Note that with GT-known metrics, our re-implementations produce better performances than the previously reported results (*e.g.* 62.7 $\rightarrow$ 65.3 for CAM on ImageNet with Inception backbone). This is perhaps because `MaxBoxAcc` and `PxAP` are based on the best operating thresholds.

Top-1 localization accuracies are reproduced well in general, except for the ADL: *e.g.* 52.4 $\rightarrow$ 39.2 for ADL on CUB with VGG backbone. This is due to the reduced training epochs compared to the original paper [6], which results in a decreased classification accuracies.

C.4. Score calibration and thresholding

The operating threshold τ for the score map s is an important parameter for WSOL. We show additional plots and visualizations to aid understanding. In Figure 15, we show the `BoxAcc` and `PxPrec-PxRec` performances at different operating thresholds with diverse architectures and datasets. It extends the main paper Figure 5. We observe again that the optimal operating thresholds $\tau^\star$ are vastly different across data and architectures for `BoxAcc`. The `MaxBoxAcc` in each method are relatively stable across methods especially on ImageNet. OpenImages `PxPrec-PxRec` curves do not exhibit big differences amongst WSOL methods, likewise.

In Figure 13, we visualize score distributions for different WSOL methods. We observe that methods have different distributions of scores; ACoL in particular tends to generate flatter score maps. Comparing dataset, we observe that OpenImages tends to have more peaky score distributions. It is therefore important to find the optimal operating point for each method and dataset for fair comparison.

We visualize more score maps in Figure 16, 17, 18; it extends the main paper Figure 4. We observe qualitatively different score maps in general across methods. However, the optimal IoU values are not as different and hard to predict just by visually looking at the samples. Again, it is important to have an objective way to set the thresholds τ in each case.

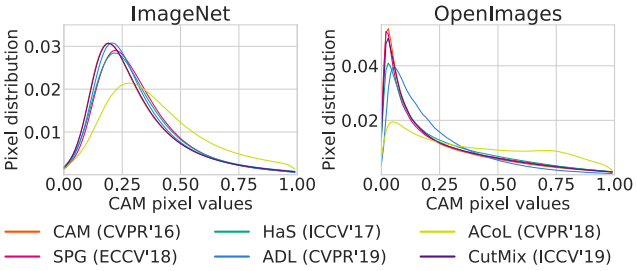


Figure 13. **CAM pixel value distributions.** On ImageNet and OpenImages `test`.

C.5. Hyperparameter analysis

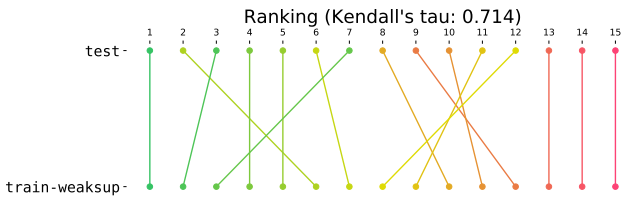
We show the performance of all 30 hyperparameter search iterations on three datasets with three backbone architectures in Figure 19; it extends the main paper Figure 6. We observe similar trends as in the main paper. (1) Performances do vary according to the hyperparameter choice, so the hyperparameter optimization is necessary for the optimal performances. (2) CAM is among the more stable WSOL methods. (3) ACoL and ADL show greater sensitivity to hyperparameters in general. (4) CUB is a difficult benchmark where random choice of hyperparameters is highly likely to lead to performances worse than the center-Gaussian baseline.

C.6. Learning Curves.

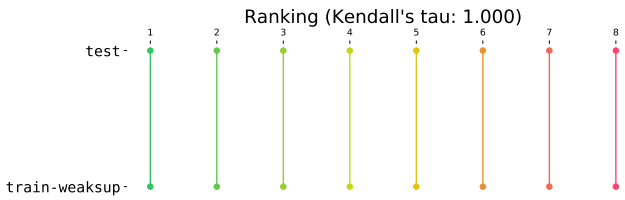
We show the learning curves for CUB, ImageNet, OpenImages in Figure 20. We observe that the performances converge at epochs (10, 50, 10) for (ImageNet, CUB, OpenImages); the number of epochs for training models is thus sufficient for all methods. Note that learning rates are decayed by 10 every (3, 15, 3) epochs for (ImageNet, CUB, OpenImages). For the most cases, performance increases as the training progresses. However, in some cases (*e.g.* CUB, SPG and OpenImages, CAM cases), best performances have already achieved at early iteration. That is, classification training rather hurts localization performance. Analyzing this is an interesting future study.

C.7. Evaluating WSOL methods with `MaxBoxAccV2`

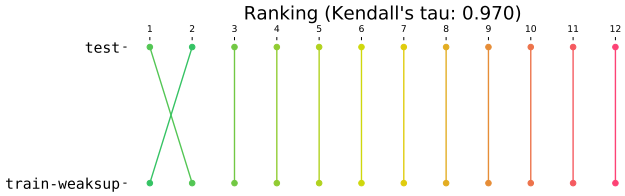
We re-evaluate six recently proposed WSOL methods with `MaxBoxAccV2`, and the results are shown in Table 8. All training configurations but evaluation metrics are the same as those of the main paper. We evaluate the last checkpoint of each training session. We obtain the same conclusions as with the original metric, `MaxBoxAcc`: (1) there has been no significant progress in WSOL performances beyond vanilla CAM [60] and (2) with the same amount of fully-supervised samples, FSL baselines provide better performances than the existing WSOL methods.



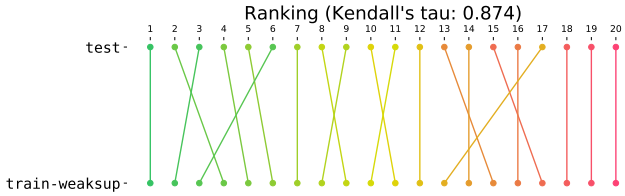
(a) CAM



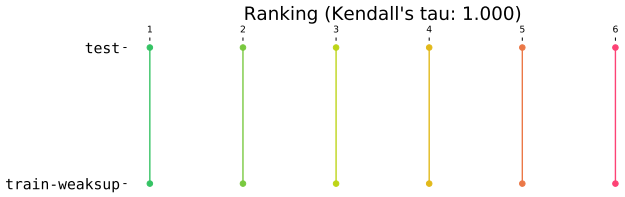
(b) HaS



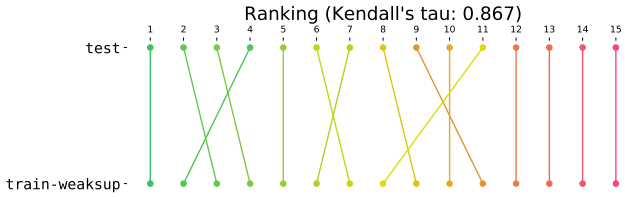
(c) ACoL



(d) SPG



(e) ADL



(f) CutMix

Figure 14. **Preservation of hyperparameter rankings.** Ranking of hyperparameters is largely preserved between `train-weaksup` and `test`. We only show the converged sessions (§5.4). Results on ResNet50, OpenImages.

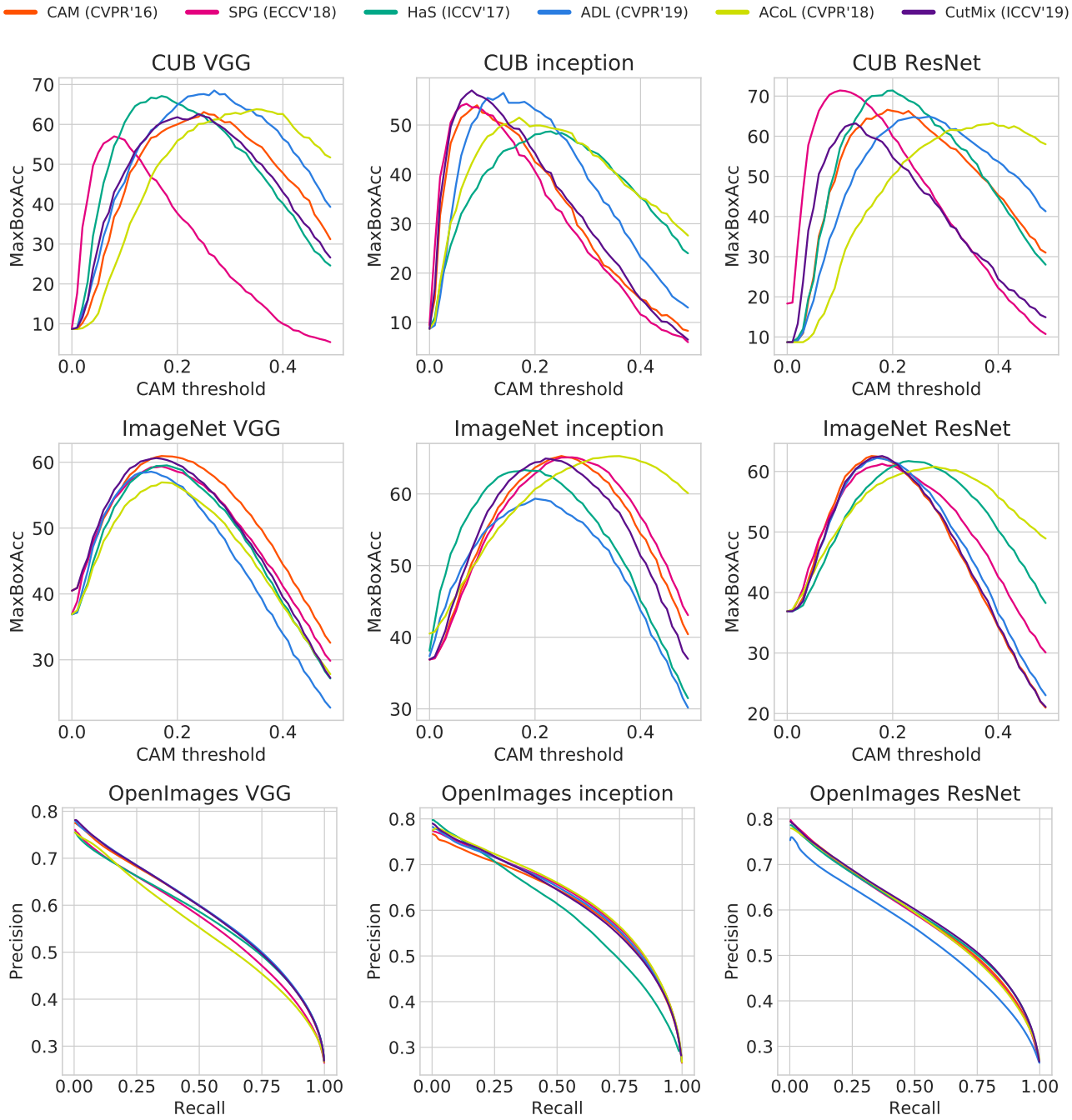
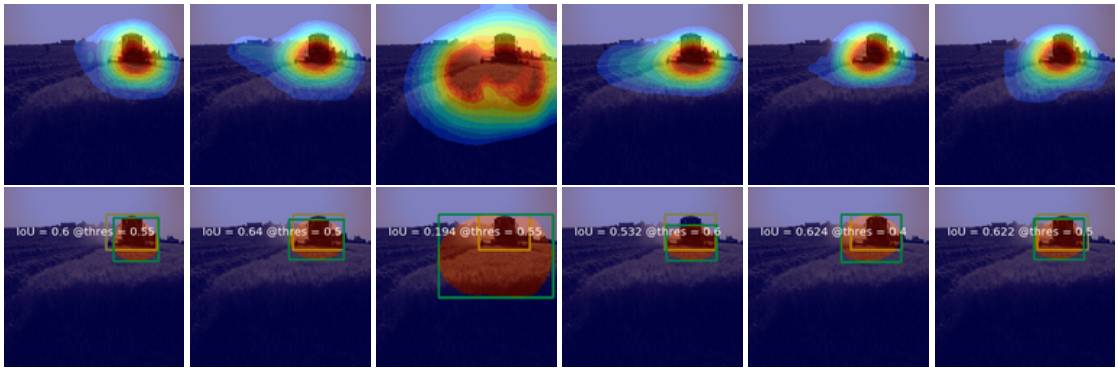
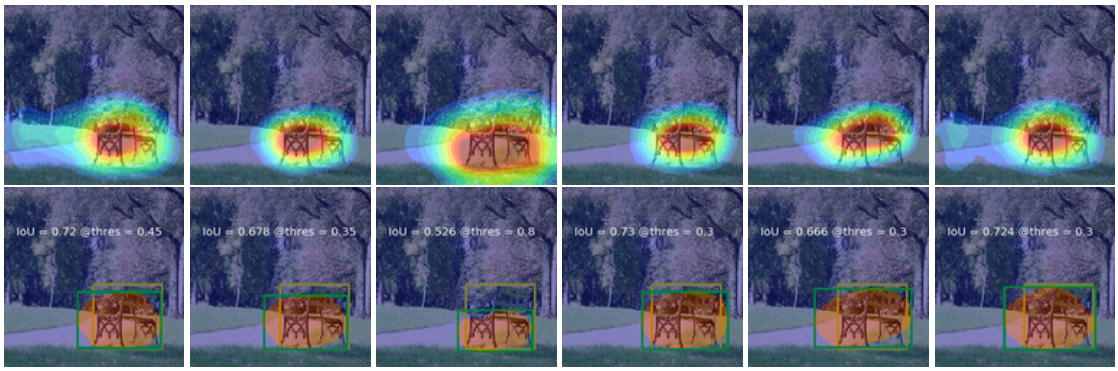


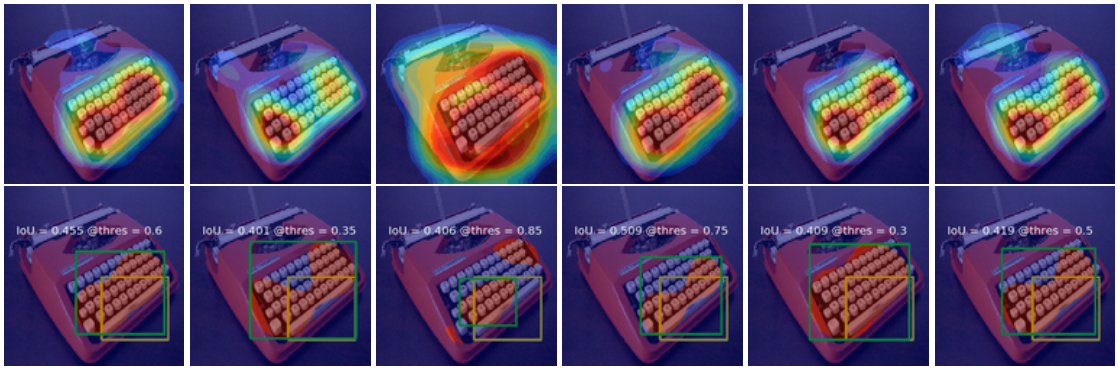
Figure 15. **Performance by operating threshold τ.** CUB and ImageNet: `BoxAcc` versus τ, OpenImages: `PxPrec` versus `PxRec`. ResNet, VGG, and Inception architecture results are used. This is the extension of Figure 5 in the main paper.



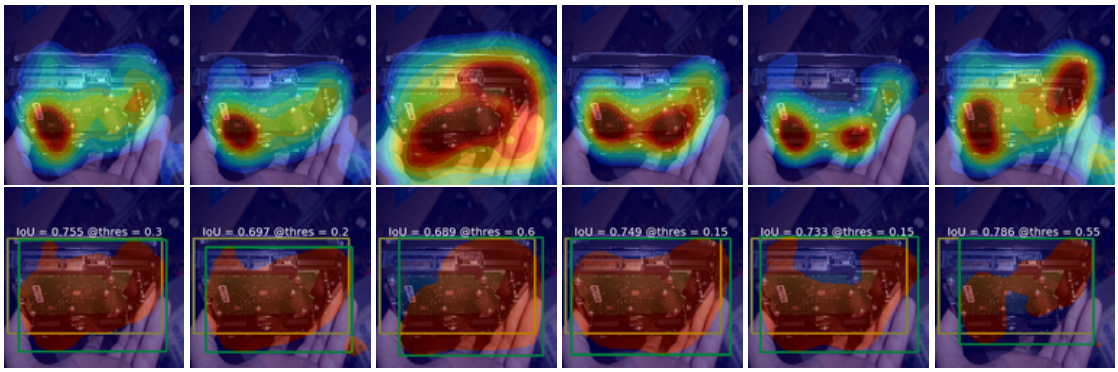
(a) Label: Harvester



(b) Label: Park bench

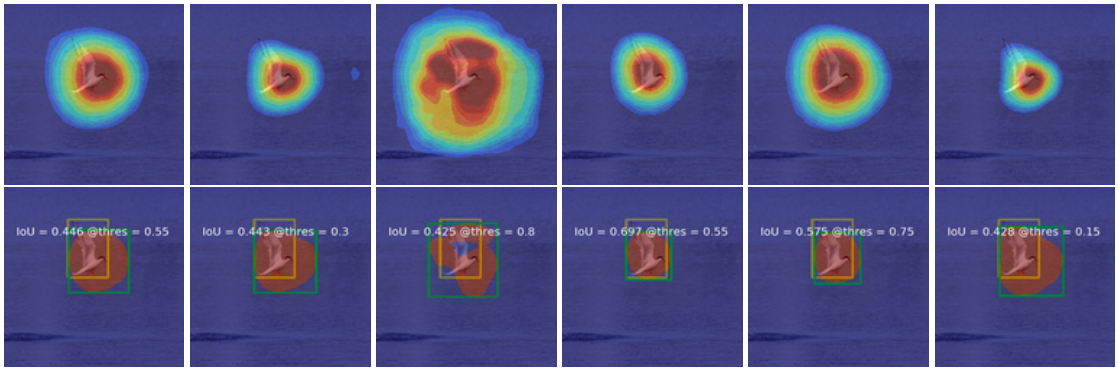


(c) Label: Space bar

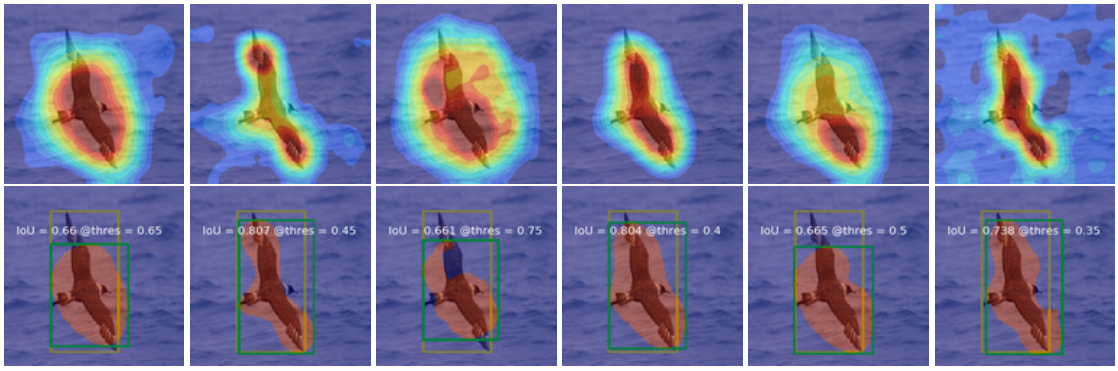


(d) Label: Hard disk

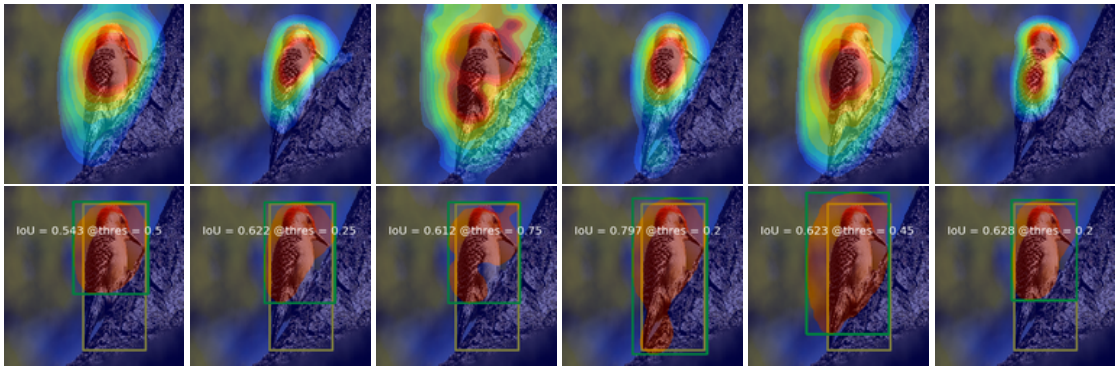
Figure 16. **ImageNet score maps.** Score maps of CAM, HaS, ACoL, SPG, ADL, CutMix from ImageNet.



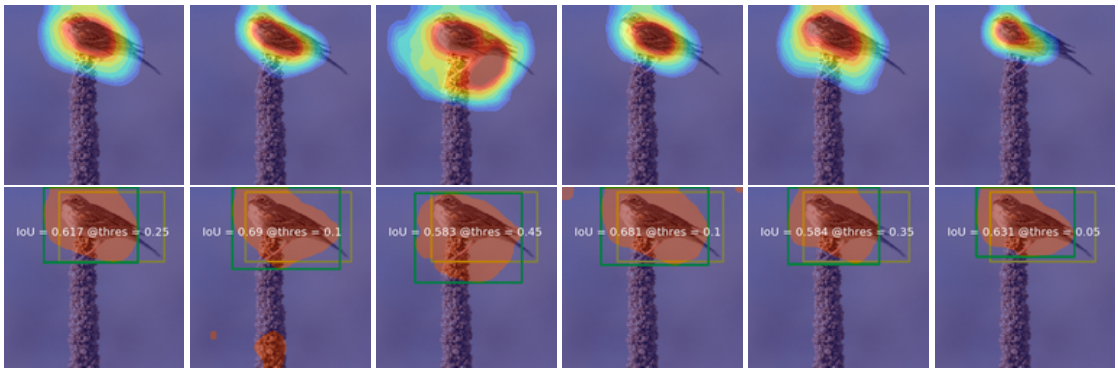
(a) Label: Common Tern



(b) Label: Pomarine Jaeger

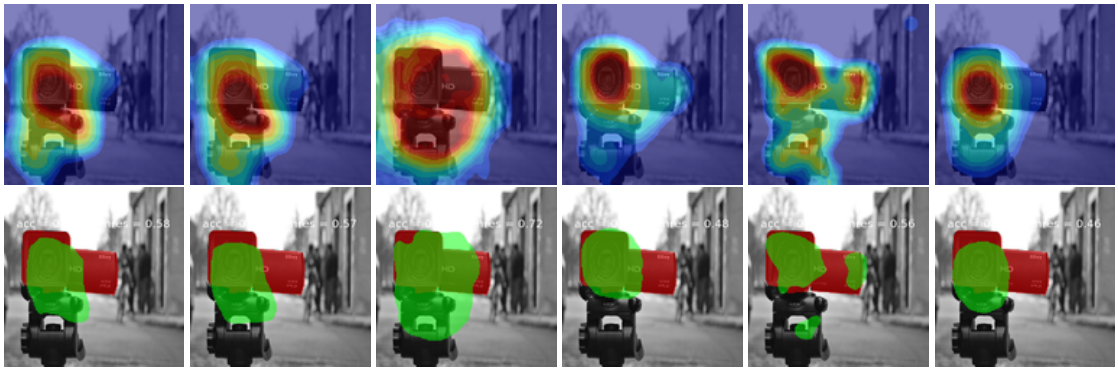


(c) Label: Red bellied Woodpecker

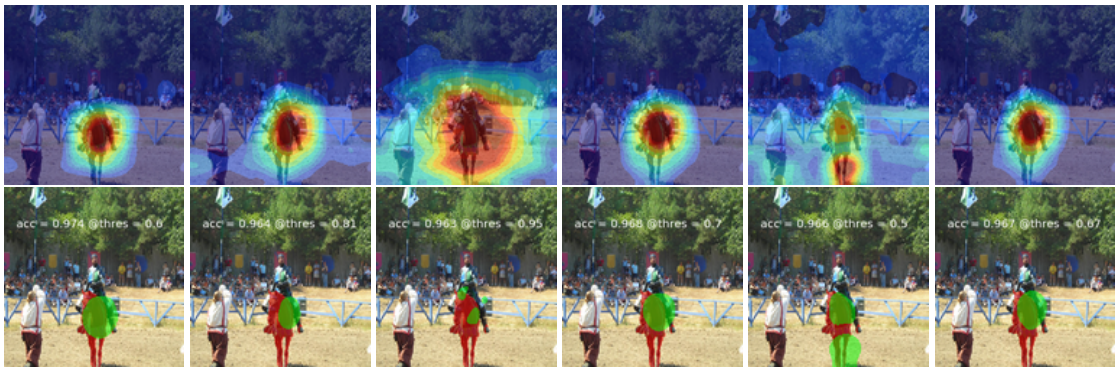


(d) Label: Vesper Sparrow

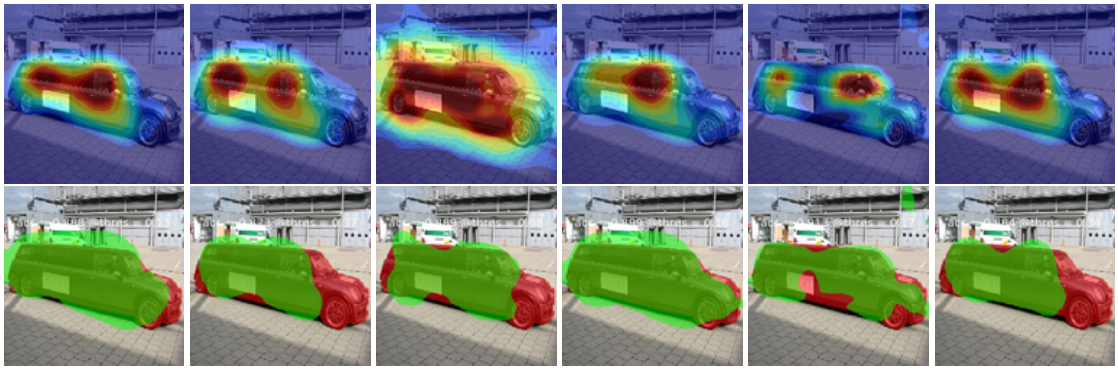
Figure 17. **CUB score maps.** Score maps of CAM, HaS, ACoL, SPG, ADL, CutMix from CUB.



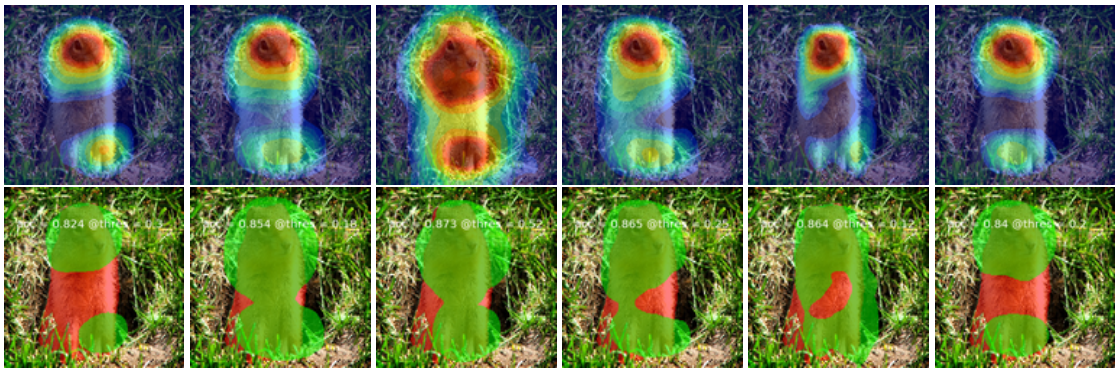
(a) Label: Camera



(b) Label: Horse

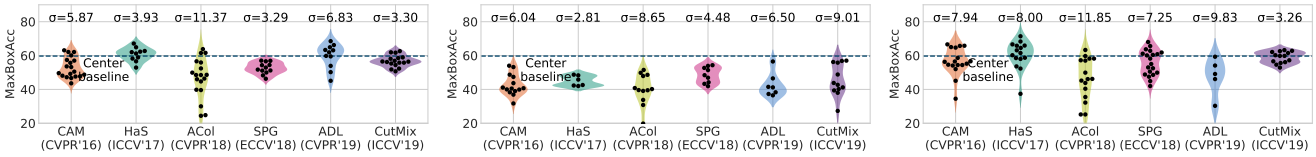


(c) Label: Limousine



(d) Label: Squirrel

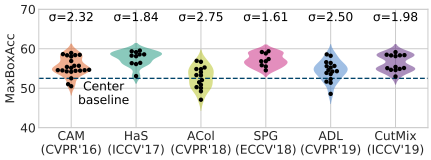
Figure 18. **OpenImages score maps.** Score maps of CAM, HaS, ACoL, SPG, ADL, CutMix from OpenImages.



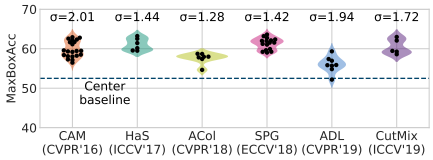
(a) CUB, VGG

(b) CUB, Inception

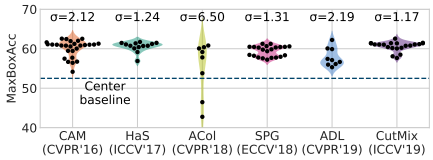
(c) CUB, ResNet



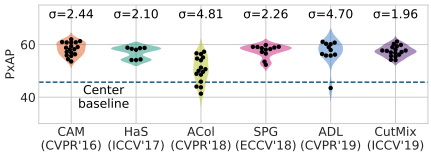
(d) ImageNet, VGG



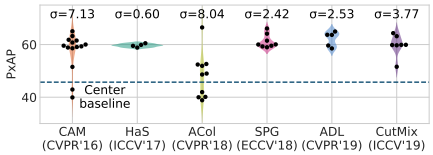
(e) ImageNet, Inception



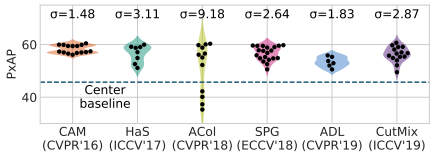
(f) ImageNet, ResNet



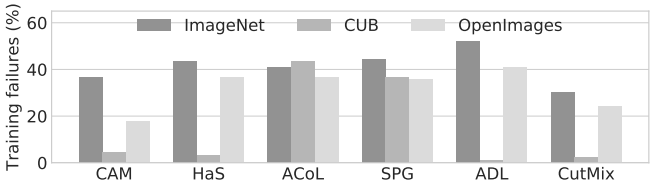
(g) OpenImages, VGG



(h) OpenImages, Inception



(i) OpenImages, ResNet



(j) Ratio of training failures

Figure 19. **All results of the 30 hyperparameter trials.** CUB, ImageNet, OpenImages performances of all 30 randomly chosen hyperparameter combinations for each method. This is the extension of Figure 6 in the main content.

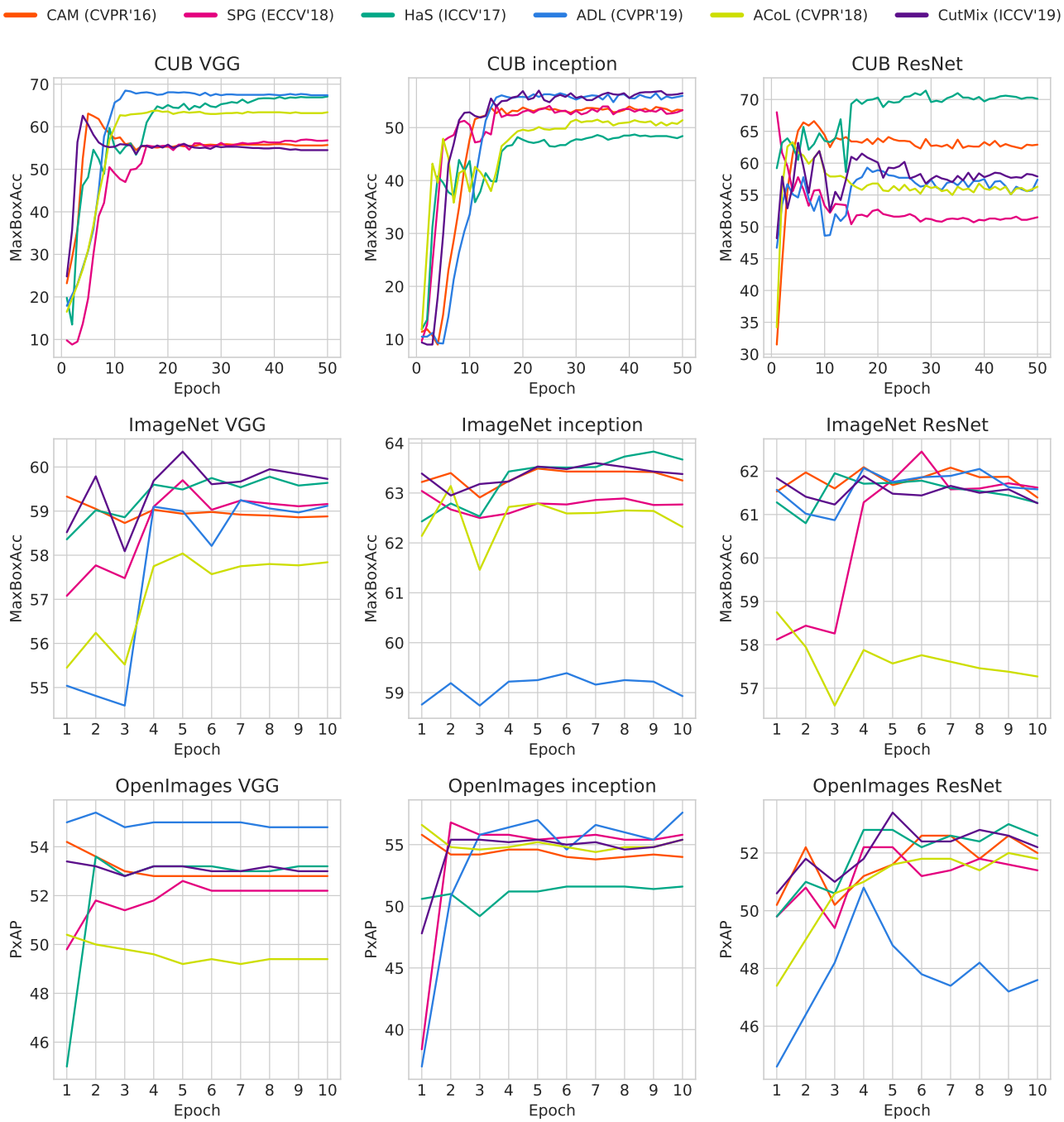


Figure 20. **Learning curves.** Results on three datasets with VGG, Inception and ResNet architectures.