Instance Adaptive Self-Training for Unsupervised Domain Adaptation

Ke Mei[1], Chuang Zhu[1] *, Jiaqi Zou[1], and Shanghang Zhang[2]

[1] School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China
{raykoo,czhu,jqzou}@bupt.edu.cn
[2] EECS, University of California, Berkeley, Berkeley, CA 94720, USA
shz@eecs.berkeley.edu

Abstract. The divergence between labeled training data and unlabeled testing data is a significant challenge for recent deep learning models. Unsupervised domain adaptation (UDA) attempts to solve such a problem. Recent works show that self-training is a powerful approach to UDA. However, existing methods have difficulty in balancing scalability and performance. In this paper, we propose an instance adaptive self-training framework for UDA on the task of semantic segmentation. To effectively improve the quality of pseudo-labels, we develop a novel pseudo-label generation strategy with an instance adaptive selector. Besides, we propose the region-guided regularization to smooth the pseudo-label region and sharpen the non-pseudo-label region. Our method is so concise and efficient that it is easy to be generalized to other unsupervised domain adaptation methods. Experiments on 'GTA5 to Cityscapes' and 'SYNTHIA to Cityscapes' demonstrate the superior performance of our approach compared with the state-of-the-art methods.

Keywords: domain adaptation, semantic segmentation, self-training, regularization

1 Introduction

Domain shifts refer to the divergence between the training data (source domain) and the testing data (target domain), induced by factors such as the variance in illumination, object viewpoints, and image background [27,4]. Such domain shifts often lead to the phenomenon that the trained model suffers from a significant performance drop in the unlabeled target domain. The unsupervised domain adaptation (UDA) methods aim to improve the model generalization performance by transferring knowledge from labeled source domain to unlabeled target domain.

Recently, the adversarial training (AT) methods have received significant attention for semantic segmentation [27,9,10,21,6,29]. These methods aim to minimize a series of adversarial losses to align source and target feature distributions.

* The corresponding author: Chuang Zhu.

More recently, an alternative research line to reduce domain shift focuses on building schemes based on the self-training (ST) framework [36,2,34,35,20,31]. These works iteratively train the model by using both the labeled source domain data and the generated pseudo-labels for the target domain and thus achieve the alignment between source and target domains. Besides, several works [19,28,34] have explored to combine AT and ST methods, which shows great potential on semantic segmentation UDA. Through carefully designed network structure, these methods achieve state-of-the-art performance on the benchmark.

Table 1. Performance comparison of AT and ST. *AT*: adversarial training based methods; *ST*: self-training based methods; *AT + ST*: the mixed methods

Method	BLF[19]	AdaptMR[34]	AdaptSeg[27]	AdvEnt[29]	PyCDA[20]	CRST[35]	Ours	mean
AT	44.3	42.7	42.4	45.5	-	-	43.8	43.7
ST	-	-	-	-	47.4	47.1	48.8	47.8
AT+ST	48.5	48.3	-	-	-	-	**50.2**	**49.0**

Despite the success of these AT and ST methods, a natural question comes up: what is the most effective one among these methods? AT or ST? Table 1 lists some of the above representative methods performance on the GTA5 to Cityscapes benchmark. All these methods use the same segmentation network for a fair comparison. In terms of performance, an explicit conclusion is: AT + ST (49.0) [19,34] > ST (47.8) [20,35] > AT (43.7) [27,29]. The mixed methods, such as BLF [19] and AdaptMR [34], both have achieved great performance gains (+ 4.2, + 5.6) after using ST. However, in order to achieve better performance, these mixed methods generally have serious coupling between sub-modules (such as network structure dependency), thus losing scalability and flexibility.

This paper aims to propose a self-training framework for semantic segmentation UDA, which has good scalability that can be easily applied to other non-self-training methods and achieves state-of-the-art performance. To achieve this, we locate the main obstacle of existing self-training methods is how to generate high-quality pseudo-labels. This paper designs a new pseudo-label generation strategy and model regularization to solve this obstacle.

Fig. 1. Pseudo-label results. Columns correspond to original images with ground truth labels, class-balanced method, and our method

The pseudo-label generation suffers from information redundancy and noise. The generator tends to keep pixels with high confidence as pseudo-labels and ignore pixels with low confidence. Because of this conservative threshold selection, they are inefficient when more similar samples with high confidence are applied to training. The existing class-balanced self-training (CBST) [36] utilized rank-based reference confidence for each class among all related images. This results in the ignorance of key information from the hard images with most of the pixels having low prediction scores. For example, in Fig. 1, the pseudo-labels generated by CBST are concentrated on the road, while pedestrians and trucks are ignored, which loses much learnable information. Therefore, we try to design a pseudo-label generation that can be adjusted adaptively according to the instance strategy to reduce data redundancy and increase the diversity of pseudo-labels.

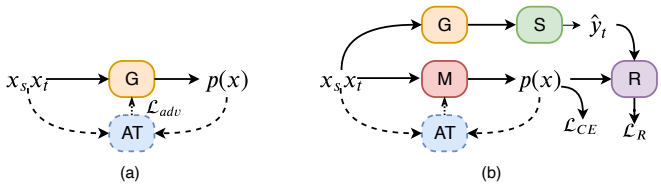

Fig. 2. IAST framework. (a) Warm-up phase, an initial model G is trained using any existing non-self-training method (eg. AT). (b) Self-training phase, the selector S filters the pseudo-labels generated by G, and R is the regularization

In this work, we propose an instance adaptive self-training framework (IAST) for semantic segmentation UDA, as shown in Fig. 2. We employ an instance adaptive selector in considering pseudo-label diversity during the training process. Besides, we design region-guided regularization in our framework, which has different roles in the pseudo-label region and the non-pseudo-label region. The main contributions of our work are summarized as follows:

- We propose a new self-training framework. Our methods significantly outperform the current state-of-the-art methods on the public semantic segmentation UDA benchmark.
- We design an instance adaptive selector to involve more useful information for training. It effectively improves the quality of pseudo-labels. Besides, region-based regularization is designed to smooth the prediction of the pseudo-label region and sharpen the prediction of the non-pseudo-label region.
- We propose a general approach that makes it easy to apply other non-self-training methods to our framework. Moreover, our framework can also be extended to semi-supervised semantic segmentation tasks.

2 Related Works

Adversarial training for UDA: A large number of UDA schemes [1,13,16,17] are proposed to reduce the domain gap by building shared embedding space to both the source and target domain. Following the same idea, many adversarial training based UDA methods are proposed by adding a domain discriminator in recent years [27,10,21,6,29,33,32]. With adversarial training, the domain adversarial loss can be minimized to directly align features between two domains. Motivated by the recent image-to-image translation works, some works [9,19] regard the mapping from the source domain to the target domain as the image synthesis problem that reduce the domain discrepancy before training.

Self-training: Self-training schemes are commonly used in semi-supervised learning (SSL) areas [18].These works iteratively train the model by using both the labeled source domain data and the generated pseudo-labels in the target domain and thus achieve the alignment between the source and target domain [26]. However, these methods directly choose pseudo-labels with high prediction confidence, which will result in the model bias towards easy classes and thus ruin the transforming performance for the hard classes. To solve this problem, the authors in [36] proposed a class-balanced self-training (CBST) scheme for semantic segmentation, which shows comparable domain adaptation performance to the best adversarial training based methods. [20] proposed a self-motivated pyramid curriculum domain adaptation method using self-training. More recently, CRST [35] further integrated a variety of confidence regularizers to CBST, producing better domain adaption results.

Regularization: Regularization refers to schemes that are intended to reduce the testing error and thus make the trained model generalize well to unseen data [7,15]. For deep neural network learning, different kinds of regularization schemes such as weight decay [14] and label smoothing [25] are proposed. The recent work [35] designed labels and model regularization under self-training architecture for UDA. However, the proposed regularization scheme is just applied to the pseudo-label region.

3 Preliminary

3.1 UDA for Semantic Segmentation

It is assumed that there are two domains: source domain S and target domain T. The source domain includes image $\mathbb{X}_S = \{x_s\}$, semantic mask $\mathbb{Y}_S = \{y_s\}$, and the target domain only has image $\mathbb{X}_T = \{x_t\}$. In UDA, the semantic segmentation model is trained only from the ground truth $\mathbb{Y}_S$ as the supervisory signal. UDA semantic segmentation model can be defined as follows:

$$\{\mathbb{X}_S, \mathbb{Y}_S, \mathbb{X}_T\} \Rightarrow \mathbf{M}_{UDA}$$

$\mathbf{M}_{UDA}$ uses some special losses and domain adaptation methods to align the distribution of two domains to learn domain-invariant feature representation.

3.2 Self-training for UDA

Because the ground truth labels of target domain are not available, we can treat the target domain as an extra unlabeled dataset. In this case, the UDA task can be transformed into a semi-supervised learning (SSL) task. Self-training is an effective method for SSL. The problem can be described as the following forms:

$$\begin{aligned}\min_{\mathbf{w}} \mathcal{L}_{CE} = &- \frac{1}{|\mathbb{X}_S|} \sum_{\mathbf{x_s} \in \mathbb{X}_S} \sum_{c=1}^{C} y_s^{(c)} \log p(c|\mathbf{x}_s, \mathbf{w}) \\ &- \frac{1}{|\mathbb{X}_T|} \sum_{\mathbf{x}_t \in \mathbb{X}_T} \sum_{c=1}^{C} \hat{y}_t^{(c)} \log p(c|\mathbf{x}_t, \mathbf{w})\end{aligned} \tag{1}$$

where C is the number of classes, $y_s^{(c)}$ indicates the label of class c in source domain, and $\hat{y}_t^{(c)}$ indicates the pseudo-label of class c in target domain. $\mathbf{x}_s$ and $\mathbf{x}_t$ are input images, $\mathbf{w}$ indicates weights of $\mathbf{M}$, $p(c|\mathbf{x}, \mathbf{w})$ is the probability of class c in softmax output, and $|\mathbb{X}|$ indicates the number of images.

In particular, $\hat{\mathbb{Y}}_T = \{\hat{y}_t\}$ are the "pseudo-labels" generated according to the existing model, which is limited to a one-hot vector (only single 1 and all the others 0) or an all-zero vector. The pseudo-labels can be used as approximate target ground truth labels.

3.3 Adversarial training for UDA

Adversarial training uses an additional discriminator to align feature distributions. The discriminator $\mathbf{D}$ attempts to distinguish the feature distribution in the output space of the source and target. The segmentation model $\mathbf{M}$ attempts to fool the discriminator to confuse the feature distributions of the source and target, thereby aligning the feature distributions. The optimization process is as follows:

$$\begin{aligned}\min_{\mathbf{w}} \max_{\mathbf{D}} \mathcal{L}_{AT} = &- \frac{1}{|\mathbb{X}_S|} \sum_{\mathbf{x}_s \in \mathbb{X}_S} \sum_{c=1}^{C} y_s^{(c)} \log p(c|\mathbf{x}_s, \mathbf{w}) \\ &+ \frac{\lambda_{adv}}{|\mathbb{X}_T|} \sum_{\mathbf{x}_t \in \mathbb{X}_T} \left[\mathbf{D}(\mathbf{M}(\mathbf{x}_t, \mathbf{w})) - \mathbf{1}\right]^2\end{aligned} \tag{2}$$

The first term is the cross-entropy loss of source, and the second term uses a mean squared error as the adversarial loss, where λ_{adv} is the weight of the adversarial loss. Eq. (2) is used to optimize $\mathbf{M}$ and $\mathbf{D}$ alternately.

4 Proposed Method

An overview of our framework is shown in Fig. 3. We propose an instance adaptive self-training framework (IAST) with instance adaptive selector (IAS) and region-guided regularization. IAS selects an adaptive pseudo-label threshold for each semantic category in units of images and dynamically reduces the proportion of "hard" classes, to eliminate noise in the pseudo-labels. Besides, region-guided regularization is designed to smooth the prediction of the confident region and sharpen the prediction of the ignored region. Our overall objective function is as follows:

$$\begin{aligned} &\min_{\mathbf{w}} \mathcal{L}_{CE}(\mathbf{w}, \hat{\mathbb{Y}}_T) + \mathcal{L}_R(\mathbf{w}) \\ &= \mathcal{L}_{CE}(\mathbf{w}, \hat{\mathbb{Y}}_T) + (\lambda_i \mathcal{R}_i(\mathbf{w}) + \lambda_c \mathcal{R}_c(\mathbf{w})) \end{aligned} \tag{3}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss, which is different from Eq.(1) and only calculates the cross-entropy loss of the target domain images. $\hat{\mathbb{Y}}_T$ is the set of pseudo-labels, and the detailed generation process is described in Section 4.1. $\mathcal{R}_i$ and $\mathcal{R}_c$ are regularization of the ignored and confidence regions, which is described in Section 4.2. And λ_i, λ_c are regularization weights.

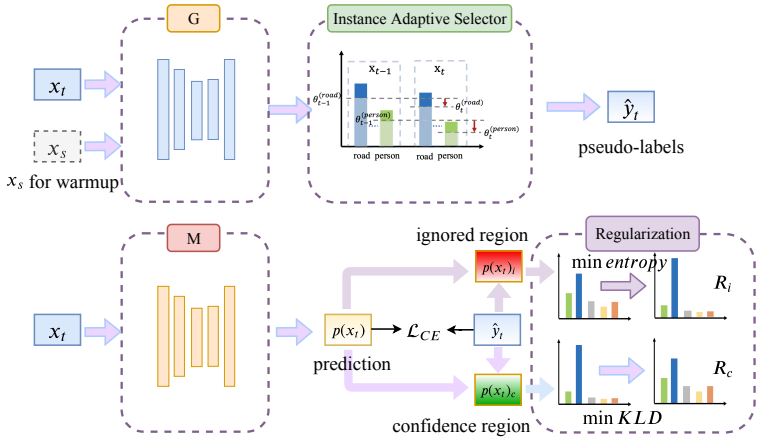

Fig. 3. Proposed IAST framework overview

The IAST training process consists of three phases.

- (a) In the *warm-up phase*, a non-self-training method uses both the source data and the target data to train an initial segmentation model $\mathbf{M}_0$ as the initial pseudo-label generator $\mathbf{G}_0$.
- (b) In the *pseudo-label generation phase*, $\mathbf{G}$ is used to obtain the prediction result of the target data, and a pseudo-label is generated by an instance adaptive selector.

– (c) In the *self-training phase*, according to Eq.(3), the segmentation model **M** is trained using the target data.

Why warm-up? Before self-training, we expect to have a stable pre-trained model so that IAST can be trained in the right direction and avoid disturbances caused by constant fitting the noise of pseudo-labels. We use the adversarial training method described in Section 3.3 to obtain a stable model by roughly aligning the output of the source and target. In addition, in the warm-up phase, we can optionally apply any other semantic segmentation UDA method as the basic method, and it can be retained even in the (c) phase. In fact, we can use IAST as a decorator to decorate other basic methods.

Multi-round self-training. Performing (b) phase and (c) phase once counts as one round. As with other self-training tasks, in this experiment we performed a total of three rounds. At the end of each round, the parameters of model **M** will be copied into model **G** to generate better target domain prediction results in the next round.

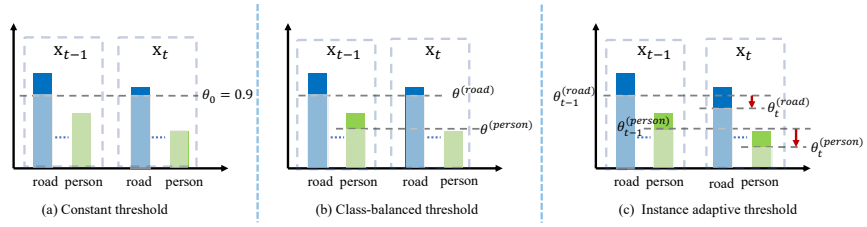

Fig. 4. Illustration of three different thresholding methods. $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ represent two consecutive instances, the bars approximately represent the probabilities of each class. (a) A constant threshold is used for all instances. (b) class-balanced thresholds are used for all instances. (c) Our method adaptively adjusts the threshold of each class based on the instance

4.1 Pseudo-Label Generation Strategy with an Instance Adaptive Selector

Pseudo-labels $\hat{\mathbb{Y}}_T$ have a decisive effect on the quality of self-training. The generic pseudo-label generation strategy can be simplified to the following form when segmentation model parameter $\mathbf{w}$ is fixed:

$$\min_{\hat{\mathbb{Y}}_T} -\frac{1}{|\mathbb{X}_T|}\sum_{\mathbf{x_t}\in\mathbb{X}_T}\sum_{c=1}^{C}\hat{y}_t^{(c)}\log\frac{p(c|\mathbf{x}_t,\mathbf{w})}{\theta_t^{(c)}} \\ s.t.\ \hat{\mathbf{y}}_t \in \{[onehot]^C\}\cup\mathbf{0}\ ,\ \forall\hat{\mathbf{y}}_t\in\hat{\mathbb{Y}}_T \tag{4}$$

Algorithm 1 pseudo-labels generation

Input: Model $\mathbf{M}$, target instance $\{\mathbf{x}_t\}^T$,
Parameter: proportion α, momentum β, weight decay γ,
Output: target pseudo-labels

init $\boldsymbol{\theta}_0 = \mathbf{0.9}$
for $t = 1$ **to** T **do**
  $\mathbf{P}_{index} = \arg\max(\mathbf{M}(\mathbf{x}_t))$
  $\mathbf{P}_{value} = \max(\mathbf{M}(\mathbf{x}_t))$
  for $c = 1$ **to** C **do**
    $\mathbb{P}_{\mathbf{x}_t}^{(c)} = \text{sort}(\mathbf{P}_{value}[\mathbf{P}_{index} = c], \text{descending})$
    $\theta_{\mathbf{x}_t}^{(c)} = \Psi(\mathbf{x}_t, \theta_{t-1}^{(c)})$ Eq.(7)
  end for
  $\boldsymbol{\theta}_t = \beta\boldsymbol{\theta}_{t-1} + (1-\beta)\boldsymbol{\theta}_{\mathbf{x}_t}$ Eq.(6)
  $\hat{\mathbf{y}}_t = \text{onehot}(\mathbf{P}_{index}[\mathbf{P}_{value} > \boldsymbol{\theta}_t])$
end for
return $\{\hat{\mathbf{y}}_t\}^T$

where $\theta^{(c)}$ indicates the confidence threshold for class c, and $\hat{\mathbf{y}}_\mathbf{t} = [\hat{y}_t^{(1)}, ..., \hat{y}_t^{(C)}]$ is required to be a one-hot vector or a all-zero vector. Therefore, $\hat{y}_t^{(c)}$ can be solved by Eq.(5).

$$\hat{y}_t^{(c)} = \begin{cases} 1, if\ c = \arg\max_c p(c|\mathbf{x}_t, \mathbf{w})\ and\ p(c|\mathbf{x}_t, \mathbf{w}) > \theta^{(c)} \\ 0, otherwise \end{cases} \tag{5}$$

When class c output probability $p(c|\mathbf{x}_t, \mathbf{w}) > \theta^{(c)}$, these pixels are regarded as confidence region (pseudo-label region), and the rest are ignored regions (non-pseudo-label region). Therefore, $\theta^{(c)}$ become the key to the pseudo-labels generation process. As shown in Fig.4: (a) the traditional pseudo-labels generation strategy based on a constant confidence threshold; (b) the generation strategy which uses the same class-balanced $\boldsymbol{\theta}$ for all target images; (c) we propose a data diversity-driven pseudo-labels generation strategy with an instant adaptive selector (IAS).

IAS maintains two thresholds $\{\boldsymbol{\theta}_t, \boldsymbol{\theta}_{\mathbf{x}_t}\}$, where $\boldsymbol{\theta}_t$ indicates the historical threshold and $\boldsymbol{\theta}_{\mathbf{x}_t}$ indicates the threshold of current instance $\mathbf{x}_t$. During the generation process, IAS dynamically updates $\boldsymbol{\theta}_t$ based on $\boldsymbol{\theta}_{\mathbf{x}_t}$ of the current instance $\mathbf{x}_t$, so each instance gets an adaptive threshold, combining global and local information. Specifically, for each instance $\mathbf{x}_t$, we sort the confidence probability of each class in descending order, and then take the $\alpha \times 100\%$ confidence probability as the **local** threshold $\theta_{\mathbf{x}_t}^{(c)}$ for each class in instance $\mathbf{x}_t$. Finally, we use the exponentially weighted moving average to update the threshold $\boldsymbol{\theta}_t$ containing historical information as the **global** threshold. The details are summarized in Algorithm1.

Exponential moving average (EMA) threshold. When generating pseudo-labels one by one, we use an exponential moving average method, denoted as

Eq.(6), which can smooth the threshold of each instance, introduce past historical information, and avoid noise interference. Eq.(7) $\Psi(\mathbf{x}_t, \theta_{t-1}^{(c)})$ represents the threshold for acquiring the current instance $\mathbf{x}_t$. β is a momentum factor used to preserve past threshold information. As β increases, the threshold $\theta_t^{(c)}$ becomes smoother.

$$\theta_t^{(c)} = \beta\theta_{t-1}^{(c)} + (1-\beta)\Psi(\mathbf{x}_t, \theta_{t-1}^{(c)}) \tag{6}$$

$$\Psi(\mathbf{x}_t, \theta_{t-1}^{(c)}) = \mathbb{P}_{\mathbf{x}_t}^{(c)}\left[\alpha {\theta_{t-1}^{(c)}}^{\gamma} | \mathbb{P}_{\mathbf{x}_t}^{(c)}|\right] \tag{7}$$

"Hard" classes weight decay (HWD). For "hard" classes, pseudo-labels tend to bring more noise labels. In Eq.(7), we design ${\theta_{t-1}^{(c)}}^{\gamma}$ to modify the proportion of pseudo-labels α. γ is a weight decay parameter, which is used to control the decay degree. The thresholds $\theta_{t-1}^{(c)}$ of the "hard" classes are usually smaller, so HWD reduces more pseudo-labels of "hard" classes. On the contrary the thresholds $\theta_{t-1}^{(c)}$ of easy classes is usually larger, so HWD has a weaker impact. It is easy to prove that when $\Psi(\mathbf{x}_t, \theta_{t-1}^{(c)}) = \theta_{t-1}^{(c)}$, θ will converge to a larger value, thereby reduce the amount of the "hard" classes.

4.2 Region-Guided Regularization

Confident region KLD minimization. During training, the model is prone to overfit pseudo-labels, which will damage the model. For the confidence region $\mathbb{I}_{\mathbf{x}_t} = \{\mathbf{1} \mid \hat{\mathbf{y}}_t^{(h,w)} > \mathbf{0}\}$, there are pseudo labels as supervising signals to supervise the model for learning. However, as shown in Table 4, although a series of techniques for generating high-confidence pseudo labels have been used, the quality of the pseudo labels is still not as good as the ground truth labels, which means that there are some noise labels in the pseudo-labels. How to reduce the impact of noise labels is a key issue. Zou et al. [35] has proposed various regularization for this. We use the KLD which works best in [35] to smooth the prediction results of the confidence region, so that the prediction results do not overfit the pseudo-labels.

$$\mathcal{R}_c = -\frac{1}{|\mathbb{X}_T|}\sum_{\mathbf{x}_t \in \mathbb{X}_T} \mathbb{I}_{\mathbf{x}_t} \sum_{c=1}^{C} \frac{1}{C} \log p(c|\mathbf{x}_t, \mathbf{w}) \tag{8}$$

As shown in Eq.(8), when the prediction result $\log p(c|\mathbf{x}_t, \mathbf{w})$ is approximately close to the uniform distribution (the probability of each class is $\frac{1}{C}$), $\mathcal{R}_c$ gets smaller. KLD minimization promotes smoothing of confidence regionsand avoid the model blindly trusting false labels.

Ignored region entropy minimization. On the other hand, for the ignored region $\mathbb{I}^{\mathtt{C}}_{\mathbf{x}_t} = \{\mathbf{1} \mid \hat{\mathbf{y}}_t^{(h,w)} = \mathbf{0}\}$, there is no supervision signal during the training process. Because the prediction result of the region $\mathbb{I}^{\mathtt{C}}_{\mathbf{x}_t}$ is smooth and has low confidence, we use the minimized entropy of the ignored region to prompt the model to predict the low entropy result, which makes the prediction result look more "sharper".

$$\mathcal{R}_i = -\frac{1}{|\mathbb{X}_T|} \sum_{\mathbf{x}_t \in \mathbb{X}_T} \mathbb{I}^{\mathtt{C}}_{\mathbf{x}_t} \sum_{c=1}^{C} p(c|\mathbf{x}_t, \mathbf{w}) \log p(c|\mathbf{x}_t, \mathbf{w}) \tag{9}$$

As shown in Eq.(9), sharpening the prediction result of the ignored region by minimizing $\mathcal{R}_i$ can promote the model to learn more useful features from the ignored region without any supervised signal, which has also been proved to be effective for UDA in the work [29].

5 Experiment

5.1 Experimental Settings

Network architecture and datasets. We adapt Deeplab-v2 [3], which is widely used in the semantic segmentation UDA problem, as our basic network architecture. ResNet-101[8] is selected as the backbone network of the model. All experiments in this work are carried out under this network architecture. We evaluate our UDA methods for semantic segmentation on the popular synthetic-to-real adaptation scenarios: (a) GTA5 [23] to Cityscapes [5], (b) SYNTHIA [24] to Cityscapes. The GTA5 dataset has 24966 images that are rendered from the GTA5 game and 19 classes with Cityscapes. SYNTHIA dataset includes 9400 images and 16 common classes with Cityscapes. Cityscapes is split into training set, validation set, and testing set. Following the standard protocols in [27], we use the training set which has 2975 images as the target dataset and use the validation dataset to evaluate our models with mIoU.

Implementation details. In our experiments, we implement IAST using PyTorch on an NVIDIA Tesla V100. The training images are randomly cropped and resized to 1024×512, the aspect ratio of the crop window is 2.0, and the window height is randomly selected from $[341 \sim 950]$ for GTA5 and $[341 \sim 640]$ for SYNTHIA. All weights of batch normalization layers were frozen. Deeplab-v2 is pre-trained on ImageNet. In IAST, we adopt Adam with learning rate 2.5×10^{-5}, batch size 6 for 4 epochs. The pseud-label parameters α, β, γ are set to 0.2, 0.9 and 8.0. The regularization weights λ_i and λ_c are set to 3.0 and 0.1. Our code code and pre-trained molels are available at: `https://github.com/Raykoooo/IAST`

5.2 Discussion and Ablation Study

Why IAS works? Table 2 shows a sensitivity analysis on the parameter α and β. When we set $\alpha = 0.2$ and $\beta = 0$, it means IAS takes 20% of each image as the

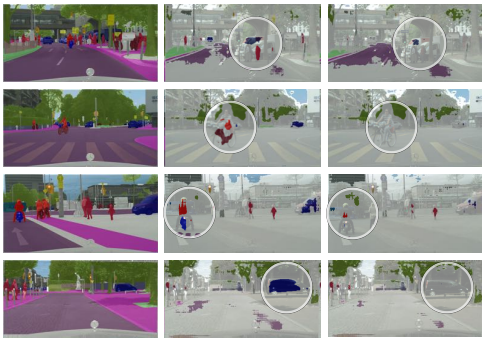

Fig. 5. Visualization of pseudo-labels. Columns correspond to original images with ground truth labels, our method, and class-balanced method [36]

confidence region. As a comparison, the class-balanced method [36] takes 20% of pixels in the whole target set as the confidence region. As shown in Fig. 5, pseudo-labels of class-balanced method miss some pixels for persons, cars and bikes. In contrast, the pseudo-labels of our method are more diverse, especially for some "hard" classes. When we set $\alpha = 0.2$ and $\beta = 0.9$, IAS combines global and local information to get more diverse content so that the model achieve the best performance.

Table 2. α and β sensitivity analysis (GTA5 to Cityscapes)

α	β	Proportion(%)	mIoU(%)
.20	.0	20.0	49.8
.20	.50	31.2	50.3
.20	.90	36.5	**50.5**
.20	.99	40.1	50.0
.30	.90	42.5	49.7
.50	.90	48.6	48.2
Constant(Fig. 4 a)		38.6	45.1
Class-balanced(Fig. 4 b)		20.0	47.9

Table 3. λ_i and λ_c sensitivity analysis (GTA5 to Cityscapes)

λ_i	λ_c	mIoU(%)
.5	.10	50.6
1.0	.10	51.1
2.0	.10	50.9
3.0	.10	**51.5**
4.0	.10	51.2
5.0	.10	51.3
3.0	.05	50.6
3.0	.15	51.0

Fig.6 shows that as the γ increases, the proportion of some easy classes (sky, car) that have a high prediction score does not decrease significantly, while the proportion of some "hard" classes (motor, wall, fence and pole) that have a low prediction score decreases sharply. This proves that Eq.(7) can effectively reduce the pseudo-labels of "hard" classes and suppress noise interference in the pseudo-labels. Table 4 shows a sensitivity analysis on the parameter γ. We find that as the γ increases, pseudo-labels have smaller proportions but have better quality. Therefore, we let $\gamma = 8$ as the trade-off between the proportion and the quality of pseudo-labels. On the contrary, moderate regularization helps the model to improve the prediction accuracy and avoid overfitting the noise labels.

Table 3 shows a sensitivity analysis of the parameter λ_i and λ_c. We performed multiple sets of experiments with fixed λ_i and λ_c, respectively. When $\lambda_c = 0.1$ is fixed and λ_i is gradually increased, the overall model performance tends to improve until $\lambda_i = 4$. It can be expected that when the low entropy prediction is excessively performed in the non-pseudo-label region, the influence of noise will be amplified and the model will be damaged.

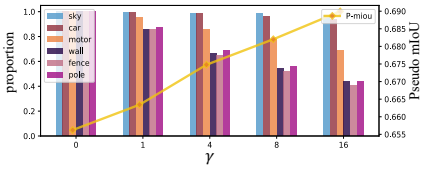

Fig. 6. Relationship between the pseudo-labels proportion and γ

Table 4. γ sensitivity analysis ($\alpha = 0.2, \beta = 1.0$). *P-mIoU* means mIoU of pseudo-labels

γ	0	1	4	8	16
Proportion	0.36	0.34	0.30	0.28	0.25
P-mIoU(%)	65.6	66.3	67.4	68.2	69.0
mIoU (%)	50.5	50.8	51.2	**51.5**	50.9

Ablation studies. The results of the ablation studies are reported in Table 5. We attempt the methods proposed in Section 4.1 and Section 4.2 one by one to study their performance in the test set. From the data in Table 5, after using self-training (Fig. 4 a) without using any other techniques, the model performance has a gain of 1.3%. After adding IAST modules (IAS, $\mathcal{R}_i$, $\mathcal{R}_c$), the performance of the model is gradually and steadily improved, and finally, 51.5% mIoU is achieved. In addition, we also try multi-scale testing and the combined result achieved the best 52.2% mIoU.

Table 5. Results of ablation study (GTA5 to Cityscapes)

Method	ST	IAS	$\mathcal{R}_c$	$\mathcal{R}_i$	mIoU	Δ
Source	-	-	-	-	35.6	0
Warm-up	-	-	-	-	43.8	+8.2
+ Constant ST(Fig. 4 a)	✓				45.1	+1.3
+ Instance adaptive selector	✓	✓			49.8	+4.7
+ Confidence region R.	✓	✓	✓		50.7	+0.9
+ Ignored region R.	✓	✓	✓	✓	51.5	+0.8

5.3 Experimental Results

Comparison with the state-of-the-art methods: The results of IAST and some other state-of-the-art methods on GTA5 to Cityscapes are present in Table6. From the overall results, IAST has the best mIoU 52.2% and has obvious advantages over other methods. Compared with some adversarial training methods AdaptSegNet [27] and SIBAN [22], IAST improves by 9.6% mIoU and have significant gains in almost all classes. Compared with the same self-training methods such as MRKLD [35], IAST improves by 4.8% mIoU. In addition, BLF [19] is a method that combines adversarial training and self-training, which has

Table 6. Results of our proposed method IAST and other state-of-the-art methods (GTA5 to Cityscapes). A&S means a mixed method of AT and ST

Method	Arch.	Road	SW	Build	Wall	Fence	Pole	TL	TS	Veg.	Terrain	Sky	PR	Rider	Car	Truck	Bus	Train	Motor	Bike	mIoU
Source [27]	AT	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	36.0	36.6
AdaptSegNet [27]		86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.7	32.5	35.4	3.9	30.1	28.1	42.4
SIBAN [22]		88.5	35.4	79.5	26.3	24.3	28.5	32.5	18.3	81.2	40.0	76.5	58.1	25.8	82.6	30.3	34.3	3.4	21.6	21.5	42.6
SSF-DAN [6]		90.3	38.9	81.7	24.8	22.9	30.5	37.0	21.2	84.8	38.8	76.9	58.8	30.7	85.7	30.6	38.1	5.9	28.3	36.9	45.4
AdvEnt [29]		89.4	33.1	81.0	26.6	26.8	27.2	33.5	24.7	83.9	36.7	78.8	58.7	30.5	84.8	38.5	44.5	1.7	31.6	32.4	45.4
APODA [30]		85.6	32.8	79.0	29.5	25.5	26.8	34.6	19.9	83.7	40.6	77.9	59.2	28.3	84.6	34.6	49.2	8.0	32.6	39.6	45.9
Source [36]	ST	71.3	19.2	69.1	18.4	10.0	35.7	27.3	6.8	79.6	24.8	72.1	57.6	19.5	55.5	15.5	15.1	11.7	21.1	12.0	33.8
CBST [36]		91.8	53.5	80.5	32.7	21.0	34.0	28.9	20.4	83.9	34.2	80.9	53.1	24.0	82.7	30.3	35.9	16.0	25.9	42.8	45.9
PyCDA[20]		90.5	36.3	84.4	32.4	28.7	34.6	36.4	31.5	**86.8**	37.9	78.5	62.3	21.5	85.6	27.9	34.8	18.0	22.9	**49.3**	47.4
MRKLD [35]		91.0	55.4	80.0	33.7	21.4	**37.3**	32.9	24.5	85.0	34.1	80.8	57.7	24.6	84.1	27.8	30.1	**26.9**	26.0	42.3	47.1
BLF [19]	A&S	91.0	44.7	84.2	34.6	27.6	30.2	36.0	36.0	85.0	**43.6**	83.0	58.6	31.6	83.3	35.3	49.7	3.3	28.8	35.6	48.5
AdaptMR [34]		90.5	35.0	84.6	34.3	24.0	36.8	**44.1**	**42.7**	84.5	33.6	82.5	**63.1**	**34.4**	85.8	32.9	38.2	2.0	27.1	41.8	48.3
PatchAlign [28]		92.3	51.9	82.1	29.2	25.1	24.5	33.8	33.0	82.4	32.8	82.2	58.6	27.2	84.3	33.4	26.3	2.2	29.5	32.3	46.5
Source(ours)	A&S	64.8	21.7	74.3	15.4	21.2	18.2	30.7	13.0	80.9	33.7	76.3	55.6	20.0	43.9	27.0	35.5	4.4	24.9	14.3	35.6
IAST(ours)		93.8	57.8	85.1	39.5	26.7	26.2	43.1	34.7	84.9	32.9	88.0	62.6	29.0	87.3	39.2	49.6	23.2	34.7	39.6	51.5
IAST-MST(ours)		**94.1**	**58.8**	**85.4**	**39.7**	**29.2**	25.1	43.1	34.2	84.8	34.6	**88.7**	62.7	30.3	**87.6**	**42.3**	**50.3**	24.7	**35.2**	40.2	**52.2**

Table 7. Results of our proposed method IAST and other state-of-the-art methods (SYNTHIA to Cityscapes)

Method	Arch.	Road	SW	Build	Wall*	Fence*	Pole*	TL	TS	Veg.	Sky	PR	Rider	Car	Bus	Motor	Bike	mIoU	mIoU*
Source [27]	AT	55.6	23.8	74.6	-	-	-	6.1	12.1	74.8	79.0	55.3	19.1	39.6	23.3	13.7	25.0	-	38.6
AdaptSegNet [27]		84.3	42.7	77.5	-	-	-	4.7	7.0	77.9	82.5	54.3	21.0	72.3	32.2	18.9	32.3	-	46.7
SIBAN [22]		82.5	24.0	79.4	-	-	-	16.5	12.7	79.2	82.8	58.3	18.0	79.3	25.3	17.6	25.9	-	46.3
SSF-DAN [6]		84.6	41.7	80.8	-	-	-	11.5	14.7	80.8	**85.3**	57.5	21.6	82.0	36.0	19.3	34.5	-	50.0
AdvEnt [29]		85.6	42.2	79.7	8.7	0.4	25.9	5.4	8.1	80.4	84.1	57.9	23.8	73.3	36.4	14.2	33.0	41.2	48.0
APODA [30]		**86.4**	41.3	79.3	-	-	-	22.6	17.3	80.3	81.6	56.9	21.0	84.1	**49.1**	24.6	45.7	-	53.1
Source [36]	ST	64.3	21.3	73.1	2.4	1.1	31.4	7.0	27.7	63.1	67.6	42.2	19.9	73.1	15.3	10.5	38.9	34.9	40.3
CBST [36]		68.0	29.9	76.3	10.8	1.4	33.9	22.8	29.5	77.6	78.3	60.6	28.3	81.6	23.5	18.8	39.8	42.6	48.9
PyCDA [20]		75.5	30.9	83.3	**20.8**	0.7	32.7	27.3	33.5	**84.7**	85.0	64.1	25.4	85.0	45.2	21.2	32.0	46.7	53.3
MRKLD [35]		67.7	32.2	73.9	10.7	1.6	**37.4**	22.2	**31.2**	80.8	80.5	60.8	29.1	82.8	25.0	19.4	45.3	43.8	50.1
BLF [19]	A&S	86.0	**46.7**	80.3	-	-	-	14.1	11.6	79.2	81.3	54.1	27.9	73.7	42.2	25.7	45.3	-	51.4
AdaptMR [34]		83.1	38.2	81.7	9.3	1.0	35.1	30.3	19.9	82.0	80.1	62.8	21.1	84.4	37.8	24.5	**53.3**	46.5	53.8
PatchAlign [28]		82.4	38.0	78.6	8.7	0.6	26.0	3.9	11.1	75.5	84.6	53.5	21.6	71.4	32.6	19.3	31.7	40.0	46.5
Source(ours)	A&S	63.4	24.1	66.7	7.1	0.1	28.4	11.6	16.8	77.0	74.6	60.4	20.5	75.6	22.0	14.4	21.2	36.5	42.2
IAST(ours)		81.9	41.5	**83.3**	17.7	**4.6**	32.3	**30.9**	28.8	83.4	85.0	**65.5**	**30.8**	**86.5**	38.2	**33.1**	52.7	**49.8**	**57.0**

the second-best 48.5% mIoU. Compared to BLF, IAST still has a significant improvement.

Table 7 is the results of the SYNTHIA to Cityscapes dataset. For a comprehensive comparison, as in the previous work, we also report two mIoU metrics: 13 classes of mIoU* and 16 classes of mIoU. The domain gap between SYNTHIA and Cityscapes is much larger than the domain gap between GTA5 and Cityscapes. Many of the methods that performed well on GTA5 to Cityscapes have experienced a significant performance degradation on this dataset. Correspondingly, the performance gap between different methods is becoming more

apparent. IAST also achieves the best results, which are 49.8% mIoU and 57.0% mIoU* and significantly higher than all recent state-of-the-art methods.

Table 8. Semi-supervised learning results on the Cityscapes val set. 1/8, 1/4 and 1/2 mean the proportion of labeled images

Method	Data Amount			
	1/8	1/4	1/2	Full
Baseline	57.3	59.0	61.2	70.2
Univ-full[12]	55.9	-	-	-
AdvSemi[11]	58.8	62.3	65.7	67.7
IAST(ours)	**64.6**	**66.7**	**69.8**	**70.2**

Table 9. Extension analysis, applying IAST to non-self-learning UDA methods [27,29] (test on Cityscapes), and *Source* means training IAST without warmup

Method	GTA5			SYNTHIA		
	Base	+IAST	Δ	Base	+IAST	Δ
AdaptSeg[27]	42.4	50.2	+7.8	46.7	54.7	+8.0
AdvEnt[29]	45.4	49.8	+4.4	48.0	55.1	+7.1
Source	35.6	48.8	+13.2	42.2	54.2	+12.0

Apply to other UDA methods. Because IAST has no special structure or model dependencies, it can be directly used to decorate other UDA methods. We chose two typical adversarial training methods, AdaptSeg[27] and AdvEnt[29] for experiments. As shown in Table 9, these two methods have significantly improved performance under the IAST framework.

Extension: other tasks. The self-training method can also be applied to semi-supervised semantic segmentation task. We use the same configuration as [11] in Cityscapes for semi-supervised training with different proportions of data as labeled data. As shown in Table 8, we have significantly better performance than [11] and [12].

6 Conclusions

In this paper, we propose an instance adaptive self-training framework for semantic segmentation UDA. Compared with other popular UDA methods, IAST still has a significant improvement in performance. Moreover, IAST is a method with no model or special structure dependency, which means that it can be easily applied to other UDA methods with almost no additional cost to improve performance. In addition, IAST can also be applied to semi-supervised semantic segmentation tasks, which also achieves state-of-the-art performance. We hope this work will prompt people to rethink the potential of self-training on UDA or semi-supervised learning tasks.

Acknowledgement

This work was supported in part by the Natural Science Foundation of Beijing Municipality under Grant 4182044.

References

1. Baktashmotlagh, M., Harandi, M.T., Lovell, B.C., Salzmann, M.: Unsupervised domain adaptation by domain invariant projection. In: ICCV. pp. 769–776 (2013) 2
2. Chen, C., Xie, W., Huang, W., Rong, Y., Ding, X., Huang, Y., Xu, T., Huang, J.: Progressive feature alignment for unsupervised domain adaptation. In: CVPR. pp. 627–636 (2019) 1
3. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. TPAMI **40**(4), 834–848 (2017) 5.1
4. Chen, Y., Li, W., Sakaridis, C., Dai, D., Van Gool, L.: Domain adaptive faster r-cnn for object detection in the wild. In: CVPR. pp. 3339–3348 (2018) 1
5. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR. pp. 3213–3223 (2016) 5.1
6. Du, L., Tan, J., Yang, H., Feng, J., Xue, X., Zheng, Q., Ye, X., Zhang, X.: Ssf-dan: Separated semantic feature based domain adaptation network for semantic segmentation. In: ICCV (2019) 1, 2, 6, 7
7. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016) 2
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016) 5.1
9. Hoffman, J., Tzeng, E., Park, T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A., Darrell, T.: Cycada: Cycle-consistent adversarial domain adaptation. In: ICML (2018) 1, 2
10. Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: ECCV. pp. 172–189 (2018) 1, 2
11. Hung, W.C., Tsai, Y.H., Liou, Y.T., Lin, Y.Y., Yang, M.H.: Adversarial learning for semi-supervised semantic segmentation. In: BMVC (2019) 8, 5.3
12. Kalluri, T., Varma, G., Chandraker, M., Jawahar, C.: Universal semi-supervised semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5259–5270 (2019) 8, 5.3
13. Kan, M., Shan, S., Chen, X.: Bi-shifting auto-encoder for unsupervised domain adaptation. In: ICCV. pp. 3846–3854 (2015) 2
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1097–1105 (2012) 2
15. Kukačka, J., Golkov, V., Cremers, D.: Regularization for deep learning: A taxonomy. arXiv preprint arXiv:1710.10686 (2017) 2
16. Lee, C.Y., Batra, T., Baig, M.H., Ulbricht, D.: Sliced wasserstein discrepancy for unsupervised domain adaptation. In: CVPR. pp. 10285–10295 (2019) 2
17. Li, J., Zhao, J., Chen, Y., Roy, S., Yan, S., Feng, J., Sim, T.: Multi-human parsing machines. In: ACM Multimedia. pp. 45–53 (2018) 2
18. Li, M., Zhou, Z.H.: Setred: Self-training with editing. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 611–621. Springer (2005) 2
19. Li, Y., Yuan, L., Vasconcelos, N.: Bidirectional learning for domain adaptation of semantic segmentation. In: CVPR (2019) 1, 1, 2, 5.3, 6, 7
20. Lian, Q., Lv, F., Duan, L., Gong, B.: Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In: CVPR. pp. 6758–6767 (2019) 1, 1, 2, 6, 7

21. Long, M., Cao, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: NIPS. pp. 1640–1650 (2018) 1, 2
22. Luo, Y., Liu, P., Guan, T., Yu, J., Yang, Y.: Significance-aware information bottleneck for domain adaptive semantic segmentation. In: ICCV (2019) 5.3, 6, 7
23. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: ECCV. pp. 102–118. Springer (2016) 5.1
24. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: CVPR. pp. 3234–3243 (2016) 5.1
25. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR. pp. 2818–2826 (2016) 2
26. Triguero, I., García, S., Herrera, F.: Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. KAIS **42**(2), 245–284 (2015) 2
27. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. In: CVPR. pp. 7472–7481 (2018) 1, 1, 2, 5.1, 5.3, 6, 7, 9, 5.3
28. Tsai, Y.H., Sohn, K., Schulter, S., Chandraker, M.: Domain adaptation for structured output via discriminative patch representations. In: CVPR (2019) 1, 6, 7
29. Vu, T.H., Jain, H., Bucher, M., Cord, M., Pérez, P.: Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In: CVPR (2019) 1, 1, 2, 4.2, 6, 7, 9, 5.3
30. Yang, J., Xu, R., Li, R., Qi, X., Shen, X., Li, G., Lin, L.: An adversarial perturbation oriented domain adaptation approach for semantic segmentation. In: AAAI (2020) 6, 7
31. Zhang, Q., Zhang, J., Liu, W., Tao, D.: Category anchor-guided unsupervised domain adaptation for semantic segmentation. In: NeurIPS. pp. 433–443 (2019) 1
32. Zhao, J., Cheng, Y., Xu, Y., Xiong, L., Li, J., Zhao, F., Jayashree, K., Pranata, S., Shen, S., Xing, J., et al.: Towards pose invariant face recognition in the wild. In: CVPR. pp. 2207–2216 (2018) 2
33. Zhao, J., Li, J., Nie, X., Zhao, F., Chen, Y., Wang, Z., Feng, J., Yan, S.: Self-supervised neural aggregation networks for human parsing. In: CVPRW. pp. 7–15 (2017) 2
34. Zheng, Z., Yang, Y.: Unsupervised scene adaptation with memory regularization in vivo. arXiv preprint arXiv:1912.11164 (2019) 1, 1, 6, 7
35. Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J.: Confidence regularized self-training. In: ICCV. pp. 5982–5991 (2019) 1, 1, 2, 4.2, 5.3, 6, 7
36. Zou, Y., Yu, Z., Vijaya Kumar, B., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: ECCV. pp. 289–305 (2018) 1, 1, 2, 5, 5.2, 6, 7