Supplemental Material
Tune it the Right Way: Unsupervised Validation of Domain Adaptation via Soft Neighborhood Density

We first describe the details of the experiments. Then we show additional experimental results and analysis.

1. Experimental Details

Dataset. In partial DA setting using OfficeHome, we choose the first 25 classes (in alphabetic order) as the target private classes following [1]. In the experiments using DomainNet, we choose 126 classes out of 345 classes following [6]. This is to remove classes that include some outlier objects or objects of multiple classes.

Implementation. As we mention in the main paper, we utilize official implementations to perform experiments. Specifically, we use the following implementations; NC [7] [1], CDAN [4] [2], MCC [3] [3], AdaptSeg [9] [4], and ADVENT [10] [5]. We employ the configurations used by these implementations and tune the hyper-parameters described in the main paper. We will publish our implementation including these code.

For NC [7], we tune the temperature parameter (Eq. 4 and 5 in [7]) used to compute similarity distribution. We multiple the τ with [0.5, 0.8, 1.0, 1.5] to find a optimal one.

Semantic Segmentation. We describe the detail of an experiment on semantic segmentation. First, we aim to tune a weight of trade-off between the source classification loss and domain confusion loss in this experiment.

In AdaptSegNet [9], the implementation defines two weights for two domain classifiers individually. We aim to tune a weight called *lambda-adv-target1*. We aim to select the hyper-parameters from ($\lambda = 5.0 \times 10^{-4}$, 3.0×10^{-4}, 2.0×10^{-4}, 1.0×10^{-4}, 1.0×10^{-3}, with 2.0×10^{-4} as its default setting). Similarly, for ADVENT [10], we aim to pick a trade-off parameter called *LAMBDA-ADV-MAIN* from ($\lambda = 5.0 \times 10^{-2}$, 1.0×10^{-2}, 1.0×10^{-3}, 5.0×10^{-4}, and 1.0×10^{-4}, with 1.0×10^{-2} as its default setting).

[1] https://github.com/VisionLearningGroup/DANCE
[2] https://github.com/thuml/CDAN
[3] https://github.com/thuml/Versatile-Domain-Adaptation
[4] https://github.com/wasidennis/AdaptSegNet
[5] https://github.com/valeoai/ADVENT

To compute source risk, we utilize 1,000 training source images as a source validation set and track mIoU over training iterations.

Toy Dataset. We utilize the implementation of DANN [2] [6]. We simply use their network architecture and other configurations. We generate source data from two Gaussian distributions with different means ((0,0) and (5,5)), which we regard as two classes. Then, we obtain target data by shifting the mean of one of the Gaussians.

We will also publish the implementation modified for our experiment.

2. Analysis in Toy Dataset

Using the toy dataset, we show several characteristics of SND. First, SND gets large if features have small within-class variance compared to the distance between classes. Second, SND outputs a large value when samples are generated from a single cluster.

SND and Within-class Variance. Soft Neighborhood Density is designed to be large if the neighborhood samples are densely clustered, which means the feature variance within each cluster is small. We aim to confirm the relationship between SND and the within-class variance using the toy dataset. As shown in Fig. A, we vary the variance of the Gaussian distributions that generate data of two classes while fixing their means. Note that we train and test a model on the same distribution since our goal here is to observe the behavior of SND for the different variance of features. The right of Fig. A illustrates the result. As we expect, as the variance is increased, SND gets smaller. The accuracy also drops with the increase of the variance since the increase makes many hard-to-classify samples.

SND and the Mode of the Data. In this experiment, we investigate the relationship between SND and the number of clusters in the target. Note that we assume we have a fixed number of target samples. As the number of clusters gets smaller, more target samples get similar since the total number of target samples is the same, and SND gets larger.

[6] https://github.com/GRAAL-Research/domain_adversarial_neural_network

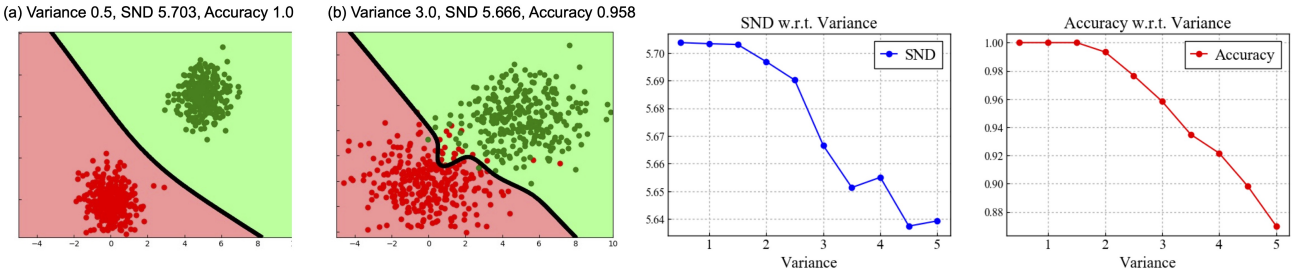

Figure A: SND with respect to with-in class variance. Left (a)(b): Plots of changing the variance. We generate the data of two classes from two Gaussians with different means. As we show in the plot, we increase the variance of them while fixing their means. In this way, we observe the behavior of SND by the change of the feature density. Right: The change of SND and accuracy with respect to the variance. Since the concentration degree of features decreases with the increase of the variance, SND gets smaller with the increase.

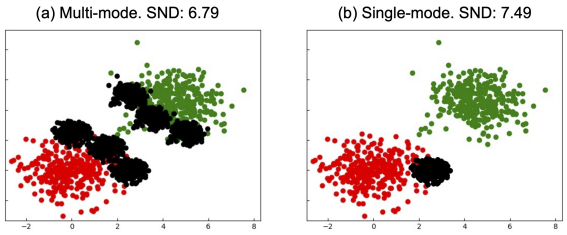

Figure B: SND shows a large value for the data with a single mode. Black: Target samples. Red: Source samples of class 0. Green: Source samples of class 1. Left: Target samples are generated from 6 modes. Right: The same number of target samples as the left are generated from a single mode. SND of the right case is much larger than the left (7.49 vs 6.79).

Fig. B shows the result. In this experiment, we generate target data (Black dots in Fig. B) by shifting the source distributions. In the left, we generate the target samples from 6 modes. In the right, the same number of target samples are generated from a single-mode. SND of the right case is much larger than that of the left (7.49 vs 6.79). If all target samples are from the green class, then SND picks the better model, but if they are actually from the red class, then SND picks the worse model. The failure case is hard to avoid as we discuss in Sec. 4.

3. Additional Results

We show results removed from our main paper due to limited space.

Semantic Segmentation. Fig. C shows iteration versus mIoU and HPO criterion. SND performs better than others on average in picking a better-adapted model (*i.e.*, better target accuracy). Besides, we can observe that the performance of segmentation models is sensitive to hyper-parameters and training iterations.

Image Classification. Fig. D shows iteration versus accuracy and HPO criterion in image classification experiments. We show results of Pseudo-labeling (PL), CDAN [4], and MCC [3]. SND performs better than others on average in picking a better-adapted model. Although SND does not always select the best model, SND shows a good correlation with accuracy. Entropy [5] shows a similar behavior to SND in PL, but behaves in a totally different way in CDAN [4].

Results of MCD [8]. We conduct experiments on tuning λ of MCD [8]. MCD is a popular approach that employs the disagreement of two task-specific classifiers' output. As shown in the Table A, SND shows the best performance on average. The result indicates the effectiveness of SND to tune classifier discrepancy-based adaptation methods.

4. Additional Analysis

In this section, we show the detailed analysis of Soft Neighborhood Density and other criterion.

Failure Case. In Sec 4.4 in the main paper, we explain possible failure cases: One can fool SND by training a model to collapse all target samples into a single point. We analyze the behavior of metrics in this setting. Specifically, we train a network to correctly classify source samples and to classify all unlabeled target samples into one class. We call the models *degenerated models*. Note that we will not employ this kind of a degenerated model in reality, but we train the models just to see the behavior of metrics. We vary λ for the target loss and compare the model with a non-adapted model. Fig. E shows the accuracy and the behavior of each metric. Since the model is trained to move all target samples to a single class, SND of degenerated models gets much larger than that of a non-adapted model(Blue). Other metrics are also not useful to identify the best model. Interestingly, training degenerated models for target does not decrease the accuracy of the source domain ((D) Source Risk). This is probably because the representational power

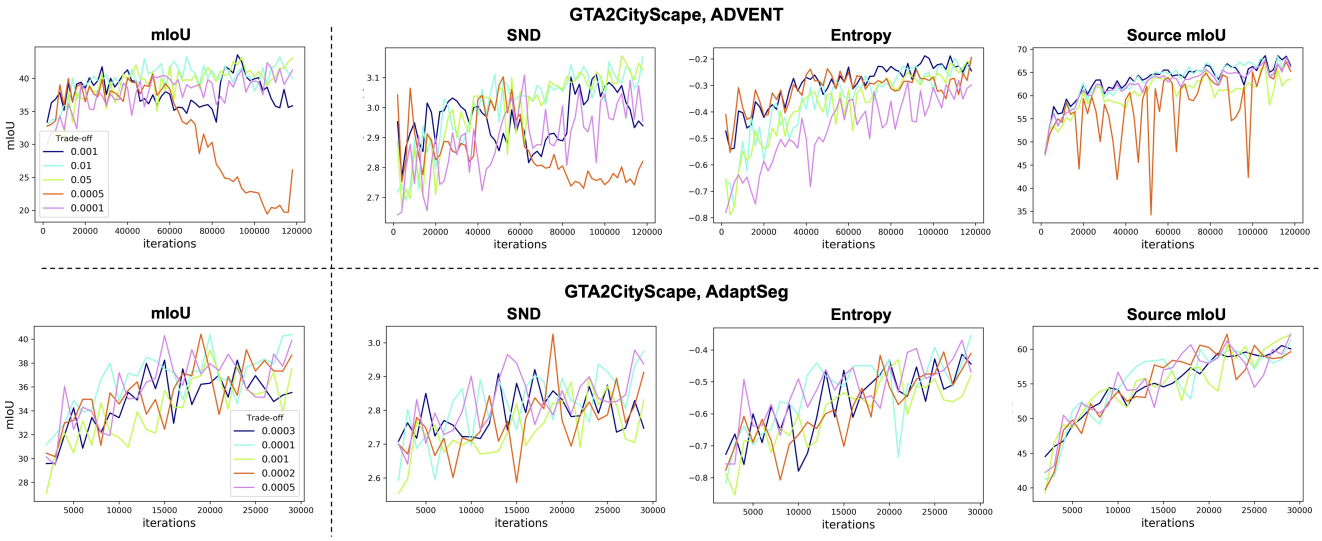

Figure C: Semantic segmentation experiments (GTA5 to CityScaple) using AdaptSeg [9] and ADVENT [10]. Different colors indicate different hyper-parameters. We validate the trade-off parameters between the source classification loss and domain-confusion loss. SND has a good correlation with mIoU (ground truth performance).

Method	Office		OH CDA		OH PDA			Avg
	A2D	W2A	R2A	A2P	R2A	A2P	P2C	
Lower Bound	78.8	62.8	61.9	66.4	69.2	67.3	37.2	63.4
Source Risk	**81.3**	**67.3**	62.4	68.0	69.3	68.4	42.2	65.6
DEV [11]	**81.3**	66.2	65.3	67.8	70.4	70.2	44.2	66.5
Entropy [5]	**81.3**	**67.3**	62.8	**68.9**	69.6	70.5	41.1	65.9
SND (Ours)	81.1	**67.3**	**66.1**	68.0	**72.2**	**71.0**	**44.2**	**67.1**
Upper Bound	84.7	68.9	66.9	68.9	72.2	71.3	45.3	68.3

Table A: **Results of MCD [8].** SND performs the best on average.

of neural networks is rich enough to learn both the degenerated solution for the target and a good solution for the source domain. One possible solution to this problem is to compare the feature visualizations of the degenerated and a non-adapted model. We leave further analysis to future work.

Varying the Number of Target Samples. We show analysis on the number of target samples necessary for Soft Neighborhood Density. Then, in the OfficeHome Real to Art closed adaptation, we employ NC [7] and reduce the number of target samples used to calculate SND. We randomly sample a certain proportion of the target domain and compute SND. As shown in Fig. F, SND is not very sensitive to the number of target samples. However, when we sample a small number of samples (10% case), Soft Neighborhood Density becomes a little unstable.

Temperature Parameter. We fix the temperature parameter (τ) in Eq. 2 (See our main draft.) as 0.05 in all of our experiments. Then, in the OfficeHome Art to Product partial domain adaptation, we employ NC [7] and vary the value of τ. In Fig. G, we compare the resulting curve of SND with the accuracy curve. We have two observations: SND is not very sensitive to the value of τ in selecting the best model; but, the large temperature can make SND inconsistent with the accuracy as the rightmost ($\tau = 0.1$) result indicates. This result indicates the necessity of the temperature scaling. The scaling enables to ignore samples embedded far away and to compute the density of neighbors.

Soft Neighborhood Density Versus Validation with a Few Labeled Target Samples. Some papers propose to utilize a few labeled target samples to tune hyper-parameters. Although the way of tuning violates the assumption of UDA, we investigate how well the criterion is effective to pick a good hyper-parameter in Fig. H. We employ the OfficeHome Real to Art closed adaptation using NC [7]. We increase the number of validation target samples per class from 1 to 20 and compare the result with SND. When the number of labeled target samples is small, the validation accuracies are not stable and have high variance. To obtain stable and reliable results, we need to have many labeled target samples whereas SND is an unsuper-

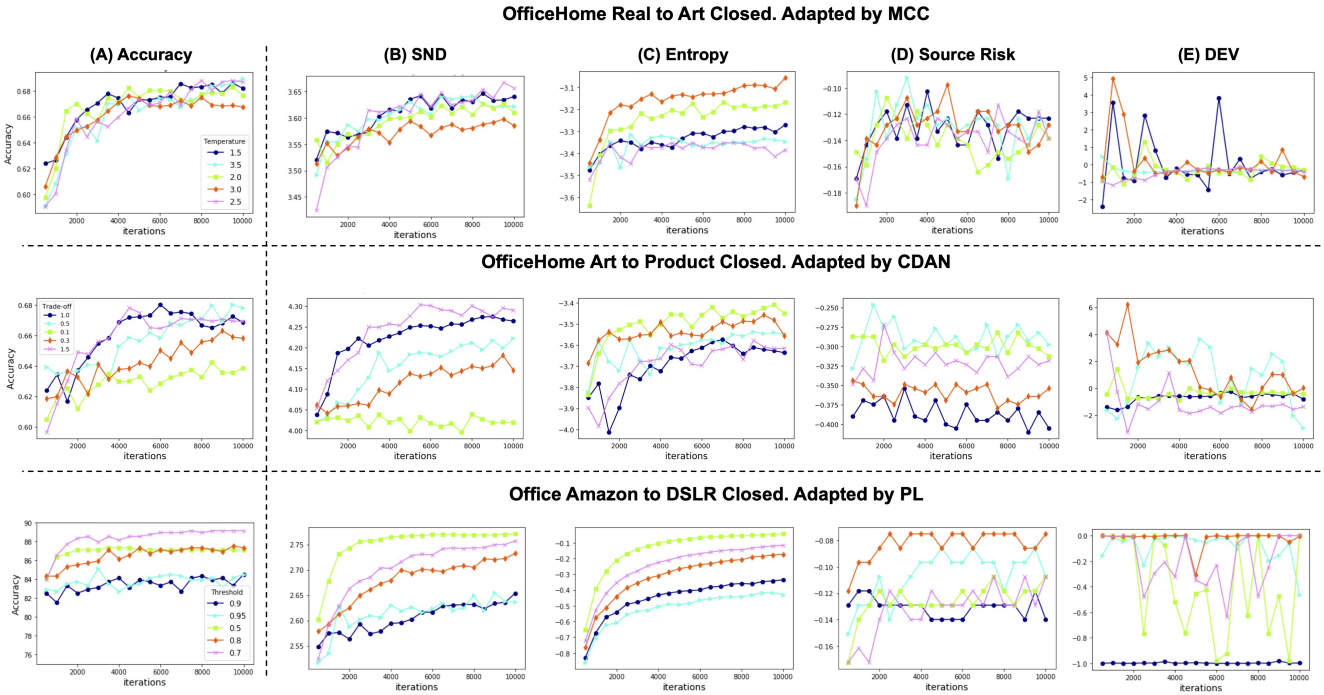

Figure D: Iteration versus accuracy and HPO criteria. Different colors indicate different hyper-parameters. To ease comparison between accuracy and criteria, we flip the sign of criteria for Entropy, Source risk, and DEV.

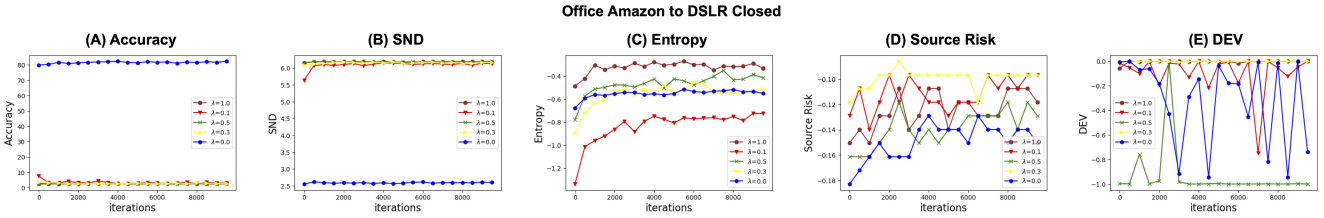

Figure E: **Analysis of a possible failure case.** We train a network to correctly classify source samples and to classify all unlabeled target samples into one class. **Blue**: A model trained only with source classification loss. **Others**: Models trained to classify all target samples into a single class as well as trained to correctly classify source samples. Different colors indicate different weights, λ, for the target loss. No metric is able to identify the non-adapted model.

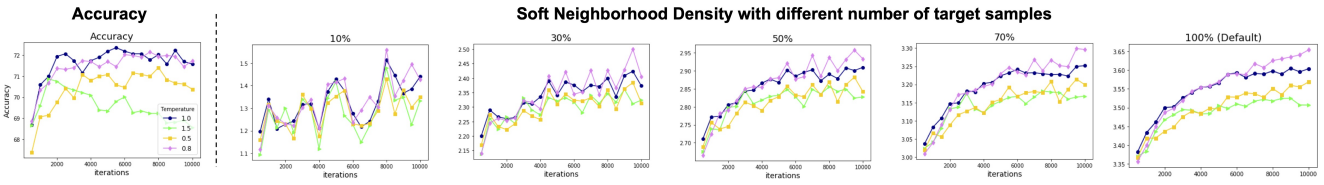

Figure F: Analysis of the number of target samples used to compute SND. Different colors indicate different hyper-parameters. We vary the number of target samples from $\frac{1}{10}N_t$ to N_t, where $N_t = 2427$ is the number of target samples. We randomly sample the target samples. To reduce variance of SND, we need to sample certain number of target samples.

vised criterion and shows reliable results. In a real application, having a few labeled target samples may not be always hard as stated in [6]. However, as this result indicates, monitoring only the accuracy of few samples may not provide a good model. Even in such a setting, combining SND will be a good way to tune hyper-parameters.

Analysis of the Number of Source Validation Samples on Source Risk and DEV [11]. We further analyze the cause of failures of source risk and DEV [11]. We increase the number of labeled source samples and observe the behavior of two criteria. We use the Amazon to DSLR setting adapted by CDAN [4]. Even when we use a large

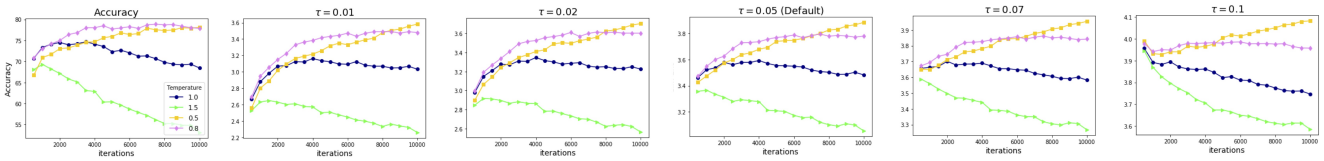

Figure G: Analysis of the temperature value used to compute SND. Different colors indicate different hyper-parameters. We vary the value of the temperature of Eq. 2, *i.e.*, 0.01, 0.03, 0.05 (default), 0.07, 0.1. The result indicates that SND shows consistent results across different temperature values.

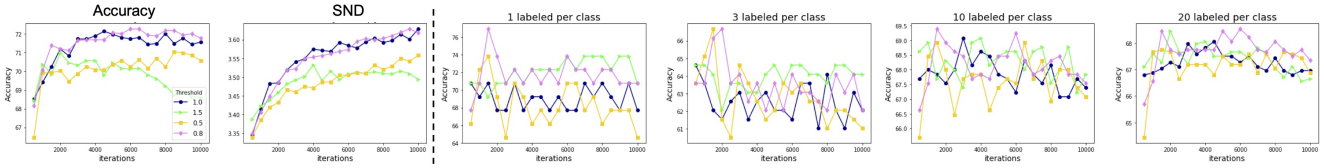

Figure H: Iteration versus accuracy and accuracy of the subset of a target domain. Different colors indicate different hyper-parameters. We subsample labeled target samples (1, 3, 10, 20 samples per class) and compute the accuracy. Many number of labeled samples is necessary to resemble the performance of a whole target domain.

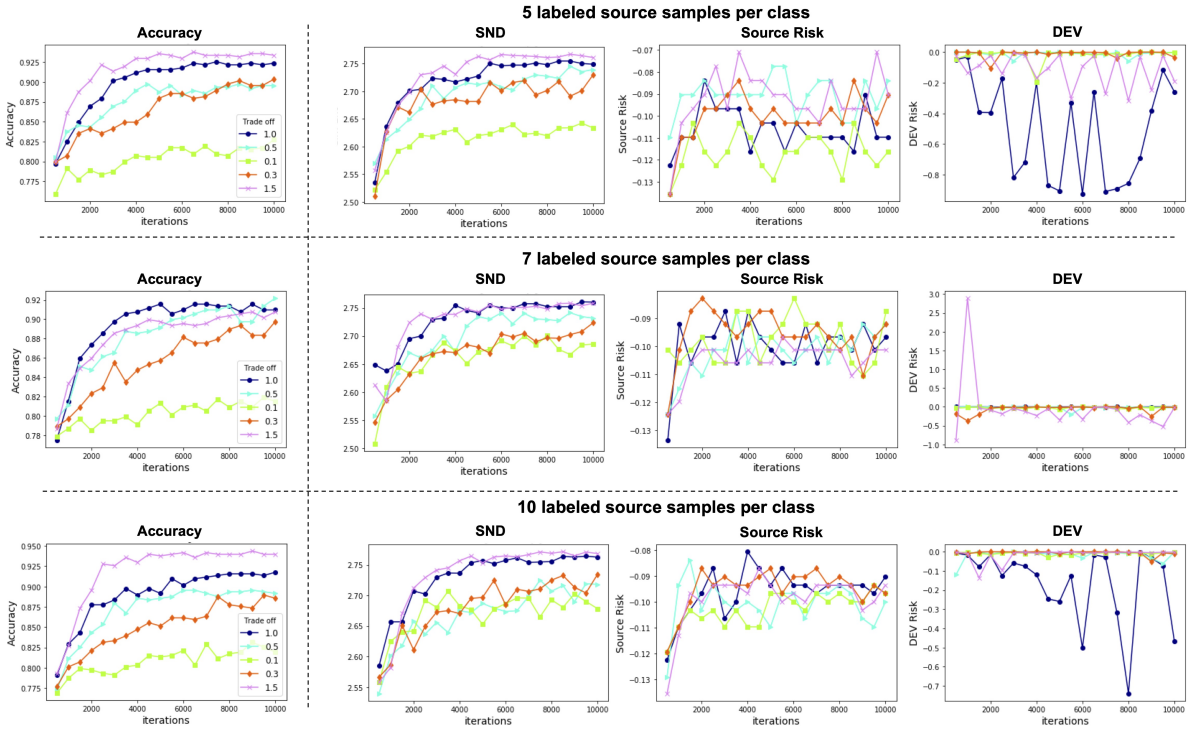

Figure I: Analysis of the number of labeled source samples used for validation. We vary the number of the labeled source samples to compute source risk and DEV risk. Different colors indicate different hyper-parameters. The result indicates that even though we increase the number of source validation samples, the risks are not reliable to select hyper-parameters.

proportion of source samples as a validation set (We utilize more than 10 % of source samples in the case of 10 labeled samples per class.), the two criteria are not well correlated with the accuracy of the target domain. This result indicates the using source risk is limited to choosing good hyper-parameters.

References

[1] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Eur. Conf. Comput. Vis.*, 2018. 1

[2] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(59):1–35, 2016. 1

[3] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *Eur. Conf. Comput. Vis.*, 2020. 1, 2

[4] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Adv. Neural Inform. Process. Syst.*, 2018. 1, 2, 4

[5] Pietro Morerio, Jacopo Cavazza, and Vittorio Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. *arXiv preprint arXiv:1711.10288*, 2017. 2, 3

[6] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Int. Conf. Comput. Vis.*, 2019. 1, 4

[7] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. *arXiv preprint arXiv:2002.07953*, 2020. 1, 3

[8] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2, 3

[9] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 1, 3

[10] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Mathieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1, 3

[11] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards accurate model selection in deep unsupervised domain adaptation. In *Int. Conf. Mach Learn.*, 2019. 3, 4