This WACV 2022 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Semi-supervised Domain Adaptation via Sample-to-Sample Self-Distillation

Jeongbeen Yoon Dahyun Kang Minsu Cho
Pohang University of Science and Technology (POSTECH), South Korea
{jeongbeen,dahyun.kang,mscho}@postech.ac.kr

Abstract

*Semi-supervised domain adaptation (SSDA) is to adapt a learner to a new domain with only a small set of labeled samples when a large labeled dataset is given on a source domain. In this paper, we propose a pair-based SSDA method that adapts a model to the target domain using self-distillation with sample pairs. Each sample pair is composed of a teacher sample from a labeled dataset (*i.e.*, source or labeled target) and its student sample from an unlabeled dataset (*i.e.*, unlabeled target). Our method generates an assistant feature by transferring an intermediate style between the teacher and the student, and then train the model by minimizing the output discrepancy between the student and the assistant. During training, the assistants gradually bridge the discrepancy between the two domains, thus allowing the student to easily learn from the teacher. Experimental evaluation on standard benchmarks shows that our method effectively minimizes both the inter-domain and intra-domain discrepancies, thus achieving significant improvements over recent methods.*

1. Introduction

Deep neural networks have shown impressive performance in learning tasks on a domain where a large number of labeled data are available for training [14, 20, 33]. However, they often fail to generalize to a new domain where the distribution of input data significantly deviates from the original domain, *i.e.*, when a domain gap arises. The goal of domain adaptation is to adapt a learner to the new domain (*target*) using the labeled data available from the original domain (*source*). Unsupervised domain adaptation (UDA) attempts to tackle this inter-domain discrepancy problem without any supervision on the target domain, assuming that no labels for samples are available from the target domain in training [11, 16, 25, 32]. In contrast, semi-supervised domain adaptation (SSDA) relaxes the strict constraint, using a small number of additional labels on the target data, *e.g.*, a few labels per class [31]. As we are able to obtain such additional labels easily on the target data, it renders the adaptation problem more practical and better situated in learning.

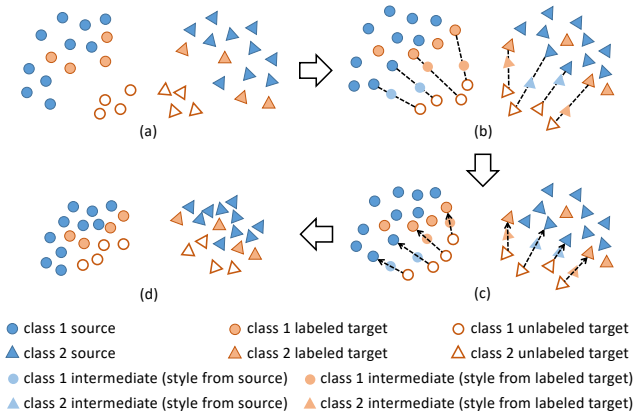


Figure 1: Sample-to-sample self-distillation (S^3D). (a) Before adaptation, both inter-domain and intra-domain discrepancy are present. (b) S^3D composes sample pairs of a teacher (labeled one) and a student (unlabeled one), and generates assistants by transferring an intermediate style of each pair. (c) The assistants bridge the discrepancy between the two domains, thus facilitating domain adaptation by self-distillation. (d) As the result, S^3D reduces both the inter- and intra-domain discrepancy. See Section 3.2 for detail.

Empirical results [31] show that a naïve adaptation of UDA to SSDA, *e.g.*, considering the labeled samples on the target domain as a part of those on the source domain, suffers from the effect of target intra-domain discrepancy, *i.e.*, the distribution of labeled samples on the target domain is separated from that of unlabeled samples during training. We consider the intra-domain discrepancy and the aforementioned inter-domain discrepancy as major challenges of SSDA, and we illustrate them in Figure 2. Previous methods for SSDA [19, 31] aims to address the issue using a prototype-based approach; they create a prototype representation for each class and reduce the distance between each prototype and its nearby unlabeled samples.

In this paper, we propose a new SSDA approach, dubbed *sample-to-sample self-distillation* (S^3D), that leverages rich sample-to-sample relations rather than prototype-to-data relations. Our method takes a labeled sample as a teacher

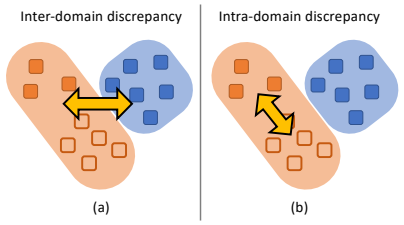


Figure 2: Two problems of SSDA. An Orange region represents a target domain and a blue region represents a source domain.

on either source or target domain and an unlabeled sample as a student. When the teacher comes from the source domain, it minimizes the inter-domain discrepancy between the source and the target. When the teacher is a labeled sample on the target domain, it effectively suppresses the intra-domain discrepancy within the target. For the reason that naïvely reducing the domain gap is demanding, we generate assistant features which support to bridge the domain gap. It is known that the domain and the style of an image are closely related and training with mixed style features helps to bridge the domain discrepancy [12,45]. Inspired by the fact, the assistant features are created by transferring intermediate styles between the teacher and the student. Then a model is trained by minimizing the output discrepancy between the assistant and the student. The assistant features smoothly bridge the discrepancy between the two domains, thus making the student easily learns from the teacher.

To generate reliable pairs of the teacher and the student, we employ pseudo-labeling [21] and present a new form of reliability evaluation on the pseudo-label motivated by [44]. Compared to the previous prototype-based approach, our pair-based approach fully exploits rich and diverse supervisory signals via data-to-data distillation and effectively adapts to the target domain by minimizing both the intra-domain and inter-domain discrepancy. The contributions of this paper are summarized as follows:

- We propose *sample-to-sample self-distillation* (S^3D) that exploits rich sample-to-sample relations using self-distillation.
- We generate assistant features of which the style is represented by an intermediate of the source and the target to fill the domain gap, thus facilitating the adaptation.
- We show that S^3D effectively adapts a network to a target domain by alleviating both the inter- and intra-domain discrepancy and S^3D sets a new state of the art.

2. Related work

Semi-supervised domain adaptation. The goal of semi-supervised domain adaptation (SSDA) is to adapt a model on the target domain with a few labels of target data [31]. Although SSDA has been considered in [1, 9, 40], most recent research has explored unsupervised domain adaptation (UDA). The main issue of domain adaptation is the gap between the source and the target domain distributions. Previous UDA methods focus on aligning the two domain distributions. Adversarial learning between a domain-classifier and a feature extractor is one of the representative UDA approaches [11, 23, 25, 32, 39]. Learning with pseudo-labels [21] is another approach in UDA [4, 8, 38, 44]. To supplement the absence of target domain labels, the network assigns labels to the target data in a certain standard. The network then utilizes the obtained pseudo-labels as supervision for training using the target domain data. SSDA is re-examined in Minimax Entropy (MME) [31] for taking the advantage of extra supervision. With a minor effort, the model benefits from just a few target labels. MME discovers the ineffectiveness of previous UDA methods in SSDA, and proposes a new approach for the task. They minimize the distance between the class prototypes and nearby unlabeled target samples by minimax entropy. After MME, several new SSDA methods are followed. [18] generate bidirectional adversarial samples from source to target domain and from target to source domain to fill the domain gap. Attract, Perturb, and Explore (APE) [19] analyzes the target intra-domain discrepancy issue, and suggests to minimize the gap using Maximum Mean Discrepancy, perturbation loss, and the class prototypes. Among the previous work, MME and APE use the class prototypes and adapt to the target domain for SSDA. We tackle the issues of SSDA in a simple pair-based way by applying self-distillation different from previous work. Dissimilar to the prototype-based way, our pair-based method enables unlabeled target samples to be trained with more abundant supervision.

Style Manipulation. The style of images has been manipulated to increase the recognition ability of neural networks [12, 16, 17, 34, 45]. Previous work of style transfer [17,34] find that mean and standard deviation of an intermediate feature from neural networks are closely related to the style of an image. Further, [45] reveals that domain is related to the image style and suggest to mix the style of given source domains to generalize the model in domain generalization. Our method is motivated by the fact that the intermediate domain style helps to minimize the domain discrepancy [12,45]. Unlike [45], which trains the model by exposing it to various styles from source domains, our method generates intermediate style features (assistants) from labeled (teachers) and unlabeled samples (students) to guide students. Also, we use the assistant only for matching its soft output with the student's one. Therefore, our method forces to produce the same results between two features of the same content with different styles.

Knowledge distillation. The idea of knowledge distillation (KD) is to train a model *(student)* by transferring knowledge

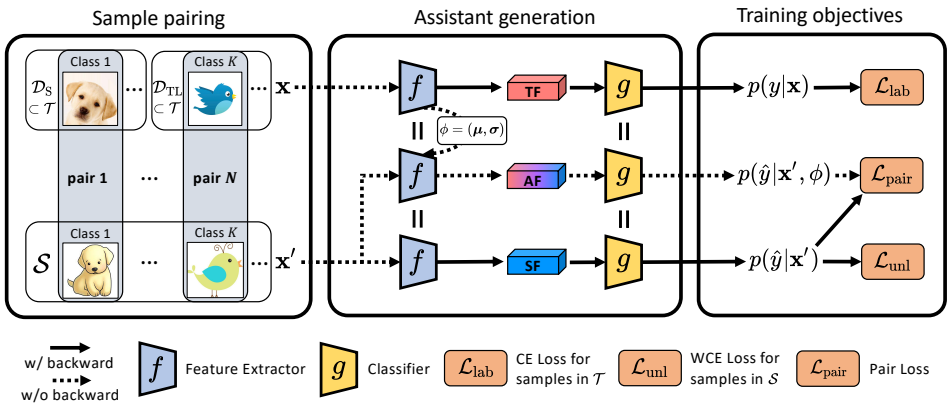


Figure 3: Overview of the Sample-to-Sample Self-Distillation (S^{3}D). We use a feature extractor and a classifier trained in the pre-training stage (Section 3.1). S^{3}D consists of the reliable student-set generation step (RSS), and the sample pairing, assistant feature generation and self-distillation step. We omit the reliable student-set generation step (RSS) in the figure. The abbreviations of TF, AF, and SF represent the teacher, the assistant, and the student feature, respectively. In the sample pairing step, the model pairs a student and a teacher of the same (pseudo-)label. In the assistant generation step, the teacher is forwarded and the set of mean and standard deviation of the teacher is extracted. The extracted ϕ is utilized for adjusting intermediate style to the student to generate the assistant feature. The $\mathcal{L}_{\text{pair}}$ is then calculated to minimize the difference between the outputs of the assistant and the student. We omit feature normalization, temperature scaling, and softmax operation for simplicity.

extracted from another model *(teacher)* that is more powerful than the student [2, 3]. A series of study on KD has shown its attractive characteristics such as regularizing the student [41], stabilizing training [6], and preventing models to be overconfident [42]. One line of work on KD assumes two independent teacher and student networks sharing an input sample, and maps the output of the student to that of the teacher [15, 27, 30, 43]. This branch of work motivates GDSDA [1], which proposes to use multiple pre-trained source models to give predictions to a target model for domain adaptation tasks. The other interesting line of work on KD investigates self-knowledge distillation; a single network is trained by the knowledge from itself [10, 37, 42]. Our design resorts to the second line of work. We propose to minimize Kullback–Leibler divergence of two predictions between an intermediate style feature and its corresponding unlabeled target sample in the form of self-distillation. This learning objective naturally conforms to the goal of domain adaptation: adapting a learner to a target domain by aligning two samples sharing semantics yet visually diverse.

3. Method

The task of semi-supervised domain adaptation is formulated as to classify unlabeled samples on a target domain using labeled samples on a source domain together with a limited number of labeled samples on the target domain [19, 24, 31]. Let us consider three datasets given in this context: a source dataset $\mathcal{D}_{\text{S}} = \{(\mathbf{x}_{\text{S}}^{(i)}, y_{\text{S}}^{(i)})\}_{i=1}^{N_{\text{S}}}$, a labeled target dataset $\mathcal{D}_{\text{TL}} = \{(\mathbf{x}_{\text{TL}}^{(j)}, y_{\text{TL}}^{(j)})\}_{j=1}^{N_{\text{TL}}}$, and an unlabeled target dataset $\mathcal{D}_{\text{TU}} = \{\mathbf{x}_{\text{TU}}^{(k)}\}_{k=1}^{N_{\text{TU}}}$, where $\mathbf{x}$, y, and N denote a sample, its corresponding label, and the number of samples, respectively. Here, we are given only a limited number of labeled samples per class on the target domain, *i.e.*, $N_{\text{TL}} \ll N_{\text{TU}}$. The two domains share the same label space $y \in \{1, \cdots, K\}$ but with different input distribution. In this setup, we train a model on $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{S}} \cup \mathcal{D}_{\text{TL}}$, and $\mathcal{D}_{\text{TU}}$, and then evaluate it on $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{TU}}$ with its ground-truth labels. In training, we validate models on additional labeled target set $\mathcal{D}_{\text{val}}$ of $\mathcal{D}_{\text{val}} \cap \mathcal{D}_{\text{train}} = \mathcal{D}_{\text{val}} \cap \mathcal{D}_{\text{test}} = \emptyset$. We select the best model and search hyper-parameters on the validation set.

3.1. Classifier model and its pre-training

Our model consists of two parts: a feature extractor $f(\cdot; \boldsymbol{\theta})$ and a classifier $g(\cdot; \mathbf{W})$, where $\boldsymbol{\theta}$ and $\mathbf{W}$ denote trainable parameters. We use a convolutional neural network for $f(\cdot; \boldsymbol{\theta})$, and a distance-based classifier for $g(\cdot; \mathbf{W})$ [5, 36]. The distance-based classifier computes its output as the cosine similarity between the input feature $\mathbf{h}$ and each column $\mathbf{w}_k$ of $\mathbf{W}$:

$$p(y|\mathbf{x}) = \text{softmax}\left(\frac{g(f(\mathbf{x}; \boldsymbol{\theta}); \mathbf{W})}{T}\right),$$
$$\text{where } g(\mathbf{h}; \mathbf{W}) = \left[\frac{\mathbf{w}_1}{\|\mathbf{w}_1\|}; \cdots; \frac{\mathbf{w}_K}{\|\mathbf{w}_K\|}\right]^\top \frac{\mathbf{h}}{\|\mathbf{h}\|}, \quad (1)$$

where the final prediction $p(y|\mathbf{x})$ is obtained via softmax operation with a temperature T. In the following subsections, we often omit the function parameters, $\boldsymbol{\theta}$ and $\mathbf{W}$, for notational simplicity.

We pre-train the model with labeled samples in $(\mathbf{x}, y) \in \mathcal{T}$, where $\mathcal{T} = \mathcal{D}_{\text{S}} \cup \mathcal{D}_{\text{TL}}$, via minimizing the cross-entropy loss:

$$\mathcal{L}_{\text{lab}} = \mathbb{E}_{(\mathbf{x},y)}[-\log p(y|\mathbf{x})]. \quad (2)$$

Algorithm 1 Sample-to-Sample Self-Distillation.

Input: $\mathcal{T}$: Teacher set
Input: $\boldsymbol{\theta}$ and $\mathbf{W}$: Pre-trained weights
Input: M: student-set generation interval
for $e \leftarrow$ 1 to `max_steps` **do**
  if $e \bmod M$ is 0 **then**
    Update student set $\mathcal{S}$ ▷ Eq. (4)
  end if
  $(\mathbf{x}, y) \sim \mathcal{T}$
  $(\mathbf{x}', \hat{y}) \sim \mathcal{S}$ such that $\hat{y} = y$
  $\phi = (\boldsymbol{\mu}, \boldsymbol{\sigma}) \leftarrow$ get feature statistics from $f(\mathbf{x}; \boldsymbol{\theta})$
  $p(y|\mathbf{x}) \leftarrow \text{softmax}(g(f(\mathbf{x}; \boldsymbol{\theta}); \mathbf{W})/T)$
  $p(\hat{y}|\mathbf{x}', \phi) \leftarrow \text{softmax}(g(f'(\mathbf{x}'; \boldsymbol{\theta}, \phi); \mathbf{W})/T)$
  $p(\hat{y}|\mathbf{x}') \leftarrow \text{softmax}(g(f(\mathbf{x}'; \boldsymbol{\theta}); \mathbf{W})/T)$
  $\mathcal{L}_{\text{lab}} \leftarrow \text{CE}(p(y|\mathbf{x}), y)$ ▷ Eq. (2).
  $\mathcal{L}_{\text{unl}} \leftarrow \text{WCE}(p(\hat{y}|\mathbf{x}'), \hat{y})$ ▷ Eq. (9).
  $\mathcal{L}_{\text{pair}} \leftarrow D_{\text{KL}}(p(\hat{y}|\mathbf{x}', \phi), p(\hat{y}|\mathbf{x}'))$ ▷ Eq. (8).
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{lab}} + \mathcal{L}_{\text{unl}} + \lambda \mathcal{L}_{\text{pair}}$
  update $\boldsymbol{\theta}$ and $\mathbf{W}$ with $\mathcal{L}$ using SGD
  if $e \bmod$ `val_freq` is 0 **then**
    validate and early-stop
  end if
end for

This pre-training improves the performance of S^{3}D and also speeds up its convergence.

3.2. Sample-to-sample self-distillation (S^{3}D)

The *sample-to-sample self-distillation* (S^{3}D) is designed to perform SSDA by simultaneously minimizing both the inter-domain discrepancy (between the source and the target) and the intra-domain discrepancy (within the target). It achieves the goal by alternating two steps: the *student-set generation step*, and the training step. The training step consists of *sample pairing, assistant generation and self-distillation*. At the student-set generation step, we pseudo-label samples from unlabeled target dataset $\mathcal{D}_{\text{TU}}$ and select reliable ones using reliability evaluation. The resultant set $\mathcal{S}$ is used for student samples in self-distillation. At the training step, we randomly produce teacher-student pairs with the same class label, generating corresponding assistant features, and perform self-distillation by minimizing the distance between the assistant and the student predictions. In pairing, we take one sample from $\mathcal{T}$ (as a teacher) and the other from $\mathcal{S}$ (as a student). The overall procedure is summarized in Algorithm 1 and also illustrated in Figure 3. In the following, we explain the details of each step and describe the overall training objective.

Reliable student-set generation. This step consists of pseudo-labeling and reliability evaluation. We assign a class label $\hat{y}$ to each unlabeled sample $\mathbf{x}' \in \mathcal{D}_{\text{TU}}$, and construct a pseudo-labeled set of the student samples $\{(\mathbf{x}'^{(l)}, \hat{y}^{(l)})\}_{l=1}^{N_{\text{TU}}}$; we simply take a pseudo-label $\hat{y}$ of $\mathbf{x}'$ as the class index k of the maximum prediction value:

$$\hat{y} = \underset{k \in \{1,2,\ldots,K\}}{\text{argmax}} \; p(y = k|\mathbf{x}'). \tag{3}$$

Although pseudo-labeling enables supervised training on unlabeled samples, pseudo-labels are often incorrect, in particular, in an early stage of training. We thus drop unreliable samples to compose $\mathcal{S}$ for pairing. Let $\pi_j(\cdot)$ be a selection operator that selects j^{th} largest value. We construct a student set by reliability evaluation:

$$\begin{aligned}\mathcal{S} = \{(\mathbf{x}', \hat{y}) | (\pi_1(g(f(\mathbf{x}'))) - \pi_2(g(f(\mathbf{x}'))) > \delta) \\ \vee \, (\pi_1(p(\hat{y}|\mathbf{x}')) > \alpha); \forall \mathbf{x}' \in \mathcal{D}_{\text{TU}}\},\end{aligned} \tag{4}$$

where δ is an average margin between the first and the second highest logits of all unlabeled target on the pre-trained model, and α is a hyper-parameter. We include analysis on the margin δ in the supplementary materials in detail. The first condition is met when a margin between the largest and the second largest value of the logit is high enough [44]. The second condition is met when the absolute largest class probability score is high enough. In this way, the model assigns pseudo-labels to confident samples only and generates reliable student-set (RSS) so that the model can take reliable pairs.

Sample pairing and assistant generation. After obtaining $\mathcal{S}$, we construct a pair of a labeled sample $(\mathbf{x}, y) \in \mathcal{T}$ and a pseudo-labeled sample $(\mathbf{x}', \hat{y}) \in \mathcal{S}$. We then set a labeled sample as a teacher sample, and set an unlabeled sample as a student sample. An assistant feature is generated by an assistant generation (AG) module, which transfers an intermediate style of the teacher and the student. The content of the assistant follows the one of the student. To blend the style of the pair, we construct new style statistics (*i.e.*, mean and standard deviation) by interpolating between the style statistics of the teacher and the student [17, 45]. Let $f_n(\cdot; \boldsymbol{\theta}_n)$ be the n-th layer of $f(\cdot; \boldsymbol{\theta})$ and the intermediate feature $\mathbf{z}_n = f_n(\mathbf{x}; \boldsymbol{\theta}_n)$, where $\mathbf{z}_n \in R^{C \times H \times W}$. The style fused feature is calculated as

$$\begin{aligned}\text{AG}(\mathbf{z}'_n) &= \gamma \frac{\mathbf{z}'_n - \mu(\mathbf{z}'_n)}{\sigma(\mathbf{z}'_n)} + \beta, \\ \text{where } \beta &= \epsilon \mu(\mathbf{z}_n) + (1 - \epsilon)\mu(\mathbf{z}'_n), \\ \gamma &= \epsilon \sigma(\mathbf{z}_n) + (1 - \epsilon)\sigma(\mathbf{z}'_n),\end{aligned} \tag{5}$$

where ϵ is extracted from the Beta distribution $Beta(\rho, \rho)$ with a hyper-parameter ρ and $\mathbf{z}'_n$ is extracted from the student sample $\mathbf{x}'$. We use $\rho = 0.1$ following [45]. $\mu(\mathbf{z})$, $\sigma(\mathbf{z}) \in R^C$ denote mean and standard deviation of the feature $\mathbf{z}$ across the spatial dimension, respectively:

Net	Method	R to C		R to P		P to C		C to S		S to P		R to S		P to R		MEAN	
		1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
AlexNet	S+T	43.3	47.1	42.4	45.0	40.1	44.9	33.6	36.4	35.7	38.4	29.1	33.3	55.8	58.7	40.0	43.4
	DANN	43.3	46.1	41.6	43.8	39.1	41.0	35.9	36.5	36.9	38.9	32.5	33.4	53.6	57.3	40.4	42.4
	ADR	43.1	46.2	41.4	44.4	39.3	43.6	32.8	36.4	33.1	38.9	29.1	32.4	55.9	57.3	39.2	42.7
	CDAN	46.3	46.8	45.7	45.0	38.3	42.3	27.5	29.5	30.2	33.7	28.8	31.3	56.7	58.7	39.1	41.0
	ENT	37.0	45.5	35.6	42.6	26.8	40.4	18.9	31.1	15.1	29.6	18.0	29.6	52.2	60.0	29.1	39.8
	MME	48.9	55.6	48.0	49.0	46.7	51.7	36.3	39.4	39.4	43.0	33.3	37.9	56.8	60.7	44.2	48.2
	APE	47.7	54.6	49.0	50.5	46.9	52.1	38.5	42.6	38.5	42.2	33.8	38.7	57.5	61.4	44.6	48.9
	S^3D w/o AF	52.3	56.2	48.7	51.2	48.0	51.3	39.2	43.5	40.6	46.5	37.4	**39.8**	59.5	65.1	46.5	50.5
	S^3D (ours)	**53.5**	**56.5**	**51.8**	**52.2**	**49.1**	**53.9**	**40.1**	**44.4**	**44.9**	**48.7**	**39.9**	39.2	**61.7**	**65.4**	**48.7**	**51.5**
ResNet34	S+T	55.6	60.0	60.6	62.2	56.8	59.4	50.8	55.0	56.0	59.5	46.3	50.1	71.8	73.9	56.9	60.0
	DANN	58.2	59.8	61.4	62.8	56.3	59.6	52.8	55.4	57.4	59.9	52.2	54.9	70.3	72.2	58.4	60.7
	ADR	57.1	60.7	61.3	61.9	57.0	60.7	51.0	54.4	56.0	59.9	49.0	51.1	72.0	74.2	57.6	60.4
	CDAN	65.0	69.0	64.9	67.3	63.7	68.4	53.1	57.8	63.4	65.3	54.5	59.0	73.2	78.5	62.5	66.5
	ENT	65.2	71.0	65.9	69.2	65.4	71.1	54.6	60.0	59.7	62.1	52.1	61.1	75.0	78.6	62.6	67.6
	MME	70.0	72.2	67.7	69.7	69.0	71.7	56.3	61.8	64.8	66.8	61.0	61.9	76.1	78.5	66.4	68.9
	APE	70.4	76.6	70.8	**72.1**	72.9	76.7	56.7	63.1	64.5	66.1	63.0	67.8	76.6	79.4	67.6	71.7
	S^3D w/o AF	**73.4**	75.3	69.2	70.8	**73.4**	74.4	60.2	63.1	66.1	69.1	62.8	64.7	79.3	79.7	69.2	71.0
	S^3D (ours)	73.3	**75.9**	68.9	**72.1**	**73.4**	**75.1**	**60.8**	**64.4**	**68.2**	**70.0**	**65.1**	**66.7**	**79.5**	**80.3**	**69.9**	**72.1**

Table 1: Classification accuracy on the DomainNet dataset (%) for one-shot and three-shot on four domains (R: Real, C: Clipart, P: Painting, S: Sketch). † denotes that we reproduce the baseline.

$$\mu(\mathbf{z}) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}\mathbf{z}_{hw}, \tag{6}$$

$$\sigma(\mathbf{z}) = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}(\mathbf{z}_{hw}-\mu(\mathbf{z}))^2}. \tag{7}$$

AG operation is applied to the intervals of the neural network. See supplementary material for more details. Note that, unlike [45] that directly trains the model with stylized features (*i.e.*, the gradients are back-propagated through the features), the gradients are blocked when we apply AG module to the intermediate features and generate assistant features. $f'(\cdot;\boldsymbol{\theta},\phi)$ is the feature extractor f with AG operation. Thus the assistant feature is denoted as $f'(\mathbf{x}';\boldsymbol{\theta},\phi)$, where $\phi=(\boldsymbol{\mu},\boldsymbol{\sigma})\in\{\mu(\mathbf{z}_n),\sigma(\mathbf{z}_n)\}_{n=1}^{N}$.

Self-distillation and training objectives. When implementing a pair loss, we cannot sample all the pairs from two sets, since their cardinalities are large enough. Thus, we uniformly sample the pairs from the set of all possible pairs. The work [22] shows this sampling technique is an unbiased estimator of true expectation. The pair loss is calculated as

$$\mathcal{L}_{\text{pair}} = \mathbb{E}_{(\mathbf{x},y),(\mathbf{x}',\hat{y})}\left[[\![\hat{y}=y]\!]D_{\text{KL}}(p(\hat{y}|\mathbf{x}',\phi)\|p(\hat{y}|\mathbf{x}'))\right], \tag{8}$$

where $[\![\cdot]\!]$ denotes Iverson brackets. ϕ is calculated when the teacher sample $\mathbf{x}$ is forwarded through the feature extractor f. It effectively reduces the inter-domain discrepancy using pairs between $\mathcal{D}_{\text{S}}$ and $\mathcal{S}$, while suppressing the intra-domain discrepancy using pairs between $\mathcal{D}_{\text{TL}}$ and $\mathcal{S}$. This effect is validated in Figure 5.

To improve our training, we introduce an additional loss using student samples with pseudo-labels. We utilize the latest prediction of student samples to decide the reliability of pseudo-labels and multiply it to the cross-entropy loss of each student sample assuming its pseudo-label as its true label, *i.e.*, we use a weighted cross-entropy loss (WCE) for training student samples:

$$\mathcal{L}_{\text{unl}} = \mathbb{E}_{(\mathbf{x}',\hat{y})}[-\omega\log p(\hat{y}|\mathbf{x}')], \quad \text{where } \omega = p(\hat{y}|\mathbf{x}'). \tag{9}$$

By doing so, ω plays a role in weighting prediction so that less confident samples from $\mathcal{S}$ give less effect on updating the model.

Our total loss in training thus consists of three terms:

$$\mathcal{L} = \mathcal{L}_{\text{lab}} + \mathcal{L}_{\text{unl}} + \lambda\mathcal{L}_{\text{pair}}, \tag{10}$$

where λ is a weighting hyper-parameter for the pair loss. The model is updated by minimizing $\mathcal{L}$ for M iterations. We iterate alternating the student-set generation step and the sample pairing, assistant generation and self-distillation step until the model converges on the validation set.

4. Experiments

We compare S^3D with current state-of-the-art methods on two standard SSDA benchmarks. We demonstrate that S^3D is generally applicable when there are zero or many target domain labels are available. Through extensive ablation studies, we verify the effectiveness of each proposed component in detail. For more experimental setups and results, refer to the supplementary material. We will make our code publicly available upon acceptance.

Net	Method	R to C	R to P	R to A	P to R	P to C	P to A	A to P	A to C	A to R	C to R	C to A	C to P	MEAN
AlexNet	S+T	37.5	63.1	44.8	54.3	31.7	31.5	48.8	31.1	53.3	48.5	33.9	50.8	44.1
	DANN	42.5	64.2	45.1	56.4	36.6	32.7	43.5	34.4	51.9	51.0	33.8	49.4	45.1
	ADR	37.8	63.5	45.4	53.5	32.5	32.2	49.5	31.8	53.4	49.7	34.2	50.4	44.5
	CDAN	36.1	62.3	42.2	52.7	28.0	27.8	48.7	28.0	51.3	41.0	26.8	49.9	41.2
	ENT	26.8	25.8	45.8	56.3	23.5	21.9	47.4	22.1	53.4	30.8	18.1	53.6	38.8
	MME	42.0	**69.6**	48.3	58.7	**37.8**	34.9	52.5	**36.4**	57.0	54.1	**39.5**	59.1	49.2
	APE †	42.1	**69.6**	**49.8**	57.7	35.5	**35.9**	49.2	32.1	55.0	52.7	37.8	57.6	47.9
	S^3D w/o AF	**45.3**	69.5	48.0	58.5	34.8	34.5	55.9	34.6	**57.2**	**56.7**	37.0	**60.3**	49.4
	S^3D (ours)	43.0	70.1	48.4	**60.3**	35.6	35.3	**56.9**	35.5	56.8	55.9	37.5	59.1	**49.5**
ResNet34	S+T	50.9	78.7	65.9	73.6	46.5	54.4	68.6	48.7	73.2	67.1	55.2	64.9	62.3
	MME	60.3	**82.6**	71.0	**79.1**	**57.9**	63.6	74.6	59.2	77.3	73.5	64.1	75.1	69.9
	APE †	60.1	82.4	**73.0**	78.5	53.3	64.6	74.7	53.4	75.7	70.6	61.6	69.1	68.1
	S^3D w/o AF	61.8	82.5	70.3	78.7	54.8	62.5	**75.6**	58.2	**78.3**	**74.9**	**65.3**	**77.5**	70.0
	S^3D (ours)	**63.2**	82.3	71.0	79.0	56.8	**64.7**	75.3	**59.3**	77.4	73.6	64.6	76.1	**70.3**

Table 2: Classification accuracy on the Office-Home dataset (%) for one-shot on four domains (R: Real, C: Clipart, P: Product, A: Art).

4.1. Setup

Datasets. We evaluate our method using two benchmark datasets: DomainNet [29] and Office-Home [35]. DomainNet contains 6 domains each of which has 345 classes. Among them, we use 4 domains (Real, Clipart, Painting, and Sketch) and 126 classes. We use 145,145 images from all 4 domains for our experiment. We choose seven source-to-target domain scenarios following the work of [31]. Office-Home consists of 4 domains (Real, Clipart, Product, and Art) of 65 classes, and 15,588 images in total. We conduct Office-Home experiments on all possible source-to-target domain scenarios.

Implementation details. We follow most of the implementation details of [31] for a fair comparison. We select AlexNet [20] and ResNet-34 [14], both of which are pre-trained on ImageNet [7]. A mini-batch consists of samples from $\mathcal{T}$ and $\mathcal{S}$ at the ratio of 1 to 1. We choose the same number of source and labeled target data to construct teacher samples in the batch. Specifically, we use 128 and 96 samples in AlexNet and ResNet-34 respectively as done in MME. We adopt the SGD optimizer with an initial learning rate of 0.01, a momentum of 0.9, and a weight decay of 0.0005. In the reliable student-set generation step, we set α to 0.95. We set the student-set generation interval M as 100. We validate the model every 1000 iterations during training, and we early-stop training when the model shows no more improvement in 5 validation steps. The details of searching the hyper-parameter λ are described in the supplementary material. We use PyTorch [28] for our experiments.

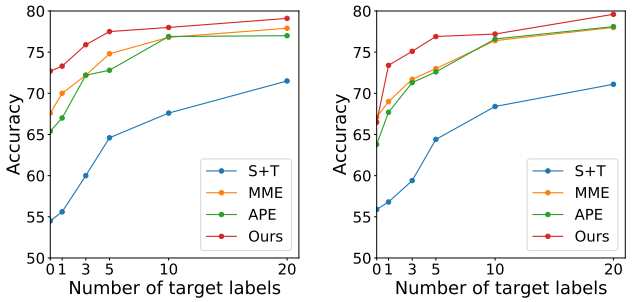
Baselines. We compare our method to competitive SSDA baselines: MME [31], and APE [19]. Additionally, we bring S+T that simply minimizes the cross-entropy loss on the labeled dataset. S^3D w/o AF is our method without the assistant, *i.e.* directly distills the teacher prediction to the student. DANN [11], ADR [32], and CDAN [25], which are the well-known methods in UDA, are also described as a comparison. Further, we include the accuracy of ENT [13].

4.2. Main results

We conduct experiments on both one-shot and three-shot settings with AlexNet and ResNet-34. For a fair comparison, we select the best model on the validation set for all experiments.

Comparison on DomainNet. Table 1 compares the classification accuracy of our method and other baselines on the DomainNet dataset. S^3D achieves higher accuracy than the previous methods in most domain adaptation scenarios. Notably, S^3D is effective where the domain gap between the source and target domain is substantial. In comparison to S+T, for example, S^3D increases accuracy by 17.7%p on Real to Clipart with ResNet in the one-shot setting. Real and Clipart domains appear considerably distinctive to each other because samples in Real domain are photos, whereas, the samples in Clipart domain are artificial illustrations. The mixed style assistant seems to be effective when comparing the accuracy between S^3D and S^3D w/o AF.

Comparison on Office-Home. Table 2 compares the results of our method and others on the Office-Home dataset. Our method shows comparable results with other baselines. As S^3D in DomainNet, the method is powerful in adapting to the quite different domain, for example, on Real to Clipart and Art to Product scenarios. However, the accuracy enhancement of S^3D on Office-Home is not as high as that on DomainNet. We suppose that the small dataset size of the Office-Home derives less performance improvement compared to that on the DomainNet dataset. Note that the dataset size of the DomainNet is ten times larger than that of the Office-Home. Considering that student samples acquire



(a) Real to Clipart. **(b)** Painting to Clipart.

Figure 4: Classification accuracy (%) of two scenarios on DomainNet with increasing number of target labels.

Method	R-C	R-P	P-C	C-S	S-P	R-S	P-R	MEAN
AlexNet								
Source	41.1	42.6	37.4	30.6	30.0	26.3	52.3	37.2
DANN	44.7	36.1	35.8	33.8	35.9	27.6	49.3	37.6
ADR	40.2	40.1	36.7	29.9	30.6	25.9	51.5	36.4
CDAN	44.2	39.1	37.8	26.2	24.8	24.3	54.6	35.9
ENT	33.8	43.0	23.0	22.9	13.9	12.0	51.2	28.5
MME	47.6	44.7	39.9	34.0	33.0	29.0	53.5	40.2
APE †	45.9	47.0	42.0	36.5	37.0	30.3	54.1	41.8
S^{3}D w/o AF	49.3	49.2	42.7	38.1	41.7	**38.0**	54.1	44.7
S^{3}D (ours)	**53.4**	**51.9**	**46.3**	**38.7**	**44.0**	36.4	**57.6**	**46.9**
ResNet34								
Source	54.5	60.2	55.9	49.7	50.1	44.1	72.1	55.2
MME	67.6	66.9	**67.1**	56.4	62.9	58.2	74.5	64.8
APE †	65.4	68.6	63.8	56.4	**65.1**	60.4	**75.3**	65.0
S^{3}D w/o AF	70.1	69.5	66.8	56.3	61.8	59.0	73.4	65.3
S^{3}D (ours)	**72.7**	**70.2**	66.5	**57.2**	63.8	**62.6**	71.2	**66.3**

Table 3: Classification accuracy on the DomainNet dataset (%) in the unsupervised domain adaptation setting.

diverse guidance from source and labeled target dataset using our method, a larger dataset is more advantageous for S^{3}D to give rich guidance to unlabeled targets.

4.3. Varying the number of target labels

Many-shot semi-supervised domain adaptation. We examine our method with increasing target labels per class. Figure 4 reports the many-shot experiment results. S^{3}D consistently outperforms current state-of-the-art along with the increasing number of target domain ground truths. Note that the S+T baseline corresponds to the pre-training stage in our context. This experiment emphasizes the strength of the sample-to-sample training even when abundant target labels are given in the pre-training stage. In the 20-shot case, for example, the pre-training is supervised by 2,520 target domain examples, and yet the following sample-to-sample training stage gains additional accuracy improvement of 8.5%p from the pre-trained model.

Unsupervised domain adaptation. In Table 3, S^{3}D is also shown to be effective even though no target ground truth is available. For UDA experiments, we compose training batch samples from the source dataset and the unlabeled target dataset, and set validation batches as the same as the one used in the semi-supervised setup. S^{3}D outperforms counterparts in most scenarios. It is impressive that S^{3}D excels methods that have been proposed for UDA [11,25,32], each of which involves a domain-adversarial learning objective.

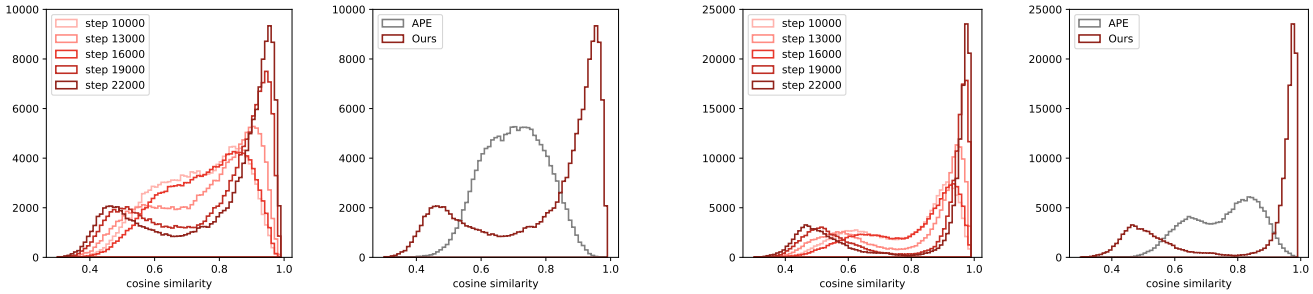
4.4. Ablation study

Ablation study on proposed components. We conduct extensive ablation studies on our major contributions: the pair loss $\mathcal{L}_{\text{pair}}$, the weighted-cross entropy loss $\mathcal{L}_{\text{unl}}$ and the reliable student-set generation (RSS) in Table 4. The top row of Table 4 represents the pre-training stage which is equivalent to the S+T baseline. It attempts to train a network only with labeled samples without any proposed components. The check mark ✓ on the RSS column indicates that we filter out unreliable pseudo-labeled samples using Eq. (4), otherwise we utilize all pseudo-labeled samples into training.

As can be seen in Table 4, the pair loss contributes to performance improvement. Using the pair loss and the weighted cross-entropy together significantly increases the accuracy by a large margin from the top-row baseline on all scenarios. Additionally, RSS further improves performance by preventing harmful alignment from mis-matched pairs. The bottom row completes our method, and it validates that all components are complementary.

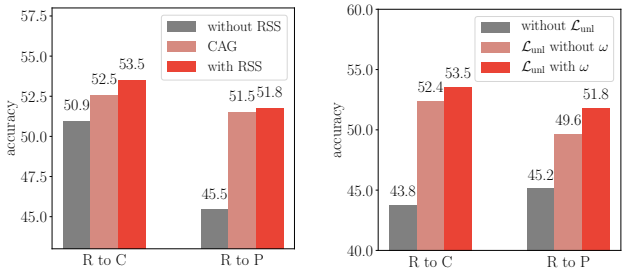
Ablation study on evaluating reliability of pseudo-labels. We examine that discarding unreliable pseudo-labels is crucial in training with pseudo-labels. We compare our RSS with other pseudo-label reliability evaluation scheme. In Figure 6a, *without RSS* indicates that we set all pseudo-labeled samples to a student set regardless of their reliability, and *CAG* [44] indicates that we measure the reliability using the first condition of Eq. (4). While CAG [44] searches for an optimal margin δ, our model sets the δ to the average margin value of all student set thus eliminating a hyper-parameter. Our RSS method effectively constructs reliable pairs thus improving performance over two baselines. Compared to the performance gain from $\mathcal{L}_{\text{unl}}$, which will be reported in the following paragraph with Figure 6b, the gain of reliable paring is clearer. This comparison witnesses again that the pair-based learning brings the major performance improvement in our pipeline.

Ablation study on weighted cross-entropy. In Figure 6b, we verify the effect of the confidence score ω in the weighted cross-entropy of Eq. (9). In Figure 6b, *without* $\mathcal{L}_{\text{unl}}$ indicates that we exclude $\mathcal{L}_{\text{unl}}$ in the overall loss term, and $\mathcal{L}_{\text{unl}}$ *without* ω indicates that we eliminate ω in $\mathcal{L}_{\text{unl}}$. We show that multiplying ω to the cross-entropy term achieves additional performance growth.



(a) **(b)** **(c)** **(d)**

Figure 5: (a) Histograms of cosine similarity between a source and a target embedding (inter-domain similarity) from the pre-training stage until convergence. (b) Inter-domain similarity histograms of APE and S^{3}D. (c) Histograms of cosine similarity between target embeddings (intra-domain similarity) from the pre-training stage until convergence. (d) Intra-domain similarity histograms of APE and S^{3}D.



(a) Ablation of RSS. **(b)** Ablation of $\mathcal{L}_{\text{unl}}$.

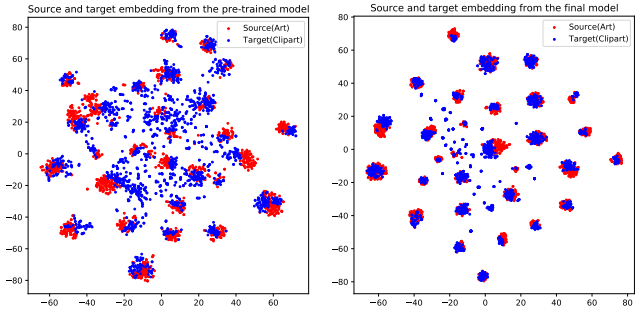
Figure 6: Ablation study of proposed components.

$\mathcal{L}_{\text{unl}}$	$\mathcal{L}_{\text{pair}}$	RSS	R-C	R-P	P-C	C-S	S-P	R-S	P-R	MEAN
✗	✗	✗	56.8	60.5	55.4	51.7	55.5	47.5	72.0	57.1
✓	✗	✗	68.7	65.6	68.8	59.2	64.1	61.6	78.4	66.6
✗	✓	✗	67.4	65.0	67.1	61.2	64.9	62.7	77.5	66.5
✓	✓	✗	69.4	65.7	69.7	61.3	65.5	61.7	78.6	67.4
✓	✓	✓	73.3	68.9	73.4	60.8	68.2	65.1	79.5	69.9

Table 4: Comprehensive ablation study of S^{3}D on the DomainNet dataset for one-shot setting.

4.5. Analysis

Inter-domain and intra-domain discrepancies. Figure 5 visualizes that S^{3}D progressively clusters instances of the same classes by overcoming inter- and intra-domain discrepancies. Figure 5a plots a histogram of cosine similarity between a source and a target embedding from the same class for all classes. Figure 5c plots the one between labeled and unlabeled target embeddings. Figures 5b and 5d visualize cosine similarity histograms from the final model of APE [19] and S^{3}D. The similarity population gradually moves toward 1.0 over iterations, which proves that method guides to map two semantically similar samples to nearby points in the embedding space. While a majority of the same-class embeddings moves close to each other, we observe that a small portion of embeddings pushes apart. This is considered one limitation of leveraging pseudo-labels; wrong pairs misguide the learning process.



(a) **(b)**

Figure 7: t-SNE visualization on the A-C scenario in the Office-Home dataset. (a) Embedding space from the pre-trained model. (b) Embedding space from the final model.

Embedding space visualization. Figure 7 visualizes how S^{3}D clusters instances from two domains over iterations using t-SNE [26]. We observe that S^{3}D clearly enhances the embedding quality from the pre-training stage. We include more qualitative results in the supplementary material.

5. Conclusion

We propose a novel sample-to-sample self-distillation (S^{3}D) by exploiting rich and diverse relations for semi-supervised domain adaptation. By exploiting an assistant feature, the style of which is mixed, S^{3}D efficiently reduce the domain shift. The experiments demonstrate that S^{3}D effectively adapts to a target domain using a single architecture given an extremely few number of labeled target domain samples.

Acknowledgements. This work was supported by Samsung Electronics Co., Ltd. (IO201208-07822-01) and the IITP grants (No.2019-0-01906, AI Graduate School Program - POSTECH) (No.2021-0-00537, Visual common sense through self-supervised learning for restoration of invisible parts in images) funded by Ministry of Science and ICT, Korea.

References

[1] Shuang Ao, Xiang Li, and Charles X Ling. Fast generalized distillation for semi-supervised domain adaptation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 1719–1725, 2017.

[2] Leo Breiman and Nong Shang. Born again trees. *University of California, Berkeley, Berkeley, CA, Technical Report*, 1:2, 1996.

[3] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535–541, New York, NY, USA, 2006. Association for Computing Machinery.

[4] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7354–7362, 2019.

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019.

[6] Xu Cheng, Zhefan Rao, Yilan Chen, and Quanshi Zhang. Explaining knowledge distillation by quantifying the knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12925–12935, 2020.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Zhijie Deng, Yucen Luo, and Jun Zhu. Cluster alignment with a teacher for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9944–9953, 2019.

[9] Jeff Donahue, Judy Hoffman, Erik Rodner, Kate Saenko, and Trevor Darrell. Semi-supervised domain adaptation with instance constraints. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 668–675, 2013.

[10] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. *arXiv preprint arXiv:1805.04770*, 2018.

[11] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[12] Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2477–2486, 2019.

[13] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[16] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998, 2018.

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[18] Pin Jiang, Aming Wu, Yahong Han, Yunfeng Shao, Meiyu Qi, and Bingshuai Li. Bidirectional adversarial training for semi-supervised domain adaptation. *Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020.

[19] Taekyung Kim and Changick Kim. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation. In *European conference on computer vision*, 2020.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] 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*, volume 3, 2013.

[22] Justin Lee. *U-statistics: Theory and Practice*. Marcel Dekker, Inc., New York, 1990.

[23] Seungmin Lee, Dongwan Kim, Namil Kim, and Seong-Gyun Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 91–100, 2019.

[24] Da Li and Timothy Hospedales. Online meta-learning for multi-source and semi-supervised domain adaptation. *arXiv preprint arXiv:2004.04398*, 2020.

[25] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018.

[26] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[27] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[29] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source

domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[30] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[31] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8050–8058, 2019.

[32] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. *arXiv preprint arXiv:1711.01575*, 2017.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[35] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[36] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274, 2018.

[37] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[38] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 5423–5432, 2018.

[39] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. *arXiv preprint arXiv:1912.01805*, 2019.

[40] Ting Yao, Yingwei Pan, Chong-Wah Ngo, Houqiang Li, and Tao Mei. Semi-supervised domain adaptation with subspace learning for visual recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2142–2150, 2015.

[41] Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting knowledge distillation via label smoothing regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3903–3911, 2020.

[42] Sukmin Yun, Jongjin Park, Kimin Lee, and Jinwoo Shin. Regularizing class-wise predictions via self-knowledge distillation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[43] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.

[44] Qiming Zhang, Jing Zhang, Wei Liu, and Dacheng Tao. Category anchor-guided unsupervised domain adaptation for semantic segmentation. In *Advances in Neural Information Processing Systems*, pages 435–445, 2019.

[45] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *International Conference on Learning Representations*, 2021.