Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-training

Yang Zou[1](✉), Zhiding Yu[2], B. V. K. Vijaya Kumar[1], and Jinsong Wang[3]

[1] Carnegie Mellon University, Pittsburgh, PA 15213, USA
yzou2@andrew.cmu.edu, kumar@ece.cmu.edu
[2] NVIDIA, Santa Clara, CA 95051, USA
zhidingy@nvidia.com
[3] General Motors R & D, Warren, MI 48092, USA
jinsong.wang@gm.com

Abstract. Recent deep networks achieved state of the art performance on a variety of semantic segmentation tasks. Despite such progress, these models often face challenges in real world "wild tasks" where large difference between labeled training/source data and unseen test/target data exists. In particular, such difference is often referred to as "domain gap", and could cause significantly decreased performance which cannot be easily remedied by further increasing the representation power. Unsupervised domain adaptation (UDA) seeks to overcome such problem without target domain labels. In this paper, we propose a novel UDA framework based on an iterative self-training (ST) procedure, where the problem is formulated as latent variable loss minimization, and can be solved by alternatively generating pseudo labels on target data and re-training the model with these labels. On top of ST, we also propose a novel class-balanced self-training (CBST) framework to avoid the gradual dominance of large classes on pseudo-label generation, and introduce spatial priors to refine generated labels. Comprehensive experiments show that the proposed methods achieve state of the art semantic segmentation performance under multiple major UDA settings.

1 Introduction

Semantic segmentation is a core computer vision task where one aims to densely assign labels to each pixel in the input image. In the past decade, significant amount of effort has been devoted to this area [1,5,6,9,10,13,20,38,39,44,45], leading to considerable progress with the recent advance of deep representation learning [15,19,31]. The competition on major open benchmark datasets [10]

Y. Zou and Z. Yu—Equal contribution.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01219-9_18) contains supplementary material, which is available to authorized users.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11207, pp. 297–313, 2018.
https://doi.org/10.1007/978-3-030-01219-9_18

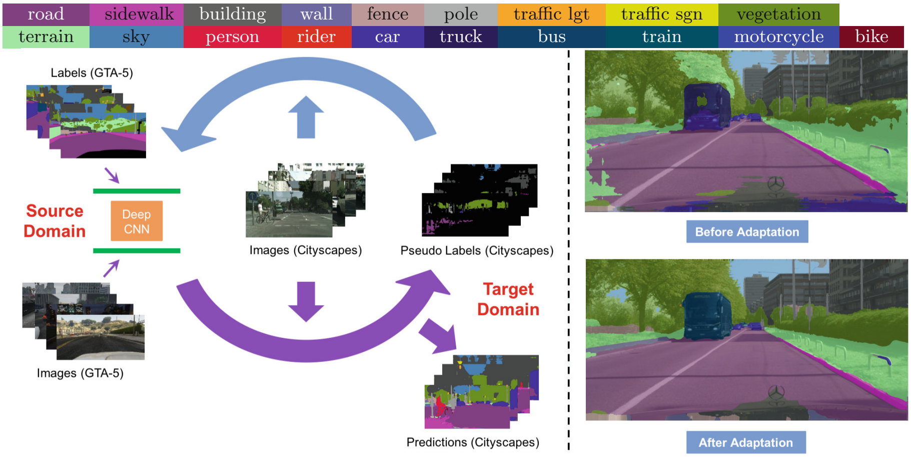

Fig. 1. Illustration of the proposed itertive self-training framework for unsupervised domain adaptation. Left: algorithm workflow. Right figure: semantic segmentation results on Cityscapes before and after adaptation.

have resulted in a number of more powerful models that tend to overfit to the benchmark data. While the boundaries of benchmark performance have been pushed to new limits, these models often encounter challenges in practical applications such as autonomous driving, where one needs ubiquitous good performance of the perception module. This is because benchmark datasets are usually biased to specific environments, while the testing scenario may encounter large domain differences caused by a number of factors, including change of geological position, illumination, camera, weather condition, etc. In this case, even the performance of a powerful model often drops dramatically, and such issue can not be easily remediated by further building up the model power [9,16,17].

A natural idea to improve network's generalization ability is to collect and annotate data covering more diverse scenes. However, densely annotating image is time-consuming and labor-intensive. For example, each Cityscapes image on average takes about 90 min to annotate [10]. To overcome the limitation, efforts were made to efficiently generate densely annotated images from rendered scenes, such as the Grand Theft Auto V (GTA5) [24] and SYNTHIA [26]. However, the large appearance gap across simulated/real domains significantly degrades the performance of synthetically trained models.

In light of the above issues, in this paper we focus on the challenging problem of unsupervised domain adaptation for semantic segmentation, aiming to unsupervisedly adapt a segmentation model trained on a labeled source domain to a target domain without knowing target labels. Recently, unsupervised domain adaptation has been widely explored for classification and detection tasks. There is a predominant trend to use adversarial training based methods for matching the distributions of both source and target features [3,9,12,17,29]. In particular, these methods aim to minimize a domain adversarial loss to reduce both the global and class-wise discrepancy between source and target feature distributions, while retaining good performance on source domain task by minimizing the task-specific loss.

Adversarial training based domain adaptation methods have recently achieved great success. However, in this work we show that similar or even better adaptation performance can be achieved by taking an alternative way without using adversarial training. Rather than trying to adapt by confusing the domain discriminator, our method kind of unifies the feature space alignment and the task itself together under a single, unified loss, which is given in Sect. 4. Under the single unified loss, we incorporate both the global and class-wise feature alignment as parts of our unified task, instead of considering feature matching and classification task separately.

Traditional self-training methods with handcrafted features is a common semi-supervised learning method that can learn better decision boundary for source and target data. Usually these approaches do not consider feature distribution matching. But combined with CNN, self-training becomes a powerful domain adaptation method that can not only learn better decision boundary, but also find a feature space of matched source and target distribution. In essence, the feature learning in self-training guided by softmax cross-entropy loss not only encourages global closeness of source and target features but also the class-wise feature alignment. The CNN based self-training methods share the same goal of adversarial training based global and class-wise feature alignment methods [9,17], but it try to solve domain adaption by a simpler and more elegant way.

The area of self-training based domain adaptation for semantic segmentation is underdevelopment. We propose a typical CNN based self-training (ST) framework for domain adaptation in semantic segmentation of which workflow is shown in Fig. 1, taking adapting from GTA5 → Cityscapes as an example. ST is carried out by alternately generating a set of pseudo-labels corresponding to large selection scores (i.e., softmax probability) in target domain, and then fine tuning network based on these pseudo-labels and labeled source data. It should be mentioned that ST assumes that target samples with larger prediction probability have better prediction accuracy.

The visual (e.g., appearance, scale, etc.) domain gap between source and target domains are usually different between classes. This can result in different difficulty degree for the network to learn transferable knowledge for each class. For instance, different countries may have different construction views and plants, but traffic lights and vehicles are similar. So it's harder for the source pre-trained models to learn transferable knowledge for construction and plants than for traffic lights and vehicles. Moreover, the imbalanced class distribution of source domain, and difference between source distribution and target distribution can also cause different degree of difficulty in transferring knowledge among different classes. This causes different prediction confidence levels for various classes in target domain. Since ST selects pseudo-labels with large confidence, it tends to be biased towards easy-to-transfer classes ignoring other classes and have inferior adaptation performance.

In summary, we focus on self-training based adaptation methods for semantic segmentation in this work. Our contributions are as follows.

– Building on deep nets, we introduce a self-training (ST) with self-paced learning adaptation framework for segmentation. We formulate it as a loss minimization problem in the form of mixed integer nonlinear program, which can be solved in an end-to-end way. Both domain-invariant features and classifier are expected be learned.
– To solve the class imbalance problem of pseudo-labels in ST, we propose a novel class-balanced self-training (CBST) adaptation for semantic segmentation. The proposed CBST utilizes confidence scores normalized classwise to select and generate pseudo-labels with balanced class distribution.
– Moreover, we observe that a traffic scene has its own spatial structure and introduce the concept of spatial priors (SP). We incorporate spatial priors into proposed self-training leading to class-balanced self-training with spatial priors (CBST-SP). The probability scores weighted by spatial priors are used for pseudo-label generation metric.
– We comprehensively evaluate our approaches in adapting large-scale rendered image dataset SYNTHIA/GTA5, to real image dataset, Cityscapes, and achieve state-of-the-art performance, outperforming other methods by a large margin. Also we test our methods in cross city adaptation settings, Cityscapes to NTHU dataset, and achieve state-of-the-art performance.

2 Related Works

The revolution of deep learning inspired broad interest in deep neural network based semantic segmentation. Long et al. [20] proposed a fully convolutional network for pixel-level classification. Recently several researchers proposed powerful segmentation nets, such as ResNet-38, PSPNet, etc. [38,39,44].

Unsupervised domain adaptation has been widely investigated in computer vision primarily for classification and detection tasks. In the era of deep neural network, the main adaption idea is to learn domain invariant features by minimizing difference between source and target feature distributions in an end-to-end way [11,12,14,21,32,35,37]. Among them, several methods utilize Maximum Mean Discrepancy (MMD) and its kernel variants to achieve the goal of feature distribution difference minimization. Recently there is an increasing interest in utilizing adversarial learning based methods to reduce the gap between source and target domains [14,21,36,37].

Another important strategy for unsupervised domain adaptation is based on self-training [4,47], which has many applications in vision and natural language processing [22,25,40,47]. Tang et al. [33] proposed a self-paced adaptation to shift object detection model from images to videos by learning labeled source samples and target data with pseudo-labels in an easy-to-hard way. Chen et al. [7] proposed a adaptation framework by slowly adapting its training set from the source to the target domain, using ideas from co-training. Bekker [2] et al. tackle the noisy labels problem.

As pointed out in [43], approaches addressing classification do not translate well to the semantic segmentation problem. So recently domain adaptation for semantic segmentation has emerged as a hot topic. Several researchers have focused on utilizing adversarial learning to minimize the domain gap of feature spaces. [9,17] proposed pixel level adversarial domain adaptation methods to reduce domain gap in feature spaces. Based on the domain adversarial training, [28] introduced a critic network detecting samples near the boundary and a generator that can generate discriminative features for target domain. [43] proposed a curriculum adaption method to regularize the predicted label distribution in the target domain to follow label distributions in source domain. Another possible direction to solve the domain adaptation problem is to utilize style transfer technique to stylize annotated source domain images as target domain images. Following this idea, based on the style transfer network Cycle-GAN [16,46] proposed a cycle-consistent adaptation framework combining the cycle-consistent loss with adversarial loss to minimize both pixel level and feature level domain gap.

3 Preliminaries

3.1 Fine-Tuning for Supervised Domain Adaptation

If the labels for the same task in both source and target are available, possibly the most direct way to perform domain adaptation is supervised fine-tuning the model on both domains. For semantic segmentation nets with softmax output, the adaptation problem can be formulated as minimizing the following loss function:

$$\min_{\mathbf{w}} \mathcal{L}_S(\mathbf{w}) = -\sum_{s=1}^{S}\sum_{n=1}^{N} \mathbf{y}_{s,n}^{\top} \log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_s)) - \sum_{t=1}^{T}\sum_{n=1}^{N} \mathbf{y}_{t,n}^{\top} \log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_t)) \tag{1}$$

where $\mathbf{I}_s$ denotes the image in source domain indexed by $s = 1, 2, ..., S$, $\mathbf{y}_{s,n}$ the ground truth label for the n-th pixel ($n = 1, 2, ..., N$) in I_s, and $\mathbf{w}$ contains the network weights. $\mathbf{p}_n(\mathbf{w}, \mathbf{I}_s)$ is the softmax output containing the class probabilities at pixel n. Similar definitions apply for $\mathbf{I}_t$, $\mathbf{y}_{t,n}$ and $\mathbf{p}_n(\mathbf{w}, \mathbf{I}_t)$.

3.2 Self-training for Unsupervised Domain Adaptation

In the case of unsupervised domain adaptation, the target ground truth labels are not available. An alternate way to fine-tune the segmentation model is to consider the target labels as hidden variables that can be learned. Accordingly, the problem can be formulated as follows:

$$\begin{aligned}\min_{\mathbf{w},\hat{\mathbf{y}}} \mathcal{L}_U(\mathbf{w},\hat{\mathbf{y}}) = -\sum_{s=1}^{S}\sum_{n=1}^{N} \mathbf{y}_{s,n}^{\top} \log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_s)) - \sum_{t=1}^{T}\sum_{n=1}^{N} \hat{\mathbf{y}}_{t,n}^{\top} \log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_t)) \\ s.t.\ \hat{\mathbf{y}}_{t,n} \in \{\mathbf{e}^{(i)} | \mathbf{e}^{(i)} \in \mathbb{R}^C\}, \forall t, n\end{aligned} \tag{2}$$

where $\hat{\mathbf{y}}$ indicates the set of target labels, C is the number of classes, and $\mathbf{e}^{(i)}$ a one-hot vector. By minimizing the loss in Eq. (2) with respect to $\hat{\mathbf{y}}$, the optimized $\hat{\mathbf{y}}$ should approximate the underlying true target ground truth. Domain adaptation can then be performed similarly to Eq. (1). We call $\hat{\mathbf{y}}$ "pseudo-labels", and regard such training strategy as self-training.

4 Proposed Methods

4.1 Self-training (ST) with Self-paced Learning

Jointly learning the model and optimizing pseudo-labels on unlabeled data is naturally difficult as it is not possible to completely guarantee the correctness of the generated pseudo-labels. A better strategy is to follow an "easy-to-hard" scheme via self-paced curriculum learning, where one seeks to generate pseudo-labels from the most confident predictions and hope they are mostly correct. Once the model is updated and better adapted to the target domain, the scheme then explores the remaining pseudo-labels with less confidence. To incorporate curriculum learning, we consider the following revised self-training formulation:

$$\begin{aligned}\min_{\mathbf{w},\hat{\mathbf{y}}} \mathcal{L}_{ST}(\mathbf{w},\hat{\mathbf{y}}) = & -\sum_{s=1}^{S}\sum_{n=1}^{N} \mathbf{y}_{s,n}^{\top}\log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_s)) \\ & -\sum_{t=1}^{T}\sum_{n=1}^{N}\left[\hat{\mathbf{y}}_{t,n}^{\top}\log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_t)) + k|\hat{\mathbf{y}}_{t,n}|_1\right] \\ & s.t.\ \hat{\mathbf{y}}_{t,n} \in \{\{\mathbf{e}^{(i)}|\mathbf{e}^{(i)} \in \mathbb{R}^C\} \cup \mathbf{0}\}, \forall t, n \\ & \quad k > 0\end{aligned} \tag{3}$$

where assigning $\mathbf{y}_{s,n}$ as $\mathbf{0}$ leads to ignoring this pseudo-label in model training, and the L_1 regularization serves as a negative sparse promoting term to prevent the trivial solution of ignoring all pseudo-labels. k is a hyperparameter controlling the amount of ignored pseudo-labels. A larger k encourages the selection of more pseudo-labels for model training. To minimize the loss in Eq. (3), we take the following alternative block coordinate descent algorithm:

- (a) Fix (initialize) $\mathbf{w}$ and minimize the loss in Eq. 3 with respect to $\hat{\mathbf{y}}_{t,n}$.
- (b) Fix $\hat{\mathbf{y}}_{t,n}$ and optimize the objective in Eq. 3 with respect to $\mathbf{w}$.

We call one step of (a) followed by one step of (b) as one **round**. In this work, we propose a self-training algorithm where step (a) and step (b) are alternately repeated for multiple rounds. Intuitively, step (a) selects a certain portion of most confident pseudo-labels from the target domain, while step (b) trains the network model given the pseudo-labels selected in step (a). Figure 1 illustrates the proposed algorithm flow in the domain adaptation example of GTA5 → Cityscapes.

Solving step (b) leads to network learning with stochastic gradient descent. However, solving step (a) requires a nonlinear integer programming given the optimization over discrete variables. Given $k > 0$, step (a) can be rewritten as:

$$\begin{aligned}&\min_{\hat{\mathbf{y}}} -\sum_{t=1}^{T}\sum_{n=1}^{N}\Big[\sum_{c=1}^{C}\hat{y}_{t,n}^{(c)}\log(p_n(c|\mathbf{w},\mathbf{I}_t)) + k|\hat{\mathbf{y}}_{t,n}|_1\Big]\\ &\text{s.t. } \hat{\mathbf{y}}_{t,n} = \big[\hat{y}_{t,n}^{(1)},...,\hat{y}_{t,n}^{(C)}\big] \in \{\{\mathbf{e}^{(i)}|\mathbf{e}^{(i)} \in \mathbb{R}^C\} \cup \mathbf{0}\}, \forall\, t,n\\ &\quad\; k > 0\end{aligned} \tag{4}$$

Since $\hat{\mathbf{y}}_{t,n}$ is required to be either a discrete one-hot vector or a zero vector, the pseudo-label configuration can be optimized via the following solver:

$$\hat{y}_{t,n}^{(c)*} = \begin{cases} 1, \textbf{ if } c = \arg\max_{c} p_n(c|\mathbf{w},\mathbf{I}_t), \\ \qquad p_n(c|\mathbf{w},\mathbf{I}_t) > \exp(-k) \\ 0, \text{otherwise}\end{cases} \tag{5}$$

Unlike traditional self-training adaptation with handcrafted features that learn a domain-invariant classifier, CNN based self-training can learn not only domain-invariant classifier but also domain-invariant features. The softmax loss implicitly tries to reduce the domain difference in feature space. In addition, the self-training also has the missing value (pseudo-label) problem, similar to EM algorithm. The proposed alternate optimization method can learn the weights of models without prior observation of target domain labels.

One may note that the proposed framework is similar to [33] and several other related works. However, the proposed method presents a more generalized model for self-training and self-paced learning, in the sense that pseudo-label generation is unified with curriculum learning under a single learning framework. More importantly, in terms of the specific application, the above self-training framework sheds light on a relatively new direction for adapting semantic segmentation models. We will show that self-training based methods lead to considerably better or competitive performance compared to many current state of the art methods that are predominantly based on adversarial training.

4.2 Class-Balanced Self-training (CBST)

As mentioned in Sect. 1, the difference in visual domain gap and class distribution can cause different domain-transfer difficulty among classes, resulting in relatively higher prediction confidence scores for easy-to-transfer classes in target domain. Since ST generates pseudo-labels corresponding to large confidence, an issue comes out that model tends to be biased towards these initially well-transferred classes and ignore other hard classes along the training process. Thus it is difficult for ST to perform well in multi-class segmentation adaptation problem. To overcome this issue, we propose the following class-balanced self-training framework where class-wise confidence levels are normalized:

$$\begin{aligned}\min_{\mathbf{w},\hat{\mathbf{y}}} \mathcal{L}_{CB}(\mathbf{w},\hat{\mathbf{y}}) = &-\sum_{s=1}^{S}\sum_{n=1}^{N}\mathbf{y}_{s,n}^{\top}\log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_s)) \\ &-\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C}\left[\hat{y}_{t,n}^{(c)}\log(p_n(c|\mathbf{w},\mathbf{I}_t)) + k_c\hat{y}_{t,n}^{(c)}\right] \\ s.t.\ \hat{\mathbf{y}}_{t,n} = &\left[\hat{y}_{t,n}^{(1)},...,\hat{y}_{t,n}^{(C)}\right] \in \{\{\mathbf{e}^{(i)}|\mathbf{e}^{(i)}\in\mathbb{R}^C\}\cup\mathbf{0}\}, \forall t,n \\ k_c > 0, \forall c&\end{aligned} \tag{6}$$

where each k_c is a separate parameter determining the proportion of selected pseudo-labels in class c. As one may observe, it is the difference between k_c that introduces different levels of class-wise bias for pseudo-label selection, and addresses the issue of inter-class balance.

The optimization flow of class-balanced self-training is the same as in Eq. (3) except for pseudo-label generation. Again, we can rewrite the step of pseudo-label optimization as:

$$\begin{aligned}&\min_{\hat{\mathbf{y}}} -\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C}\left[\hat{y}_{t,n}^{(c)}\log(p_n(c|\mathbf{w},\mathbf{I}_t)) + k_c\hat{y}_{t,n}^{(c)}\right] \\ &s.t.\ \hat{\mathbf{y}}_{t,n} = \left[\hat{y}_{t,n}^{(1)},...,\hat{y}_{t,n}^{(C)}\right] \in \{\{\mathbf{e}|\mathbf{e}\in\mathbb{R}^C\}\cup\mathbf{0}\}, \forall\ t,n \\ &\quad k_c > 0, \forall\ c\end{aligned} \tag{7}$$

Note that the loss function in Eq. (7) can not be trivially minimized by the solver of Eq. (3). Instead, optimizing Eq. (7) requires the following class-balanced solver:

$$\hat{y}_{t,n}^{(c)*} = \begin{cases} 1, \textbf{ if } c = \arg\max\limits_{c}\dfrac{p_n(c|\mathbf{w},\mathbf{I}_t)}{\exp(-k_c)}, \\ \qquad \dfrac{p_n(c|\mathbf{w},\mathbf{I}_t)}{\exp(-k_c)} > 1 \\ 0, \text{otherwise} \end{cases} \tag{8}$$

From Eq. (8), one can see that pseudo-label generation in Eq. (6) is no longer dependent on the output $p_n(c|\mathbf{w},\mathbf{I}_t)$, but hinges on the normalized output $\frac{p_n(c|\mathbf{w},\mathbf{I}_t)}{\exp(-k_c)}$. Pseudo-label assignment using this normalized output owns the benefit of balancing towards the class with relatively low score but having high within-class confidence. As a result, k_c should be set in a way that $\exp(-k_c)$ encodes the response strength of each class to balance different classes. In addition, for CBST, the pseudo-label of any pixel is only filtered when all the balanced responses are smaller than 1. There could also be multiple classes with $\frac{p_n(c|\mathbf{w},\mathbf{I}_t)}{\exp(-k_c)} > 1$. In this case, the class with the maximum balanced response is selected.

4.3 Self-paced Learning Policy Design

Determination of k in ST. From previous sections, we know that k plays a key role in filtering out pseudo-labels with probabilities smaller than k. To control the proportion of selected pseudo-labels in each round, we set k based on the following strategy:

We take the maximum output probability at each pixel, and sort such probabilities across all pixel locations and all target images in descending order. We then set k such that $\exp(-k)$ equals the probability ranked at $\text{round}(p * T * N)$, where p is a proportion number between $[0, 1]$. In this case, pseudo-label optimization produces $p \times 100\%$ most confident pseudo-labels for network training. The above policy can be summarized in Algorithm 1.

Algorithm 1. Determination of k in ST

```
Input  : Neural network $P(\mathbf{w})$, all target images $\mathbf{I}_t$, portion $p$ of selected
         pseudo-labels
Output: k
for $t=1$ to $T$ do
    $\mathrm{P}_{\mathbf{I}_t} = \mathrm{P}(\mathbf{w},\mathbf{I}_t)$
    $\mathrm{MP}_{\mathbf{I}_t} = \max(\mathrm{P}_{\mathbf{I}_t},\text{axis=0})$
    $\mathrm{M} = [\mathrm{M}, \text{matrix\_to\_vector}(\mathrm{MP}_{\mathbf{I}_t})]$
end
M = sort(M,order=descending)
$\mathrm{len}_{th} = \mathrm{length(M)} \times \mathrm{p}$
$\mathrm{k} = -\log(\mathrm{M}[\mathrm{len}_{th}])$
return k
```

We design the self-paced learning policy such that more pseudo-labels are incorporated for each additional round. In particular, we start p from 20%, and empirically add 5% to p in each additional round of pseudo-label generation. The maximum portion is set to be 50%.

Determination of k_c in CBST. The policy of k_c in CBST is similarly defined. Although CBST seemingly introduce much more parameters than ST, we propose a strategy to easily determine k_c, and effectively encode the class-wise confidence levels.

Note that Algorithm 2 determines k_c by ranking the class c probabilities on all pixels predicted as class c, and setting k_c such that $\exp(-k_c)$ equals to the probability ranked at $\text{round}(p * N_c)$, where N_c indicates the number of pixels predicted as class c. Such a strategy basically takes the probability ranked at $p \times 100\%$ separately from each class as a reference for both thresholding and confidence normalization. The proportion variable p and its increasing policy is defined exactly the same to ST.

4.4 Incorporating Spatial Priors

For adapting models in the case of street scenes, we could take advantage of the spatial prior knowledge. Traffic scenes have common structures. For example, sky is not likely to appear at the bottom and road is not likely to appear at the top. If the image views in source domain and target domain are similar, we believe this knowledge can help to adapt source model. Thus we introduce spatial priors, similar to [30], by counting the class frequencies in the source domain, followed by smoothing with a 70×70 Gaussian kernel. In particular, we use $q_n(c)$ to indicate the frequency of class c at pixel n. Upon obtaining the class frequencies, we also normalize them by requiring $\sum_{i=1}^{N} q_n(c) = 1$. Figure 2 shows the heat map of spatial priors, calculated from GTA5 dataset, where yellow color indicates higher energy and blue color indicates lower energy.

To incorporate spatial priors into proposed CBST, we multiply the softmax output with the spatial priors, and consider the resulting potential as selection metric in pseudo-label generation:

$$\begin{aligned}
\min_{\mathbf{w},\hat{\mathbf{y}}} \mathcal{L}_{SP}(\mathbf{w},\hat{\mathbf{y}}) = & -\sum_{s=1}^{S}\sum_{n=1}^{N} \mathbf{y}_{s,n}^{\top} \log(\mathbf{p}_n(\mathbf{w},\mathbf{I}_s)) \\
& -\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C} \left[\hat{y}_{t,n}^{(c)} \log(q_n(c) p_n(c|\mathbf{w},\mathbf{I}_t)) + k_c \hat{y}_{t,n}^{(c)}\right] \\
& s.t.\ \hat{\mathbf{y}}_{t,n} \in \{\{\mathbf{e}|\mathbf{e} \in \mathbb{R}^C\} \cup \mathbf{0}\}, \forall t, n \\
& \quad k_c > 0, \forall c
\end{aligned} \tag{9}$$

Algorithm 2. Determination of k_c in CBST

```
Input  : Neural network $f(\mathbf{w})$, all target images $\mathbf{I}_t$, portion $p$ of selected
         pseudo-labels
Output: $\mathbf{k}_c$
for $t$=1 to $T$ do
    $\mathrm{P}_{\mathbf{I}_t}$ = P($\mathbf{w}$,$\mathbf{I}_t$)
    $\mathrm{LP}_{\mathbf{I}_t}$ = argmax(P,axis=0)
    $\mathrm{MP}_{\mathbf{I}_t}$ = max(P,axis=0)
    for $c$=1 to $C$ do
        $\mathrm{MP}_{c,\mathbf{I}_t}$ = $\mathrm{MP}_{\mathbf{I}_t}$($\mathrm{LP}_{\mathbf{I}_t}$ == c)
        $\mathrm{M}_c$ = [$\mathrm{M}_c$, matrix_to_vector($\mathrm{MP}_{c,\mathbf{I}_t}$)]
    end
end
for $c$=1 to $C$ do
    $\mathrm{M}_c$ = sort($\mathrm{M}_c$,order=descending)
    $\mathrm{len}_{c,th}$ = length($\mathrm{M}_c$) × p
    $\mathrm{k}_c$ = -log($\mathrm{M}_c$[$\mathrm{len}_{c,th}$])
end
return $\mathbf{k}_c$
```

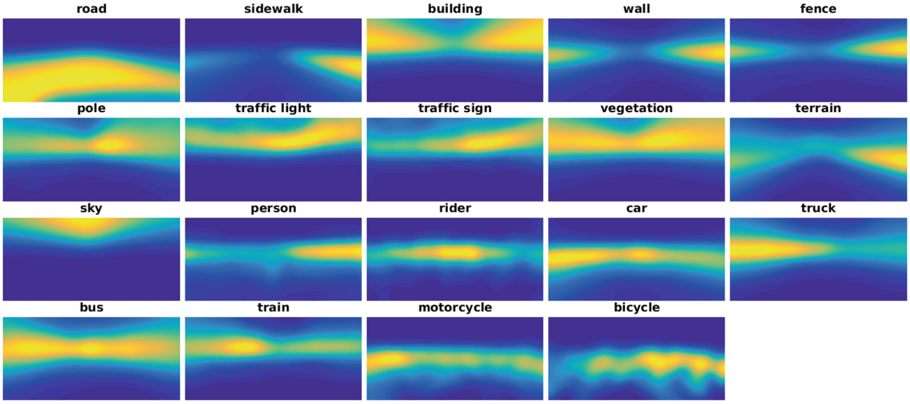

Fig. 2. Spatial priors on GTA5

We denote the above algorithm as CBST-SP. The workflow and self-paced learning policy are identical to CBST, except that the potential $q_n(c)p_n(c|\mathbf{w}, \mathbf{I}_t)$ is used to replace $p_n(c|\mathbf{w}, \mathbf{I}_t)$ in CBST. It should be noted that incorporating the spatial prior does not change network training, since $q_n(c)$ can be taken out of $\log(\cdot)$ as constants.

5 Numerical Experiments

In this section, we provide a comprehensive evaluation of proposed methods by performing experiments on three benchmark datasets. We firstly consider a cross-city adaptation case of shifting from Cityscapes to NTHU dataset [9]. Following [9], we choose the training set of Cityscapes as source. The NTHU dataset contains 400 $1,024 \times 2,048$ from 4 different cities: Rome, Rio, Tokyo and Taipei. Also we consider two challenging problems: from SYNTHIA [26] to Cityscapes [10] and from GTA5 [24] to Cityscapes. We use SYNTHIA-RAND-CITYSCAPES subset including labeled 9,400 760×1280 images. GTA5 dataset includes annotated 24,966 $1,052 \times 1,914$ images captured from the GTA5. The validation set of Cityscapes is treated as target domain.

Implementation Details. We use FCN8s-VGG16 [20] as our base network in SYNTHIA to Cityscapes and GTA5 to Cityscapes to give fair comparison with other methods utilizing the same base net. Further we boost our methods' performance via a better model ResNet-38 [39]. In the cross-city setting, we show state-of-the-art performance via CBST with ResNet-38. The networks were pre-trained on ImageNet [27]. SGD has been used to train all the models by MXNET [8]. We use NVIDIA Titan Xp. In the CBST and CBST-SP experiments of GTA5 to Cityscapes and Cityscapes to NTHU, we use a hard sample mining strategy which mines the least prediction classes according to target prediction

Table 1. Experimental results for Cityscapes → NTHU dataset

City	Method	Road	SW	Build	TL	TS	Veg.	Sky	PR	Rider	Car	Bus	Motor	Bike	Mean
Rome	Source Dilation-Frontend [9]	77.7	21.9	83.5	0.1	10.7	78.9	88.1	21.6	10.0	67.2	30.4	6.1	0.6	38.2
	GCAA [9]	79.5	29.3	84.5	0.0	22.2	80.6	82.8	29.5	13.0	71.7	37.5	25.9	1.0	42.9
	DeepLab-v2 [34]	83.9	34.3	87.7	13.0	41.9	84.6	92.5	37.7	22.4	80.8	38.1	39.1	5.3	50.9
	MAA [34]	83.9	34.2	88.3	**18.8**	40.2	**86.2**	**93.1**	47.8	21.7	80.9	**47.8**	48.3	8.6	**53.8**
	Source Resnet-38	86.0	21.4	81.5	14.3	47.4	82.9	59.8	30.8	20.9	83.1	20.2	40.0	5.6	45.7
	ST	85.9	20.2	84.3	15.0	46.4	84.9	73.5	**48.5**	21.6	84.6	17.6	46.2	6.7	48.9
	CBST	**87.1**	**43.9**	**89.7**	14.8	**47.7**	85.4	90.3	45.4	**26.6**	**85.4**	20.5	**49.8**	**10.3**	53.6
Rio	Source Dilation-Frontend [9]	69.0	31.8	77.0	4.7	3.7	71.8	80.8	38.2	8.0	61.2	38.9	11.5	3.4	38.5
	GCAA [9]	74.2	43.9	79.0	2.4	7.5	77.8	69.5	39.3	10.3	67.9	**41.2**	27.9	10.9	42.5
	DeepLab-v2 [34]	76.6	47.3	82.5	12.6	22.5	77.9	86.5	43.0	19.8	74.5	36.8	29.4	16.7	48.2
	MAA [34]	76.2	44.7	84.6	9.3	25.5	**81.8**	**87.3**	55.3	**32.7**	74.3	28.9	**43.0**	**27.6**	51.6
	Source Resnet-38	80.6	36.0	81.8	**21.0**	33.1	79.0	64.7	36.0	21.0	73.1	33.6	22.5	7.8	45.4
	ST	80.1	41.4	83.8	19.1	**39.1**	80.8	71.2	**56.3**	27.7	**79.9**	32.7	36.4	12.2	50.8
	CBST	**84.3**	**55.2**	**85.4**	19.6	30.1	80.5	77.9	55.2	28.6	79.7	33.2	37.6	11.5	**52.2**
Tokyo	Source Dilation-Frontend [9]	81.2	26.7	71.7	8.7	5.6	73.2	75.7	39.3	14.9	57.6	19.0	1.6	33.8	39.2
	GCAA [9]	83.4	**35.4**	72.8	12.3	12.7	77.4	64.3	42.7	21.5	64.1	**20.8**	8.9	40.3	42.8
	DeepLab-v2 [34]	83.4	35.4	72.8	12.3	12.7	77.4	64.3	42.7	21.5	64.1	**20.8**	8.9	40.3	42.8
	MAA [34]	81.5	26.0	77.8	**17.8**	26.8	82.7	**90.9**	55.8	**38.0**	72.1	4.2	24.5	**50.8**	**49.9**
	Source Resnet-38	83.8	26.4	73.0	6.5	27.0	80.5	46.6	35.6	22.8	71.3	4.2	10.5	36.1	40.3
	ST	83.1	27.7	74.8	7.1	29.4	**84.4**	48.5	**57.2**	23.3	**73.3**	3.3	22.7	45.8	44.6
	CBST	**85.2**	33.6	**80.4**	8.3	**31.1**	83.9	78.2	53.2	28.9	72.7	4.4	**27.0**	47.0	48.8
Taipei	Source Dilation-Frontend [9]	77.2	20.9	76.0	5.9	4.3	60.3	81.4	10.9	11.0	54.9	32.6	15.3	5.2	35.1
	GCAA [9]	78.6	28.6	80.0	13.1	7.6	68.2	82.1	16.8	9.4	60.4	34.0	26.5	9.9	39.6
	DeepLab-v2 [34]	78.6	28.6	80.0	13.1	7.6	68.2	82.1	16.8	9.4	60.4	34.0	26.5	9.9	39.6
	MAA [34]	81.7	29.5	**85.2**	**26.4**	15.6	**76.7**	**91.7**	31.0	12.5	71.5	**41.1**	47.3	27.7	49.1
	Source Resnet-38	84.9	26.0	80.1	8.3	**28.0**	73.9	54.4	18.9	26.8	71.6	26.0	48.2	14.7	43.2
	ST	83.1	23.5	78.2	9.6	25.4	74.8	35.9	**33.2**	27.3	75.2	32.3	52.2	28.8	44.6
	CBST	**86.1**	**35.2**	84.2	15.0	22.2	75.6	74.9	22.7	**33.1**	**78.0**	37.6	**58.0**	**30.9**	**50.3**

portions. The mining classes are the worst 5 classes and top priority are given to classes whose portions are smaller than 0.1%. Other more details are provided in supplementary document.

5.1 Small Shift: Cross City Adaptation

NTHU dataset contains 13 classes shared with Cityscapes. We follow the same protocol as [9] to use a 10-fold cross validation. The IoU (Intersection-over-Union) of each class and the mIoU (mean IoU) are reported. Table 1 shows the results. Our CBST achieves superior or competitive performance compared with state-of-the-art.

5.2 Large Shift: Synthetic to Real Adaptation

From SYNTHIA to Cityscapes. We follow the same evaluation protocol as other works [17,43], we choose 16 common classes between SYNTHIA and CITYSCAPES as our valid labels. There is another setting only considering 13 classes excluding wall, fence and pole [34].

Table 2 reports the results. mIoU* is the mean IoU of 13 classes, excluding the classes with *. With FCN8s-VGG16 as base model, our CBST provides competitive performance compared with other methods. Equipped with a better base net ResNet-38, CBST achieves the superior performance outperforming

Table 2. Experimental results for SYNTHIA → Cityscapes

Method	Base Net	Road	SW	Build	Wall*	Fence*	Pole*	TL	TS	Veg.	Sky	PR	Rider	Car	Bus	Motor	Bike	mIoU	mIoU*
Source only [17]	Dilation-Frontend	6.4	17.7	29.7	1.2	0.0	15.1	0.0	7.2	30.3	66.8	51.1	1.5	47.3	3.9	0.1	0.0	17.4	20.2
FCN wild [17]	[41]	11.5	19.6	30.8	4.4	0.0	20.3	0.1	11.7	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6	20.2	22.1
Source only [43]	FCN8s-VGG16	5.6	11.2	59.6	8.0	**0.5**	21.5	8.0	5.3	72.4	75.6	35.1	9.0	23.6	4.5	0.5	18.0	22.0	27.6
Curr. DA [43]	[20]	65.2	26.1	74.9	0.1	**0.5**	10.7	3.5	3.0	76.1	70.6	47.1	8.2	43.2	20.7	0.7	13.1	29.0	34.8
Source only	FCN8s-VGG16	24.1	19.1	68.5	0.9	0.3	16.4	5.7	10.8	75.2	76.3	43.2	15.2	26.7	15.0	5.9	8.5	25.7	30.3
GAN DA	[20]	79.1	31.1	77.1	3.0	0.2	22.8	6.6	15.2	77.4	78.9	47.0	14.8	67.5	16.3	6.9	13.0	34.8	40.8
Source only	DeepLab-v2 [34]	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
MAA	[34]	**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
Source only	FCN8s-VGG16	17.2	19.7	47.3	1.1	0.0	19.1	3.0	9.1	71.8	78.3	37.6	4.7	42.2	9.0	0.1	0.9	22.6	26.2
ST	[20]	0.2	14.5	53.8	1.6	0.0	18.9	0.9	7.8	72.2	80.3	48.1	6.3	67.7	4.7	0.2	4.5	23.9	27.8
CBST		69.6	28.7	69.5	12.1	0.1	25.4	11.9	13.6	82.0	81.9	49.1	14.5	66.0	6.6	3.7	32.4	35.4	36.1
Source only	ResNet-38	32.6	21.5	46.5	4.8	0.1	26.5	14.8	13.1	70.8	60.3	56.6	3.5	74.1	20.4	8.9	13.1	29.2	33.6
ST	[39]	38.2	19.6	70.2	3.9	0.0	31.9	17.6	17.2	82.4	68.3	63.1	5.3	78.4	11.2	0.8	7.5	32.2	36.9
CBST		53.6	23.7	75.0	**12.5**	0.3	**36.4**	**23.5**	**26.3**	**84.8**	74.7	**67.2**	17.5	**84.5**	28.4	15.2	**55.8**	**42.5**	**48.4**

Table 3. Experimental results for GTA5 → Cityscapes

Method	Base Net	Road	SW	Build	Wall	Fence	Pole	TL	TS	Veg.	Terrain	Sky	PR	Rider	Car	Truck	Bus	Train	Motor	Bike	mIoU
Source only [17]	Dilation-Frontend	31.9	18.9	47.7	7.4	3.1	16.0	10.4	1.0	76.5	13.0	58.9	36.0	1.0	67.1	9.5	3.7	0.0	0.0	0.0	21.2
FCN wild [17]	[41]	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0	27.1
Source only [43]	FCN8s-VGG16	18.1	6.8	64.1	7.3	8.7	21.0	14.9	16.8	45.9	2.4	64.4	41.6	17.5	55.3	8.4	5.0	6.9	4.3	13.8	22.3
Curr. DA [43]	[20]	74.9	22.0	71.7	6.0	11.9	8.4	16.3	11.1	75.7	13.3	66.5	38.0	9.3	55.2	18.8	18.9	0.0	16.8	16.6	28.9
Source only [16]	FCN8s-VGG16	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0	17.9
CyCADA [16]	[20]	85.2	37.2	76.5	21.8	15.0	23.8	22.9	21.5	80.5	31.3	60.7	50.5	9.0	76.9	17.1	28.2	4.5	9.8	0.0	35.4
Source only [16]	Dilated ResNet-26	42.7	26.3	51.7	5.5	6.8	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4	21.7
CyCADA [16]	[42]	79.1	33.1	77.9	23.4	17.3	32.1	33.3	31.8	81.5	26.7	69.0	62.8	14.7	74.5	20.9	25.6	6.9	18.8	20.4	39.5
Source only [28]	ResNet-50	64.5	24.9	73.7	14.8	2.5	18.0	15.9	0	74.9	16.4	72.0	42.3	0.0	39.5	8.6	13.4	0.0	0.0	0.0	25.3
ADR [28]	[15]	87.8	15.6	77.4	20.6	9.7	19.0	19.9	7.7	82.0	31.5	74.3	43.5	9.0	77.8	17.5	27.7	1.8	9.7	0.0	33.3
Source only [23]	DenseNet	67.3	23.1	69.4	13.9	14.4	21.6	19.2	12.4	78.7	24.5	74.8	49.3	3.7	54.1	8.7	5.3	2.6	6.2	1.9	29.0
I2I Adapt [23]	[18]	85.8	37.5	80.2	23.3	16.1	23.0	14.5	9.8	79.2	**36.5**	**76.4**	53.4	7.4	82.8	19.1	15.7	2.8	13.4	1.7	35.7
Source only [34]	DeepLab-v2	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
MAA [34]	[18]	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
Source only	FCN8s-VGG16	64.0	22.1	68.6	13.3	8.7	19.9	15.5	5.9	74.9	13.4	37.0	37.7	10.3	48.2	6.1	1.2	1.8	10.8	2.9	24.3
ST	[17]	83.8	17.4	72.1	14.3	2.9	16.5	16.0	6.8	81.4	24.2	47.2	40.7	7.6	71.7	10.2	7.6	0.5	11.1	0.9	28.1
CBST		66.7	26.8	73.7	14.8	9.5	28.3	25.9	10.1	75.5	15.7	51.6	47.2	6.2	71.9	3.7	2.2	5.4	18.9	32.4	30.9
CBST-SP		**90.4**	50.8	72.0	18.3	9.5	27.2	28.6	14.1	82.4	25.1	70.8	42.6	14.5	76.9	5.9	12.5	1.2	14.0	28.6	36.1
Source only	ResNet-38	70.0	23.7	67.8	15.4	18.1	40.2	41.9	25.3	78.8	11.7	31.4	**62.9**	**29.8**	60.1	21.5	26.8	7.7	28.1	12.0	35.4
ST	[39]	90.1	56.8	77.9	28.5	23.0	41.5	45.2	39.6	84.8	26.4	49.2	59.0	27.4	82.3	39.7	45.6	**20.9**	**34.8**	**46.2**	41.5
CBST		86.8	46.7	76.9	26.3	**24.8**	42.0	46.0	38.6	80.7	15.7	48.0	57.3	27.9	78.2	24.5	49.6	17.7	25.5	45.1	45.2
CBST-SP		88.0	56.2	77.0	27.4	22.4	40.7	47.3	**40.9**	82.4	21.6	60.3	50.2	20.4	**83.8**	35.0	**51.0**	15.2	20.6	37.0	46.2
CBST-SP+MST		89.6	**58.9**	78.5	**33.0**	22.3	**41.4**	**48.2**	39.2	**83.6**	24.3	65.4	49.3	20.2	83.3	**39.0**	48.6	12.5	20.3	35.3	**47.0**

state-of-the-art by 1.7. Compared with ST, CBST with either FCN8s-VGG16 or ResNet-38 achieves better performance for mIoU and IoU of these initially not well-transfered classes, such as wall, rider, motorcycle and bike. The appearance of fence in SYNTHIA (car barriers) is extremely different from the fence in Cityscapes (pedestrian barriers) and it's very hard for the model to learn transferable knowledge for fence from SYNTHIA to Cityscapes. Figure 3 gives the visualization segmentation results in Cityscapes.

From GTA5 to Cityscapes. Table 3 gives experimental results of the shared 19 classes. For the results with FCN8s-VGG16 as base model, the performance of ST demonstrates that the adapted model can be easily biased towards initial easy-to-transfer classes. However, the CBST not only achieves better mIoU than ST, but also better IoU for these initial hard-to-transfer classes. Moreover, since images from GTA5 and Cityscapes have similar view structure, we evaluate our proposed CBST-SP, achieving mIoU 36.1, which is better than the results using powerful base models, ResNet-50 [28] and DenseNet [23]. Equipped with a powerful model ResNet-38, our method get a much better score 46.2, outper-

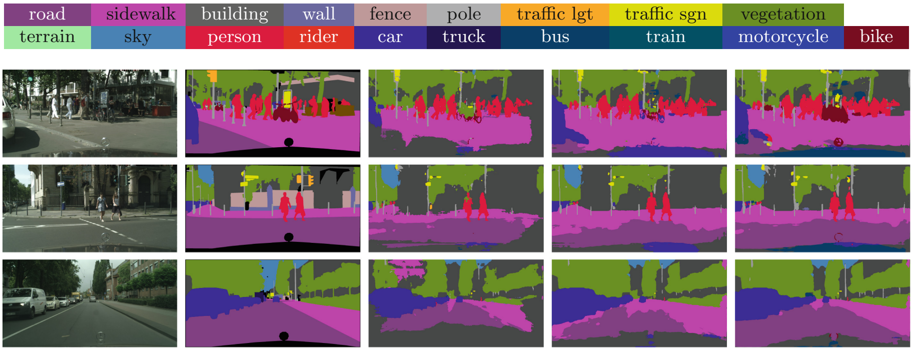

Fig. 3. Adaptation results on SYNTHIA → Cityscapes. Rows correspond to predictions for sample images in Cityscapes. Columns correspond to original images, ground truth, and results of source ResNet-38, ST, CBST

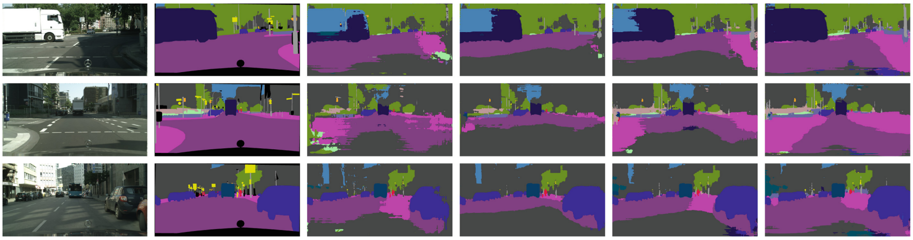

Fig. 4. Adaptation results on GTA5 → Cityscapes. Rows correspond to predictions for sample images in Cityscapes. Columns correspond to original images, ground truth, and results of source ResNet-38, ST, CBST and CBST-SP

forming other methods by a large margin. The multi-scale testing (0.5,0.75,1.0) boosts the mIoU to 47.0. Figure 4 gives the visualization segmentation results in Cityscapes.

6 Conclusions

In this paper, we proposed a deep neural network based self-training (ST) framework for unsupervised domain adaptation in the context of semantic segmentation. The ST is formulated as a loss minimization problem allowing learning of domain-invariant features and classifier in an end-to-end way. A class-balanced self-training (CBST) is introduced to overcome the imbalance issue of transferring difficulty among classes via generating pseudo-labels with balanced class distribution. Moreover, if there is a small domain difference in image view, we could incorporate spatial priors (SP) into CBST, resulting in CBST-SP. We experimentally demonstrate that our proposed methods achieve superior results outperforming other state-of-the-art methods by a large margin. We empirically show our proposed methods is compatible with adversarial domain adaptation methods.

References

1. http://data-bdd.berkeley.edu/
2. Bekker, A.J., Goldberger, J.: Training deep neural-networks based on unreliable labels. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2682–2686. IEEE (2016)
3. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: Advances in Neural Information Processing Systems, pp. 343–351 (2016)
4. Chapelle, O., Scholkopf, B., Zien, A.: Semi-supervised learning. IEEE Trans. Neural Netw. **20**(3), 542–542 (2009). (Chapelle, O. et al. (eds.) 2006) [book reviews]
5. 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. arXiv preprint arXiv:1606.00915 (2016)
6. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
7. Chen, M., Weinberger, K.Q., Blitzer, J.: Co-training for domain adaptation. In: Advances in Neural Information Processing Systems, pp. 2456–2464 (2011)
8. Chen, T., et al.: MXNet: a flexible and efficient machine learning library for heterogeneous distributed systems. arXiv preprint arXiv:1512.01274 (2015)
9. Chen, Y.H., Chen, W.Y., Chen, Y.T., Tsai, B.C., Frank Wang, Y.C., Sun, M.: No more discrimination: cross city adaptation of road scene segmenters. In: The IEEE International Conference on Computer Vision (ICCV), October 2017
10. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3213–3223 (2016)
11. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International Conference on Machine Learning, pp. 1180–1189 (2015)
12. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(59), 1–35 (2016)
13. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3354–3361. IEEE (2012)
14. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B.: Covariate shift and local learning by distribution matching, pp. 131–160. MIT Press, Cambridge (2009)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
16. Hoffman, J., et al.: CyCADA: cycle-consistent adversarial domain adaptation. arXiv preprint arXiv:1711.03213 (2017)
17. Hoffman, J., Wang, D., Yu, F., Darrell, T.: FCNs in the wild: pixel-level adversarial and constraint-based adaptation. arXiv preprint arXiv:1612.02649 (2016)
18. Huang, G., Liu, Z.: Densely connected convolutional networks
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (2012)

20. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
21. Long, M., Cao, Y., Wang, J., Jordan, M.: Learning transferable features with deep adaptation networks. In: International Conference on Machine Learning, pp. 97–105 (2015)
22. Maeireizo, B., Litman, D., Hwa, R.: Co-training for predicting emotions with spoken dialogue data. In: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, p. 28. Association for Computational Linguistics (2004)
23. Murez, Z., Kolouri, S., Kriegman, D., Ramamoorthi, R., Kim, K.: Image to image translation for domain adaptation. arXiv preprint arXiv:1712.00479 (2017)
24. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: ground truth from computer games. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 102–118. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_7
25. Riloff, E., Wiebe, J., Wilson, T.: Learning subjective nouns using extraction pattern bootstrapping. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, vol. 4, pp. 25–32. Association for Computational Linguistics (2003)
26. 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3234–3243 (2016)
27. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
28. Saito, K., Ushiku, Y., Harada, T., Saenko, K.: Adversarial dropout regularization. arXiv preprint arXiv:1711.01575 (2017)
29. Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S.N., Chellappa, R.: Unsupervised domain adaptation for semantic segmentation with gans. arXiv preprint arXiv:1711.06969 (2017)
30. Silberman, N., Fergus, R.: Indoor scene segmentation using a structured light sensor. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp. 601–608. IEEE (2011)
31. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015)
32. Sun, B., Saenko, K.: Deep CORAL: correlation alignment for deep domain adaptation. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 443–450. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_35
33. Tang, K., Ramanathan, V., Fei-Fei, L., Koller, D.: Shifting weights: adapting object detectors from image to video. In: Advances in Neural Information Processing Systems, pp. 638–646 (2012)
34. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. arXiv preprint arXiv:1802.10349 (2018)
35. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4068–4076 (2015)
36. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Adversarial discriminative domain adaptation. In: Computer Vision and Pattern Recognition (CVPR) (2017)

37. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)
38. Wang, P., et al.: Understanding convolution for semantic segmentation. arXiv preprint arXiv:1702.08502 (2017)
39. Wu, Z., Shen, C., van den Hengel, A.: Wider or deeper: revisiting the ResNet model for visual recognition. arXiv preprint arXiv:1611.10080 (2016)
40. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, pp. 189–196. Association for Computational Linguistics (1995)
41. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122 (2015)
42. Yu, F., Koltun, V., Funkhouser, T.: Dilated residual networks. In: Computer Vision and Pattern Recognition, vol. 1 (2017)
43. Zhang, Y., David, P., Gong, B.: Curriculum domain adaptation for semantic segmentation of urban scenes. In: The IEEE International Conference on Computer Vision (ICCV), October 2017
44. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017
45. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ADE20K dataset. arXiv preprint arXiv:1608.05442 (2016)
46. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks
47. Zhu, X.: Semi-supervised learning literature survey (2005)