Prototypical Cross-domain Self-supervised Learning for Few-shot Unsupervised Domain Adaptation

Anonymous CVPR 2021 submission

Paper ID xxxx

Abstract

Unsupervised Domain Adaptation (UDA) transfers predictive models from a fully-labeled source domain to an unlabeled target domain. In some applications, however, it is expensive even to collect labels in the source domain, making most previous works impractical. To cope with this problem, recent work performed instance-wise cross-domain self-supervised learning, followed by an additional fine-tuning stage. However, the instance-wise self-supervised learning only learns and aligns low-level discriminative features. In this paper, we propose an end-to-end Prototypical Cross-domain Self-Supervised Learning (PCS) framework for Few-shot Unsupervised Domain Adaptation (FUDA). PCS not only performs cross-domain low-level feature alignment, but it also encodes and aligns semantic structures in the shared embedding space across domains. Our framework captures category-wise semantic structures of the data by in-domain prototypical contrastive learning; and performs feature alignment through cross-domain prototypical self-supervision. Compared with state-of-the-art methods, PCS improves the mean classification accuracy over different domain pairs on FUDA by 10.5%, 4.3%, 9.0%, and 13.2% on Office, Office-Home, VisDA-2017, and DomainNet, respectively.

1. Introduction

Deep Learning has achieved remarkable performance in various computer vision tasks, such as image classification [29, 31] and semantic segmentation [40, 72, 8]. Despite high accuracy, deep neural networks trained on specific datasets often fail to generalize to new domains owing to the *domain shift* problem [64, 14, 65]. Unsupervised domain adaptation (UDA) transfers predictive models from a fully-labeled source domain to an unlabeled target domain. Although it is challenging with no label information in the target domain, many UDA methods [65, 30, 41, 17] could achieve high accuracy on the target domain using the abundant explicit supervised information from source domain, together with the unlabeled target samples for domain alignment.

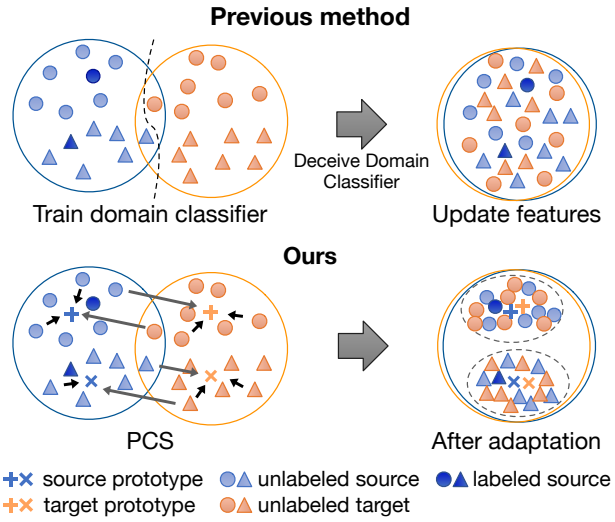


Figure 1: We address the task of few-shot unsupervised domain adaptation. Top: Existing domain-classifier based methods align source and target distributions but fail to extract discriminative features due to lack of labeled data. Bottom: Our method estimates prototype for each class and performs cross-domain self-supervised learning to extract domain-aligned discriminative features.

In some real-world applications, however, providing large-scale annotations even in the source domain is often challenging due to the high cost and difficulty of annotation. Taking medical imaging for instance, each image of the Diabetic Retinopathy dataset [27] is annotated by a panel of 7 or 8 U.S. board-certified ophthalmologists, with a total group of 54 doctors. Thus practically it is too stringent to assume the availability of source data with abundant labels.

In this paper, to cope with the labeling costs of the source domain, we instead consider a few-shot unsupervised domain adaptation (FUDA) setting, where only an extremely small fraction of source samples are labeled, while

all the rest source and target samples remain unlabeled. Most state-of-the-art UDA methods align source and target features by minimizing some form of distribution distances [41, 42, 61, 17], and learn the discriminative representation by minimizing the supervision loss on fully-labeled source domain data. In FUDA, however, since we have a very limited number of labeled source samples, it is much harder to learn discriminative features in the source domain, not to mention in the target domain.

Several recent papers [28, 9, 26, 49, 68] on self-supervised learning (SSL) present promising representation learning results on images from a single domain and [36] further extended to perform SSL across two domains for better domain adaptation performance. Despite the improved performance, the instance-based method in [36] has some fundamental weaknesses. First, the semantic structure of the data is not encoded by the learned structure. This is because the in-domain self-supervision in [36] treats two instances as negative pairs as long as they are from different samples, regardless of the semantic similarity. Consequently, many instances sharing the same semantic are undesirably pushed apart in the feature space. Second, the cross-domain instance-to-instance matching in [36] is very sensitive to abnormal samples. Imagine a case where the embeddings of source and target samples are far apart (*i.e.* the domain gap is large) and one abnormal source sample is mapped closer to all target samples than any other source sample. Then the method in [36] would match all target samples to the same source sample (cf. Figure 3). For a given sample, the matched sample in the other domain may change drastically during the training procedure, making the optimization harder to converge. Third, the two-stage pipeline (*i.e.* SSL followed by domain adaptation) is complicated and experiments show that the optimal DA methods for different datasets are different. As a result, the training is rather cumbersome and it is unclear how to choose the optimal DA method in the second stage for different datasets.

In this paper, we propose *Prototypical Cross-domain Self-supervised learning*, a novel single-stage framework for FUDA that unifies representation learning and domain alignment with few-shot labeled source images. PCS contains three major components to learn both discriminative and domain-invariant features. First, PCS performs in-domain prototypical self-supervision to implicitly encode the semantic structure of data into the embedding space. This is motivated by [38], but we further leverage the known semantic information of the task and learn better semantic structure in each domain. Second, PCS performs cross-domain instance-to-prototype matching to transfer knowledge from source to the target in a more robust manner. Instead of instance-to-instance matching, matching a sample to a prototype (*i.e.* representative embedding for a group of instances that are semantically similar) is more robust to the abnormal instances in the other domain and makes the optimization converge faster and more smoothly. In order to further mitigate the effect of cross-domain mismatching, we perform entropy maximization to obtain a more diversified output. We show that together with entropy minimization, this is equivalent to maximizing the mutual information (MI) between input image and the network prediction. Third, PCS unifies prototype learning with cosine classifier and adaptively transfers from source prototypes to target prototypes for better performance on the target domain.

To summarize, our contributions are three-fold:

- We propose a novel Prototypical Cross-domain Self-supervised learning framework (PCS) for few-shot unsupervised Domain Adaptation.
- We propose to leverage prototypes to perform better semantic structure learning, discriminative feature learning, and cross-domain alignment in a unified, unsupervised and adaptive manner.
- While it is hard to choose the optimal domain adaptation method in the complex two-stage framework [36], PCS can be easily trained in an end-to-end matter, and outperforms all state-of-the-art methods by a large margin across multiple benchmark datasets.

2. Related Work

Domain Adaptation. Unsupervised Domain Adaptation (UDA) [23] addresses the problem of transferring knowledge from a fully-labeled source domain to an unlabeled target domain. Most UDA methods have focused on feature distribution alignment. Discrepancy-based methods explicitly compute the Maximum Mean Discrepancy (MMD) [25] between the source and the target to align the two domains [41, 66, 43]. Long *et al*. [44] proposed to align the joint distributions using the Joint MMD criterion. Sun *et al*. [61] and Zhuo *et al*. [74] further proposed to align second-order statistics of source and target features. With the development of Generative Adversarial Networks [22], additional papers proposed to perform domain alignment in the feature space with adversarial learning [16, 65, 30, 69, 42, 59]. Recently, image translation methods, *e.g.* [73, 39] have been adopted to further improve domain adaptation by performing pixel-level alignment [30, 5, 55, 47, 70, 58, 60]. Instead of explicit feature alignment, Saito *et al*. [57] proposed minimax entropy for adaptation. While these methods have full supervision on the source domain, similar to [36], we focus on a new adaptation setting with few source labels.

Self-supervised Learning. Self-supervised learning (SSL) is a subset of unsupervised learning methods where supervision is automatically generated from the

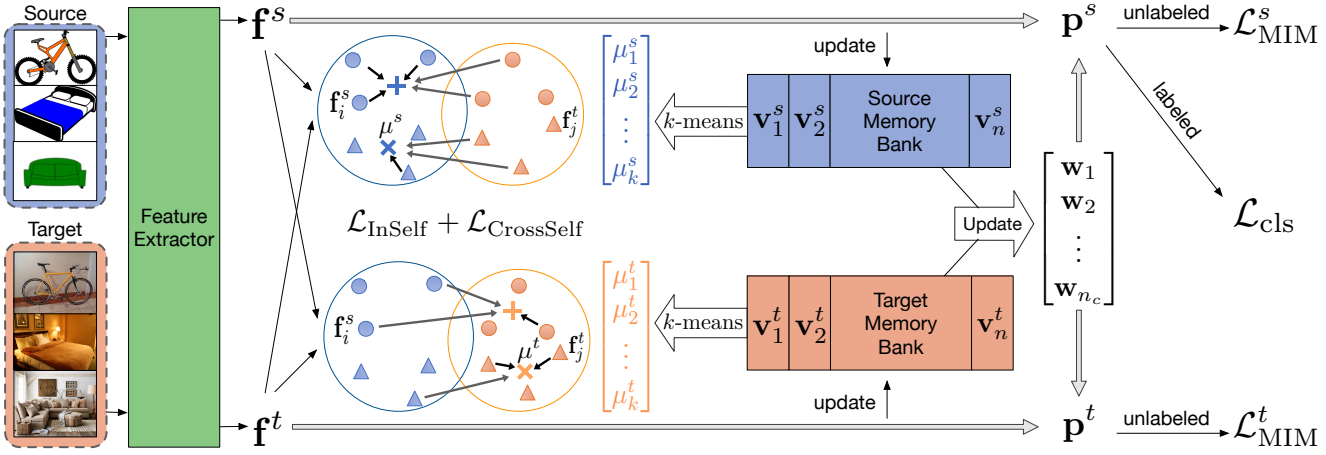


Figure 2: An overview of the PCS framework. In-domain and cross-domain self-supervision are performed between normalized feature vectors $\mathbf{f}$ and prototypes μ computed by clustering vectors $\mathbf{v}$ in memory banks. Features with confident predictions ($\mathbf{p}$) are used to adaptively update classifier vectors $\mathbf{w}$. MI maximization and classification loss are further used to extract discriminative features.

data [33, 12, 71, 48, 21]. One of the most common strategies for SSL is handcrafting auxiliary pretext tasks predicting future, missing or contextual information. In particular, image colorization [71, 37], patch location prediction [12, 13], image jigsaw puzzle [48], image inpainting [51] and geometric transformations [21, 15] have been shown to be helpful. Currently, contrastive learning [3, 28, 49, 63, 46] has achieved state-of-the-art performance on representation learning [26, 9, 11, 10]. Most contrastive methods are instance-wise, aiming to learn an embedding space where samples from the same instance are pulled closer and samples from different instances are pushed apart [68, 9]. Recently, contrastive learning based on prototypes has shown promising results in representation learning [38, 2, 7, 18].

Self-supervised Learning for Domain Adaptation Self-supervision-based methods incorporate SSL losses into the original task network [19, 20]. Reconstruction was first utilized as self-supervised task in some early works [19, 20], in which source and target data share the same encoder to extract domain-invariant features. To further consider individual characteristics of each domain, Bousmalis *et al*. [5] explicitly extracts image representations into two spaces, one private for each domain and one shared across domains to capture both domain-specific and shared properties. In [6], solving jigsaw puzzle [48] was leveraged as a self-supervision task to solve domain adaptation and generalization. Sun *et al*. [62] further proposed to perform domain adaptation by jointly learning multiple self-supervision tasks. The same feature encoder is shared by both source and target images, and the extracted features are then fed into different self-supervision task heads: flip prediction, image rotation prediction [21], and patch location prediction [12]. Recently, based on instance discrimination [68], Kim *et al*. [36] proposed a cross-domain SSL approach for adaptation with few source labels. SSL has also been incorporated in point-cloud adaptation [1], in which region reconstruction is introduced as a new pretext task.

3. Approach

In few-shot unsupervised domain adaptation, we are given a very limited number of labeled source images $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{N_s}$, as well as unlabeled source images $\mathcal{D}_{su} = \{(\mathbf{x}_i^{su})\}_{i=1}^{N_{su}}$. In the target domain, we are only given unlabeled target images $\mathcal{D}_{tu} = \{(\mathbf{x}_i^{tu})\}_{i=1}^{N_{tu}}$. The goal is to train a model on $\mathcal{D}_s, \mathcal{D}_{su}$, and $\mathcal{D}_{tu}$; and evaluate on $\mathcal{D}_{tu}$.

The base model consists of a feature encoder F, a ℓ_2 normalization layer, which outputs a normalized feature vector $\mathbf{f} \in \mathbb{R}^d$ and a cosine similarity-based classifier C.

3.1. In-domain Prototypical Contrastive Learning

We learn a shared feature encoder F that extracts discriminative features in both domains. Instance Discrimination [68] is employed in [36] to learn discriminative features. As an instance-wise contrastive learning method, it results in an embedding space where all instances are well separated. Despite promising results, instance discrimination has a fundamental weakness: the semantic structure of the data is not encoded by the learned representations. This is because two instances are treated as negative pairs as long as they are from different samples, regardless of their semantics. For a single domain, ProtoNCE [38] is proposed to learn semantic structure of the data by performing iterative clustering and representation learning. The goal is to drive features within the same cluster to become more aggregated and features in different clusters further apart.

However, naively applying ProtoNCE to $\mathcal{D}_s \cup \mathcal{D}_{su} \cup \mathcal{D}_{tu}$ in our domain adaptation setting would cause potential problems. Primarily due to the domain shift, images of different classes from different domains can be incorrectly aggregated into the same cluster, and images of the same class from different domains can be mapped into clusters that are far apart. In order to mitigate these problems, we propose to perform prototypical contrastive learning separately in $\mathcal{D}_s \cup \mathcal{D}_{su}$ and in $\mathcal{D}_{tu}$. This aims to prevent the incorrect clustering of images across domains and indiscriminative feature learning.

Specifically, we maintain two memory banks $\boldsymbol{V^s}$ and $\boldsymbol{V^t}$ for source and target respectively:

$$\boldsymbol{V^s} = [\mathbf{v}_1^s, \cdots, \mathbf{v}_{(N_s+N_{su})}^s], \ \boldsymbol{V^t} = [\mathbf{v}_1^t, \cdots, \mathbf{v}_{N_{tu}}^t], \quad (1)$$

where $\mathbf{v}_i$ is the stored feature vector of $\mathbf{x}_i$, initialized with $\mathbf{f}_i$ and updated with a momentum m in each batch:

$$\mathbf{v}_i \leftarrow m\mathbf{v}_i + (1-m)\mathbf{f}_i. \quad (2)$$

In order for in-domain prototypical contrastive learning, k-means clustering is performed on $\boldsymbol{V^s}$ and $\boldsymbol{V^t}$ to get source clusters $\boldsymbol{C^s} = \{C_1^{(s)}, C_2^{(s)}, \ldots, C_k^{(s)}\}$ and similarly $\boldsymbol{C^t}$ with normalized source prototypes $\{\mu_j^s\}_{j=1}^k$ and normalized target prototypes $\{\mu_j^t\}_{j=1}^k$. Specifically, $\mu_j^s = \frac{\mathbf{u}_j^s}{\|\mathbf{u}_j^s\|}$, where $\mathbf{u}_j^s = \frac{1}{|C_j^{(s)}|} \sum_{\mathbf{v}_i^s \in C_j^{(s)}} \mathbf{v}_i^s$. We explain only on the source domain for succinct notation, all operations are performed on target similarly.

During training, with the feature encoder F, we compute a feature vector $\mathbf{f}_i^s = F(\mathbf{x}_i^s)$. To perform in-domain prototypical contrastive learning, we compute the similarity distribution vector between $\mathbf{f}_i^s$ and $\{\mu_j^s\}_{j=1}^k$ as $P_i^s = [P_{i,1}^s, P_{i,2}^s, \ldots, P_{i,k}^s]$, with

$$P_{i,j}^s = \frac{\exp(\mu_j^s \cdot \mathbf{f}_i^s/\phi)}{\sum_{r=1}^k \exp(\mu_r^s \cdot \mathbf{f}_i^s/\phi)}, \quad (3)$$

where ϕ is a temperature value determining the level of concentration. Then the in-domain prototypical contrastive loss can be written as:

$$\mathcal{L}_{\text{PC}} = \sum_{i=1}^{N_s+N_{su}} \mathcal{L}_{CE}(P_i^s, c_s(i)) + \sum_{i=1}^{N_{tu}} \mathcal{L}_{CE}(P_i^t, c_t(i)) \quad (4)$$

where $c_s(\cdot)$ and $c_t(\cdot)$ return the cluster index of the instance. Due to the randomness in clustering, we perform k-means on the samples M times with different number of clusters $\{k_m\}_{m=1}^M$. Moreover, in the FUDA setting, since the number of classes n_c is known, we set $k_m = n_c$ for most m.

To further alleviate the problem of instance discrimination and promote more accurate clustering, we propose to apply InfoNCE loss only on the labeled set $\mathcal{D}_s$:

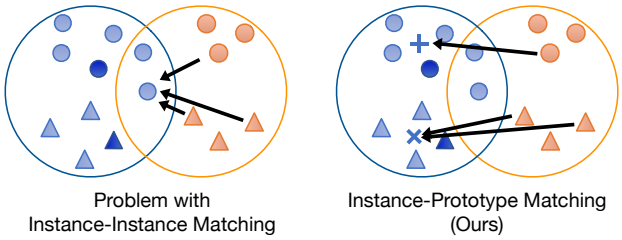


Figure 3: Comparison of cross-domain instance-instance (I-I) matching [36] (left) and our cross-domain instance-prototype (I-P) matching (right). Left: I-I incorrectly matches all orange samples to the same blue sample. Right: I-P robustly matches samples to the correct prototypes.

$$\mathcal{L}_{\text{Info}} = \sum_{i=1}^{N_s} -\log \frac{\exp(\mathbf{f}_i^s \cdot \hat{\mathbf{f}}_i^s/\tau)}{\sum_{j=0}^r \exp(\mathbf{f}_i^s \cdot \hat{\mathbf{f}}_j^s/\tau)}, \quad (5)$$

where $\hat{\mathbf{f}}_i^s$ is a positive embedding (augmented from the same image or from other images of same class) for instance i, and $\hat{\mathbf{f}}_j^s$ includes one positive embedding and r negative embeddings from other instances whose classes are different from i. The overall loss for in-domain self-supervision is

$$\mathcal{L}_{\text{InSelf}} = \mathcal{L}_{\text{Info}} + \lambda \cdot \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}_{\text{PC}}^{(m)} \quad (6)$$

3.2. Cross-domain Instance-Prototype SSL

To explicitly enforce learning domain-aligned and more discriminative features in source and target domains, we perform cross-domain instance-prototype SSL.

Many previous works focus on domain alignment via discrepancy minimization or adversarial learning. However, these methods provide inferior performance or have unstable training. Moreover, most of them focus on distribution matching, without considering semantic similarity matching across domains. Instance-instance matching [36] is proposed to match an instance i to another instance j in the other domain with the most similar representation. However, due to the domain gap, instances can be easily mapped to instances of different classes in the other domain. In some cases, if an outlier in one domain is extremely close to the other domain, it will be matched to all the instances in the other domain, as illustrated in Figure 3.

Instead, our method discovers positive matching as well as negative matchings between instance and cluster prototypes in different domains. To find a matching for an instance i, we perform entropy minimization on the similarity distribution vector between its representation, *e.g.* $\mathbf{f}_i^s$ and the centroids of the other domain, *e.g.* $\{\mu_j^t\}_{j=1}^k$.

Specifically, given feature vector $\mathbf{f}_i^s$ in the source domain, and centroids $\{\mu_j^t\}_{j=1}^k$ in the target domain, we first compute the similarity distribution vector $P_i^{s\to t} =$

$[P_{i,1}^{s\to t}, \dots, P_{i,k}^{s\to t}]$ in which

$$P_{i,j}^{s\to t} = \frac{\exp(\mu_j^t \cdot \mathbf{f}_i^s/\tau)}{\sum_{r=1}^{k} \exp(\mu_r^t \cdot \mathbf{f}_i^s/\tau)}. \quad (7)$$

Then we minimize the entropy of $P_i^{s\to t}$, which is:

$$H(P_i^{s\to t}) = -\sum_{j=1}^{k} P_{i,j}^{s\to t} \log P_{i,j}^{s\to t}. \quad (8)$$

Similarly, we can compute $H(P_i^{t\to s})$, and the final loss for cross-domain instance-prototype SSL is:

$$\mathcal{L}_{\text{CrossSelf}} = \sum_{i=1}^{N_s+N_{su}} H(P_i^{s\to t}) + \sum_{i=1}^{N_{tu}} H(P_i^{t\to s}) \quad (9)$$

3.3. Adaptive Prototypical Classifier Learning

The goal of this section is to learn a better domain-aligned, discriminative feature encoder F and more importantly, a cosine classifier C that could achieve high accuracy on the target domain.

The cosine classifier C consists of weight vectors $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{n_c}]$, where n_c denotes the total number of classes, and a temperature T. The output of C, $\frac{1}{T}\mathbf{W}^{\mathrm{T}}\mathbf{f}$ is fed into a softmax layer σ to obtain the final probabilistic output $\mathbf{p}(\mathbf{x}) = \sigma(\frac{1}{T}\mathbf{W}^{\mathrm{T}}\mathbf{f})$. With the availability of the labeled set $\mathcal{D}_s$, it is straightforward to train F and C with a standard cross-entropy loss for classification:

$$\mathcal{L}_{\text{cls}} = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_s} \mathcal{L}_{CE}(\mathbf{p}(\mathbf{x}), y) \quad (10)$$

However, since $\mathcal{D}_s$ is quite small under FUDA setting, only training with $\mathcal{L}_{\text{cls}}$ is hard to get a C with high performance on the target.

Adaptive Prototype-Classifier Update (APCU) Note that for C to classify samples correctly, the direction of a weight vector $\mathbf{w}_i$ needs to be representative of features of the corresponding class i. This indicates that the meaning of $\mathbf{w}_i$ coincide with the ideal cluster prototype of class i. We propose to use an estimate of the ideal cluster prototypes to update $\mathbf{W}$. Yet the computed $\{\mu_j^s\}$ and $\{\mu_j^t\}$ cannot be naively used for this purpose, not only because the correspondence between $\{\mathbf{w}_i\}$ and $\{\mu_j\}$ is unknown, but also the k-means result may contain very impure clusters, leading to non-representative prototypes.

We use the few-shot labeled data as well as samples with high-confident predictions to estimate the prototype for each class. Formally, we define $\mathcal{D}_s^{(i)} = \{\mathbf{x}|(\mathbf{x}, y) \in \mathcal{D}_s, y = i\}$ and denote by $\mathcal{D}_{su}^{(i)}$ and $\mathcal{D}_{tu}^{(i)}$ the set of samples with high-confident label i in source and target, respectively. With $\mathbf{p}(\mathbf{x}) = [\mathbf{p}(\mathbf{x})_1, \dots, \mathbf{p}(\mathbf{x})_{n_c}]$, $\mathcal{D}_{su}^{(i)} = \{\mathbf{x}|\mathbf{x} \in \mathcal{D}_{su}, \mathbf{p}(\mathbf{x})_i > t\}$, where t is a confidence threshold; and similarly for $\mathcal{D}_{tu}^{(i)}$. Then the estimate of $\mathbf{w}_i$ from source and target domain can be computed as:

$$\hat{\mathbf{w}}_i^s = \frac{1}{|\mathcal{D}_{s^+}^{(i)}|} \sum_{\mathbf{x}\in\mathcal{D}_{s^+}^{(i)}} m(\mathbf{x}); \hat{\mathbf{w}}_i^t = \frac{1}{|\mathcal{D}_{tu}^{(i)}|} \sum_{\mathbf{x}\in\mathcal{D}_{tu}^{(i)}} m(\mathbf{x}) \quad (11)$$

where $\mathcal{D}_{s^+}^{(i)} = \mathcal{D}_s^{(i)} \cup \mathcal{D}_{su}^{(i)}$ and $m(\mathbf{x})$ returns the representation in memory bank corresponding to $\mathbf{x}$.

With only few labeled samples in source, it is hard to learn a representative prototype shared across domains. Instead of directly employing a global prototype for a class i, we further propose to update $\mathbf{w}_i$ in an domain adaptive manner, with $\hat{\mathbf{w}}_i^s$ during early training stage and with $\hat{\mathbf{w}}_i^t$ at later stage. This is because that $\hat{\mathbf{w}}_i^s$ is more robust in early training stage due to the few labeled source samples, while $\hat{\mathbf{w}}_i^t$ would be more representative later for target domain to get better adaptation performance. Specifically, we use $|\mathcal{D}_{tu}^{(i)}|$ to determine whether $\hat{\mathbf{w}}_i^t$ is robust to use:

$$\mathbf{w}_i = \begin{cases} unit(\hat{\mathbf{w}}_i^s) & \text{if } |\mathcal{D}_{tu}^{(i)}| < t_w \\ unit(\hat{\mathbf{w}}_i^t) & \text{otherwise} \end{cases} \quad (12)$$

where $unit(\cdot)$ normalizes the input vector and t_w is a threshold hyper-parameter.

Mutual Information Maximization In order for the above unified prototype-classifier learning paradigm to work well, the network is desired to have enough confident predictions, *e.g.* $|\mathcal{D}^{(i)}| > t_w$, for all classes to get robust $\hat{\mathbf{w}}_i^s$ and $\hat{\mathbf{w}}_i^t$ for $i = 1, \dots, n_c$. First, to promote the network to have diversified outputs over the dataset, we maximize the entropy of expected network prediction $\mathcal{H}(\mathbb{E}_{\mathbf{x}\in\mathcal{D}}[p(y|\mathbf{x};\theta)])$, where θ denotes learnable parameters in both F and C, and $\mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_{su} \cup \mathcal{D}_{tu}$. Second, in order to get high-confident prediction for each sample, we leverage entropy minimization on the network output which has shown efficacy in label-scarce scenarios [24, 4]. These two desired behaviors turn out to be equivalent to maximizing the mutual information between input and output:

$$\mathcal{I}(y;\mathbf{x}) = \mathcal{H}(\mathbf{p}_0) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))], \quad (13)$$

where the prior distribution $\mathbf{p}_0$ is given by $\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]$, and the detailed derivation is presented in the supplementary material. We can get the objective as:

$$\mathcal{L}_{\text{MIM}} = -\mathcal{I}(y;\mathbf{x}) \quad (14)$$

3.4. PCS Learning for FUDA

The PCS learning framework performs in-domain prototypical contrastive learning, cross-domain instance-prototype self-supervised learning, and unified adaptive

Table 1: Adaptation accuracy (%) comparison on 1-shot and 3-shots per class on the Office dataset.

Method	Office: Target Acc. on 1-shot / 3-shots						
	A→D	A→W	D→A	D→W	W→A	W→D	Avg
SO	27.5 / 49.2	28.7 / 46.3	40.9 / 55.3	65.2 / 85.5	41.1 / 53.8	62.0 / 86.1	44.2 / 62.7
MME [57]	21.5 / 51.0	12.2 / 54.6	23.1 / 60.2	60.9 / 89.7	14.0 / 52.3	62.4 / 91.4	32.3 / 66.5
CDAN [42]	11.2 / 43.7	6.2 / 50.1	9.1 / 65.1	54.8 / 91.6	10.4 / 57.0	41.6 / 89.8	22.2 / 66.2
CAN [35]	25.3 / 48.6	26.4 / 45.3	23.9 / 41.2	69.4 / 78.2	21.2 / 39.3	67.3 / 82.3	38.9 / 55.8
MDDIA [32]	45.0 / 62.9	54.5 / 65.4	55.6 / 67.9	84.4 / 93.3	53.4 / 70.3	79.5 / 93.2	62.1 / 75.5
CDS [36]	33.3 / 57.0	35.2 / 58.6	52.0 / 67.6	59.0 / 86.0	46.5 / 65.7	57.4 / 81.3	47.2 / 69.3
DANN + ENT [17]	32.5 / 57.6	37.2 / 54.1	36.9 / 54.1	70.1 / 87.4	43.0 / 51.4	58.8 / 89.4	46.4 / 65.7
MME + ENT	37.6 / 69.5	42.5 / 68.3	48.6 / 66.7	73.5 / 89.8	47.2 / 63.2	62.4 / 95.4	52.0 / 74.1
CDAN + ENT	31.5 / 68.3	26.4 / 71.8	39.1 / 57.3	70.4 / 88.2	37.5 / 61.5	61.9 / 93.8	44.5 / 73.5
CDS + ENT	40.4 / 61.2	44.7 / 66.7	66.4 / 73.1	71.6 / 90.6	58.6 / 71.8	69.3 / 86.1	58.5 / 74.9
CDS + MME + ENT	39.4 / 61.6	43.6 / 66.3	66.0 / 74.5	75.7 / 92.1	53.1 / 73.0	70.9 / 90.6	58.5 / 76.3
CDS / MME + ENT†	55.4 / 75.7	57.2 / 77.2	62.8 / 69.7	84.9 / 92.1	62.6 / 69.9	77.7 / 95.4	65.3 / 80.0
CDS / CDAN + ENT†	53.8 / 78.1	65.6 / 79.8	59.5 / 70.7	83.0 / 93.2	57.4 / 64.5	77.1 / 97.4	66.1 / 80.6
PCS (Ours)	**60.2 / 78.2**	**69.8 / 82.9**	**76.1 / 76.4**	**90.6 / 94.1**	**71.2 / 76.3**	**91.8** / 96.0	**76.6 / 84.0**
Improvement	+4.8 / +0.1	+4.2 / +3.1	+9.7 / +1.9	+5.7 / +0.9	+8.6 / +6.4	+14.1 / -1.4	+10.5 / +3.4

† Two-stage training results reported in [36].

Table 2: Performance contribution of each part in PCS framework on Office.

Method	Office: Target Acc. on 1-shot / 3-shots						
	A→D	A→W	D→A	D→W	W→A	W→D	Avg
$\mathcal{L}_{\text{cls}}$	27.5 / 49.2	28.7 / 46.3	40.9 / 55.3	65.2 / 85.5	41.1 / 53.8	62.0 / 86.1	44.2 / 62.7
$+\mathcal{L}_{\text{InSelf}}$	39.0 / 55.6	38.6 / 55.1	47.2 / 68.5	71.7 / 89.4	50.9 / 68.4	65.1 / 90.6	52.1 / 71.3
$+\mathcal{L}_{\text{CrossSelf}}$	47.2 / 71.1	52.7 / 70.6	59.0 / 75.5	76.4 / 90.3	58.5 / 74.1	66.9 / 91.8	60.1 / 78.9
$+\mathcal{L}_{\text{MIM}}$	52.8 / 73.5	57.5 / 71.2	67.2 / 76.3	78.9 / 91.4	64.2 / 74.3	68.7 / 92.2	64.9 / 79.8
+APCU (PCS)	**60.2 / 78.2**	**69.8 / 82.9**	**76.1 / 76.4**	**90.6 / 94.1**	**71.2 / 76.3**	**91.8 / 96.0**	**76.6 / 84.0**

prototype-classifier learning. Together with APCU in Eq. 12, the overall learning objective is:

$$\mathcal{L}_{PCS} = \mathcal{L}_{\text{cls}} + \lambda_{\text{in}} \cdot \mathcal{L}_{\text{InSelf}} + \lambda_{\text{cross}} \cdot \mathcal{L}_{\text{CrossSelf}} + \lambda_{\text{mim}} \cdot \mathcal{L}_{\text{MIM}} \quad (15)$$

4. Experiments

4.1. Experimental Setting

Datasets. We evaluate our approach on four public datasets and choose labeled images in source domain based on previous work [36]. **Office** [56] is a real-world dataset for domain adaptation tasks. It contains 3 domains (Amazon, DSLR, Webcam) with 31 classes. Experiments are conducted with 1-shot and 3-shots source labels per class in this dataset. **Office-Home** [67] is a more difficult dataset than Office, which consists of 4 domains (Art, Clipart, Product, Real) in 65 classes. Following [36], we look into the settings with 3% and 6% labeled source images per class, which means each class has 2 to 4 labeled images on average. **VisDA-2017** [53] is a challenging simulation-to-real dataset containing over 280K images across 12 classes. We validate our model on settings with 0.1% and 1% labeled source images per class as suggested in [36]. **DomainNet** [52] is a large-scale domain adaptation benchmark. Since some domains and classes are noisy, we follow [57] and use a subset containing four domains (Clipart, Real, Painting, Sketch) with 126 classes. We show results on settings with 1-shot and 3-shots source labels on this dataset.

Implementation Details. We use ResNet-101 [29] (for DomainNet) and ResNet-50 (for other datasets) pre-trained on ImageNet [54] as our backbones. To enable a fair comparison with [36], we replaced the last FC layer with a 512-D randomly initialized linear layer. L2-normalizing are performed on the output features. We use k-means GPU implementation in faiss [34] for efficient clustering. We use SGD with momentum of 0.9, a learning rate of 0.01, a batch size of 64. More implementation details can be found in the supplementary material.

4.2. Results on FUDA

Baselines. **SO** is a model only trained using the labeled source images. **CDAN** [42] and **MDDIA** [32] are both state-of-the-art methods in UDA with a domain classifier to perform domain alignment. **MME** [57] minimizes the conditional entropy of unlabeled target data with respect to the feature extractor and maximizes it with respect to the classifier. **CAN** [35] uses clustering information to con-

Table 3: Adaptation accuracy (%) comparison on 3% and 6% labeled samples per class on the Office-Home dataset.

Method	Office-Home: Target Acc. (%)												
	Ar →Cl	Ar →Pr	Ar →Rw	Cl →Ar	Cl →Pr	Cl →Rw	Pr →Ar	Pr →Cl	Pr →Rw	Rw →Ar	Rw →Cl	Rw →Pr	Avg
3% labeled source													
SO	24.4	38.3	43.1	26.4	34.7	33.7	27.5	26.5	42.6	41.2	29.0	52.3	35.0
MME [57]	4.5	15.4	25.0	28.7	34.1	37.0	25.6	25.4	44.9	39.3	29.0	52.0	30.1
CDAN [42]	5.0	8.4	11.8	20.6	26.1	27.5	26.6	27.0	40.3	38.7	25.5	44.9	25.2
MDDIA [32]	21.7	37.3	42.8	29.4	43.9	44.2	37.7	29.5	51.0	47.1	29.2	56.4	39.1
CAN [35]	17.1	30.5	33.2	22.5	34.5	36.0	18.5	19.4	41.3	28.7	18.6	43.2	28.6
CDS [36]	33.5	41.1	41.9	45.9	46.0	49.3	44.7	37.8	51.0	51.6	35.7	53.8	44.4
DANN + ENT [17]	19.5	30.2	38.1	18.1	21.8	24.2	31.6	23.5	48.1	40.7	28.1	50.2	31.2
MME + ENT	31.2	35.2	40.2	37.3	39.5	37.4	48.7	42.9	60.9	59.3	46.4	58.6	44.8
CDAN + ENT	20.6	31.4	41.2	20.6	24.9	30.6	33.5	26.5	56.7	46.9	29.5	48.4	34.2
CDS + ENT	39.2	46.1	47.8	49.9	50.7	54.1	48.0	43.5	59.3	58.6	44.3	59.3	50.1
CDS + MME + ENT	39.4	48.0	52.1	50.0	52.3	56.4	47.8	44.2	60.6	61.1	45.3	62.1	51.6
CDS / MME + ENT†	41.7	49.4	57.8	51.8	52.3	55.9	54.3	46.2	69.0	65.6	52.2	68.2	55.4
CDS / CDAN + ENT†	37.7	49.2	56.5	49.8	51.9	55.9	50.0	42.3	68.1	63.1	48.7	67.5	53.4
PCS (Ours)	**42.1**	**61.5**	**63.9**	**52.3**	**61.5**	**61.4**	**58.0**	**47.6**	**73.9**	**66.0**	**52.5**	**75.6**	**59.7**
Improvement	+0.4	+12.1	+6.1	+0.5	+9.2	+5.5	+3.7	+1.4	+4.9	+0.4	+0.3	+7.4	+4.3
6% labeled source													
SO	28.7	45.7	51.2	31.9	39.8	44.1	37.6	30.8	54.6	49.9	36.0	61.8	42.7
MME [57]	27.6	43.2	49.5	41.1	46.6	49.5	43.7	30.5	61.3	54.9	37.3	66.8	46.0
CDAN [42]	26.2	33.7	44.5	34.8	42.9	44.7	42.9	36.0	59.3	54.9	40.1	63.6	43.6
MDDIA [32]	25.1	44.5	51.9	35.6	46.7	50.3	48.3	37.1	64.5	58.2	36.9	68.4	50.3
CAN [35]	20.4	34.7	44.7	29.0	40.4	38.6	33.3	21.1	53.4	36.8	19.1	58.0	35.8
CDS [36]	38.8	51.7	54.8	53.2	53.3	57.0	53.4	44.2	65.2	63.7	45.3	68.6	54.1
DANN + ENT [17]	22.4	32.9	43.5	23.2	30.9	33.3	33.2	26.9	54.6	46.8	32.7	55.1	36.3
MME + ENT	37.2	42.4	50.9	46.1	46.6	49.1	53.5	45.6	67.2	63.4	48.1	71.2	51.8
CDAN + ENT	23.1	35.5	49.2	26.1	39.2	43.8	44.7	33.8	61.7	55.1	34.7	67.9	42.9
CDS + ENT	42.9	55.5	59.5	55.2	55.1	59.1	54.3	46.9	68.1	65.7	50.6	71.5	57.0
CDS + MME + ENT	41.7	58.1	61.7	55.7	56.2	61.3	54.6	47.3	68.6	66.4	50.3	72.1	57.8
CDS / MME + ENT†	44.1	51.6	63.3	53.9	55.2	62.0	56.5	46.6	70.9	67.7	54.7	74.7	58.4
CDS / CDAN + ENT†	39.0	51.3	63.1	51.0	55.0	63.6	57.8	45.9	72.8	65.8	50.4	73.5	57.4
PCS (Ours)	**46.1**	**65.7**	**69.2**	**57.1**	**64.7**	**66.2**	**61.4**	**47.9**	**75.2**	67.0	53.9	**76.6**	**62.6**
Improvement	+2.0	+14.1	+5.9	+3.2	+9.5	+2.6	+3.6	+1.3	+2.4	-0.7	-0.8	+1.9	+4.2

† Two-stage training results reported in [36].

Table 4: Adaptation accuracy (%) comparison on 0.1% and 1% labeled samples per class on the VisDA-2017 dataset.

Method	VisDA: Target Acc. (%)	
	0.1% Labeled	1% Labeled
SO	47.9	51.4
MME [57]	55.6	69.4
CDAN [42]	58.0	61.5
MDDIA [32]	68.9	71.3
CAN [35]	51.3	57.2
CDS [36]	34.2	67.5
DANN + ENT [17]	44.5	50.2
MME + ENT	54.0	66.1
CDAN + ENT	57.7	58.1
CDS + ENT	49.8	75.3
CDS + ENT + MME	60.0	78.3
CDS / MME + ENT†	62.5	69.4
CDS / CDAN + ENT†	69.0	69.1
PCS (Ours)	**78.0**	**79.0**
Improvement	+9.0	+0.7

† Two-stage training results reported in [36].

Table 5: Adaptation accuracy (%) comparison on 1-shot and 3-shots per class on the DomainNet dataset.

Method	DomainNet: Target Acc. (%)							
	R→C	R→P	R→S	P→C	P→R	C→S	S→P	Avg
1-shot labeled source								
SO	18.4	30.6	16.7	16.2	28.9	12.7	10.5	19.1
MME [57]	13.8	29.2	9.7	16.0	26.0	13.4	14.4	17.5
CDAN [42]	16.0	25.7	12.9	12.6	19.5	7.2	8.0	14.6
MDDIA [32]	18.0	30.6	15.9	15.4	27.4	9.3	10.2	18.1
CAN [35]	18.3	22.1	16.7	13.2	23.9	11.1	12.1	16.8
CDS [36]	16.7	24.4	11.1	14.1	15.9	13.4	19.0	16.4
CDS + ENT	21.7	30.1	18.2	17.4	20.5	18.6	22.7	21.5
CDS + MME + ENT	21.2	28.8	15.5	15.8	17.6	19.0	20.7	19.8
PCS (Ours)	**39.0**	**51.7**	**39.8**	**26.4**	**38.8**	**23.7**	**23.6**	**34.7**
Improvement	+17.3	+21.1	+21.6	+9.0	+9.9	+4.7	+0.9	+13.2
3-shots labeled source								
SO	30.2	44.2	25.7	24.6	49.8	24.2	23.2	31.7
MME [57]	22.8	46.5	14.5	25.1	50.0	20.1	24.9	29.1
CDAN [42]	30.0	40.1	21.7	21.4	40.8	17.1	19.7	27.3
MDDIA [32]	41.4	50.7	37.4	31.4	52.9	23.1	24.1	37.3
CAN [35]	28.1	33.5	25	24.7	46.9	23.3	20.1	28.8
CDS [36]	35.0	43.8	36.7	34.1	36.8	31.1	34.5	36.0
CDS + ENT	44.5	52.2	40.9	40.0	47.2	33.0	40.1	42.5
CDS + MME + ENT	43.8	54.9	41.1	38.9	45.9	32.8	38.7	42.3
PCS (Ours)	**45.2**	**59.1**	**41.9**	**41.0**	**66.6**	31.9	37.4	**46.1**
Improvement	+0.7	+6.9	+0.8	+1.0	+13.7	-0.9	-2.7	+3.6

trast discrepancy of source and target domain. **CDS** [36] is a instance-based cross-domain self-supervised pre-training, which can be used for other domain adaptation methods and form *two-stage* methods, such as **CDS / CDAN** and **CDS / MME**. We re-implement CDS into an end-to-end version by adding losses in two stage together and tuning the weight for different losses. We also investigate the *one-stage* version of the methods above (**CDS + CDAN**, **CDS + MME**). Following [36], entropy minimization (**ENT**) on source is

Table 6: Performance contribution of each part in PCS framework on Office-Home.

Method	Office-Home: Target Acc.												
	Ar →Cl	Ar →Pr	Ar →Rw	Cl →Ar	Cl →Pr	Cl →Rw	Pr →Ar	Pr →Cl	Pr →Rw	Rw →Ar	Rw →Cl	Rw →Pr	Avg
3% labeled source													
$\mathcal{L}_{\text{cls}}$	24.4	38.3	43.1	26.4	34.7	33.7	27.5	26.5	42.6	41.2	29.0	52.3	35.0
$+\mathcal{L}_{\text{InSelf}}$	34.6	48.3	54.7	49.2	53.1	57.1	48.2	40.6	62.9	57.9	44.9	68.8	51.7
$+\mathcal{L}_{\text{CrossSelf}}$	36.5	53.7	56.6	51.2	57.9	58.8	51.2	42.8	66.2	61.5	50.1	72.2	54.9
$+\mathcal{L}_{\text{MIM}}$	37.2	55.9	58.8	51.5	59.4	59.0	53.2	43.0	68.2	62.0	50.2	72.5	55.9
+APCU (PCS)	**42.1**	**61.5**	**63.9**	**52.3**	**61.5**	**61.4**	**58.0**	**47.6**	**73.9**	**66.0**	**52.5**	**75.6**	**59.7**
6% labeled source													
$\mathcal{L}_{\text{cls}}$	28.7	45.7	51.2	31.9	39.8	44.1	37.6	30.8	54.6	49.9	36.0	61.8	42.7
$+\mathcal{L}_{\text{InSelf}}$	40.8	57.6	65.5	54.5	62.4	62.7	54.6	43.1	73.6	64.2	44.7	75.9	58.3
$+\mathcal{L}_{\text{CrossSelf}}$	40.8	59.5	66.9	55.5	64.1	63.1	57.2	46.2	73.9	65.0	52.0	76.9	60.1
$+\mathcal{L}_{\text{MIM}}$	42.1	60.2	68.5	55.9	64.4	63.5	59.1	47.1	74.4	66.6	52.1	77.0	60.9
+APCU (PCS)	**46.1**	**65.7**	**69.2**	**57.1**	**64.7**	**66.2**	**61.4**	**47.9**	**75.2**	**67.0**	**53.9**	**76.6**	**62.6**

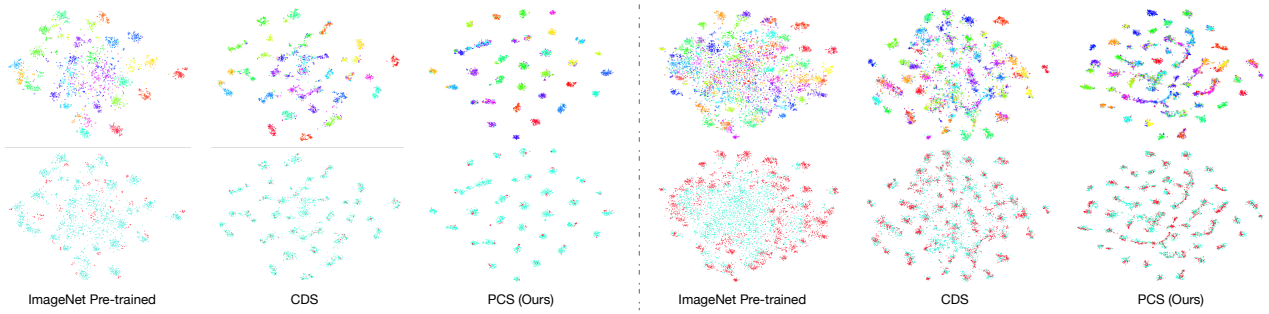


Figure 4: t-SNE visualization of ours and baselines on Office (left) and Office-Home (right). Top row: Coloring represents the class of each sample. Features with PCS are more discriminative than the ones with other methods. Bottom row: Cyan represents source features and Red represents target features. Feature from PCS are better-aligned between domains compared to other methods.

added to previous DA methods to obtain better baseline performance.

We compare the proposed PCS with state-of-the-art methods on FUDA (adaptation with few source labels). Extensive experiments are conducted on Office, Office-Home, VisDA-2017 and DomainNet, with the results presented in Table 1, 3, 4, and 5, respectively. We can see that PCS outperforms the best state-of-the-arts in all the benchmarks, with large improvements: 10.5% and 3.4% on Office, 4.3% and 4.2% on Office-Home, 9.0% and 0.7% on VisDA, 13.2% and 3.6% on DomainNet. If we look at the result of each domain pair in each dataset (*e.g.* D → A in Office), PCS outperforms previous best in 47 out of 52 settings. Finally, as the number of labeled samples decreases, PCS shows larger performance improvements against the previous best methods, which demonstrates PCS is extremely beneficial in label-scarce adaptation scenarios.

4.3. Ablation Study

Next, we investigate the effectiveness of different components in PCS on both Office and Office-Home. Table 2 and 6 show the results after adding each component incrementally. We can observe that adding each component contributes to the finally results without any performance degradation.

4.4. Analysis

We plot the learned features with t-SNE [45] on the DSLR-to-Amazon setting in Office, and Real-to-Clipart in Office-Home respectively in left and right of Figure 4. In the top row, the color represents the class of each sample; while in the bottom row, cyan represents source samples and red represents target samples. Compared to ImageNet pre-training and CDS, it qualitatively shows that PCS well clusters samples with the same class in the feature space; thus, PCS favors more discriminative features. Also, the features from PCS are more closely aggregated than ImageNet pre-training and CDS, which demonstrates that PCS learns a better semantic structure of the datasets.

5. Conclusion

In this paper, we investigated FUDA where only few-labeled source samples are available. We proposed a novel Prototypical Cross-domain Self-supervised learning (PCS) framework that leverages prototypes for in-domain and cross-domain self-supervised learning, as well as adaptive prototype-classifier learning. We demonstrated the superiority of PCS over previous best methods with extensive experimental results, setting a new state of the art for FUDA.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point-clouds. *arXiv preprint arXiv:2003.12641*, 2020. 3

[2] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations (ICLR)*, 2020. 3

[3] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15535–15545, 2019. 3

[4] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059, 2019. 5

[5] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017. 2, 3

[6] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019. 3

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020. 3

[8] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 1

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 2, 3

[10] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020. 3

[11] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3

[12] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 3

[13] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017. 3

[14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 1

[15] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pages 766–774, 2014. 3

[16] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015. 2

[17] 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. 1, 2, 6, 7, 16

[18] Vivien Sainte Fare Garnot and Loic Landrieu. Metric-guided prototype learning. *arXiv preprint arXiv:2007.03047*, 2020. 3

[19] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pages 2551–2559, 2015. 3

[20] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016. 3

[21] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 3

[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[23] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *2011 international conference on computer vision*, pages 999–1006. IEEE, 2011. 2

[24] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005. 5

[25] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012. 2

[26] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 2, 3

[27] Varun Gulshan, Lily Peng, Marc Coram, Martin C Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama*, 316(22):2402–2410, 2016. 1

[28] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 2, 3

[29] 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. 1, 6

[30] 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. PMLR, 2018. 1, 2

[31] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 1

[32] Xiang Jiang, Qicheng Lao, Stan Matwin, and Mohammad Havaei. Implicit class-conditioned domain alignment for unsupervised domain adaptation. In *International Conference on Machine Learning*, 2020. 6, 7, 16

[33] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3

[34] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017. 6

[35] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 6, 7

[36] Donghyun Kim, Kuniaki Saito, Tae-Hyun Oh, Bryan A Plummer, Stan Sclaroff, and Kate Saenko. Cross-domain self-supervised learning for domain adaptation with few source labels. *arXiv preprint arXiv:2003.08264*, 2020. 2, 3, 4, 6, 7, 12, 13, 14, 16

[37] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593. Springer, 2016. 3

[38] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020. 2, 3, 12, 13

[39] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016. 2

[40] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[41] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015. 1, 2

[42] 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. 2, 6, 7, 16

[43] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in neural information processing systems*, pages 136–144, 2016. 2

[44] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*, pages 2208–2217. PMLR, 2017. 2

[45] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 8, 14

[46] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020. 3

[47] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4500–4509, 2018. 2

[48] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 3

[49] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2, 3

[50] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019. 12

[51] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 3

[52] 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. 6, 12, 13

[53] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain

adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 6, 12, 13

[54] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6

[55] Paolo Russo, Fabio M Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8099–8108, 2018. 2

[56] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010. 6, 12, 13

[57] 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. 2, 6, 7, 12, 16

[58] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2018. 2

[59] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*, 2017. 2

[60] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017. 2

[61] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer, 2017. 2

[62] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019. 3

[63] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 3

[64] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011. 1

[65] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017. 1, 2

[66] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 2

[67] 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. 6, 12, 13

[68] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. 2, 3, 13

[69] 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. 2

[70] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2100–2110, 2019. 2

[71] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 3

[72] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 1

[73] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2

[74] Junbao Zhuo, Shuhui Wang, Weigang Zhang, and Qingming Huang. Deep unsupervised convolutional domain adaptation. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 261–269, 2017. 2

Appendix

A. Proof of Equation (13)

As mentioned in Section 3.2 of the main paper, in order for the prototype-classifier learning paradigm to work well, the network is desired to have enough confident predictions for all classes to get robust $\hat{\mathbf{w}}_i^s$ and $\hat{\mathbf{w}}_i^t$. First, to promote the network to have diversified outputs, we propose to maximize the entropy of expected network prediction $\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)])$. Second, to get high-confident prediction for each sample, we perform entropy minimization on the network output. So the overall objective is:

$$\max \mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))]. \tag{16}$$

Now we show that this objective equals maximizing the mutual information between input and output, *i.e.* $\mathcal{I}(y;\mathbf{x})$:

$$\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))] \tag{17}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log p(y_i|\mathbf{x})\right] - \sum_{i=1}^{L} \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})] \log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]] \tag{18}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log p(y_i|\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]\right] \tag{19}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log \frac{p(y_i|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]}\right] \tag{20}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x})]} \, \mathrm{d}y\right] \tag{21}$$

$$= \int p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{\int p(\mathbf{x}) p(y|\mathbf{x}) \, \mathrm{d}\mathbf{x}} \, \mathrm{d}y \tag{22}$$

$$= \int p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{p(y)} \, \mathrm{d}y \tag{23}$$

$$= \iint p(y,\mathbf{x}) \log \frac{p(y,\mathbf{x})}{p(y)p(\mathbf{x})} \, \mathrm{d}y \, \mathrm{d}\mathbf{x} \;=\; \mathcal{I}(y;\mathbf{x}) \tag{24}$$

In addition, we estimate $\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)])$ with $\sum_{x\in\mathcal{D}} p(y|\mathbf{x};\theta) \log \hat{\mathbf{p}}_0$, where $\hat{\mathbf{p}}_0$ is a moving average of $p(y|\mathbf{x};\theta)$.

B. Additional Datasets Details

Overall statistics of the datasets and the number of labeled source examples used in our experiments can be found in Table 7. For Office [56], Office-Home [67] and VisDA [53], we follow the same setting in [36], randomly sampling labeled images from the source domain and ensure that each class has at least one labeled example. For DomainNet [52], we use the same split files as [57] and further select 1-shot and 3-shots labeled samples in the training set for each class.

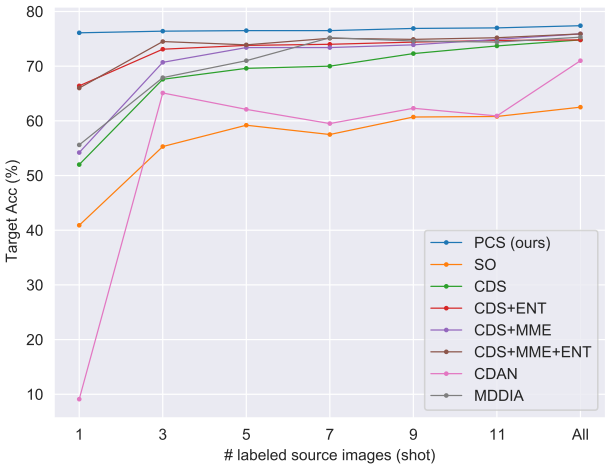


Figure 5: Sample efficiency comparison from DSLR to Amazon in Office.

C. Additional Implementation Details

We implemented our model in PyTorch [50]. We choose batch size of 64 for both source and target in self-supervised learning and batch size of 32 for the classification loss. The learning rate ratio between linear layer and convolution layer is set to $1:0.1$. We use SGD with weight decay rate $5e^{-4}$. For Office and Office-Home, we adaptively set temperature ϕ according to [38]. For VisDA and DomainNet, we fix ϕ to be 0.1 for more stable training. We set temperature τ to be 0.1 in all experiments. We choose hyper-parameters λ_{in} and $\lambda_{\text{cross}} \in \{1, 0.5\}$, and the weight $\lambda_{\text{mim}} \in \{0.05, 0.01\}$. As for parameters m (momentum for memory bank update) and M (number of k-means in $\mathcal{L}_{\text{InSelf}}$), we set $m = 0.5$ and $M = 20$.

We use spherical k-means for clustering and set half of the number of clusters in k-means to be the number of the classes n_c, and the rest to be $2n_c$. We compute the weight for cosine classifier only using source images for the first 5 epochs and set t_w to be around half of the average number of images per class. New prototypes (*i.e.* centroids of clusters and weights of cosine classifier) are computed per epoch for both self-supervised learning and classification.

Our implementation will be open-source released.

D. Sample Efficiency

We compare our method with other state-of-the-art methods on Office dataset (DSLR as source and Amazon

Table 7: Dataset statistics and labeled source used

Dataset	Domain	# total image	# labeled images	# classes
Office [56]	Amazon (A)	2817	1-shot and 3-shots labeled source	31
	DSLR (D)	498		
	Webcam (W)	795		
Office-Home [67]	Art (Ar)	2427	3% and 6% labeled source	65
	Clipart (Cl)	4365		
	Product (Pr)	4439		
	Real (Rw)	4357		
VisDA [53]	Synthetic (Syn)	152K	0.1% and 1% labeled source	12
	Real (Rw)	55K		
DomainNet [52]	Clipart (C)	18703	1-shot and 3-shots labeled source	126
	Painting (P)	31502		
	Real (R)	70358		
	Sketch (S)	24582		

Table 8: Accuracy of cross-domain weighted kNN with different SSL methods.

Method	D→A	Rw→Cl
ImageNet pre-train	62.5	40.6
ID [68]	70.3	51.9
CDS [36]	72.5	53.7
protoNCE [38]	72.3	49.3
$\mathcal{L}_{\text{InSelf}} + \mathcal{L}_{\text{CrossSelf}}$	**75.5**	**55.3**

Table 9: Accuracy of cross-domain weighted kNN with different FUDA methods.

Method	D→A (1-shot)	Rw→Cl (3%)
CDS [36]	72.3	57.6
CDS + ENT	72.8	58.6
CDS + MME + ENT	60.8	59.2
PCS (Ours)	**76.0**	**59.3**

as target) with a varying number of source labels. From Figure 5, we can see that PCS outperforms all SOTA methods in all settings with different number of labeled samples. Moreover, our method is very label-efficient: a) For 1-shot image per class (31 labeled source images in total), PCS can achieve **76.1%** accuracy. b) For the fully-labeled setting (498 labeled source images in total), PCS can achieve **77.4%** accuracy. c) With **94%** less labeled source images, the performance degradation of our method is only **1.3%**.

E. Quantitative Feature Analysis

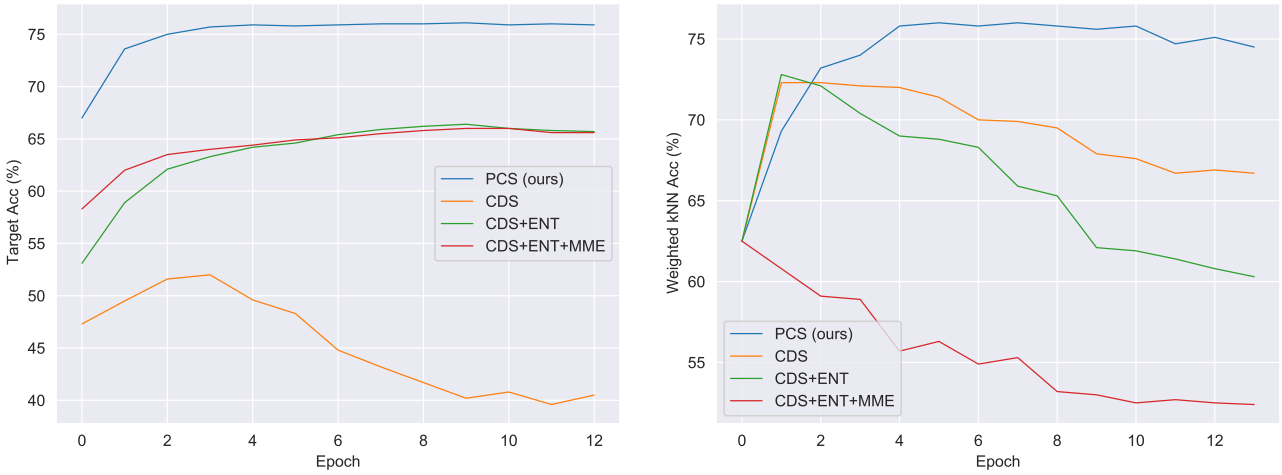
To quantitatively compare the quality of learned features with different approaches, we perform classification with weighted k-nearest neighbor (kNN) classifier proposed by Wu *et al*. [68] in a cross-domain manner. Specifically, given a test image $\mathbf{x}^t$, we first compute its normalized feature $\mathbf{f}^t = F(\mathbf{x}^t)$, and then compare it again embeddings of all source images in the source memory bank $\boldsymbol{V}^s$ using cosine similarity $s_i = \cos(\mathbf{f}^t, \mathbf{v}_i^s)$. The top k nearest neighbors in the source domain, $\mathcal{N}_k$, would be used to make the final prediction with weighted voting. Specifically, class c would get weight $w_c = \sum_{i \in \mathcal{N}_k} \alpha_i \cdot 1(c_i = c)$, in which α_i is the contributing weight of neighbor $\mathbf{v}_i^s$ defined as $\alpha_i = \exp(s_i/\tau)$. We set $\tau = 0.07$ and $k = 200$ as in [68].

We perform the above cross-domain kNN classification on models trained with 1) only cross-domain self-supervised learning methods, and 2) Few-shot Unsupervised Domain Adaptation methods, with the results shown in Table 8 and Table 9, respectively. From the results, we can see that both the proposed cross-domain prototypical self-supervised learning method and the whole PCS framework outperforms previous approaches.

F. Stability Analysis of PCS

To show the performance stability of PCS, we conduct multiple runs with three different random seeds. Table 10 reports the averaged accuracy and standard deviation of the three runs on the 1-shot and 3-shots settings of Office.

Figure 6 shows adaptation accuracy vs. training epochs using cosine classifier (Figure 6a) and weighted kNN classifier (Figure 6b). From the plots, we can see that the target



a Target Acc. with cosine classifier vs. training epochs

b Target Acc. with Weighted kNN vs. training epochs

Figure 6: Stability of Target Accuracy during training procedure.

Table 10: Averaged accuracy and standard deviation of PCS on three runs of 1-shot and 3-shots on Office dataset.

Labeled Source	A→D	A→W	D→A	D→W	W→A	W→D
1-shot	60.2±1.9	69.8±0.8	76.1±0.4	90.6±0.8	71.2±1.0	91.8±1.9
3-shots	78.2±1.8	82.9±1.1	76.4±0.5	94.1±0.1	76.3±0.7	96.0±0.7

Table 11: Sum of pair-wise cosine-similarity between prototypes in Office and Office-Home.

Method	D→A (1-shot)	Rw→Pr (3%)
SO	0.44	-0.71
CDS [36]	0.43	-0.71
PCS w/o APCU	-53.3	-22.8
PCS (Ours)	**-58.4**	**-26.5**

accuracy of PCS increases more steadily and robustly compared to other methods. In addition, PCS converges faster than other methods.

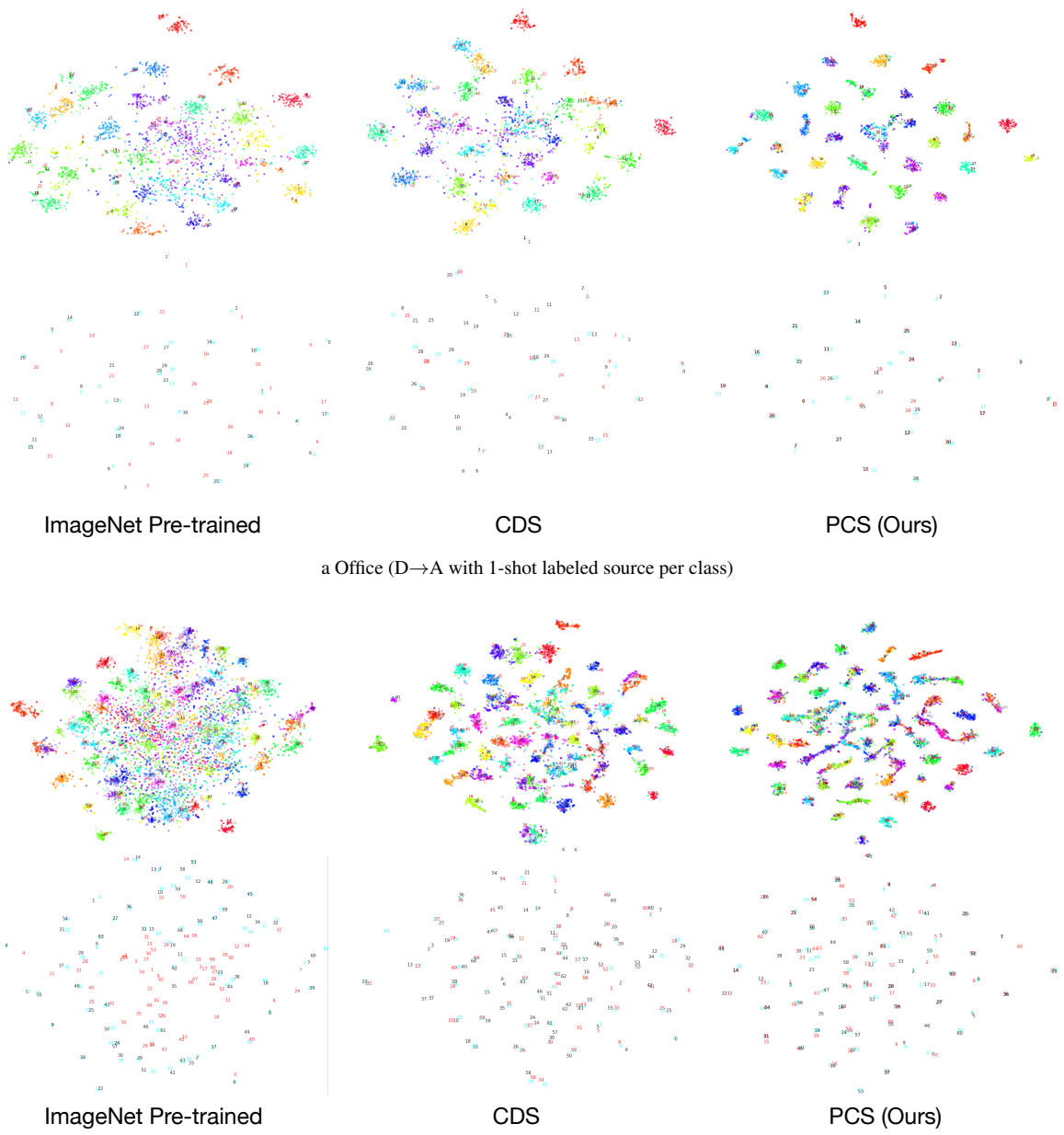
G. Prototype Quality Comparison

To further compare how well source and target are aligned, we provide more t-SNE [45] visualizations on Office (D→A) and Office-Home (Rw→Cl) in Figure 7a and 7b, comparing ImageNet Pre-training, CDS [36] and PCS. Specifically, we plot representations for all samples (top in both Figures), as well as the prototypes (normalized average representation) for each class. In top rows of both figures, the color of a sample represents its class, and samples from different domains are represented by different shapes (circles for source and crosses for target). In bottom rows of both figures, the number of a prototype represents its class index, and color represent the domain of the prototype (Cyan for source, Red for target, and Black for prototype weight of the classifier). As we can see from Figure 7, for each class, the prototypes of source, target and the weight vector of classifier gets more aggregated with PCS than other methods, which demonstrates that PCS could better align source and target domains.

In a well-learned feature embedding space, prototypes of different classes should be far / different from each other. To quantitatively measure the similarity of the learned prototypes, we compute the sum of cosine similarities between all pairs of prototypes. From the results shown in Table 11, we can see that the prototypes learned with PCS have the least similarities, indicating that PCS learns an embedding space with better semantic structure.

H. Image Retrieval Results

We present cross-domain image retrieval results in Figure 8. Given a query feature $\mathbf{f}_q$ in the target domain, we measure the pairwise cosine similarity between $\mathbf{f}_q$ and all features in the source domain. The source images with the most similar features as $\mathbf{f}_q$ are returned as the top retrieval results. We compare image retrieval results of PCS with CDS in Figure 8. In Figure 8, features from model trained with CDS are biased to some wrong color, texture and other visual clues; and quantitatively similar features do not correspond to semantically similar images in differ-



a Office (D→A with 1-shot labeled source per class)

b Office-Home (Rw→Cl with 3% labeled source per class)

Figure 7: t-SNE visualization of ours and baselines on Office (a) and Office-Home (b). Top row: Coloring represents the class of each sample, and shape represents domain (circle for source and cross for target). Features with PCS are more discriminative than the ones with other methods. Bottom row: each number represents a centroid for corresponding class. **Cyan** represents centroids of source images based on ground truth and **Red** for target. **Black** represents prototypes of the classifier. Centroids from PCS are better-aligned between domains compared to other methods. (Zoom in for more details).

ent domains. We can see that PCS could extract features that are more discriminative and semantically meaningful across domains.

I. Performance Comparison with UDA Methods using Full Source Labels

We have shown the superiority of PCS in label-scarce setting (FUDA), and we further conduct experiments with fully-labeled source domain (UDA). The performance comparison with other UDA methods on Office and Office-

Figure 8: Image retrieval examples of the closest cross-domain neighbors using CDS (a) and PCS (b) in Office-Home (Target: Real, Source: Art).

Table 12: Adaptation accuracy (%) comparison on fully-labeled setting on the Office-Home dataset.

Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
SO	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DANN [17]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
CDAN [42]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
MMDIA [32]	56.2	77.9	79.2	64.4	73.1	74.4	64.2	54.2	79.9	71.2	58.1	83.1	69.5
MME [57]	54.2	72.8	78.3	57.9	70.2	71.8	58.5	52.9	77.9	72.7	58.1	81.8	67.3
CDS / MME [36]	56.9	73.3	76.5	62.8	73.1	71.1	63.0	57.9	79.4	72.5	62.5	83.0	69.3
PCS (Ours)	55.8	76.9	**80.3**	**67.9**	**74.0**	**75.7**	**67.0**	52.9	**81.0**	**74.5**	**58.3**	82.8	**70.6**

Table 13: Adaptation accuracy (%) comparison on fully-labeled setting on the Office dataset.

Method	A→D	A→W	D→A	D→W	W→A	W→D	Avg
SO	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DANN [17]	79.7	82	68.2	96.9	67.4	99.1	82.2
CDAN [42]	92.9	94.1	71	98.6	69.3	100	87.7
MMDIA [32]	92.1	90.3	75.3	98.7	74.9	99.8	88.8
MME [57]	88.8	87.3	69.2	98.7	65.6	100	84.9
CDS + MME [36]	86.9	88.3	75.9	98.6	73.3	100	87.1
PCS (Ours)	**94.6**	92.1	**77.4**	97.7	**77.0**	99.8	**89.8**

Home are presented in Table 13 and Table 12, respectively. We can see that PCS achieves the best results even with fully-labeled source, and this shows the proposed PCS could potentially be applied to a wide range of adaptation settings.