Not All Labels Are Equal: Rationalizing The Labeling Costs for Training Object Detection

Ismail Elezi[1]* Zhiding Yu[2] Anima Anandkumar[2,3] Laura Leal-Taixé[1] Jose M. Alvarez[2]
[1]TUM [2]NVIDIA [3]CALTECH

Abstract

Deep neural networks have reached high accuracy on object detection but their success hinges on large amounts of labeled data. To reduce the labels dependency, various active learning strategies have been proposed, based on the confidence of the detector. However, these methods are biased towards high-performing classes and lead to acquired datasets that are not good representatives of the testing set data. In this work, we propose a unified framework for active learning, that considers both the uncertainty and the robustness of the detector, ensuring that the network performs well in all classes. Furthermore, our method leverages auto-labeling to suppress a potential distribution drift while boosting the performance of the model. Experiments on PASCAL VOC07+12 and MS-COCO show that our method consistently outperforms a wide range of active learning methods, yielding up to a 7.7% improvement in mAP, or up to 82% reduction in labeling cost. Code is available at `https://github.com/NVlabs/AL-SSL`.

1. Introduction

The performance of deep object detection networks [1,2] depends heavily on the size of the labeled dataset. Adding more labeled data helps, yet adding more data costs. Therefore, it is imperative to adopt active learning (AL) strategies to select the most informative samples in the dataset for labeling, and self and semi-supervised learning (SSL) approaches to leverage unlabeled data whenever possible.

Consistency-based Semi-Supervised Learning (SSL) methods for object detection [3] train a network to minimize the inconsistency between its predictions. However, as shown in Fig. 1, some images still give inconsistent predictions, and thus the network does not learn from them. Autolabeling uses self-learning to label high confident predictions, *i.e.* pseudo-label (PL), but, since networks are miscalibrated, they can generate wrong labels, potentially harming the training. Moreover, by targetting high-confident predictions, they ignore objects of low-performing classes.

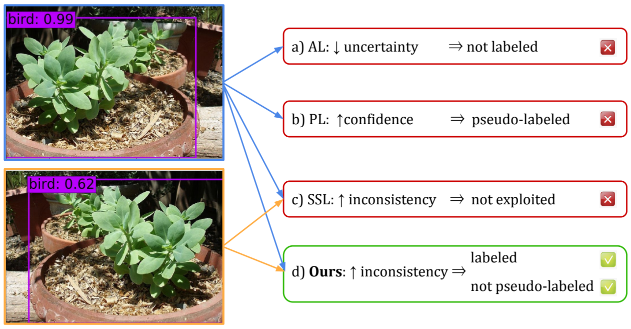

bird: 0.99
bird: 0.62
a) AL: ↓ uncertainty ⇒ not labeled ☒
b) PL: ↑confidence ⇒ pseudo-labeled ☒
c) SSL: ↑ inconsistency ⇒ not exploited ☒
d) **Ours**: ↑ inconsistency ⇒
labeled ✓
not pseudo-labeled ✓

Figure 1. An object from a low-performing class (Pottedplant), original image in blue, augmented version in orange. a) Because of its low-entropy, uncertainty-based AL methods do not label the image. b) Because of its high confidence, pseudo-label SSL methods wrongly pseudo-label the image and thus harm the training. c) Because of its high-inconsistency, consistency-based SSL methods cannot learn from it. d) Our method selects the image for labeling and prevents it from getting pseudo-labeled.

When samples cannot be pseudo-labeled, an alternative is to obtain the ground truth via manual labeling. Active learning (AL) for object detection [4, 5] is a common approach to select the most promising samples for labeling to reduce labeling costs. The selection is based on an acquisition function to assess the informativeness of an image, typically computed based on the network's uncertainty. However, the acquisition function is only meaningful if the network is already well-trained for the task, which is not always the case, especially in the early AL cycles. Even if the network performs well in most classes, significant intra-class variance can lead to a low accuracy on a particular class, see Fig. 1a. In those cases, using the network predictions to compute the acquisition function can lead to worse performance than random sampling. Furthermore, we change the dataset distribution at every AL cycle by selecting only the most uncertain (hard) samples until they no longer resemble the test distribution.

In this work, we advocate for a holistic view of the labeling problem, that is, a unified strategy to choose which samples to manually label and which samples can be automatically labeled. We start from an *uncertainty*-based AL

[*] Work performed while interning at NVIDIA.

framework and generalize its acquisition function by introducing the concept of *robustness*, which is commonly not present on AL. If a network's predictions of an image and its random augmentations, i.e., horizontal flipping, are not consistent, the image needs to be manually labeled. This simple yet effective change allows us to select informative samples for both low and high-performing classes. This is unlike classic SSL settings, where samples with inconsistent predictions would neither be labeled nor pseudo-labeled, hence, the information contained in them would not be used.

We are still left with the dataset distribution drift, for which we propose to use auto-labeling to not increase the labeling costs. For every active learning cycle, we use the previously trained network to mine easy samples, i.e., samples where the network is confident about its prediction, and use the network's own prediction as labels. Note, that easy samples are typically not used in AL cycles. Only by holistically thinking about which samples to manually label and which to auto-label can we take full advantage of the entire dataset. In summary, our **contributions** are the following:

- We propose a novel class-agnostic active learning score based on the *robustness* of the network, using a novel *inconsistency* score.
- We use auto-labeling to leverage the less informative samples, expanding the labeled dataset for free.
- We demonstrate the benefits of our method in two publicly available datasets: PASCAL VOC07+12 and MS-COCO. Compared to state-of-the-art methods [4–8], our approach yields up to a 7.7% and 7% relative mAP improvement for PASCAL-VOC and MS-COCO, respectively. Importantly, we can achieve the same performance as the baseline but reduce up to 82% of the labeling costs.

2. Related Work

Deep Active Learning (AL) for Object Detection. The traditional way of doing AL is by training an ensemble of neural networks [9] and then selecting the samples with the highest score defined by some acquisition function, i.e., entropy [10], or BALD [11]. Concurrent works [12, 13] explore a similar direction by approximating the uncertainty via Monte-Carlo dropout [14]. These approaches are compared [9], with the authors concluding that the ensemble approach reaches higher results at the cost of more computational power. Another Bayesian approach [15] trains a variational autoencoder (VAE) [16] on both real and augmented samples, and then chooses to label the samples with the highest reconstruction error. A different approach is the core-set [6] that chooses to label a set of points such that a model trained over the selected subset is competitive for the remaining data points. In the work of [17] the authors use the inconsistency method of [18] to do AL. A similar approach is that of [19] where the authors use an inconsistency score that is based on the difference of predictions based on different checkpoints of the network. Similar to each other, and different to us, these methods are designed for classification-based AL, do not explore the best usage of inconsistency, its unification with entropy or the use of pseudo-labeling. Other methods [20, 21] combine AL with SSL, however they are different to our work considering that they use label propagation instead of inconsistency, do not use auto-labeling to balance active learning, and are focused solely in the problem of image classification

Recently, several methods have been adapted specifically for the task of object detection [6, 7, 22–25], some of which are based on the core-set approaches where the diversity of the training examples is taken into account. However, the state-of-the-art approaches are based on the uncertainty [4, 5, 8, 26]. The work of [26] consists of an ensemble of object detectors that provide bounding boxes and probabilities for each class of interest. Then, a scoring function is used to obtain a single value representing the informativeness of each unlabeled image. Similar to that is the work of [8] where the authors compute the instance-based uncertainty. Another work [4] gives an elegant solution, reaching promising results compared with other single-model methods. The authors train a network in the task of detection while learning to predict the final loss. In the sample acquisition stage, samples with the highest prediction loss are considered the most interesting ones and are chosen to be labeled. In the state-of-the-art approach [5], authors define the aleatoric and epistemic uncertainty, in both class and bounding box level, and use the combined score to determine the images that need labeling. Our work is related but different from the above-mentioned works. Similarly, we consider the uncertainty of the detector as part of the solution. Unlike them, we find that the robustness of the detector is even more reliable as an acquisition function, especially for the low-performing classes. We then unify these two scores to reach high performance in the majority of classes.

Deep Semi-Supervised Learning (SSL) for Object Detection is a deep learning approach that combines a small amount of labeled data with a large amount of unlabeled data during neural network training. Unlike in AL, where the unlabeled data is used only during the acquiring stage, in SSL, the unlabeled data are used during the training. Several methods have shown excellent results [27–30] by casting the problem of semi-supervised learning as a regularization problem, in effect adding a new loss for the unlabeled samples. Follow-up works significantly improved the performance of SSL in object classification [18, 31–35].

While going from semi-supervised image classification to semi-supervised object detection is challenging, some promising directions are given in recent works [3, 36, 37]

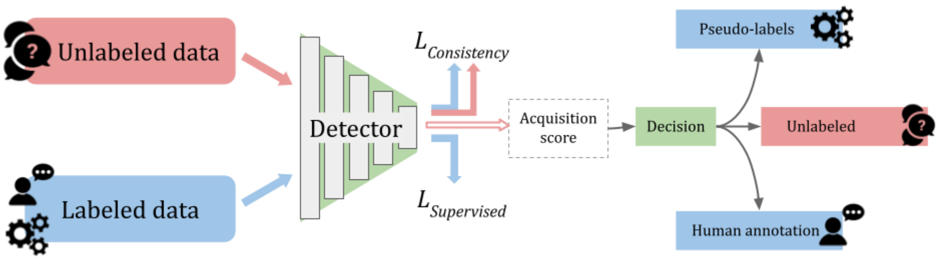


Figure 2. Overview of our method. We first train the network in a semi-supervised manner. During active learning, for each image, we use the network to compute the acquisition function and based on it decide if we actively label the image, if we pseudo-label it, or if we only use it as part of the unlabeled data for the next training cycle.

where the authors develop loss functions that minimize the inconsistency between images and their augmented version. The methods significantly improved the mAP score in Pascal VOC dataset [38]. Our work is inspired by [3] but we instead develop an acquisition function that computes the inconsistency between an image and an augmented version of it, and show that such a score is more reliable than the uncertainty, especially for low-performing classes. Furthermore, we add a pseudo-labeling module that labels the easy images for free. Together with our developed AL method, it ensures that our acquired dataset is a good representative of the original dataset, in turn, improving the results.

3. Method

Let D be a dataset divided into a labeled set L and a pool of unlabeled data U. We describe our acquisition function for AL in Sec. 3.1. This consists of mining a subset of samples from the pool of unlabeled data U and transferring them to the labeled set L, incurring a labeling cost. However, arbitrarily augmenting the set L with only hard samples creates a distribution drift in our training data. Hence, we propose to include in training the easy samples, i.e., objects for which the network's confidence is high, by using pseudo-labeling (Sec. 3.2). Finally, we describe the training procedure in Sec. 3.3. We show a high-level description of the method in Fig. 2.

Notation. Let $\boldsymbol{\Delta}$ be the object predictions for an image, and let $\boldsymbol{\Delta}_i$ be its i-th object prediction. $\boldsymbol{\Delta}_i$ consists of the bounding box $\boldsymbol{b}_i$ and $\boldsymbol{c}_i$ represents the probability distribution after the softmax layer of the neural network. We denote the p-th category of the distribution with $\boldsymbol{c}_i^p$. The bounding box $\boldsymbol{b}_i$ consists of the *displacement of the center* and *scale coefficients*, represented by the tuple $[\boldsymbol{\delta x}, \boldsymbol{\delta y}, \boldsymbol{w}, \boldsymbol{h}]$. Given a weakly augmented version of the original image, e.g., by doing a horizontal flip, we define $\hat{\boldsymbol{\Delta}}$ to be the set of its object predictions, and $\hat{\boldsymbol{\Delta}}_{\boldsymbol{i}}$ consisting of the bounding box $\hat{\boldsymbol{b}}_i$ and $\hat{\boldsymbol{c}}_i$, its i-th prediction.

3.1. Inconsistency-based AL

Most AL methods use some measure of uncertainty, e.g., the entropy, to compute the acquisition function. A prediction that has a high entropy suggests that the object is highly dissimilar to the images the network is trained on. Thus, if labeled, it will provide different information to the ones we have. However, we empirically find that using only an *uncertainty*-based acquisition function is not an ideal solution, especially for images coming from low-performing classes. As we show in the experiments, if the network's predictions for a class are incorrect, they are also unreliable to compute the acquisition function.

Inconsistency-based acquisition function. To solve this issue, we propose a *robustness*-based score for AL based on the *consistency* between an image and its augmented version. If the predictions from an image and its augmented version are very similar, then we say that the network is robust for that image. On the other side, images where the network is inconsistent provide information that has not been captured in the training process and are prime candidates to be labeled. By focusing on robustness, the method is class-agnostic and performs well in most classes.

To measure the robustness of the network, we first define the inconsistency acquisition function $\mathcal{L}_{con_C}$. To do so, we feed the image and an augmented version of it to the detector. In our case, we use horizontal flip as augmentation. Given the sets of predictions for the original and augmented image, we first need to match the predictions $\boldsymbol{\Delta}$ with $\hat{\boldsymbol{\Delta}}$. We do so by computing their intersection over union (IoU):

$$\boldsymbol{\Delta}_i' = \text{argmax}_{\boldsymbol{b}_i \in \{\boldsymbol{b}\}} IoU(\boldsymbol{b}_i, \hat{\boldsymbol{b}}_i). \qquad (1)$$

For each matched pair, $\boldsymbol{\Delta}_i'$ and $\hat{\boldsymbol{\Delta}}_{\boldsymbol{i}}$, we define their inconsistency as:

$$\mathcal{L}_{con_C}(\boldsymbol{c}_i', \hat{\boldsymbol{c}}_i) = \frac{1}{2}[KL(\boldsymbol{c}_i', \hat{\boldsymbol{c}}_i) + KL(\hat{\boldsymbol{c}}_i, \boldsymbol{c}_i')], \qquad (2)$$

where KL represents the Kullback-Leibler divergence. The higher the inconsistency, the more informative the sample is for training and therefore potentially worth labeling.

Aggregating object scores for image selection. Given $\mathcal{L}_{con_C}$, the inconsistency for each object prediction in an image, we define the inconsistency of an image by aggregating the scores over $\boldsymbol{\Delta}$. Specifically, we first apply non-maximum suppression over its predictions, and then, define its inconsistency as:

$$I(\boldsymbol{\Delta}) = max_i\{\mathcal{L}_{con_C}(\boldsymbol{c}_i', \hat{\boldsymbol{c}}_i)\}. \qquad (3)$$

Similarly, we define the uncertainty of the image as:

$$H(\boldsymbol{\Delta}) = max_i\{H(\boldsymbol{c}_i)\}, \tag{4}$$

where $H(\boldsymbol{c}_i)$ represents the entropy over distribution $\boldsymbol{c}_i$. The intuition behind using the *maximum* score instead of some other score, such as the *average*, is that labeling an image that has at least one *difficult* object, independently of the number of *easy* objects is beneficial because of the difficult object. Considering that the inconsistency and the uncertainty scores are on different scales, we unify them by multiplying them. For $\boldsymbol{\Delta}$, we formulate our unified acquisition score as:

$$A(\boldsymbol{\Delta}) = H(\boldsymbol{\Delta}) \times I(\boldsymbol{\Delta}). \tag{5}$$

Having scored each image in U, we then sort all the images based on their acquisition score and select to label the N/T images with the highest score, where N corresponds to the acquisition budget, and T corresponds to the number of active learning steps. Note that we annotate every bounding box that belongs to a *selected image* regardless of whether the box has a high score or not. We repeat this procedure for T active learning cycles.

3.2. Pseudo-labeling to prevent distribution drift

The active learning pipeline described above targets the most informative (hard) samples, ignoring the confident samples. We argue that the network should see some representative easy samples in order to ensure that no distribution drift happens. At the same time, we want to avoid labeling confident samples to not spend labeling resources. Hence, we propose to use pseudo-labeling, where the network trained in the previous AL cycle provides pseudo-labels for the network that is being trained on this cycle. We pseudo-label an object if the network is confident, the confidence is above some threshold τ:

$$\hat{y}_i^p = \begin{cases} 1, & \text{if } p = argmax(\boldsymbol{c}_i) \text{ and } \boldsymbol{c}_i^p \geq \tau \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

We then use the one-hot pseudo-labels as ground truth for the training of the current network. We note that in an image, the network might be confident for some predicted bounding box, and not confident for the others. As a toy example, given an image containing a cat and a dog, the network might be confident for the cat and pseudo-label it, but not confident for the dog. If we do not consider this, the standard loss functions will penalize the predictions of the network for the region of the image which contains the dog. However, because the region is unlabeled, the ground truth of that object will be set to background, in turn giving a high loss even if the network makes accurate prediction (as dog). In the next section, we describe how to fix this issue.

3.3. Deep Object Detection Training

In this section, we describe the different losses used in our unified framework for training the deep object detector. First, we describe the multibox, consistency, and pseudo-labeling losses and finally the overall training loss.

Multibox loss for labeled samples. For the labeled images, the network is trained with the standard MultiBox loss for class predictions, and a smooth $\mathcal{L}1$ loss for bounding box predictions. Given the network's class predictions $\boldsymbol{c}$ and an indicator $\boldsymbol{y_{ij}^p} = \{0, 1\}$ for matching the i-th box to its corresponding j-th ground truth box of category p, the MultiBox loss is defined as [2]:

$$\mathcal{L}_{conf}(\boldsymbol{c}, \boldsymbol{y}) = -\sum_{i \in Pos} \sum_{p=1}^{|\text{classes}|} \boldsymbol{y}_{ij}^p log(\boldsymbol{c}_i^p) - \sum_{i \in Neg} log(\boldsymbol{c}_i^0), \tag{7}$$

where Pos defines positive bounding boxes (containing objects), Neg defines bounding boxes of class *background*.

Consistency loss for unlabeled samples. Our approach leverages the inconsistency of the detector in the acquisition function. Intuitively, if the detector has high inconsistency in an image, it can not learn from it in a self-supervised manner, and the only way to learn from that image is to label it during the AL cycle. During training, we need to guide the detector to provide consistent predictions. To this end, we mirror the active learning procedure and feed an image and its augmented version to the detector using horizontal flips. After matching the predictions, as described in Eq. 1, we use the class acquisition function, $\mathcal{L}_{con_C}$, as the loss function for class inconsistency. To stabilize the training, we compute the localization inconsistency loss as [3]:

$$\begin{aligned} \mathcal{L}_{con_L}(\boldsymbol{b}_i', \hat{\boldsymbol{b}}_i) =& \frac{1}{4}(||\boldsymbol{\delta x}_i' - (-\hat{\boldsymbol{\delta x}}_i)||^2 + ||\boldsymbol{y0}_i' - \hat{\boldsymbol{y0}}_i||^2 + \\ & ||\boldsymbol{w}_i' - \hat{\boldsymbol{w}}_i||^2 + ||\boldsymbol{h}_i' - \hat{\boldsymbol{h}}_i||^2), \end{aligned} \tag{8}$$

where, as we use horizontal flipping, we apply the negation on the displacement of the center $\hat{\boldsymbol{\delta x}}_i$.

We compute the total consistency loss by averaging the losses from all matched pairs of predictions:

$$\mathcal{L}_{con} = \mathbb{E}[\mathcal{L}_{con_C}(\boldsymbol{c}', \hat{\boldsymbol{c}})] + \mathbb{E}[\mathcal{L}_{con_L}(\boldsymbol{b}', \hat{\boldsymbol{b}})]. \tag{9}$$

Pseudo-labeling loss. Our approach only pseudo-labels those objects in an image where the detector is highly confident, leaving the rest of the image unlabeled. Using the loss described in Eq. 7 would cause problems for those predictions in the regions where there are no pseudo-labels as they would be considered false positives. We thus modify the MultiBox Loss as:

$$\begin{aligned} \mathcal{L}_{conf}(\boldsymbol{c}, \boldsymbol{y}, \hat{\boldsymbol{y}}) = &- \sum_{i \in Pos} \sum_{p=1}^{|\text{classes}|} \boldsymbol{y}_{ij}^p log(\boldsymbol{c}_i^p) \\ &- \sum_{i \in Neg} log(\boldsymbol{c}_i^0) - \sum_{i \in \hat{Pos}} \sum_{p=1}^{|\text{classes}|} \hat{\boldsymbol{y}}_{ij}^p log(\boldsymbol{c}_i^p), \end{aligned} \tag{10}$$

where $\hat{y}$ and $\hat{Pos}$ represent the indicator and the positive bounding boxes for the pseudo-labels.

Overall Training loss. Finally, to train the deep detection network, we aggregate the multibox, $\mathcal{L}1$ and consistency losses as:

$$\mathcal{L}_{total} = \mathcal{L}_{conf} + \mathcal{L}_{con} + \mathcal{L}1, \tag{11}$$

where $\mathcal{L}_{con}$ is used in all samples, while $\mathcal{L}_{conf}$ and $\mathcal{L}1$, the smooth L_1 for bounding boxes, are used in the labeled and pseudo-labeled samples.

4. Experiments

In this section, we demonstrate the effectiveness of our approach to improve the performance of object detection. For all experiments, we report mean average precision @0.5 (*mAP*) as main metric, and use two public datasets: PASCAL VOC07+12 (VOC07+12) [38] and MS-COCO train2014 [39]. VOC07+12 consists of $16,551$ images for training, and $4,952$ testing images taken from VOC07 test-set. MS-COCO consists of $83K$ images for training, and *valset2017* contains $5,000$ images for testing.

Following [4, 5], on VOC07+12 we start by randomly sampling $2,000$ images. On the larger MS-COCO, we start by randomly sampling $5,000$ images. We perform 5 active learning cycles, and in each cycle, we choose $1,000$ images to label. To ensure that the network does not diverge, we define each mini-batch to have half the images labeled. We set the confidence threshold for the pseudo-label threshold to $\tau = 0.99$ for VOC07+12 and $\tau = 0.75$ for MS-COCO, based on the results of the zeroth active learning step. We set the IoU threshold at ≥ 0.5.

For a fair comparison with [4–7], we use the Single-Shot Detector 300 (SSD300) [2] based on a VGG [40] backbone for all our experiments. We train the model for $120,000$ iterations using SGD with momentum. We set the initial learning rate to 0.001 and divide it by 10 after $80,000$ and $100,000$ iterations, respectively. We use batches of size 32 and a constant L2 regularization parameter set to 0.0005. We use the same model, hyperparameters, and the same public implementation[1]. We train all networks using four NVIDIA V100 GPUs. In all experiments, we train three independent networks using the same initial split of randomly sampled images and report the mean. We give the numbers of the mean and standard deviation in the supplementary.

4.1. Main results: Comparison with other methods

We compare our method with two baselines, random and entropy sampling, in addition to five state-of-the-art single-model methods: Coreset [6], Learning Loss [4], CDAL-AL [7], MI-ALD [8] and PM [5]. The last method uses a mixture of SSD, thus adds extra parameters. We also compare with two multimodel approaches, MC-dropout [12] and ensemble-based [9] active learning (consisting of three neural networks). Finally, we compare to the consistency-based SSL method and to a pseudo-labeling method.

We present the comparisons with AL methods for VOC07+12 in Fig. 3a. We observe that in the first active learning cycle, our method has a relative improvement over the random baseline by 10.5%, and over 8.2% compared to the best overall active learning method [8]. We see that the performance improvement of our method is maintained in the other active learning cycles. In the last one, where we use $7,000$ samples, $5,000$ of which are actively labeled, our method outperforms the random baseline by 9.1% and the best existing active learning methods by more than 2.8% [5]. Multi-model active learning networks, namely, ensemble [9] or MC-dropout [12] outperform single models at the cost of longer training and active learning time, and in the case of the ensemble has 3 times more training parameters. Nonetheless, our proposed single model still reaches better results than multi-model methods, outperforming the ensembles by 8% in the first AL cycle, and 1.8% in the last cycle. In Fig. 3b. we compare the results of our method, with the two semi-supervised learning methods. In the first AL cycle, our method outperforms the consistency-based SSL by 5.6%, and the pseudo-labeling method by 2%. In the last cycle, we outperform the consistency method by 3.4% and the pseudo-labeling method by more than 3%.

For MS-COCO, in Fig. 4a-b, we observe that in the first active learning cycle, our method outperforms the random baseline by 5.8%, the best-performing AL method by 2.7% [5], the semi-supervised method by 5.4%, and the ensembles by 5%. In the second cycle, our approach outperforms all the other methods, including PM [5], by almost 4% or more. We observe that this difference is maintained in the other cycles, including the last active learning cycle where our method outperforms the semi-supervised method, multi-model methods [9, 12] and the best AL method [5] by 1.6%.

4.1.1 Ablation study.

The effect of active learning and auto-labeling. We now analyze every module of our method, in order to disentangle the contribution coming from them. In Fig. 3c, we present the performance comparison of the semi-supervised model on VOC07+12 under different acquisition functions (random, entropy, inconsistency) and two instances of our unified method: with and without pseudo-labels. We see that on the first active learning cycle, neither entropy nor inconsistency significantly outperforms the results of random sampling. However, we immediately see a significant effect, i.e., a relative improvement over 0.9% using entropy

[1] https://github.com/amdegroot/ssd.pytorch

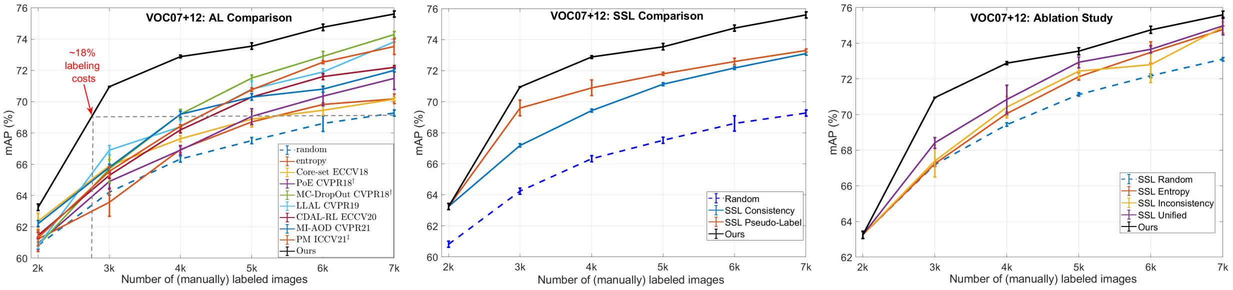


Figure 3. **VOC07+12**. **Left**: Comparison to state-of-the-art active learning methods; **Middle**: Comparison to the two SSL methods used in this work when they do not use AL; **Right**: Ablation study on the effect of entropy, inconsistency, unified score without pseudo-labeling, and our method. † denotes ensemble method; ‡ denotes mixture of SSD. Ours uses also unlabeled data during training.

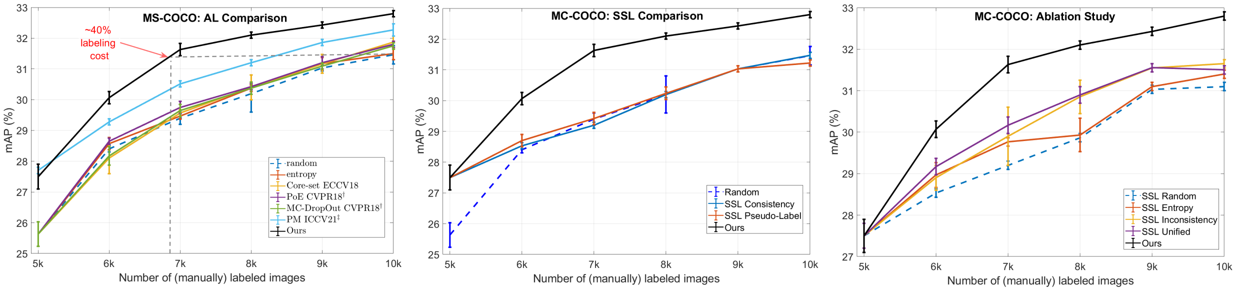


Figure 4. **MS-COCO**. **Left**: Comparison to state-of-the-art active learning methods; **Middle**: Comparison to the two SSL methods used in this work when they do not use AL; **Right**: Ablation study on the effect of entropy, inconsistency, unified score without pseudo-labeling, and our method. † denotes ensemble method; ‡ denotes mixture of SSD. Ours uses also unlabeled data during training.

and 1.4% using inconsistency, in the second active learning cycle. We see that the increase in performance gets bigger in the next AL cycle and in the fifth active learning cycle, the performance gain from entropy is 2.3% and from inconsistency is 2.4%.

We then show the results of our unified acquisition function. In the first active learning cycle, we immediately see a significant improvement in performance. While entropy ($67.24mAP$) and inconsistency (67.39mAP) reach an insignificant improvement over random sampling ($67.19mAP$), our acquisition function reaches $68.40mAP$, which is 1.5% better than random sampling. The performance improvement gets larger in the next cycles: 2% in the second cycle, 2.5% in the third cycle, and a peak improvement of 2.6% in the fifth active learning cycle. In all cases, our proposed score outperforms both active learning methods that are based on a single acquisition function.

We further study the effect of pseudo-labeling in our framework. In Fig. 3c, we observe that on the first active learning cycle, adding pseudo-labels comes with an immediate boost, improving the results by 3.7% compared to the already well-performing acquisition function for semi-supervised learning (5.3% better than the semi-supervised method that uses random sampling). We further observe that on the second cycle, it gives an improvement of 2.9% compared to using only our acquisition function (4.7% better than the semi-supervised method that uses random sampling). The Pseudo-labeling module continues to give a boost in performance in all the following AL steps.

In Fig. 4c, we provide a similar ablation study for MS-COCO. We again observe that the unified score outperforms both the entropy and inconsistency scores in isolation. However, unlike in VOC07+12, we observe that using only the entropy, the improvement is marginal over random sampling. On the other hand, we observe that inconsistency works significantly better than random sampling and entropy (we provide an explanation in the next section). We further observe the effect of pseudo-labels. We see that in the first AL cycle, adding pseudo-labels boosts the performance by 3.1% and the performance boost is maintained up to the last cycle. This is very different from the results of pseudo-labeling alone, see Fig. 4b, where the performance gain is marginal. In other words, pseudo-labeling in isolation does not work well. However, pseudo-labeling complemented with the unified score reaches high results.

Acquisition functions. We now focus on analyzing the effects of aggregating the two acquisition functions. In Fig. 5, we check the performance of every individual class in the zeroth and the last AL cycle in VOC07+12 dataset. We then focus on three best-performing classes (*"Train"*, *"Car"*, and *"Horse"*) and three worst-performing classes (*"Bottle"*, *"Pottedplant"*, and *"Chair"*). A first observation is that for the best-performing classes, entropy-based AL, on average, tends to outperform inconsistency-based AL.

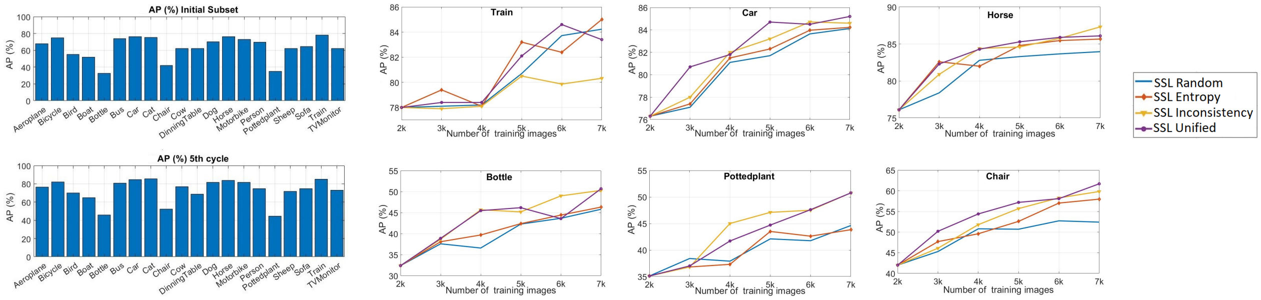
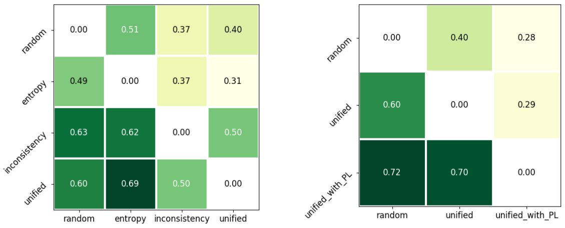


Figure 5. **VOC07+12.** In the bar plots we show the accuracy per class using random sampling in the zeroth and last cycle. We present the results of each AL method for the three best-performing (*"Train"*, *"Car"*, and *"Horse"*) and worst-performing (*"Bottle"*, *"Pottedplant"*, and *"Chair"*) classes.



(a) (b)

Figure 6. **MS-COCO.** a) The percentage of classes where one acquisition function outperforms another; b) The percentage of classes where our unified acquisition function outperforms random with and without pseudo-labels. Example: taking the entry "unified" in the y-axis, and "entropy" in the x-axis, we get the value 0.69 which means that "unified" acquisition function outperforms the "entropy" acquisition function in 69% of classes.

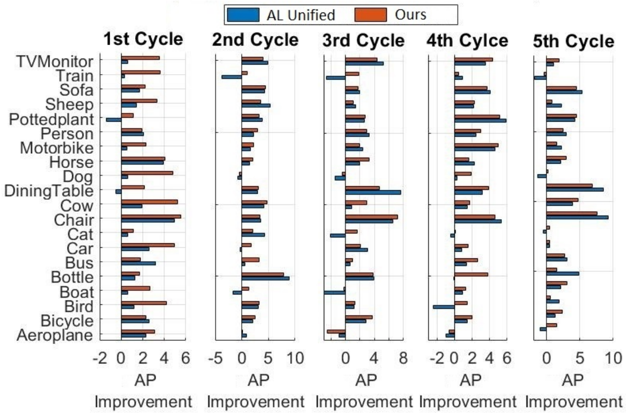


Figure 7. **VOC07+12:** Effect of pseudo-labels compared to AL alone for every class.

On the other hand, we see that inconsistency-based AL outperforms the entropy-based AL by a significant margin in all three worst-performing classes. While the entropy-based AL on average seems to only slightly outperform random sampling, the inconsistency-based AL gives a relative performance gain of up to 24%, 14% and 18% in classes *"Bottle"*, *"Pottedplant"*, and *"Chair"*. Intuitively, one can argue that this phenomenon is to be expected. The fact that the network does a poor job on its predictions leads to its class predictions being unreliable for any uncertainty-based AL method. At the same time, a more general acquisition function dependent only on the robustness of the network is better suited for low-performing classes. Finally, we show that our acquisition function reaches the best overall results.

Because of the massive number of classes, we aggregate the results on MS-COCO dataset. In Fig. 6a, we show the percentage of classes where one acquisition function outperforms the other. We see that inconsistency outperforms entropy in 62% of the classes, and our unified score outperforms entropy in 60% of the classes. This explains why in MS-COCO, which contains many more challenging classes, the robustness-based acquisition scores significantly outperform the uncertainty-based acquisition score. We provide the results for each AL cycle in the supplementary.

Do we need SSL training? The inconsistency acquisition function computes the robustness of the network for each image in the acquisition pool. If the network is inconsistent in an image, despite that it was trained to minimize the inconsistency of that image, then that image provides information that was not captured by the SSL. We now check what happens if the network is not trained in SSL, thus it has never seen the images in the labeling pool. In Tab. 1a, we show the results of inconsistency acquisition function for a network trained with and without consistency loss. As expected, the results with the SSL loss significantly outperform the results of the fully-supervised baseline. Interestingly, the results of inconsistency AL are not better than those of random. Clearly, in order to be able to exploit the robustness information in samples, the network needs to try minimizing their inconsistency during training.

PL performance boost per class. We now study if the pseudo-labels help only some particular classes, or if they help in all classes. We start the analysis on VOC07+12 dataset. In Fig. 7, we plot the performance gain coming from the module for each class and compare it with the performance gain from AL alone, and random sampling. In the first AL cycle, we see that pseudo-labels improve over random sampling in all 20 classes, with AL alone giving a negative boost in two classes: *"Pottedplant"* and *"DiningTable"*. We also find out that pseudo-labels give a boost over AL alone in all three worst-performing classes (*"Bottle"*, *"Pottedplant"*, and *"Chair"*). We see a similar pattern in the other cycles. In the second cycle, the pseudo-labels module improves over AL alone in 14 classes and gives a negative boost in only one class (*"Dog"*) compared to AL alone that gives a negative boost in four classes. In the third cycle, the pseudo-labels module improves over AL alone in 11 classes and gives a negative boost in two classes (compared to AL in five classes). In the fourth cycle, the pseudo-labels module improves over AL alone in 15 classes and gives a negative boost in one class (compared to AL in three classes). Finally, in the fifth AL cycle, the pseudo-labels module improves over AL alone in 10 classes with class *"Car"* being a tie and gives a negative boost in only one class, compared to AL in three classes. We thus conclude that by focusing only on the *hard* samples, AL alone harms the performance on several classes. However, adding pseudo-labels diminishes this effect, making the network much more robust and thus preventing a dataset drift.

We do a similar analysis on the larger MS-COCO, by aggregating the results, showing the results in Fig. 6b. While our unified acquisition function outperforms the random acquisition in 60% of the classes, adding pseudo-labels increases the number of classes this happens to 72%, showing the usefulness of pseudo-labels. We provide the experiment results for each AL cycle in the supplementary material.

Ratio of pseudo-labels. We now study the effect of increasing the number of pseudo-labels by allowing more noisy pseudo-labels. To do so, we lower the pseudo-labeling threshold τ from 0.99 to 0.9 and 0.5. We present the results in Fig. 8a. We observe that we reach the best overall results by using an extremely high threshold $\tau = 0.99$. Decreasing τ to 0.9 and thus allowing more pseudo-labels harms the performance. Further decreasing it to 0.5, hence allowing many more pseudo-labels, actually harms the entire training. We thus conclude that we need to be selective in the choice of pseudo-labels.

To understand why the performance improvement of the network trained with the pseudo-labels module diminishes in the later active learning cycles, we study the pseudo-labels gain as a function of the pseudo-labels ratio to the entire labels. As we show in Fig. 8b, in the first active learning cycle where the pseudo-labels bring a maximum gain (3.7%), roughly half of the labels are pseudo-labels. With the decrease of the number of pseudo-labels, we see a tendency for the gain to lower. Our intuition is that when the number of pseudo-labels is high, despite them being noisy, they still help the training process. An interesting fact is that while the total number of pseudo-labels decreases for each AL cycle (because $1,000$ images are removed from the labeling pool) the number of pseudo-labels for an image increases with each cycle, starting from 0.58 in the first cycle, to 0.81 in the last one. Thus, the network becomes better at selecting pseudo-labels during AL cycles.

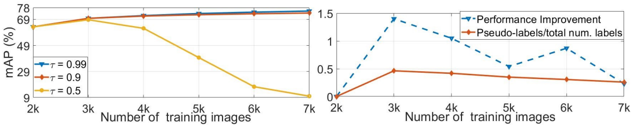
Cycle	Random	No SSL	SSL
1	64.23	63.26	**67.39**
2	66.33	65.79	**70.42**
3	67.51	67.16	**72.43**
4	68.60	68.65	**72.80**
5	69.27	70.33	**74.90**

(a)

Cycle	0.5	0.9	0.99
1	80.04	91.13	**96.68**
2	84.00	92.21	**96.01**
3	86.00	93.32	**95.80**
4	87.55	93.41	**95.61**
5	90.05	94.64	**95.57**

(b)

Table 1. **VOC07+12.** a) The effect of training on SSL. b) PL correctness with τ.



(a) (b)

Figure 8. **VOC07+12. Left**: Accuracy as a function of τ for selecting pseudo-labels. **Right**: Accuracy improvement with respect to the pseudo-labels ratio to the entire labels.

On MS-COCO, where the number of images that can be potentially pseudo-labeled is higher ($78K$ compared to $14,651$ in VOC0712), the performance gain from the pseudo-labels module does not diminish. This is because the ratio of pseudo-labels in all cycles remains high.

Pseudo-labels noise. Deep neural networks are overconfident and not well-calibrated, thus inducing pseudo-labeling errors. We consider a pseudo-label correct if the predicted class is the same as the ground truth and intersection over union with the ground truth object is over 0.5. We provide the correctness of the pseudo-labels given by our model in Tab. 1b. We see that setting the pseudo-labeling threshold to 0.99 leads to 3.7% pseudo-labeling errors. This percentage remains constant over different AL cycles suggesting the network is robust to this amount of noise.

5. Conclusions

In this work, we developed a framework that reduces the labeling costs for object detection. Our framework consists of a novel acquisition function based on the *robustness* of the neural network with respect to its predictions, and an auto-labeling scheme that prevents a potential distribution drift. In this way, our unified model chooses to actively label the most informative samples in the dataset, while it pseudo-labels the easiest samples. This allows us to use the majority of the dataset in a supervised manner while reducing the labeling costs. As we showed in the experiments, we can reduce the labeling costs by up to 82% in order to reach the same results as a fully-supervised baseline.

References

[1] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1

[2] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1, 4, 5

[3] Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. In *NeurIPS*, 2019. 1, 2, 3, 4

[4] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *CVPR*, 2019. 1, 2, 5

[5] Jiwoong Choi, Ismail Elezi, Hyuk-Jae Lee, Clément Farabet, and Jose M. Alvarez. Active learning for deep object detection via probabilistic modeling. In *ICCV*, 2021. 1, 2, 5

[6] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018. 2, 5

[7] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *ECCV*, 2020. 2, 5

[8] Tianning Yuan, Fang Wan, Mengying Fu, Jianzhuang Liu, Songcen Xu, Xiangyang Ji, and Qixiang Ye. Multiple instance active learning for object detection. In *CVPR*, 2021. 2, 5

[9] William H. Beluch, Tim Genewein, Andreas Nürnberger, and Jan M. Köhler. The power of ensembles for active learning in image classification. In *CVPR*, 2018. 2, 5

[10] Claude E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 2001. 2

[11] Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745, 2011. 2

[12] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *ICML*, 2017. 2, 5

[13] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *NeurIPS*, 2019. 2

[14] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[15] Toan Tran, Thanh-Toan Do, Ian D. Reid, and Gustavo Carneiro. Bayesian generative active deep learning. In *ICML*, 2019. 2

[16] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 2

[17] Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan Ömer Arik, Larry S. Davis, and Tomas Pfister. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In *ECCV*, 2020. 2

[18] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 2

[19] Siyu Huang, Tianyang Wang, Haoyi Xiong, Jun Huan, and Dejing Dou. Semi-supervised active learning with temporal output discrepancy. In *ICCV*, 2021. 2

[20] Jiannan Guo, Haochen Shi, Yangyang Kang, Kun Kuang, Siliang Tang, Zhuoren Jiang, Changlong Sun, Fei Wu, and Yueting Zhuang. Semi-supervised active learning for semi-supervised models: Exploit adversarial examples with graph-based virtual labels. In *ICCV*, 2021. 2

[21] Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier. Rethinking deep active learning: Using unlabeled data at model training. In *ICPR*, 2020. 2

[22] Hamed Habibi Aghdam, Abel Gonzalez-Garcia, Antonio M. López, and Joost van de Weijer. Active learning for deep detection neural networks. In *ICCV*, 2019. 2

[23] Chieh-Chi Kao, Teng-Yok Lee, Pradeep Sen, and Ming-Yu Liu. Localization-aware active learning for object detection. In *ACCV*, 2018. 2

[24] Soumya Roy, Asim Unmesh, and Vinay P. Namboodiri. Deep active learning for object detection. In *BMVC*, 2018. 2

[25] Sai Vikas Desai, Akshay Chandra Lagandula, Wei Guo, Seishi Ninomiya, and Vineeth N. Balasubramanian. An adaptive supervision framework for active learning in object detection. In *BMVC*, 2019. 2

[26] Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecky, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M Alvarez. Scalable active learning for object detection. In *IV*, 2020. 2

[27] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *ICMLW*, 2013. 2

[28] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017. 2

[29] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 2

[30] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *tPAMI*, 2019. 2

[31] Avital Oliver, Augustus Odena, Colin Raffel, Ekin Dogus Cubuk, and Ian J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, 2018. 2

[32] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *ICLR*. 2

[33] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. 2

[34] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In *NeurIPS*, 2021. 2

[35] David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alex Kurakin. Adamatch: A unified approach to semi-supervised learning and domain adaptation. *CoRR*, abs/2106.04732, 2021. 2

[36] Jisoo Jeong, Vikas Verma, Minsung Hyun, Juho Kannala, and Nojun Kwak. Interpolation-based semi-supervised learning for object detection. In *CVPR*, 2021. 2

[37] Yihe Tang, Weifeng Chen, Yijun Luo, and Yuting Zhang. Humble teachers teach better students for semi-supervised object detection. In *CVPR*, 2021. 2

[38] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 2010. 3, 5

[39] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 5

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5