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

Kuniaki Saito[1], Donghyun Kim[1], Piotr Teterwak[1], Stan Sclaroff[1], Trevor Darrell[2], and Kate Saenko[1,3]
[1]Boston University, [2]University of California, Berkeley, [3]MIT-IBM Watson AI Lab

[keisaito,donhk,piotrt,sclaroff,saenko]@bu.edu, trevor@eecs.berkeley.edu

Abstract

Unsupervised domain adaptation (UDA) methods can dramatically improve generalization on unlabeled target domains. However, optimal hyper-parameter selection is critical to achieving high accuracy and avoiding negative transfer. Supervised hyper-parameter validation is not possible without labeled target data, which raises the question: How can we validate unsupervised adaptation techniques in a realistic way? We first empirically analyze existing criteria and demonstrate that they are not very effective for tuning hyper-parameters. Intuitively, a well-trained source classifier should embed target samples of the same class nearby, forming dense neighborhoods in feature space. Based on this assumption, we propose a novel unsupervised validation criterion that measures the density of soft neighborhoods by computing the entropy of the similarity distribution between points. Our criterion is simpler than competing validation methods, yet more effective; it can tune hyper-parameters and the number of training iterations in both image classification and semantic segmentation models. The code used for the paper will be available at https://github.com/VisionLearningGroup/SND.

1. Introduction

Deep neural networks can learn highly discriminative representations for visual recognition tasks [11, 39, 19, 29, 14], but do not generalize well to out-of-domain data [47]. To improve performance on a new target domain, Unsupervised Domain Adaptation (UDA) aims to transfer representations from a label-rich source domain without additional supervision. Recent UDA methods primarily achieve this through unsupervised learning on the target domain, *e.g.*, by minimizing the feature distribution shift between source and target domains [12, 21, 41], classifier confusion [17], clustering [34], and pseudo-label based methods [56]. Promising adaptation results have been demonstrated on image classification [22, 55, 8, 52, 43], semantic segmentation [16] and object detection [9] tasks.

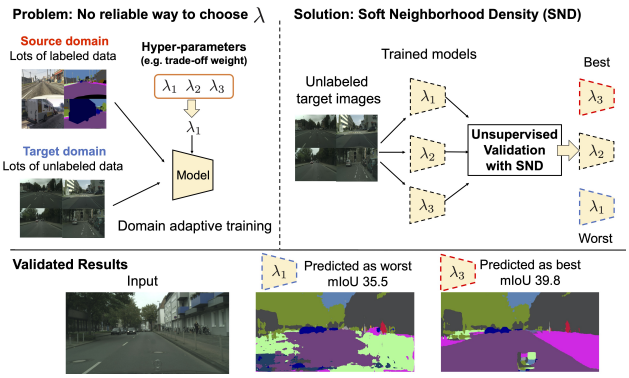


Figure 1: In unsupervised domain adaptation, validation is a significant and unsolved issue. Performance can be sensitive to hyper-parameters, yet no reliable validation criteria have been presented. In this work, we provide a new criterion, *SND*, to select proper hyper-parameters for model validation in UDA. The example shows validation of the ADVENT [50] segmentation model.

However, adaptation methods can be highly sensitive to hyper-parameters and the number of training iterations. For example, the adversarial alignment approach is popular in semantic segmentation [16, 44, 50], but can fail badly without careful tuning of the loss trade-off weight as shown in Fig. 1. In addition, many methods have other kinds of hyper-parameters, *e.g.*, defining the concentration of clusters [34], the confidence threshold on target samples [56], etc. Therefore validation of hyper-parameters is an important problem in UDA, yet it has been largely overlooked. In UDA, we assume not to have access to labeled target data, thus, hyper-parameter optimization (HPO) should be done without using target labels. In this paper, we would like to pose a question crucial to making domain adaptation more practical: *How can we reliably validate adaptation methods in an unsupervised way?*

Unsupervised validation is very challenging in practice, thus many methods do HPO in an ineffective, or even unfair, way. Evaluating accuracy (risk) in the source domain is popular [12, 41, 5, 40, 22, 5, 17, 51, 53], but this will not

necessarily reflect success in the target domain. Using the risk of the target domain [37, 2, 3, 13, 35, 38, 32] contradicts the assumption of UDA. Many works [45, 42, 44, 25, 56, 54, 49, 50] do not clearly describe their HPO method.

To the best of our knowledge, no comprehensive study has compared validation methods across tasks and adaptation approaches in a realistic evaluation protocol. Our first contribution is to empirically analyze these existing criteria and demonstrate that they are not very effective for HPO. This exposes a major barrier to the practical application of state-of-the-art unsupervised domain adaptation methods.

To tackle this problem, we first revisit an unsupervised validation criterion based on classifier entropy, C-Ent. If the classification model produces confident and low-entropy outputs on target samples, then the target features are discriminative and the prediction is likely reliable. Morerio *et al*. [24] propose to utilize C-Ent to do HPO but only evaluated it on their own adaptation method. We evaluate C-Ent extensively, across various adaptation methods, datasets, and vision tasks. We reveal that C-Ent is very effective for HPO for several adaptation methods, but also expose a critical issue in this approach, which is that it cannot detect the collapse of neighborhood structure in target samples (See Fig. 2). By *neighborhood structure*, we mean the relationship between samples in a feature space. Before any adaptation, target samples embedded nearby are highly likely in the same class (*No adaptation* in Fig. 2). A good UDA model will keep or even enhance the relationships of target samples while aligning them to the source. However, a UDA model can falsely align target samples with the source and incorrectly change the neighborhood structure ((*DANN* in Fig. 2). Even in this case, C-Ent can become small and choose a poorly-adapted model.

Natekar *et al*. [26] measure the consistency of feature embeddings within a class and their discrepancy from other classes to predict the generalization of supervised models by using labeled samples. Accounting for such relationships between points is a promising way to overcome the issues of C-Ent. But, since computing these metrics requires labeled samples, we cannot directly apply this method.

This leads us to propose a novel unsupervised validation criterion that considers the neighborhood density of the unlabeled target. Our notion of a neighborhood is soft, i.e. we do not form explicit clusters as part of our metric computation. Rather, we define soft neighborhoods of a point using the distribution of its similarity to other points, and measure density as the entropy of this distribution. We assume that a well-trained source model will embed target samples of the same class nearby and thus will form dense implicit neighborhoods. The consistency of representations within each neighborhood should be preserved or even enhanced by a well-adapted model. Monitoring the density thus enables us to detect whether a model causes the collapse of implicit neighborhood structure as shown in Fig. 2.

Method	Technical Advantages			Stability across methods
	w/ X_t	w/o X_s, Y_s	w/o HP	
Source Risk	✗	✗	Test split	✗
IWV [40, 53]	✗	✗	+Density Model	✗
Entropy [24]	✓	✓	✓	✗
SND (Ours)	✓	✓	✓	✓

Table 1: Technical comparison with other validation approaches. X_t denotes unlabeled target, and (X_s, Y_s) denote labeled source samples. SND computes a score on unlabeled target samples. Empirically, we verify that our method is stable across different datasets, methods, and tasks.

Our proposed metric, called *Soft Neighborhood Density (SND)*, is simple yet more effective than competing validation methods. Rather than focusing on the source and target relationship (like IWV [40] or DEV [53]), we measure the discriminability of target features by computing neighborhood density and choosing a model that maximizes it.

We demonstrate that target accuracy is consistent with our criterion in many cases. Empirically, we observe that SND works well for closed and partial domain adaptive image classification, as well as for domain adaptive semantic segmentation. SND is even effective in choosing a suitable source domain given an unlabeled target domain.

Our contributions are summarized as follows:

- We re-evaluate existing criteria for UDA and call for more practical validation of DA approaches.
- We propose Soft Neighborhood Density metric which considers target neighborhood structure, improves upon class entropy (C-Ent) and achieves performance close to optimal (supervised) HPO in 80% cases on closed, partial DA, and domain adaptive semantic segmentation.

2. Related Work

Domain Adaptation aims to transfer knowledge from a labeled source domain to a label-scarce target domain. Its application in vision is diverse: image classification, semantic segmentation [16, 44], and object detection [9]. One popular approach in DA is based on distribution matching by adversarial learning [12, 46, 22]. Adversarial adaptation seeks to minimize an approximate domain discrepancy distance through an adversarial objective with respect to a domain discriminator. Recently, some techniques that utilize clustering or a variant of entropy minimization have been developed [34, 18, 43, 17]. [43, 18] propose to train a model by minimizing inter-class discrepancy given the number of classes. SND computes the density of local neighborhoods and selects a model with the largest density, which allows us to select a good model without knowing the number of classes in the target. All of the existing approaches have important hyper-parameters to be tuned, such as the trade-off parameter between source classification and adaptation

loss. Another important hyper-parameter is softmax temperature [34, 17]. Given a specific target domain, the selection of a source domain is also important. If we have to choose only a single source domain, the selection process is crucial for the model's performance. However, it is questionable whether existing approaches do HPO in a realistic way.

Validation Methods for UDA. In Table 1, we summarize several prior validation methods that do not need any target labeled samples. The validation methods themselves can have hyper-parameters (HP) and other requirements.

Source Risk. Ganin *et al*. [12] considers the source risk to select hyper-parameters. But, the source risk is not a good estimator of the target risk in the presence of a large domain gap.

Importance Weighted Validation (IWV) and DEV. Sugiyama *et al*. [40] and You *et al*. [53] validate methods using the risk of source samples. If a source sample is very similar to target samples, the risk on the source sample is heavily weighted. This approach has a similar issue to Source Risk validation. Since DEV is developed to control the variance in IWV, we use it as a baseline in experiments.

Entropy (C-Ent). Morerio *et al*. [24] employ entropy of classification output. If the model has confident predictions in classifying target samples, the hyper-parameter is considered appropriate. The method is simple and does not require validation with labeled samples. But, Morerio *et al*. [24] apply the C-Ent criterion only to tune their proposed model, which makes its applicability to diverse methods unclear. We extensively evaluate this approach and reveal that while it is often useful, it has a critical failure mode. This failure mode is that domain adaptation models can output confidently incorrect predictions for target samples. In comparison, we empirically show that Soft Neighborhood Density gives the most stable results across different datasets and methods for both image classification and semantic segmentation.

Locally Linear Embedding (LLE). Roweis *et al*. [31] compute low dimensional and neighborhood-preserving embeddings of high dimensional data. The neighborhood-preserving embeddings recover global nonlinear structure. We aim to pick a model by monitoring the density of implicit neighborhoods during adaptation.

3. Approach

Problem setting. In UDA, we are provided labeled source data $\mathcal{D}_s = \{(\mathbf{x}_i^s, {y_i}^s)\}_{i=1}^{N_s}$ and unlabeled target data $\mathcal{D}_t = \{(\mathbf{x}_i^t)\}_{i=1}^{N_t}$. Generally, domain adaptation methods optimize the following loss,

$$L = L_s(x_s, y_s) + \lambda L_{adapt}(x_s, x_t, \eta), \tag{1}$$

where L_s is the classification loss for source samples and L_{adapt} is the adaptation loss computed on target samples.

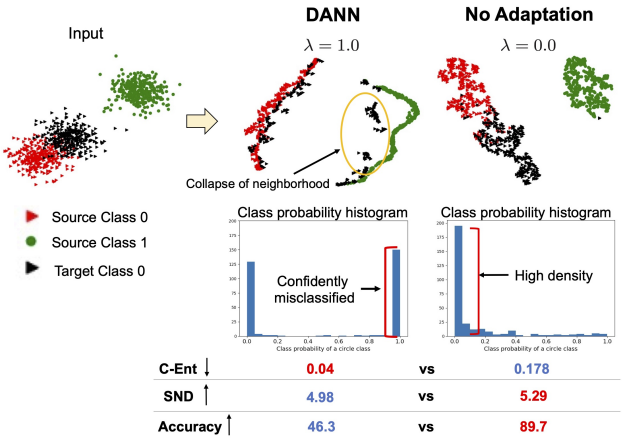


Figure 2: **An illustration of C-Ent failing to detect the collapse of target neighborhood structure.** This is the case where an adapted model ($\lambda = 1.0$) confidently misclassifies target samples and low entropy (C-Ent) cannot select a good model. The model incorrectly changes the relative distances between target samples. SND can select a better model since it can consider how well the neighborhood structure is preserved.

λ controls the trade-off between source classification and adaptation loss. η are hyper-parameters used to compute the adaptation loss. Our goal is to build a criterion that can tune λ and η. We additionally aim to choose the best training iteration since a model can be sensitive to its choice.

Assumptions. We assume that target samples embedded nearby should belong to the same class. Therefore, representations of a well-adapted model should result in dense neighborhoods of highly-similar points. We express density as the consistency of representations within each soft neighborhood, such that Soft Neighborhood Density becomes large with a well-adapted model. Models are trained with labeled source samples as well as unlabeled target samples. Since source and target are related domains, the model will have somewhat discriminative features for target samples. Before adaptation, such features will define *initial neighborhoods*, namely, samples with higher similarity to each other relative to the rest of the points. If the selection of a method and hyper-parameter is appropriate, such neighborhoods should be preserved rather than split into smaller clusters, and the similarity of features within soft neighborhoods should increase.

Motivation. First, we empirically show the necessity of considering the neighborhood structure. Class Entropy (C-Ent) [24] measures the confidence in prediction. Through domain-adaptive training with inappropriate hyper-parameters, the confidence in the target prediction can increase while drastically changing the neighborhoods from the initial neighborhood structure.

A toy example (Fig. 2) clarifies this idea. We generate source data from two Gaussian distributions with different

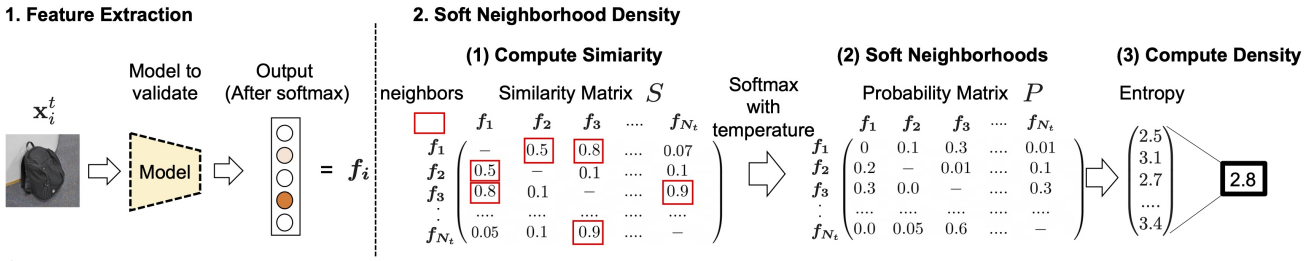


Figure 3: **Method overview.** Soft Neighborhood Density measures the density of implicit local neighborhoods in the target domain, which is used to select hyper-parameters of the adaptation model. We first extract features from the softmax layer for all target samples. We then compute the similarity distribution from the pair-wise similarities of the features (red boxes highlight similar points). Finally we use the entropy of the similarity distribution as our evaluation criterion (SND), where the higher its value, the better.

means, which we regard as two classes. Then, we obtain target data by shifting the mean of one of the Gaussians. We train two-layered neural networks in two ways: training with distribution alignment (DANN [12], $\lambda = 1.0$) and training with only source samples ($\lambda = 0$). From the input space, the target samples should not be aligned with source class 1 (green circles). But, the DANN model aligns target samples incorrectly and confidently misclassifies many target samples, resulting in a very small C-Ent. If we employ C-Ent as a criterion, we will select the left model which performs poorly. But, if we are able to consider whether the density of neighborhoods is maintained or increased, we can avoid selecting the poor model.

In this toy example, we have utilized the partial DA setting [5], where the target label set is the subset of the source, and DANN [12] to illustrate the issue of C-Ent. In fact, this kind of *wrong alignment* can happen in real data when we use distribution alignment (Fig. 4a).

3.1. Soft Neighborhood Density (SND)

We aim to design a criterion that can consider the density of implicit local neighborhoods in the target domain. We define a soft *neighborhood* for each sample by computing its similarity to other samples in the dataset, and converting it to a probability distribution. This is done via a softmax activation with temperature-scaling to ignore samples outside of the local neighborhood. Once we define the soft neighborhoods, we can estimate their density by computing the entropy of the distribution. The overall pipeline in Fig. 3 consists of 1) computing the similarity between samples, 2) applying the softmax activation with temperature-scaling (identifying soft neighborhoods), and 3) calculating soft neighborhood density.

Similarity Computation. We first compute similarity between target samples, $S \in \mathbb{R}^{N_t \times N_t}$, where N_t denotes the number of target samples. Let $S_{ij} = \langle \boldsymbol{f}_i^t, \boldsymbol{f}_j^t \rangle$, where $\boldsymbol{f}_i^t$ denotes a L_2 normalized target feature for input $\mathbf{x}_i^t$. We ignore diagonal elements of S because our goal is to compute the distances to neighbors for each sample. This matrix defines distances, but it is not clear which samples are relatively close to each other vs. far away just from this matrix.

Soft Neighborhoods. To highlight the difference between nearby points and far-away points, we convert the similarity between samples into a probabilistic distribution, P, using temperature scaling and the softmax function,

$$P_{ij} = \frac{\exp(S_{ij}/\tau)}{\sum_{j'} \exp(S_{ij'}/\tau)}, \tag{2}$$

where τ is a temperature constant. Note that the temperature scaling and the exponential function have the ability to enlarge the difference between similarity values S_{ij}. Therefore, if a sample j is relatively dissimilar from i, the value of P_{ij} will be very small, which allows us to ignore distant samples for the sample i. The temperature is the key to implicitly identifying neighborhoods; we set it to 0.05 across all experiments given the results of the toy dataset.

Soft Neighborhood Density. Next, we design a metric to consider the neighborhood density given P. The metric needs to evaluate the consistency of representations within the implicit neighborhoods. If a model extracts ideally discriminative features, the representations within a neighborhood are identical. To identify this situation, we propose to compute the entropy of P. For example, if the entropy of P_i is large, the probabilistic distribution should be uniform across the soft neighborhoods; that is, neighbors of the sample i are concentrated into very similar points. Specifically, we compute the entropy for each sample and take the average of all samples as our final metric:

$$H(P) = -\frac{1}{N_t} \sum_{i=1}^{N_t} \sum_{j=1}^{N_t} P_{ij} \log P_{ij}. \tag{3}$$

We then choose the model that has the highest entropy of all candidate models. If a model falsely separates samples into clusters as in Fig. 2, the entropy becomes small.

Input Features. The key to the success of our method is extracting implicit neighborhoods. Ideally, all samples in the same class should be embedded near each other. In that case, Eq. 3 can compute the density of each class. Hence, we need to use class-discriminative features, and the choice of features can be essential. We propose to employ the classification softmax output as our input feature vec-

tor $\boldsymbol{f}$. Since this feature represents class-specific information, it is the most likely to place target samples of the same class together. As we analyze in experiments, this feature has smaller within-class variance than either middle-layer features or classification output without softmax activation. Therefore, our hope is that the number of clusters in Eq. 2 should be equal to (closed set case) or less than (partial set case) the number of source classes. In the ideal case, the computed density should be close to the within-class density.

Discussion. Note that we assume that the classifier is well-trained on source samples. The criterion becomes very large if a model produces the same output for all target samples. To avoid this, the model needs to be well trained on source samples, which can be easily monitored by seeing the loss on source training samples.

Also, note that our assumption is that samples embedded nearby are likely to share the category labels. This is consistent with an assumption made by general domain adaptation methods [1]. If a pre-trained model provides very poor representations or the source domain is too different from the target, the assumption will not be satisfied. Under this setting, any metrics of UDA will not be a good tool to monitor the training.

Extension to Semantic Segmentation. In semantic segmentation, the outputs for each image can be as large as one million pixels. When the number of target samples is that large, the computation of the similarity matrix can be very expensive. In order to make the computation of SND more efficient, we subsample the target samples to make the similarity graph smaller and easier to compute. In our experiments, we randomly sample a hundred pixels for each image and compute SND for each, then, take the average of all target images. Although the method is straightforward, our experiments show that the resulting approximation of SND continues to be an effective criterion for selecting hyper-parameters.

4. Experiments

First, we evaluate the existing metrics and SND to choose hyper-parameters in domain adaptation for image classification and semantic segmentation. Second, we show experiments to analyze the characteristics of our method.

We evaluate the ability to choose suitable hyper-parameters including checkpoints (*i.e.*, training iterations) for unsupervised domain adaptation. Closed DA (CDA) assumes that the source and target domain share the same label set while partial DA (PDA) assumes that the target label set is the subset of the source. We perform experiments on both adaptation scenarios. The details of semantic segmentation are described in the appendix. The general design of the experiment is similar to image classification.

4.1. Adaptation Methods

For each method, we select the hyper-parameter mentioned below, plus a training iteration. See appendix for other hyper-parameters used in experiments.

Adversarial Alignment. As a representative method of domain alignment, we use CDAN [22]. We select a weight of the trade-off between domain confusion loss and source loss (λ = 0.1, 0.3, 0.5, 1.0 ,1.5, with λ = 1.0 as its default setting). To analyze the behavior in detail, we also utilize DANN [12] for OfficeHome PDA. The validation is done in the same way as CDAN.

Clustering. As a clustering-based DA method, we utilize Neighborhood Clustering (NC) [34]. NC performs clustering using similarity between features and a temperature is used to compute the similarity distribution. Since the selection of the temperature value can affect the performance, we evaluate criteria to choose the best temperature (η = 0.5, 0.8, 1.0, 1.5, with η = 1.0 as its default setting).

Classifier Confusion. As a recent state-of-the art method, we employ MCC [17], where a temperature is used to compute the classifier's confusion loss. The goal is to tune the temperature values (η = 1.5, 2.0, 2.5, 3.0, 3.5, with η = 2.5 as its default setting).

Pseudo Labeling (PL). Employing pseudo labels [20] is one of the popular approaches in DA [55, 56, 35]. One important hyper-parameter is a threshold to select confident target samples. If the output probability for a predicted class is larger than the threshold, we use the sample for training. The optimal threshold may be different for different datasets. Then, our goal is to tune the threshold values (η = 0.5, 0.7, 0.8, 0.9, 0.95).

Semantic Segmentation. AdaptSeg [44] and ADVENT [50] are employed. For both methods, the goal is to tune both trade-off parameters of adversarial alignment loss (λ) and training iterations.

4.2. Setup

Datasets. For image classification, we use Office [33] (Amazon to DSLR (A2D) and Webcam to Amazon (W2A) adaptation) with 31 categories and OfficeHome [48] (Real to Art (R2A), Art to Product (A2P) and Product to Clipart (P2C)) with 65 categories. Office is used for CDA while we use OfficeHome for both CDA and PDA. For the category split of PDA, we follow [6]. To further demonstrate applicability to large-scale datasets, we evaluate SND on VisDA [28] and DomainNet [27] in CDA. We describe the detail in the appendix. In semantic segmentation, we use GTA5 [30] as a source and Cityscape [10] as a target domain.

Baselines. Entropy [24] directly employs the entropy of the classification output. It takes the average of all samples. A smaller value should indicate a better-adapted model. For DEV [53], we need to have held-out validation source samples. Since holding out many source samples can

Method	CDAN [22]				MCC [17]				NC [34]				PL [20]				Avg
	A2D	W2A	R2A	A2P	A2D	W2A	R2A	A2P	A2D	W2A	R2A	A2P	A2D	W2A	R2A	A2P	
Lower Bound	77.7	57.4	58.7	59.9	80.4	62.6	59.0	60.5	69.8	60.2	67.7	66.5	80.4	65.1	66.8	66.6	66.4
Source Risk	87.9	65.5	62.3	62.2	**92.8**	70.2	65.6	72.6	78.1	**66.8**	71.3	70.4	84.5	67.4	69.3	67.4	71.1
DEV [53]	90.0	66.4	63.5	62.4	91.3	67.6	63.1	70.1	78.1	65.3	71.4	72.1	84.7	**67.5**	69.0	69.2	71.6
Entropy [24]	82.3	63.8	61.7	63.4	91.3	**72.4**	66.9	70.3	**88.4**	66.4	72.0	73.7	84.8	67.4	**70.1**	**69.4**	71.9
SND (Ours)	**92.9**	**67.0**	**70.8**	**67.3**	92.6	67.4	**68.8**	**72.8**	**88.4**	66.4	**72.5**	**73.9**	**85.1**	67.4	**70.1**	**69.4**	**74.3**
Upper Bound	93.3	69.8	71.1	68.1	94.5	72.9	69.5	74.4	89.6	71.2	72.8	74.1	86.8	68.1	70.8	70.2	75.5

Table 2: **Results of closed DA.** SND provides faithful results for all methods and datasets, *i.e.* Office (A2D and W2A) and OfficeHome (R2A and A2P) whereas baselines show several failure cases. Lower/Upper bounds are results obtained with the worst/best model.

Method	CDAN [22]			MCC [17]			NC [34]			PL [20]			Avg
	R2A	P2C	A2P	R2A	P2C	A2P	R2A	P2C	A2P	R2A	P2C	A2P	
Lower Bound	60.9	34.5	60.2	53.7	38.3	60.9	60.0	37.5	52.3	49.2	56.6	44.1	50.7
Source Risk	**67.6**	42.0	64.8	65.1	47.4	73.3	76.6	49.8	77.9	61.5	72.7	53.5	62.7
DEV [53]	65.6	36.6	63.9	67.7	47.3	70.3	72.3	54.5	67.0	62.4	66.3	52.1	60.5
Entropy [24]	64.7	40.3	64.8	53.8	40.4	62.0	79.1	**58.2**	**78.7**	60.1	71.7	47.0	60.1
SND (Ours)	66.3	**45.7**	**65.4**	**70.2**	**50.8**	**79.3**	**79.2**	58.1	78.2	**68.7**	**72.2**	**57.9**	**66.0**
Upper Bound	68.8	46.9	68.5	72.0	52.1	79.7	80.1	58.2	79.1	69.0	74.0	59.4	67.3

Table 3: **Results of partial DA on OfficeHome.** SND performs the best on average.

degrade the accuracy of adapted models, we take 3 source samples per class as validation sets. Increasing the number of source validation samples to more than 3 per class does not much improve the validation performance. See the appendix for more detail. A smaller risk represents a better model. Similarly, Source Risk is measured on the source validation samples. We also report lower bound and upper bound performance among all hyper-parameters.

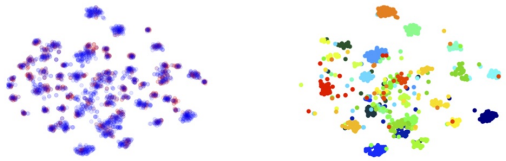
Evaluation Protocol. In image classification, we train all adaptation methods for 10,000 iterations. Although every method has different default training iterations, we keep them the same for the simplicity of experiments. Then, we select a checkpoint that shows the best value among all reported iterations and choice of hyper-parameters. In semantic segmentation, we calculate mIoU and each criterion similarly. We don't use DEV [53] for semantic segmentation since the design of the domain classifier is complicated. We run experiments three times and show their averaged accuracy.

Implementation. We utilize published official implementations for adaptation methods, CDAN, MCC, NC, AdaptSeg, and ADVENT. For the pseudo labeling method, we use the NC's implementation. These methods use ResNet50 or ResNet101 [15] as backbone networks. AdaptSeg and ADVENT employ DeepLab [7]. See the appendix for more details.

4.3. Validation Results

Image Classification. The results of image classification are summarized in Tables 2, 3, and 4. Fig. 5 shows plots of accuracy and criteria for several adaptation settings. In most cases, DA methods are sensitive to hyper-parameters and training iterations. As we can see in the Tables, our proposed method faithfully selects good checkpoints for various methods and two category shifts. Of course, there are some gaps between the upper bound and our score, but the gap is not large. Importantly, SND does not catastrophically fail in these cases while other criteria choose several bad checkpoints. Besides, the curve of accuracy and SND have very similar shapes. The results denote that SND is effective for HPO in various methods. In the experiments on VisDA and DomainNet, most methods select good checkpoints since adaptation methods are stable across different hyper-parameters.

(a) DANN tuned by Entropy [24] (Accuracy: 35.1 %).



(b) DANN tuned by SND (Accuracy: 70.8 %).

Figure 4: **Feature visualization [23].** OfficeHome partial DA using DANN [12]. **Left:** Source (Blue), Target (Red). **Right:** Target samples. Different colors denote different classes. (a) Entropy [24] does not detect the collapse of the neighborhood structure and chooses a model that wrongly aligns features. (b) SND chooses a model that keeps the structure.

Source Risk provides a good model in some cases, but also catastrophically fails in some cases such as A2D in NC Table 2. DEV [53] also sometimes catastrophically fails.

Method	CDAN [22]		MCC [17]		NC [34]		PL [20]		Avg
	VisDA	DNet	VisDA	DNet	VisDA	DNet	VisDA	DNet	
Lower bound	51.1±1.3	51.3±1.4	67.1±1.1	54.8±2.1	44.8±3.7	56.9±0.4	58.7±1.1	55.5±0.3	55.0±1.1
Source Risk	**72.6±0.8**	63.8±1.4	71.7±0.8	58.7±0.8	65.8±1.3	62.0±0.2	66.7±2.8	59.9±0.5	65.1±0.3
DEV [53]	**72.6±0.8**	57.9±4.1	72.3±3.0	58.5±0.5	65.8±1.3	59.4±0.8	66.7±2.8	59.1±0.3	64.0±1.1
Entropy [24]	69.9±1.8	61.5±1.1	68.9±1.3	**59.2±0.2**	**68.4±1.1**	62.3±0.6	68.5±0.1	60.7±0.3	64.9±0.2
SND (Ours)	70.3±0.1	**64.9±0.4**	**73.0±1.1**	58.9±2.3	66.9±3.2	**62.4±0.8**	**69.0±1.1**	**60.9±0.1**	**65.8±0.3**
Upper bound	74.1±1.1	65.6±0.1	74.5±0.6	61.2±0.5	69.2±0.2	63.2±0.2	69.2±0.8	61.0±0.1	67.2±0.1

Table 4: **VisDA [28] and DomainNet (DNet) [27] results in closed DA.** Averaged accuracy over three runs and its standard deviation are shown. We utilize Real to Clipart adaptation for DomainNet. SND performs the best on average.

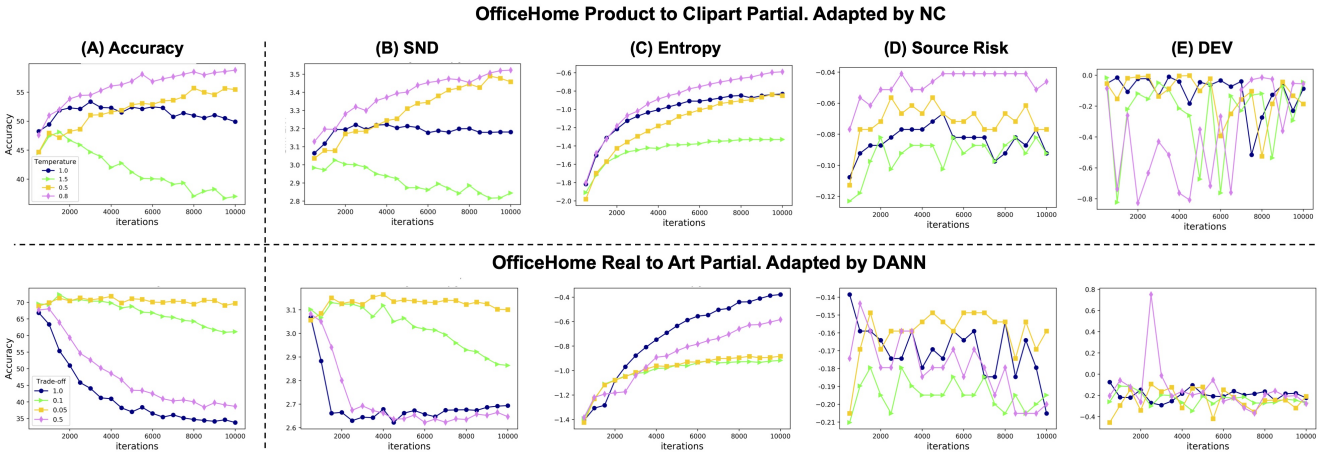


Figure 5: Iteration versus accuracy and HPO criteria. To ease comparison between accuracy and criteria, we flip the sign of criteria for Entropy, Source risk, and DEV. **Notice how SND curves track the Accuracy. More results are shown in appendix.**

GTA5 to Cityscape	AdaptSeg [44]	ADENT [50]
Lower Bound	31.1±0.5	21.0±2.5
Source Risk	35.6±2.9	37.2±1.6
Entropy [24]	**39.3±1.1**	35.6±3.7
SND (Ours)	**39.5±0.8**	**40.2±0.5**
Upper Bound	40.4±0.5	43.6±0.3

Table 5: Validation results in domain-adaptive semantic segmentation in GTA5 to Cityscapes adaptation.

From Fig. 5, these two criteria have some variance and do not necessarily reflect the accuracy in the target. We hypothesize that there are two reasons. First, Source Risk is not necessarily correlated with the performance of the target. If a model focuses on the classification of source samples (*i.e.*, setting $\lambda = 0$ in Eq. 1), the risk gets small, which does not indicate good performance on the target domain. Unless the source and target are very similar, the risk will not be reliable. Second, we may need careful design in selecting validation source samples and the domain classifier construction for DEV. However, considering the practical application, the validation methods should not have a module that requires careful design.

Entropy [24] (C-Ent) shows comparable performance to SND in NC [34] and PL [20]. This is probably because both methods are trained to maintain the target neighborhood structure. Then, by monitoring the confidence of the prediction, we can pick a good model. However, as is clear from the graph at the bottom of Fig. 5, it causes catastrophic failure by mistakenly providing confident predictions in PDA (Real to Art) adapted by DANN. We show the feature visualization of DANN results in Fig. 4. The model selected by C-Ent collapses the neighborhood structure of target samples yet matches them with source samples, which results in a lower C-Ent value. By contrast, SND selected a model that maintains the structure. Since the overconfidence issue can happen in many methods and datasets, C-Ent is not reliable for some methods. This is consistent with the results on the toy dataset in Fig. 2.

Semantic Segmentation. Table 5 describes the checkpoint selection result. SND selects good checkpoints for both methods. If we compare the performance of the upper bound, ADVENT [50] is better than AdaptSeg [44] with 3.2 points in mIoU. But, the gap becomes much smaller (only 0.7 points) if we apply unsupervised evaluation. ADVENT [50] is more sensitive to hyper-parameters such as training iterations than AdaptSeg [44]. Many current state-of-the-art models seem to select checkpoints by the target risk. But, as this result indicates, such comparisons may be misleading for real-world applications.

Source Domain Selection. We examine whether SND can select the best source domain given a target domain using the OfficeHome dataset. The task is to predict the best

Method	Target: R		Target: Ar		Target: Cl		Target: Pr	
	Acc.	S	Acc.	S	Acc.	S	Acc.	S
Lower bound	67.5	Cl	56.3	Pro	43.2	Pro	64.4	Cl
SND (Ours)	74.6	Ar	69.2	R	50.1	R	78.3	R
Upper bound	76.0	Pro	69.6	R	51.1	R	79.0	R

Table 6: Source domain selection experiments using the Office-Home dataset. We show the accuracy of the selected model and selected source domain (R: Real, Ar: Art, Cl: Clipart, Pr: Product). SND selects the best source domain except for the Real domain. Even in that case, the selected model and the oracle accuracy perform similarly.

Input Layer	A to D	W to A	A to P
Middle	0.437	0.594	0.474
Last w/o Softmax	0.215	0.388	0.301
Last w Softmax	**0.163**	**0.348**	**0.285**

Table 7: **Analysis of input features.** Within-class variance normalized by variance of all samples. Features of the last layer after softmax show the smallest variance within-class.

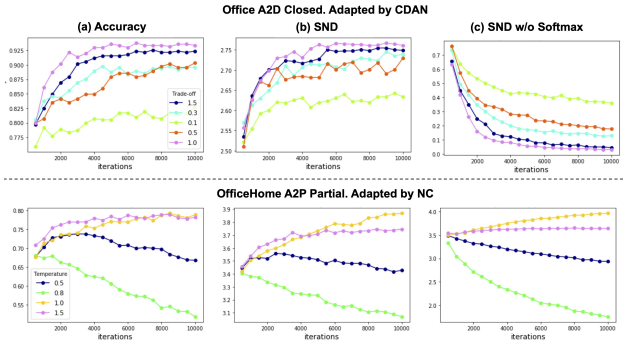


Figure 6: **Analysis of softmax features.** We remove the softmax layer to obtain target features and compute *SND w/o Softmax*. The accuracy and *SND w/o Softmax* have correlation, but the correlation depends on adaptation methods. Applying softmax makes the correlation consistent across methods.

source domain from 3 candidates given a target domain. For simplicity, we do not use any adaptation method, thus we just train a model using source samples and evaluate using unlabeled target samples. As shown in Fig. 6, though SND does not always predict the best source domain, it always returns a model with upper-bound level performance.

4.4. Analysis

Effectiveness of Softmax Normalization of Features. Here, we analyze the effect of using softmax normalized features to compute S_{ij} in Sec. 3.1. We compute the relative within-class variance, *i.e.*, within-class variance divided by the variance of all classes, and compare between different features in Table 7. The features we employ (*Last w Softmax*) show the smallest relative variance, *i.e.*, they separate each sample from other classes the best. Therefore, using the features allows us to ignore samples of other classes effectively in computing Eq. 2.

Next, we track SND without softmax normalization S_{ij} in Fig. 6. Note that we retain the softmax which normalizes the rows of the similarity matrix in Eq. 2. The accuracy and *SND w/o Softmax* have correlation, but the correlation depends on adaptation methods. The softmax normalization has the effect of highlighting difference between within-class and between class variance, which is a key to the success of SND. Other normalization methods, such as L2, didn't have the same effect.

Possible Failure Cases and How to Avoid Them. As we mention in the method section, first, if a model is not trained at all, the output does not characterize features of the target samples and SND does not work well. We can easily address this by monitoring the training loss on source samples. Second, one can also fool SND by training a model to collapse all target samples into a single point. Empirically, we find that such a degenerate solution is hard to detect with any metrics including SND. One possible solution is to compare the feature visualizations of an adapted and an initial model. We leave further analysis to future work.

5. Conclusion and Recommendations

In this paper, we studied the problem of validating unsupervised domain adaptation methods, and introduced a novel criterion that considers how well target samples are clustered. Our experiments reveal a problem in existing methods' validation protocols. Therefore our recommendations to evaluate UDA algorithms in future research are as follows:

- Report which HPO method is used and describe the detail of validation if it includes hyper-parameters, *e.g.* number of hold-out source samples.
- A space to search hyper-parameters can be defined with the scale of loss and insight from previous works, but should be clearly discussed.
- Show the curve of the metric and accuracy.
- Publish implementations, including code for HPO.

It is important to design adaptation methods considering how HPO works on them. For example, methods requiring many hyper-parameters are hard to validate and, as we see in the right of Fig. 6, the difficulty of unsupervised validation differs from method to method.

Applying this protocol may reveal methods with highly performant upper-bounds on accuracy, but which are difficult to tune with any unsupervised validation criterion, including ours. In such scenarios, it may be reasonable to use a small set of target labels. However, this should be clearly discussed in the paper.

Finally, HPO is also crucial in open-set DA [4, 36] and domain-adaptive object detection [9], and the topic of generalization in general. We leave extensions to these other tasks to future work.

Acknowledgment. This work was supported by Honda, DARPA LwLL and NSF Award No. 1535797.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010. 5

[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[3] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Adv. Neural Inform. Process. Syst.*, 2016. 2

[4] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *Int. Conf. Comput. Vis.*, 2017. 8

[5] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Eur. Conf. Comput. Vis.*, 2018. 1, 4

[6] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 5

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848, 2017. 6

[8] Xinyang Chen, Sinan Wang, Mingsheng Long, and Jianmin Wang. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *Int. Conf. Mach Learn.*, 2019. 1

[9] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 1, 2, 8

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 5

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2009. 1

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Int. Conf. Mach Learn.*, 2014. 1, 2, 3, 4, 5, 6

[13] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *Eur. Conf. Comput. Vis.*, 2016. 2

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, 2017. 1

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 6

[16] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 1, 2

[17] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *Eur. Conf. Comput. Vis.*, 2020. 1, 2, 3, 5, 6, 7

[18] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Adv. Neural Inform. Process. Syst.*, 2012. 1

[20] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013. 5, 6, 7

[21] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. In *Int. Conf. Mach Learn.*, 2015. 1

[22] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Adv. Neural Inform. Process. Syst.*, 2018. 1, 2, 5, 6, 7

[23] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(11):2579–2605, 2008. 6

[24] Pietro Morerio, Jacopo Cavazza, and Vittorio Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. *arXiv preprint arXiv:1711.10288*, 2017. 2, 3, 5, 6, 7

[25] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[26] Parth Natekar and Manik Sharma. Representation based complexity measures for predicting generalization in deep learning, 2020. 2

[27] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *Int. Conf. Comput. Vis.*, 2019. 5, 7

[28] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 5, 7

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Adv. Neural Inform. Process. Syst.*, 2015. 1

[30] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Eur. Conf. Comput. Vis.*, 2016. 5

[31] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000. 3

[32] Paolo Russo, Fabio M Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-

directional adaptive gan. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8099–8108, 2018. 2

[33] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Eur. Conf. Comput. Vis.*, 2010. 5

[34] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. *arXiv preprint arXiv:2002.07953*, 2020. 1, 2, 3, 5, 6, 7

[35] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Int. Conf. Mach Learn.*, 2017. 2, 5

[36] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Eur. Conf. Comput. Vis.*, 2018. 8

[37] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[38] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *Int. Conf. Learn. Represent.*, 2018. 2

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014. 1

[40] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. Covariate shift adaptation by importance weighted cross validation. *JMLR*, 8(May):985–1005, 2007. 1, 2, 3

[41] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016. 1

[42] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision (ECCV) Workshop*, 2016. 2

[43] Hui Tang, Ke Chen, and Kui Jia. Unsupervised domain adaptation via structurally regularized deep clustering. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8725–8735, 2020. 1, 2

[44] 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, 2, 5, 7

[45] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Int. Conf. Comput. Vis.*, pages 4068–4076, 2015. 2

[46] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[47] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv*, 2014. 1

[48] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 5

[49] Riccardo Volpi, Pietro Morerio, Silvio Savarese, and Vittorio Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[50] 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, 2, 5, 7

[51] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *Adv. Neural Inform. Process. Syst.*, 2019. 1

[52] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Int. Conf. Comput. Vis.*, 2019. 1

[53] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards accurate model selection in deep unsupervised domain adaptation. In *Int. Conf. Mach Learn.*, 2019. 1, 2, 3, 5, 6, 7

[54] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3801–3809, 2018. 2

[55] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Int. Conf. Comput. Vis.*, 2019. 1, 5

[56] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Eur. Conf. Comput. Vis.*, 2018. 1, 2, 5