PAIR-BASED SELF-DISTILLATION FOR SEMI-SUPERVISED DOMAIN ADAPTATION

Anonymous authors
Paper under double-blind review

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 learner to the target domain using self-distillation with sample pairs. Our method composes the sample pair by selecting 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), and then minimizes the output discrepancy between the two samples. We assign a reliable student to a teacher using pseudo-labeling and reliability evaluation so that the teacher sample propagates its prediction to the corresponding student sample. When the teacher sample is chosen from the source dataset, it minimizes the discrepancy between the source domain and the target domain. When the teacher sample is selected from the labeled target dataset, it reduces the discrepancy within the target domain. Experimental evaluation on standard benchmarks shows that our method effectively minimizes both the inter-domain and intra-domain discrepancies, thus achieving the state-of-the-art results.

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 (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014; He et al., 2016). 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 (Ganin et al. (2016); Saito et al. (2017); Long et al. (2018); Hoffman et al. (2018)). 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 (Saito et al. (2019)). As we are often able to obtain such additional labels easily on the target data, it renders the adaptation problem more practical and better situated in learning.

Empirical results (Saito et al. (2019)) 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 illustrate in Fig. 1a and Fig. 1b. Previous methods for SSDA (Saito et al. (2019); Kim & Kim (2020)) aims to address the issue using a proxy-based approach; they create a prototype representation for each class and reduce a distance between each prototype and its nearby unlabeled samples (Fig. 1c).

In this paper, we propose a new SSDA approach, dubbed *pair-based self-distillation* (PSD), that leverages rich data-to-data relations rather than proxy-to-data relations (Fig. 1d). Our method takes as a teacher a labeled sample on either source or target domain, and propagates its information to an unlabeled sample as a student in the form of self-knowledge distillation. 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

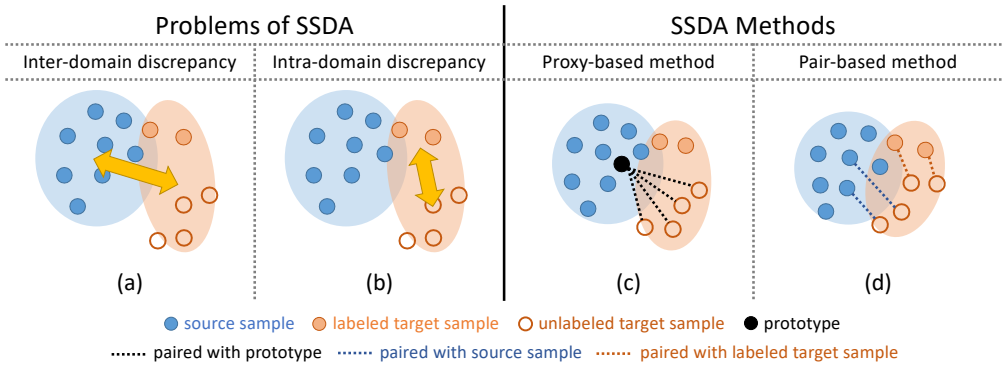


Figure 1: Two problems of SSDA and two different approaches. (a) Inter-domain discrepancy represents the discrepancy of sample distributions between a source domain and a target domain. (b) Intra-domain discrepancy indicates the discrepancy of sample distributions within the target domain. (c) Previous proxy-based methods give a prototype as a guidance to unlabeled target samples. (d) Our pair-based method uses a fine-grained sample-to-sample guidance to unlabeled target samples.

intra-domain discrepancy within the target. To generate reliable pairs of the teacher and the student, we employ pseudo-labeling (Lee, 2013), and present a new form of reliability evaluation on the pseudo-label motivated by Zhang et al. (2019). Compared to the previous proxy-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 our proposed method are summarized as follows.

- We propose *pair-based self-distillation* (PSD) that exploits rich sample-to-sample relations using self-distillation with the help of pseudo-labeling.
- We show that PSD effectively adapts a network to a target domain by alleviating both the inter- and intra-domain discrepancy issue.
- PSD sets a new state of the art on semi-supervised domain adaptation benchmarks and an unsupervised domain adaptation benchmark.

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 (Saito et al., 2019). Although SSDA has been considered in Ao et al. (2017); Donahue et al. (2013); Yao et al. (2015), 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 commonly focus on aligning the two domain distributions. Adversarial learning between a domain-classifier and a feature extractor is one of the representative UDA approaches (Ganin et al., 2016; Saito et al., 2017; Long et al., 2018; Lee et al., 2019; Xu et al., 2019). Learning with pseudo-labels (Lee, 2013) is another approach in UDA (Xie et al., 2018; Chang et al., 2019; Deng et al., 2019; Zhang et al., 2019). 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) (Saito et al., 2019) 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. Jiang et al. (2020) 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) (Kim & Kim, 2020) analyzes the target intra-domain discrepancy issue, and suggest to minimize the gap using Maximum Mean Discrepancy (MMD), perturbation loss, and the class prototypes. Among the

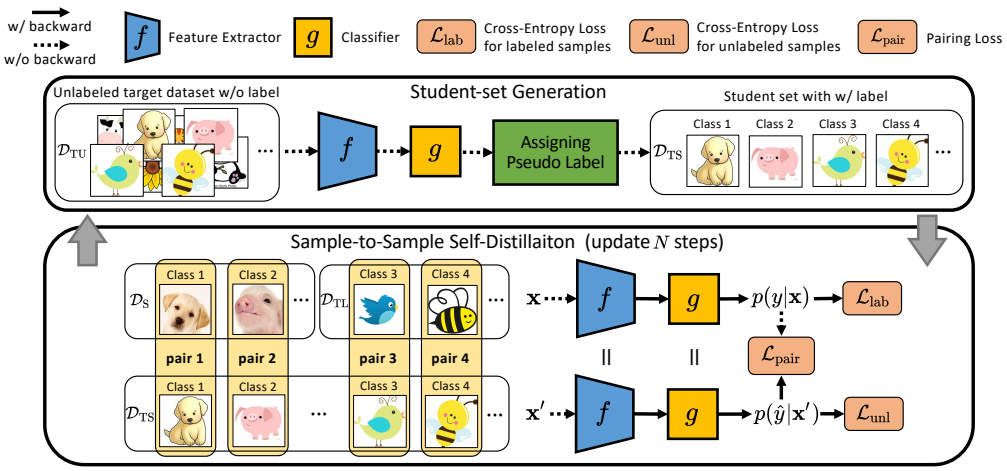


Figure 2: An overview of the Pair-based Self-Distillation (PSD) (Section 3.2). We use a feature extractor and a classifier trained in the pre-training stage (Section 3.1). PSD consists of a student-set generation procedure and a sample-to-sample distillation procedure. In the former, the model generates pseudo-labels of $\mathcal{D}_{\text{TU}}$ and completes $\mathcal{D}_{\text{TS}}$. In the latter, the model is trained with generated teacher-student pairs. In every N steps, pseudo-labels are updated using the latest model. We omit feature normalization, temperature scaling, and softmax operation for simplicity.

previous work, MME and APE adapt to the target domain, and use the class prototypes for SSDA. We tackle the issues of SSDA in a simple pair-based way by applying self-distillation different from previous work.

Knowledge distillation. The idea of knowledge distillation (KD) is to train a model *(student)* by transferring knowledge extracted from another model *(teacher)* that is more powerful than the student (Breiman & Shang, 1996; Buciluǎ et al., 2006). A series of study on KD has shown its attractive characteristics such as regularizing the student (Yuan et al., 2020), stabilizing training (Cheng et al., 2020), and preventing models to be overconfident (Yun et al., 2020). 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 (Hinton et al., 2015; Romero et al., 2014; Zagoruyko & Komodakis, 2017; Park et al., 2019). This type of work motivates GDSDA (Ao et al., 2017), 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 (Furlanello et al., 2018; Xie et al., 2020; Yun et al., 2020). Our design follows the second line of work. We present a loss to minimize Kullback–Leibler divergence of two predictions between a labeled sample and its corresponding pseudo-labeled sample using self-distillation. This learning objective naturally conforms to the goal of domain adaptation: adapting to a target domain by aligning semantically similar samples from two diverse domains.

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 (Saito et al., 2019; Kim & Kim, 2020; Li & Hospedales, 2020). 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 source and target domains share the same number of classes K. 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

Algorithm 1 Pair-based self-distillation.

Input: $\mathcal{D}_{\text{S}}, \mathcal{D}_{\text{TL}}, \mathcal{D}_{\text{TU}}$: Source domain, labeled target domain, and unlabeled target domain dataset
Input: $\boldsymbol{\theta}$ and $\mathbf{W}$: Pre-trained weights ▷ Section 3.1
Input: N: student-set generation interval
for $e \leftarrow 1$ to `max_steps` **do**
  if $e \bmod N$ is 0 **then** ▷ Student-set generation.
    Update student set $\mathcal{D}_{\text{TS}} = \{(\mathbf{x'}^{(l)}_{\text{TS}}, \hat{y}^{(l)}_{\text{TS}})\}_{l=1}^{N_{\text{TS}}}$ ▷ Equation 4 and equation 3.
  end if
  $(\mathbf{x}, y) \sim \mathcal{D}_{\text{S}} \cup \mathcal{D}_{\text{TL}}$, $(\mathbf{x}', \hat{y}) \sim \mathcal{D}_{\text{TS}}$ such that $\hat{y} = y$
  $p(y|\mathbf{x}) \leftarrow \text{softmax}(g(f(\mathbf{x}; \boldsymbol{\theta}); \mathbf{W})/T)$ ▷ Section 3.1
  $p(\hat{y}|\mathbf{x}') \leftarrow \text{softmax}(g(f(\mathbf{x}'; \boldsymbol{\theta}); \mathbf{W})/T)$ ▷ Section 3.1
  $\mathcal{L}_{\text{lab}} \leftarrow \text{CE}(p(y|\mathbf{x}), y)$ ▷ Equation 2
  $\mathcal{L}_{\text{unl}} \leftarrow \text{WCE}(p(\hat{y}|\mathbf{x}'), \hat{y})$ ▷ Equation 6
  $\mathcal{L}_{\text{pair}} \leftarrow \text{KL}(p(y|\mathbf{x}), p(\hat{y}|\mathbf{x}'))$ ▷ Sample-to-sample self-distillation. Equation 5.
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{lab}} + \mathcal{L}_{\text{unl}} + \lambda \mathcal{L}_{\text{pair}}$ ▷ Equation 7.
  update $\boldsymbol{\theta}$ and $\mathbf{W}$ with $\mathcal{L}$ using SGD
  if $e \bmod$ `val_freq` is 0 **then**
    validate and early-stop ▷ Section 3
  end if
end for

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 $g(\cdot; \mathbf{W})$ (Wang et al. (2018); Chen et al. (2019)). The distance-based classifier computes its output as the cosine similarity between the input feature $\mathbf{h}$ and each column $\mathbf{w}_k$ of $\mathbf{W}$:

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

where the final prediction $p(y|\mathbf{x})$ is obtained via softmax operation with 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 $\mathcal{D}_{\text{S}} \cup \mathcal{D}_{\text{TL}}$ via minimizing the cross-entropy loss:

$$\mathcal{L}_{\text{lab}} = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\text{S}}\cup\mathcal{D}_{\text{TL}}}[-\log p(y|\mathbf{x})]. \tag{2}$$

This pre-training improves the performance of PSD and also speeds up its convergence.

3.2 Pair-based Self-Distillation (PSD)

The *pair-based self-distillation* (PSD) 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 student-set generation and sample-to-sample 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 $\mathcal{D}_{\text{TS}}$ is used for student samples in self-distillation. At the sample-to-sample self-distillation step, we randomly produce teacher-student pairs with the same class label and perform self-distillation by miminizing the distance between their predictions. In paring, we take one sample from either $\mathcal{D}_{\text{S}}$ or $\mathcal{D}_{\text{TL}}$ (as a teacher) and the other from $\mathcal{D}_{\text{TS}}$ (as a student). It effectively reduces the inter-domain discrepancy using pairs between $\mathcal{D}_{\text{S}}$ and $\mathcal{D}_{\text{TS}}$, while suppressing the intra-domain discrepancy using pairs between $\mathcal{D}_{\text{TL}}$ and $\mathcal{D}_{\text{TS}}$. The overall procedure is summarized in Alg. 1 and also illustrated in Fig. 2. In the following, we explain the details of each step and describe the overall training objective.

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 stu-

dent samples $\{(\mathbf{x}'^{(l)}, \hat{y}^{(l)})\}_{l=1}^{N_{\mathrm{TU}}}$; we simply take a pseudo-label $\hat{y}$ of $\mathbf{x}'$ as the class index k of the maximum prediction value:

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

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 $\mathcal{D}_{\mathrm{TS}}$ for pairing. Let $\pi_j(\cdot)$ be a selection operator that selects j^{th} largest value. We construct a student set by reliability evaluation:

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

where δ is an average margin of unlabeled target logits and α is a hyper-parameter. The first condition is met when the absolute largest class probability score is high enough. The second condition is met when a margin between the largest and the second largest value of the logit is high enough (Zhang et al., 2019). In this way, the model assigns pseudo-labels to confident samples only so that the model can take reliable pairs.

Sample-to-sample self-distillation. After we obtain $\mathcal{D}_{\mathrm{TS}}$, we construct a pair of a labeled sample $(\mathbf{x}, y) \in \mathcal{D}_{\mathrm{S}} \cup \mathcal{D}_{\mathrm{TL}}$ and an pseudo-labeled sample $(\mathbf{x}', \hat{y}) \in \mathcal{D}_{\mathrm{TS}}$. We then set a labeled sample as a teacher sample, and set an unlabeled sample as a student sample. The pairing loss is calculated as

$$\mathcal{L}_{\mathrm{pair}} = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\mathrm{S}}\cup\mathcal{D}_{\mathrm{TL}},(\mathbf{x}',\hat{y})\in\mathcal{D}_{\mathrm{TS}}} \left[[\![\hat{y} = y]\!] D_{\mathrm{KL}}(p(y|\mathbf{x}) \parallel p(\hat{y}|\mathbf{x}'))\right], \tag{5}$$

where $[\![\cdot]\!]$ denotes Iverson brackets. If the teacher sample $\mathbf{x}$ is selected from $\mathcal{D}_{\mathrm{S}}$, the loss would minimize the inter-domain discrepancy, and if $\mathbf{x}$ is chosen from $\mathcal{D}_{\mathrm{TL}}$, the loss would reduce the intra-domain discrepancy of the target domain. This effect is validated in our experiments in Fig. 3.

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, *i.e.*, we use a weighted cross-entropy loss (WCE) for training student samples:

$$\mathcal{L}_{\mathrm{unl}} = \mathbb{E}_{(\mathbf{x}',\hat{y})\in\mathcal{D}_{\mathrm{TS}}}[-p(\hat{y}|\mathbf{x}') \log p(\hat{y}|\mathbf{x}')]. \tag{6}$$

Our total loss in training thus consists of three terms:

$$\mathcal{L} = \mathcal{L}_{\mathrm{lab}} + \mathcal{L}_{\mathrm{unl}} + \lambda \mathcal{L}_{\mathrm{pair}}, \tag{7}$$

where λ is a weighting hyper-parameter for the pairing loss. The model is updated by minimizing $\mathcal{L}$ for N iterations. $\mathcal{L}_{\mathrm{lab}}$ is the cross-entropy loss from equation 2.

We iterate alternating the student-set generation step and the sample-to-sample self-distillation step until the model converges on the validation set.

4 Experiments

We compare PSD with current state-of-the-art methods on two standard SSDA benchmarks. We include experiments of PSD on the UDA setup. We analyze the effectiveness our method both quantitatively and qualitatively. For more experimental results, please refer to the appendix.

4.1 Setup

Datasets. We evaluate our method using two benchmark datasets: DomainNet (Peng et al., 2019) and Office-Home (Venkateswara et al., 2017). DomainNet contains 6 domains of 345 classes each. Among them, we use 4 domains (Real, Clipart, Painting, and Sketch) and 126 classes. We choose seven source-to-target domain scenarios following the work of Saito et al. (2019). Office-Home consists of four domains (Real, Clipart, Product, and Art) of 65 classes. We conduct Office-Home experiments on all possible source-to-target domain scenarios.

Implementation details. We follow most of the implementation details of Saito et al. (2019) for a fair comparison. We select AlexNet (Krizhevsky et al., 2012) and ResNet-34 (He et al., 2016), both of which are pre-trained on ImageNet (Deng et al., 2009), for our base networks. In a mini-batch, the

Table 1: Classification accuracy of the DomainNet dataset (%) for one-shot and three-shot on 4 domains (R: Real, C: Clipart, P: Painting, S: Sketch). † denotes that we reproduced the baseline.

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
	APE †	46.3	51.5	45.5	48.5	40.6	47.6	36.2	42.2	37.1	42.2	30.3	37.8	54.2	58.5	41.5	46.7
	PSD (ours)	**51.6**	**56.2**	47.9	**51.2**	**48.0**	51.3	**39.2**	**43.5**	**40.6**	**46.5**	**37.4**	**39.8**	**59.5**	**65.1**	**46.3**	**50.5**
ResNet	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
	APE †	67.0	72.2	**70.3**	69.9	67.7	71.3	56.8	**63.8**	65.5	67.4	61.9	**65.0**	72.8	77.6	66.0	69.6
	PSD (ours)	**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**

Table 2: Classification accuracy of the Office-Home dataset (%) for one-shot on 4 domains (R: Real, C: Clipart, P: Product, A: Art).

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
	PSD (ours)	**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**

ratio of teacher samples (from $\mathcal{D}_{\mathrm{S}} \cup \mathcal{D}_{\mathrm{TL}}$) and student samples (from $\mathcal{D}_{\mathrm{TS}}$) is one to one. We choose the same number of source and labeled target data to construct teacher samples. Specifically, we use 48 teacher samples and 48 student samples in AlexNet. In ResNet-34, we use 64 teacher samples and 64 student samples like MME. We use the Stochastic Gradient Descent (SGD). The values of an initial learning rate, a momentum, and a weight decay are 0.01, 0.9, and 0.0005, respectively. In student-set generation step, we set α to 0.95, and δ is fixed to the average of all student samples' margin. We set the student-set generation interval N as 100. The hyper-parameter for the pairing loss λ is set individually on each base network according to the maximum validation accuracy. The details of searching λ are described in Section A.2. Experiments are implemented using PyTorch (Paszke et al. (2017)).

Baselines. We compare our method to competitive SSDA baselines: MME (Saito et al., 2019), and APE (Kim & Kim, 2020). Additionally, we bring S+T that simply minimizes the cross-entropy loss on the labeled dataset. DANN (Ganin et al., 2016), ADR Saito et al. (2017), and CDAN (Long et al., 2018)), which are the well-known methods in UDA, are also described as comparison. Further, we include the accuracy of ENT (Grandvalet & Bengio, 2005).

4.2 RESULTS

Comparison on DomainNet. Table 1 demonstrates the classification accuracy of our method and other baselines on DomainNet dataset. We conduct experiments on both one-shot and three-shot settings with AlexNet and ResNet. We reproduce APE based on their public codes. For a fair comparison, we select the best model, and tune hyper-parameters on the validation set for all experiments including reproduction and our method. In AlexNet one-shot and three-shot setting, our proposed method outperformed S+T with 6.3%p and 7.1%p respectively when we take an average of all adaptation scenarios. PSD also acquires improved accuracy than the previous state-of-the-art in most of domain scenarios. In ResNet experiments, our method achieves 12.3%p and 11.0%p higher mean accuracy in one-shot and three-shot setting respectively than S+T. Compared with APE, our method obtains notable accuracy improvement in one-shot and three-shot setting.

Table 3: Classification accuracy of the DomainNet dataset (%) in the UDA setting.

Net	Method	R to C	R to P	P to C	C to S	S to P	R to S	P to 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
	PSD (ours)	**49.3**	**49.2**	**42.7**	**38.1**	**41.7**	**38.0**	**54.1**	**44.7**

Table 4: Comprehensive ablation study of PSD on DomainNet dataset (%) for one-shot setting.

Net	Method	$\mathcal{L}_{\text{unl}}$	RSS	R to C	R to P	P to C	C to S	S to P	R to S	P to R	MEAN
AlexNet	S+T			43.3	42.4	40.1	33.6	35.7	29.1	55.8	40.0
	DANN			43.3	41.6	39.1	35.9	36.9	32.5	53.6	40.4
	MME			48.9	48.0	46.7	36.3	39.4	33.3	56.8	44.2
	APE			47.7	49.0	46.9	38.5	38.5	33.8	57.5	44.6
	PSD (ours)		✓	50.8	47.5	47.0	37.9	40.1	37.3	60.8	45.9
		✓		49.2	46.7	47.1	39.5	41.1	36.5	59.6	45.8
		✓	✓	51.6	47.9	48.0	39.2	40.6	37.4	59.5	46.3

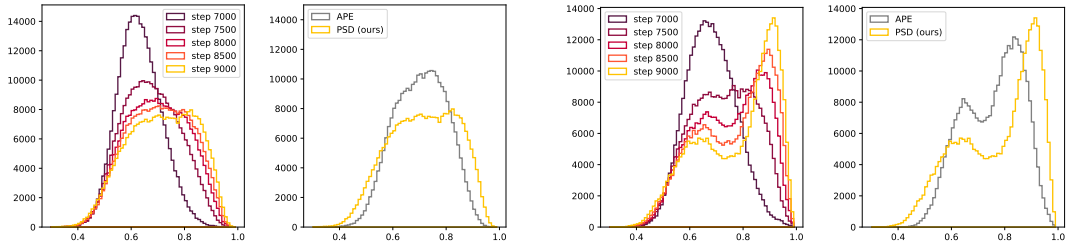
Comparison on Office-Home. Table 2 shows the results of our method and others on Office-Home. We reproduce MME and APE based on their public codes. We observe that PSD outperforms current state-or-the-art methods in most scenarios. Notably, PSD is effective where the domain gap between the source and target domain is substantial. In comparison to S+T, for example, PSD increases accuracy by 7.8%p on Real to Clipart, and 8.2%p on the other way around. Real and Clipart domains appear considerably distinctive to each other because samples in Real domain are photos from real world, on the other hand, the samples in Clipart domain are artificial illustrations. We believe that the pair-based loss explicitly drives two individual features from two visually diverse domains to be close. As a results, it achieves a clear performance gain in such challenging scenarios.

4.3 Analysis

Unsupervised domain adaptation. Table 3 summarizes performance improvement of PSD on the UDA setup on DomainNet. In this setup, we assume no labels are given from target domains, while other settings are not changed. PSD outperforms counterparts in most scenarios. It is impressive that PSD excels methods that have been proposed for UDA (Ganin et al., 2016; Saito et al., 2017; Long et al., 2018). Note that DomainNet dataset is designed to have a substantial domain gap between domains, each of which are categorized into 126 classes. We examine that PSD is powerful on such a challenging dataset even though target labels are not given at all.

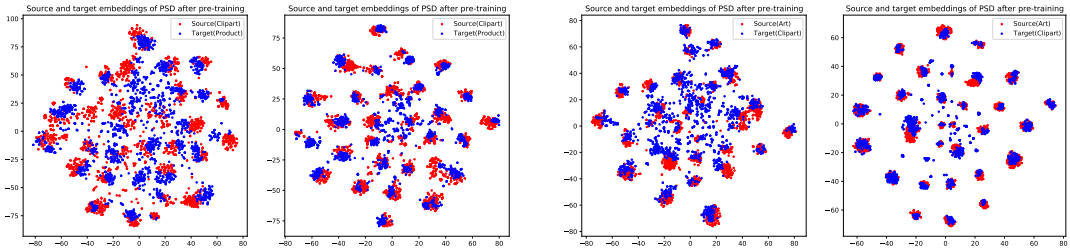
Ablation study. We conduct an ablation study on the weighted-cross entropy loss and the reliable student-set generation (RSS). The check mark of $\mathcal{L}_{\text{unl}}$ represents that the weighted cross-entropy loss for student samples (equation 6) is added to the overall loss. The check mark of RSS denotes that the reliability evaluation is utilized to generate $\mathcal{D}_{\text{TS}}$ (equation 4). If there is no check mark on the RSS column, we assign a class index of the maximum prediction value as a pseudo-label instead of equation 3. The bottom row is our complete setting. By comparing the first row and the third row of ours, the effectiveness of $\mathcal{L}_{\text{unl}}$ is verified in most of the domain scenarios. Also, by comparing the second row and the third row, it shows that our method benefits from RSS.

Inter-domain and intra-domain discrepancies. Fig. 3 visualizes that PSD progressively clusters instances of the same classes by overcoming inter- and intra-domain discrepancies. Fig. 3a plots cosine similarity between a source embedding and a target embedding from the same class for all classes. Fig. 3c plots cosine similarity between two target embeddings from the same class for all classes. Fig. 3b and Fig. 3d visualize the histograms from each final model of APE and PSD. For more implementation details, please refer to appendix. The cosine similarities gradually move toward 1.0 over iterations, which proves that a learner is guided 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 as shown



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

Figure 3: (a) Histograms of cosine similarities between a source and a target embedding (inter-domain similarity) over iterations (b) Inter-domain similarity histograms of APE and PSD. (c) Histograms of cosine similarities between target embeddings (intra-domain similarity) over iterations. (d) Intra-domain similarity histograms of APE and PSD.



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

Figure 4: t-SNE visualization on Office-Home. (a) Embedding space after pre-training stage on Clipart $\rightarrow$ Product. (b) Embedding space at the final model on Clipart $\rightarrow$ Product. (c) Embedding space after pre-training stage on Art $\rightarrow$ Clipart. (d) Embedding space at the final model on Art $\rightarrow$ Clipart.

in Fig. 3c. This is considered one limitation of leveraging pseudo-labels; wrong pairs misguide the learning process. We thus address the importance of pseudo-labeling for future work on domain adaptation.

Qualitative results. Fig. 4 visualizes how PSD clusters instances from two domains over iterations using t-SNE (Maaten & Hinton, 2008). The results respresent two main points: (1) target samples align toward source samples over iterations. (2) samples from the same class pull each other over iterations. Each point above stands for little inter-domain discrepancy and little intra-domain discrepancy on the final embedding manifold. We include more qualitative results in the appendix.

5 Conclusion

We have proposed a novel pair-based self-distillation (PSD) for semi-supervised domain adaptation. First, PSD assigns a pseudo-label to an unlabeled sample only if its prediction is reliable. Then, PSD makes a pair of two samples: one from pseudo-labeled samples and the other from labeled samples. PSD drives two predictions of the pair to be close. PSD outperforms on two semi-supervised domain adaptation benchmarks and one unsupervised domain adaptation benchmark. The experiments demonstrate that PSD effectively adapts to a target domain using a single architecture given an extremely few number of labeled target domain samples.

References

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*, pp. 1719–1725, 2017.

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

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, pp. 535–541, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933395. doi: 10.1145/1150402.1150464. URL https://doi.org/10.1145/1150402.1150464.

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*, pp. 7354–7362, 2019.

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.

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*, pp. 12925–12935, 2020.

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*, pp. 248–255. Ieee, 2009.

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*, pp. 9944–9953, 2019.

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*, pp. 668–675, 2013.

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

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.

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

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*, pp. 770–778, 2016.

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

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*, pp. 1989–1998, 2018.

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.

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.

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

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.

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*, pp. 91–100, 2019.

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

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

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

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

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.

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*, pp. 1406–1415, 2019.

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.

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

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*, pp. 8050–8058, 2019.

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

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*, pp. 5018–5027, 2017.

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*, pp. 5265–5274, 2018.

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*, pp. 10687–10698, 2020.

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

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.

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*, pp. 2142–2150, 2015.

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*, pp. 3903–3911, 2020.

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.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. URL https://arxiv.org/abs/1612.03928.

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*, pp. 435–445, 2019.

A APPENDIX

In this appendix, we provide supplementary results and details of our method.

A.1 IMPLEMENTATION DETAILS OF FIG. 3

We use ResNet34 architecture, Office-Home dataset on ont-shot setting. We use Product and Clipart domains for the source and target domains, respectively. For PSD, we start plotting after the pre-training stage (the 7,000$^{\text{th}}$ iteration) until the model converges for every 500 iterations. For APE, we plot the final model only.

A.2 THE HYPER-PARAMETER λ

Table 5: Hyper-parameter search on DomainNet using ResNet34. We choose Real to Clipart three-shot and Real to Sketch one-shot scenarios. We measure the test accuracy at the maximum validation accuracy.

Scenario	m						
	3	4	5	6	7	8	9
R to C 3-shot	75.3	75.9	73.9	75.5	75.9	75.7	75.2
R to S 1-shot	60.6	61.2	63.6	62.9	62.9	62.9	61.2
Average	68.0	68.5	68.7	69.2	**69.4**	69.3	68.2

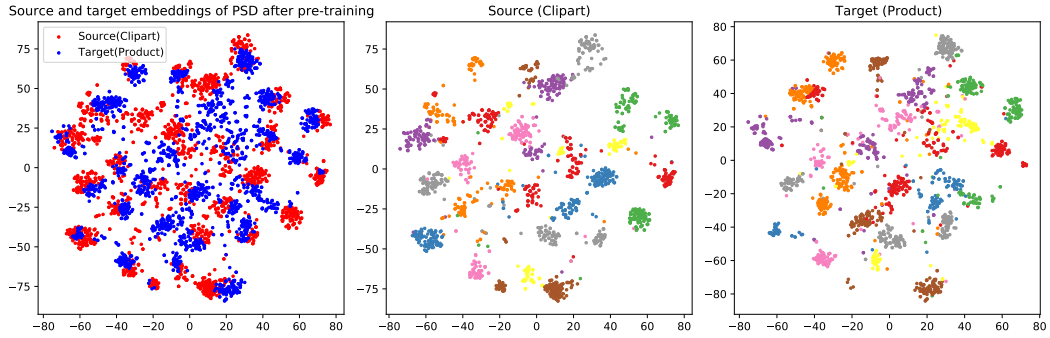
We use λ to balance $\mathcal{L}_{\text{pair}}$ in the overall loss. We set the hyper-parameter λ using a ramp-up function

$$\lambda = \frac{2}{1 + e^{-mz}} - 1, \tag{8}$$

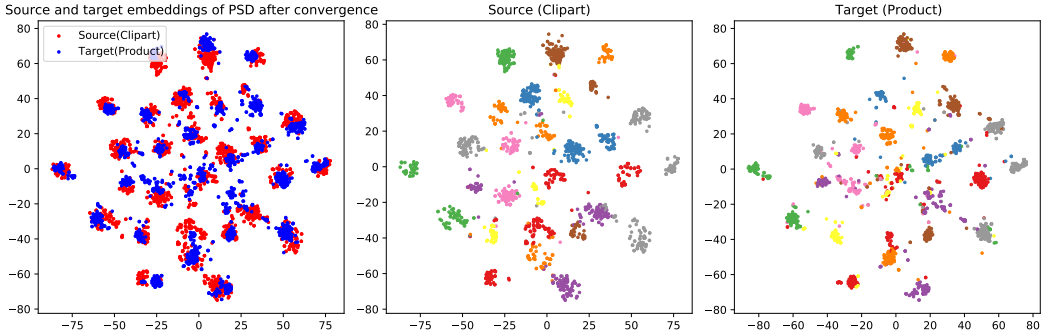
where z is a training progress calculated by the current step over maximum training step. We set the start step according to the accuracy of the pre-trained model. By changing m, we vary the slope of the ramp-up function to examine the effects of weighting $\mathcal{L}_{\text{pair}}$. We search the hyper-parameter by varying m from 3 to 9. We choose two domain scenarios, and select m at the best validation accuarcy on average of two scenarios. We set m to 6, 7, and 9 on AlexNet of Table 1, ResNet of Table 1, and Table 2 respectively. Table 5 shows the accuracy of our model when varying m.

A.3 EXTRA T-SNE VISUALIZATION

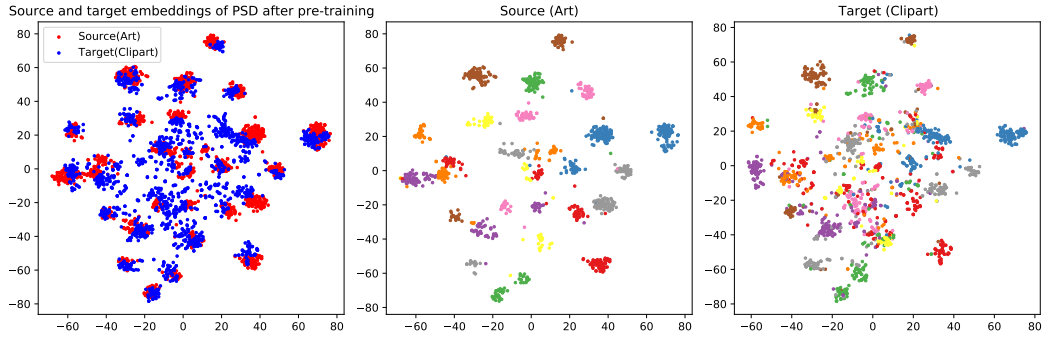
Fig. 5 visualizes how PSD clusters instances from two domains over iterations. We observe that the data-to-data self-distillation stage clearly enhances the embedding quality from the pre-training stage. Two main points of the results are: (1) target samples align toward source samples over iterations. (2) samples from the same class pull each other over iterations. We use ResNet34 architecture, Office-Home dataset on the one-shot setting. The final dimension is reduced from 512 to 2 dimensions by using t-SNE (Maaten & Hinton, 2008).



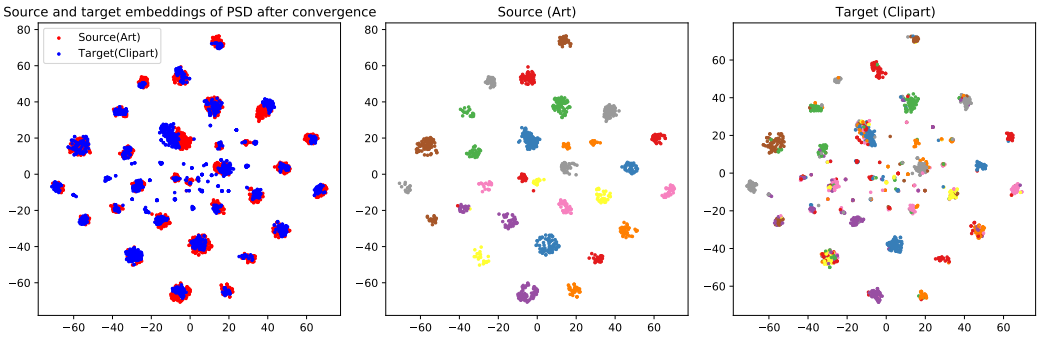
(a)



(b)



(c)



(d)

Figure 5: t-SNE visualization on Office-Home. The left column visualizes the source and target embeddings together. The middle and the right column visualizes the source and the target embeddings, respectively. The first 30 classes are visualized. (a) t-SNE visualization after pre-training stage on Clipart $\rightarrow$ Product. (b) t-SNE visualization at the final model on Clipart $\rightarrow$ Product. (c) t-SNE visualization after pre-training stage on Art $\rightarrow$ Clipart. (d) t-SNE visualization at the final model on Art $\rightarrow$ Clipart.