Self-Training and Adversarial Background Regularization for Unsupervised Domain Adaptive One-Stage Object Detection

Seunghyeon Kim  
KAIST  
Jaehoon Choi  
KAIST  
Taekyung Kim  
KAIST  
Changick Kim  
KAIST  
{seunghyeonkim, whdns44, tkkim93, changick}@kaist.ac.kr

Abstract

Deep learning-based object detectors have shown remarkable improvements. However, supervised learning-based methods perform poorly when the train data and the test data have different distributions. To address the issue, domain adaptation transfers knowledge from the label-sufficient domain (source domain) to the label-scarce domain (target domain). Self-training is one of the powerful ways to achieve domain adaptation since it helps class-wise domain adaptation. Unfortunately, a naive approach that utilizes pseudo-labels as ground-truth degenerates the performance due to incorrect pseudo-labels. In this paper, we introduce a weak self-training (WST) method and adversarial background score regularization (BSR) for domain adaptive one-stage object detection. WST diminishes the adverse effects of inaccurate pseudo-labels to stabilize the learning procedure. BSR helps the network extract discriminative features for target backgrounds to reduce the domain shift. Two components are complementary to each other as BSR enhances discrimination between foregrounds and backgrounds, whereas WST strengthen class-wise discrimination. Experimental results show that our approach effectively improves the performance of the one-stage object detection in unsupervised domain adaptation setting.

1. Introduction

Object detection is a fundamental and core problem in computer vision. Recent studies [26, 21, 25] have achieved remarkable improvements with the advances of deep neural networks and large-scale benchmarks [7, 19, 8]. Deep learning-based object detectors can be categorized as two-stage detectors or one-stage detectors. Two-stage detectors first extract object regions and then refine them through classification and regression [26, 6]. On the other hand, one-stage object detectors [25, 21, 35, 20, 39] directly estimate the coordinates and classes of objects without the Region of Interest (RoI) pooling procedure.

One limitation of these supervised learning-based methods is the assumption that test data have the same distribution as train data. However, domain shift frequently occurs in many practical applications. For example, variances of object appearance, viewpoints, backgrounds, illumination, and weather condition can degenerate the performance of the network. One possible solution is collecting labeled data for a new domain, but it is usually expensive and time-consuming. To address this issue, domain adaptation transfers knowledge from the train data domain (source domain) to the test data domain (target domain). Especially, unsupervised domain adaptation assumes there are no labels available in the target domain. The goal of domain adaptation is to train the network that performs well on the target domain dataset.

Unfortunately, domain adaptive object detection has received less attention in contrast to classification [22, 10, 33,
2. Related Work

2.1. Object Detection


On the other hand, one-stage detectors also have been researched and show impressive performance on the inference speed. YOLO [25] is a fast object detector based on a fully convolutional network. SSD [21] improves the performance of accuracy by utilizing feature maps from various scales. The authors of [18] point out that one-stage detectors suffer from the class imbalance problem between foregrounds and backgrounds, and propose focal loss which focuses on hard examples rather than easy ones. Furthermore, recent studies [35, 20, 39] have improved the performance both in accuracy and inference speed maintaining the efficiency of one-stage detectors.

2.2. Domain Adaptation

Domain adaptation reduces the domain gap between train data and test data. For classification, most recent methods manage to reduce the discrepancy between feature distributions of the source and the target. Early work [22] uses Maximum Mean Discrepancy (MMD) as a metric for the discrepancy between two distributions. The authors of [10] propose adversarial learning by adding a gradient reversal layer (GRL) and a discriminator to extract domain invariant features. Recent works [33, 23, 32, 36] are based on the domain adversarial learning framework and further improve the discriminative property on the target domain.

For semantic segmentation, GAN-based domain adaptation approaches are shown to be effective. The authors of [13] point out that earlier feature-level distribution matching approaches fail to capture pixel-level domain shifts. They propose adversarial domain adaptation model, which aligns adaptive object detection which reduces negative effects of inaccurate pseudo-labels.

- We propose adversarial background score regularization (BSR) to reduce the domain shift by extracting discriminative features for target backgrounds.
- Experimental results show that our approach improves the performance of one-stage object detection under unsupervised domain adaptation setting.
both pixel-level and feature-level distribution. In [14], conditional GAN was used to model the residual of the feature map between the source and target domain.

Some prior works use self-training [1, 17] to compensate for the lack of categorical information for either classification [2, 24, 27, 34, 9, 36, 32, 5] or segmentation [40, 4].

### 2.3. Domain Adaptive Object Detection

Compared to classification and semantic segmentation, domain adaptive object detection has received less attention. The authors of [3] present two domain adaptation components, image-level adaptation and instance-level adaptation. They adopt domain adversarial approach using a discriminator for each component. Recently, the focal loss is utilized for a weak global alignment [28]. However, it is hard to apply their algorithm to one-stage detector since the algorithms are designed for Faster R-CNN [26]. The authors of [16] utilize style transfer method, but the method needs large amount of augmented data.

For one-stage object detection, the authors of [15] propose pseudo-labeling and pixel-level adaptation on a cross-domain weakly-supervised setting, which assumes that image-level labels are available for all target images. They first fine-tune the detector with style transferred source images. After that, they generate pseudo-labels by simply choosing the top-1 confidence detections considering image-level labels. They further improve the performance via fine-tuning the network with generated pseudo-labels. However, we confirmed that this method is not valid with the unsupervised domain adaptation setting and degenerates the detection performance.

On the other hand, proposed WST achieves stable learning without image-level labels and BSR extracts discriminative features for target backgrounds instead of aligning non-transferable features.

### 3. Proposed Method

In this section, we introduce details of WST and BSR. We adopt SSD [21] as our baseline.

#### 3.1. Problem Setting

We assume that source data \((x^s, y^s)\) is drawn from the source domain \(X_s\), and target data \((x^t, y^t)\) is drawn from the target domain \(X_t\). Here, \(x\) is an image and \(y = (b, c)\) is a corresponding label, where \(b\) is the coordinates of the bounding box and \(c\) is the class to which the object belongs. We denote the distribution of domain \(X\) as \(P(X)\), and \(P(X_s) \neq P(X_t)\). We assume that both source data and target data have \(K + 1\) classes including the background. We set \(c\) to 0 for the background. We do not have access to target labels, \(y^t\).

We denote layers before conv5 of SSD by \(F\), and the others by \(C\). The output of the SSD is \(O = \{r_i\}_{i=1}^n\), where \(r_i\) is the \(i^{th}\) detection and \(n\) is the total number of detections (e.g., \(n = 8732\) for SSD300). We only take remaining detections after Non-Maximum Suppression (NMS) as final detections. We denote the final outputs as \(O^* = \{r^*_i\}_{i=1}^{n^*}\), where \(r^*_i\) is the \(i^{th}\) detection and \(n^*\) is the number of detections in the final outputs.

#### 3.2. Weak Self-Training

We propose a weak self-training scheme (WST) to compensate for the lack of categorical information on the target domain. Unfortunately, the base network often produces incorrect outputs with high confidence due to large domain shift. These misclassified outputs become false positives when we choose them for pseudo-labels. Also, false negative error occurs when the network fails to detect some objects in an image. To overcome such problems, WST is designed to omit unreliable examples from the training procedure. The framework of WST is shown in Fig. 3.

---

**Figure 3.** The framework of proposed weak self-training. First, we generate pseudo-labels using SRRS (Supporting Region-based Reliable Score) as a criterion. Second, by following traditional hard negative mining, we obtain two sets of positive examples and negative examples \((Pos\ and\ Neg\ respectively)\). Finally, since the chosen hard negatives are risky for self-training, we select easy samples among \(Neg\) and construct \(Neg^\prime\). We use examples in \(Pos\) and \(Neg^\prime\) for weak self-training.
Reducing False Negatives. We minimize the effects of false negatives by modifying the training loss for supervised learning. As defined in [21], the original loss is
\[
L_{\text{task}}(x^t, \hat{b}, \hat{c}) = - \sum_{i \in \text{Pos}} \log(p_i(\hat{c}_i|x^t)) - \sum_{i \in \text{Neg}} \log(p_i(0|x^t)) + L_{\text{loc}}(x^t, \hat{b}),
\]
where Pos and Neg are sets of positive and negative examples respectively, \(b\) is a pseudo-bounding box label, \(\hat{c}_i\) is a pseudo-class label of \(i\)th detection, \(p_i(\hat{c}_i|x^t)\) and \(p_i(0|x^t)\) are the probability values of the class \(\hat{c}_i\) and the background of \(i\)th detection, and \(L_{\text{loc}}\) is the localization loss. However, we observe that Eq. (1) is not effective for self-training. Especially, false negatives selected by hard negative mining are harmful to training. We reduce the false negatives by masking out the gradients of background examples during the training. However, it is undesirable to neglect all the background examples because the network will be biased to the foregrounds. Thus, we ignore background examples that have the potential of being foregrounds.

Negative examples in the \(\tilde{\text{Neg}}\) set have a large potential of being foregrounds since hard negative mining refers to incorrect pseudo-foregrounds and selects background examples with the highest confidence loss values. For example, in Fig. 3, conventional hard negative mining will select the false negative example of the boat as a background example. Thus, we choose \(|\tilde{\text{Neg}}|/3\) examples that have the lowest confidence loss value among negative examples in \(\text{Neg}\). We call this process as weak negative mining, and the obtained set from the process is denoted as \(\tilde{\text{Neg}}\). Additionally, we do not update the network for bounding box regression since pseudo-labels usually have inaccurate bounding box information. Finally, the modified loss function for weak self-training is defined as
\[
L_{\text{ST}}(x^t, \hat{c}) = - \sum_{i \in \text{Pos}} \log(p_i(\hat{c}_i|x^t)) - \sum_{i \in \tilde{\text{Neg}}} \log(p_i(0|x^t)).
\]
Reducing False Positives. We propose a criterion for instance-level pseudo labeling based on supporting RoIs. Here, supporting RoIs denote examples having IoU value larger than some threshold \(\delta\) with the final detection \(r^*\). Rather than using only a single confidence score of a detected box, we consider all the boxes close to the final detection \(r^*\). We define Supporting Region-based Reliable Score (SRRS) as
\[
\text{SRRS}(r^*) = \frac{1}{N_s} \sum_{i=1}^{N_s} \text{IoU}(r_i, r^*) \cdot P(c^*|r_i),
\]
where \(N_s\) is the number of supporting regions, \(\text{IoU}(A, B)\) denotes the IoU value between region A and region B, \(c^*\) is a predicted class of \(r^*\), and \(P(c^*|r_i)\) is a probability for a region \(r_i\) to belong to the \(c^*\). By thresholding the score with \(\epsilon\), we choose reliable detections. The pipeline of generating pseudo-labels is described in Algorithm 1.

3.3. Adversarial Background Score Regularization

We point out that backgrounds of the source domain and target domain share less common features compared to those of foregrounds. We claim that simple global feature alignment enforces to align non-transferable backgrounds, and makes the training procedure unstable. Motivated by [29], we propose background score regularization (BSR) in an adversarial way. The loss function for BSR can be defined as the binary cross entropy function as follows:
\[
L_{\text{adv}}(x^t) = - t \sum_i \log(p_i(0|x^t)) - (1-t) \sum_i \log(1-p_i(0|x^t)),
\]
\[
L_{\text{adv}}(x^t) = - t \sum_i \log(p_i(0|x^t)) - (1-t) \sum_i \log(1-p_i(0|x^t)),
\]
where \(i\) is the index of a detection, and \(t \in [0,1]\) is a target value of \(p_i(0|x^t)\). As we minimize the loss, the value of \(p_i(0|x^t)\) becomes close to \(t\). On the contrary, \(p_i(0|x^t)\) should be close to 0 or 1 to maximize the loss.

At the training phase, both the classifier \(C\) and the feature extractor \(F\) minimize the supervised loss \(L_{\text{task}}\) for the source inputs. For the target inputs, we regularize \(C\) to predict the value of \(p_i(0|x^t)\) close to \(t\) by minimizing \(L_{\text{adv}}\). On the other hand, \(F\) is trained to maximize the adversarial loss to make \(p_i(0|x^t)\) close to 0 or 1. Thus, \(F\) will learn discriminative features to deceive the classifier. The overall training objectives for BSR can be written as follows:
\[
\min_F L_{\text{task}}(x^s, y^s) - L_{\text{adv}}(x^t),
\]
\[
\min_C L_{\text{task}}(x^s, y^s) + L_{\text{adv}}(x^t).
\]
We enable the adversarial training using gradient reversal layer (GRL) right after relu4.3 of SSD300. Note that GRL is activated only when the target input $x^t$ is used for BSR ($L_{adv}(x^t)$). The example input image is from the target domain. Right: We present feature space representations of the adversarial learning. Each point is a detection.

(a): Initial state. (b): The network is trained with the source data. Source examples are well classified, while the boat object of the target data is misclassified as the background. (c): The classifier $C$ minimizes the adversarial loss. The boundary passes through the clusters of the target data. (d): The feature extractor $F$ maximizes the adversarial loss. The example points move away from the boundary.

4. Experiments

In this section, we present details of the implementation and compare our results with other methods.

4.1. Datasets and Evaluation

In our experiments, we used Pascal VOC2007-trainval and VOC2012-trainval dataset [8] as a source domain dataset, and Clipart1k, Watercolor2k, or Comic2k dataset [15] as a target domain dataset.

Pascal VOC [8] is a real-world image dataset. It provides both instance-level bounding box annotations and pixel-level annotations. VOC2007-trainval and VOC2012-trainval set have total 16,551 images with 20 distinct categories. Clipart1k [15] is a dataset of graphical images which have a large domain gap with real-world images. It provides 1k images and has the same categories as Pascal VOC. We used all images as a target dataset both for training and evaluation. Watercolor2k and Comic2k [15] are also unrealistic datasets. Each dataset provides 2k of images, 1k for a train set and the other 1k for a test set. Both Watercolor2k and Comic2k have 6 classes which also exist in VOC. We used the train set for training and the test set for evaluation.

For all experiments, we evaluated methods using mean average precision (mAP). We set the confidence threshold to 0.05 and the IoU threshold to 0.5.

4.2. Implementation Details

In all experiments, we used SSD300 as a base network. Following the original paper [21], inputs were resized to $300 \times 300$, and we applied all augmentations used in the original paper. SGD was used as an optimizer.

**Base Network.** For the base network, only source dataset was used. We trained the network for 120k iterations with...
Table 1. Comparison of various methods in terms of mAP. For all methods, the base network is SSD300 [21]. Pascal VOC2007 trainval and VOC2012 trainval is used for source dataset and Clipart1k is used for target dataset. Descriptions of each method is in Sec. 4.3.

<table>
<thead>
<tr>
<th>Method</th>
<th>BSR</th>
<th>WST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base [21]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>27.3</td>
<td>60.4</td>
</tr>
<tr>
<td>Dann [10]</td>
<td>27.5</td>
<td>46.1</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>26.3</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>30.8</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2. Comparisons on Watercolor2k test set.

<table>
<thead>
<tr>
<th>Method</th>
<th>BSR</th>
<th>WST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base [21]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>77.5</td>
<td>46.1</td>
</tr>
<tr>
<td>Dann [10]</td>
<td>78.9</td>
<td>48.1</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>82.8</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 3. Comparisons on Comic2k test set.

4.3. Results and Comparisons

We compared our method with the base network [21], Dann [10] and ST. Here, ST is the naive approach of self-training that utilizes pseudo-labels as ground-truth without localization loss. While PL in [15] generates pseudo-labels only once before training, ST and our method recreate pseudo-labels for every single iteration. By comparing Dann and ST with our method, we can directly confirm the effectiveness of the proposed algorithm.

Results on Clipart1k. As shown in Table 1, our weak self-training method can improve the accuracy of the object detector without any labels in the target domain. On the other hand, the naive approach (ST) degenerates the performance due to the effects of false positives and false negatives occurred in generated pseudo-labels. To validate each component of weak self-training, we did ablation study in Sec. 5.1. Applying BS shows performance gaps of 8% and 2.2% mAP compared to the baseline and Dann respectively. Without any additional networks such as discriminator, adversarial background score regularization effectively improves the performance. Using both BSR and WST further enhances the performance as they complement each other.

Self-training approaches have inferior performance on the class of sheep, because of the poor performance of the base network, while the domain adaptation methods (DANN and BSR) show improvement of nearly 9% AP.

Results on Watercolor2k and Comic2k. Comparison of performances on VOC → Watercolor2k and VOC → Comic2k is shown in Table 2 and 3.

In the case of Watercolor2k, we set the learning rate to $1.0 \times 10^{-6}$ for self-training methods, since most of the images contain single instance and thus the network is easily overfitted [15]. Furthermore, images in Watercolor2k have an initial learning rate of $1.0 \times 10^{-3}$, momentum of 0.9, weight decay of $5.0 \times 10^{-4}$. We applied learning rate decay of 0.1 at 80k and 100k, so the final learning rate became $1.0 \times 10^{-5}$. From this setting, the base network shows an accuracy of 77.43% mAP on the VOC2007 test set.

Adversarial Background Score Regularization. Both source and target data were used. Each batch is composed of 32 images, 16 from the source domain, and the other 16 from the target domain. We used $t = 0.5$ and $\gamma = 2.0$ in Eq. (4). We trained the network for 50k iterations with a learning rate of $1.0 \times 10^{-3}$, and reduced the learning rate to $1.0 \times 10^{-4}$ for another 10k.

Weak Self-training. We fine-tuned the base model with generated pseudo-labels. This method is different from PL in [15] because we update pseudo-labels for every iteration. The network was trained for 10 epochs with a batch size of 32 and a learning rate of $1.0 \times 10^{-5}$. In the process of generating pseudo-labels, we set $\epsilon$ to 0.8 in Algorithm 1 unless otherwise stated. Also, $\delta = 0.5$ was used since the evaluation metric regards an example as a positive when it has a value of IoU larger than 0.5 with the ground-truth.

BSR with WST. We followed all the settings of BSR. WST was employed after 50k iterations as the network is not reliable before then. The training was early stopped at 55k iterations since self-training is not helpful when it is overused. The threshold of $\epsilon = \frac{1}{1+e^{-\gamma p}}$ was taken where $p = \frac{\text{current iteration}}{\text{max iteration}}$.

DANN. The same configuration with BSR was used to implement DANN for SSD. We aligned distribution of features extracted from relu4_3.
Table 4. Ablation study on WST. Mask denotes that no negative example is used for learning. Weak Mask indicates that weak negative mining is used for sampling negative examples. Method A is identical to ST in Table 1 and method F is the proposed weak self-training.

<table>
<thead>
<tr>
<th>Method</th>
<th>SRRS</th>
<th>Mask</th>
<th>Weak Mask</th>
<th>mAP</th>
<th>$t$</th>
<th>$\gamma$</th>
<th>mAP</th>
<th>$\gamma$</th>
<th>$t$</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST (A)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>10.7</td>
<td>0.0</td>
<td>not converge</td>
<td>0.25</td>
<td>28.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRRS (B)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>10.5</td>
<td>1.0</td>
<td>not converge</td>
<td>0.33</td>
<td>19.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mask (C)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>16.8</td>
<td>0.5</td>
<td>2.0</td>
<td>34.0</td>
<td>2.0</td>
<td>0.5</td>
<td>34.0</td>
</tr>
<tr>
<td>SRRS+Mask (D)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>29.2</td>
<td>4.0</td>
<td>3.29</td>
<td>0.67</td>
<td>21.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weak Mask (E)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>31.3</td>
<td>5.0</td>
<td>3.11</td>
<td>0.75</td>
<td>20.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRRS+Weak Mask (F)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>33.8</td>
<td>6.0</td>
<td>3.13</td>
<td>0.81</td>
<td>21.3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5. Parameter sensitivity on BSR. The table shows the performance of BSR with various values of $\gamma$ and $t$ in Eq. (7).

From the results of method A, B, and C, self-training cannot be accomplished without either SRRS or Mask. The accuracy of the three methods rapidly decreases as training proceeds. This implies that reducing both false positives and false negatives is crucial for object detection.

Comparing with Mask, proposed Weak Mask showed remarkable performance improvement by 14.5% mAP without SRRS and 4.6% mAP with SRRS (E compared to C and F compared to D). Comparing methods A and E, only adding weak negative mining can improve the performance dramatically. From these results, we confirm that selecting reliable negative samples is even more effective than selecting positive samples. More specifically, since pseudo-labels commonly omit some foreground instances, hard negative mining tends to select those missing instances as backgrounds. Masking all the gradients of the negative examples can stabilize the learning, but the network will be biased to foregrounds as it never learns about backgrounds. Thus, the proposed weak negative mining is critical for a self-training under domain adaptation setting.

We observed that SRRS is not valuable when it is used alone. Combined with either Mask or Weak Mask, SRRS succeeded to enhance both learning stability and accuracy (D compared to C and F compared to E). Although SRRS can reduce not only false positives but also true positives, we experimentally confirmed that reducing false positives is crucial even though less true positives are used for training.
5.2. Parameter Sensitivity on BSR

Table 5 shows the results for parameter sensitivity of both $\gamma$ and $t$ in Eq. (7). The focusing parameter $\gamma$ controls the strength of the loss. As $\gamma$ gets smaller, more detections contribute to the adversarial training. More specifically, the network will have too strong regularization on backgrounds with $\gamma = 0$ and $\gamma = 1$. On the other hand, the regularization is relaxed with large values of $\gamma$. The network will ignore examples which have the probability of the background around 0.5 with large $\gamma$. See Fig. 6 for visualization of BSR with various values of $\gamma$. The performance is insensitive to the value of $\gamma$ unless it is too small.

We conducted experiments on $t \in \{\frac{1}{2}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}\}$ to verify trends of the learning according to $t$. The network trained with $t = 0.5$ shows better performance than the others. For the other values, the network is easily over-regularized and the performance rapidly drops after the learning rate decay.

5.3. Qualitative Results

We compare the qualitative results of the base network, the proposed method, and ground-truth as shown in Fig. 7. We found that the proposed method detects objects with less confidence but correctly, compared to the base network due to BSR. As shown in top left example in Fig. 7, the probabilities of two chairs that are detected by the base network are decreased while the chair between them is only detected by our method.

6. Conclusion

In this paper, we have addressed unsupervised domain adaption for one-stage object detection. We enable self-training for object detection by reducing the adverse effects of inaccurate pseudo-labels. Proposed weak self-training (WST) effectively reduces false negatives and false positives by masking the gradients of hard negative examples and utilizing SRRS as a criterion for pseudo-labeling. We have also present adversarial background score regularization (BSR) to reduce the domain shifts by enhancing discrimination between foregrounds and backgrounds of the target data.

Acknowledgements This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIP) (No. 2018-0-00198), Object information extraction and real-to-virtual mapping based AR technology)
References


