Universal Domain Adaptation through Self-Supervision

Kuniaki Saito[1] **Donghyun Kim**[1] **Stan Sclaroff**[1]

Kate Saenko[1,2]
[1]Boston University [2]MIT-IBM Watson AI Lab

[keisaito, dohnk,sclaroff,saenko]@bu.edu

Abstract

Unsupervised domain adaptation methods traditionally assume that all source categories are present in the target domain. In practice, little may be known about the category overlap between the two domains. While some methods address target settings with either partial or open-set categories, they assume that the particular setting is known a priori. We propose a more universally applicable domain adaptation framework that can handle arbitrary category shift, called Domain Adaptive Neighborhood Clustering via Entropy optimization (DANCE). DANCE combines two novel ideas: First, as we cannot fully rely on source categories to learn features discriminative for the target, we propose a novel neighborhood clustering technique to learn the structure of the target domain in a self-supervised way. Second, we use entropy-based feature alignment and rejection to align target features with the source, or reject them as unknown categories based on their entropy. We show through extensive experiments that DANCE outperforms baselines across open-set, open-partial and partial domain adaptation settings. Implementation is available at `https://github.com/VisionLearningGroup/DANCE`.

1 Introduction

Deep neural networks can learn highly discriminative representations for image recognition tasks [8, 38, 20, 32, 16], but do not generalize well to domains that are not distributed identically to the training data. Domain adaptation (DA) aims to transfer representations of source categories to novel target domains without additional supervision. Recent deep DA methods primarily do this by minimizing the feature distribution shift between the source and target samples [11, 25, 39]. However, these methods make strong assumptions about the degree to which the source categories overlap with the target domain, which limits their applicability to many real-world settings.

In this paper, we investigate the problem of *Universal DA*. Suppose L_s and L_t are the label sets in the source and target domain. In Universal DA we want to handle all of the following potential "category shifts": closed-set ($L_s = L_t$), open-set ($L_s \subset L_t$) [1, 36], partial ($L_t \subset L_s$) [2], or a mix of open and partial [44]. Existing DA methods cannot address Universal DA well because they are each designed to handle just one of the above settings. However, since the target domain is unlabeled, we may not know in advance which of these situations will occur. Thus, an unexpected category shift could lead to catastrophic misalignment. For example, using a closed-set method when the target has novel ("unknown") classes could incorrectly align them to source ("known") classes. The underlying issue at play is that existing work heavily relies on prior knowledge about the category shift.

The second problem is that the over-reliance on source supervision makes it challenging to obtain discriminative features on the target. Prior methods focus on aligning target features with source,
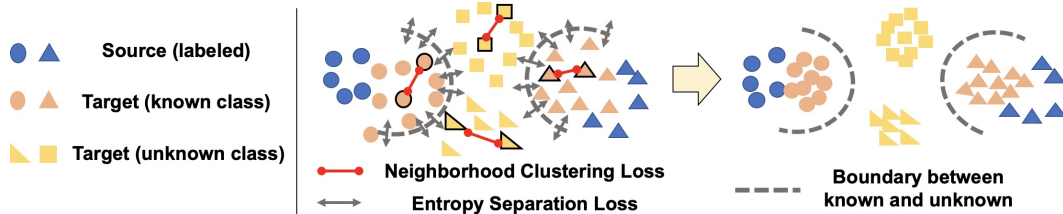
Figure 1: We propose *DANCE*, which combines a self-supervised clustering loss (red) to cluster neighboring target examples and an entropy separation loss (gray) to consider alignment with source (best viewed in color).

rather than on exploiting structure specific to the target domain. In the universal DA setting, this means that we may fail to learn features useful for discriminating "unknown" categories from the known categories, because such features may not exist in the source. Self-supervision was proposed in [5] to extract domain-generalizable features, but it is limited in that they did not utilize the cluster structure of the target domain.

We propose to overcome these challenging problems by introducing *Domain Adaptive Neighborhood Clustering via Entropy optimization (DANCE)*. An overview is shown in Fig. 1. Rather than relying only on the supervision of source categories to learn a discriminative representation, *DANCE* harnesses the cluster structure of the target domain using self-supervision. This is done with a "neighborhood clustering" technique that self-supervises feature learning in the target. At the same time, useful source features and class boundaries are preserved and adapted via distribution alignment with batch-normalization [7] and a novel partial domain alignment loss that we refer to as "entropy separation loss." This loss allows the model to either match each target example with the source, or reject it as an "unknown" category. Our contributions are summarized as follows: (i) we propose *DANCE*, a universal domain adaptation framework that can be applied out-of-the-box without prior knowledge of specific category shift, (ii) design two novel loss functions, neighborhood clustering and entropy separation, for shift-agnostic adaptation, (iii) experimentally observe that *DANCE* is the only method that outperforms the source-only model in every setting, achieving state-of-the-art performance on all open-set and open-partial DA settings, and some partial DA settings, and (iv) learn discriminative features of "unknown" target samples without any supervision.

2 Related Work

Closed-set Domain Adaptation (CDA). The main challenge in domain adaptation (DA) is to leverage unlabeled target data to improve the source classifier's performance while accounting for domain shift. Classic approaches measure the distance between feature distributions in source and target, then train a model to minimize this distance. Many DA methods utilize a domain classifier to measure the distance [11, 40, 25, 26], while others minimize utilize pseudo-labels assigned to target examples [35, 48]. Clustering-based methods are proposed by [9, 37, 15]. These and other mainstream methods assume that all target examples belong to source classes. In this sense, they rely heavily on the relationship between source and target. **Partial Domain Adaptation (PDA)** handles the case where the target classes are a subset of source classes. This task is solved by performing importance-weighting on source examples that are similar to samples in the target [2, 45, 3]. **Open set Domain Adaptation (ODA)** deals with target examples whose class is different from any of the source classes [29, 36, 24]. The drawback of ODA methods is that they assume we necessarily have unknown examples in the target domain, and can fail in closed or partial domain adaptation. The idea of **Universal Domain Adaptation (UniDA)** was proposed in [44]. However, they applied their method to a mixture of PDA and ODA, which we call **OPDA**, where the target domain contains a subset of the source classes plus some unknown classes. Our goal is to propose a method that works well on CDA, ODA, PDA, and OPDA. We call the task UniDA in our paper. **Entropy minimization** [13] for unlabeled samples is popular in semi-supervised learning. In CDA, its effectiveness is confirmed when combined with batch-normalization based domain alignment [4]. Pseudo-labeling is a similar approach since it increases the confidence of the prediction for target samples and also performs well on CDA when combined with domain-specific batch-normalization [7]. These methods simply attempt to increase the confidence for "known" classes while our entropy separation loss can decrease the confidence to reject "unknown" classes.

Self-Supervised Learning. Self-supervised learning obtains features useful for various image recognition tasks by using a large number of unlabeled images [10]. A model is trained to solve a

pretext (surrogate) task such as solve a jigsaw puzzle [28] or instance discrimination [42]. [6] trained a model to predict each sample's cluster index given by k-means clustering. Directly applying the method to UniDA is challenging, since we need to know the number of clusters in the target domain. [19, 47] proposed to perform instance discrimination and trained a model to discover neighborhoods for each example. Their result indicates that we can cluster samples without specifying their cluster centers. They calculate cross entropy loss on the probabilistic distribution of similarity between examples. Our neighborhood clustering (NC) is similar in that we aim to perform unsupervised clustering of unlabeled examples without specifying cluster centers, but different in that [19, 47] require assigning neighbors for each example. Instead, we perform entropy minimization on the similarity distribution among unlabeled target examples and source prototypes. Objectives that can cluster samples without specifying the center can possibly replace NC. But, note that one of our contributions is to provide a framework for UniDA that exploits cluster structure of the target domain and domain-alignment objective with a rejection option. Since these methods are not designed for DA, we focus on the comparison with DA baselines in our main paper. We provide the ablation study of replacing our neighborhood clustering loss with [19, 6, 5] to better understand each loss in supplemental material.

3 DANCE: Domain Adaptive Neighborhood Clustering via Entropy optimization

Our task is universal domain adaptation: given a labeled source domain $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{N_s}$ with "known" categories L_s and an unlabeled target domain $\mathcal{D}_t = \{(\mathbf{x}_i^t)\}_{i=1}^{N_t}$ which contains all or some "known" categories and possible "unknown" categories. Our goal is to label the target samples with either one of the L_s labels or the "unknown" label. We train the model on $\mathcal{D}_s \cup \mathcal{D}_t$ and evaluate on $\mathcal{D}_t$. We seek a truly universal method that can handle any possible category shift without prior knowledge of it. The key is not to force complete alignment between the entire source and target distributions, as this may result in catastrophic misalignment. Instead, the challenge is to extract well-clustered target features while performing a relaxed alignment to the source classes and potentially rejecting "unknown" points.

We adopt a prototype-based classifier that maps samples close to their true class centroid (prototype) and far from samples of other classes. We first propose to use self-supervision in the target domain to cluster target samples. We call this technique **neighborhood clustering (NC)**. Each target point is aligned either to a "known" class prototype in the source or to its neighbor in the target. This allows the model to learn a discriminative metric that maps a point to its semantically close match, whether or not its class is "known". This is achieved by minimizing the entropy of the distribution over point similarity. Second, we propose an **entropy separation loss (ES)** to either align the target point with a source prototype or reject it as "unknown". The loss is applied to the entropy of the "known" category classifier's output to force it to be either low (the sample should belong to a "known" class) or high (the sample should be far from any "known" class). In addition, we utilize domain-specific batch normalization [7, 23, 34] to eliminate domain style information as a form of weak domain alignment.

3.1 Network Architecture
We adopt the architecture used in [34], which has an L2 normalization layer before the last linear layer. We can regard the weight vectors in the last linear layer as prototype features of each class. This architecture is well-suited to our purpose of finding a clustering over both target features and source prototypes. Let G be the feature extraction network which takes an input $\mathbf{x}$ and outputs a feature vector $\boldsymbol{f}$. Let $\mathbf{W}$ be the classification network which consists of one linear layer without bias. The layer consists of weight vectors $[\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K]$ where K represents the number of classes in the source. $\mathbf{W}$ takes L2 normalized features and outputs K logits. $\boldsymbol{p}$ denotes the output of $\mathbf{W}$ after the softmax function.

3.2 Neighborhood Clustering (NC)
The principle behind our self-supervised clustering objective is to move each target point either to a "known" class prototype in the source or to its neighbor in the target. By making nearby points closer, the model learns well-clustered features. If "unknown" samples have similar characteristics with other "unknown" samples, then this clustering objective will help us extract discriminative features. This intuition is illustrated in Fig. 1. The important point is that we do not rely on strict distribution alignment with the source in order to extract discriminative target features. Instead we propose to minimize the entropy of each target point's similarity distribution to other target samples and to
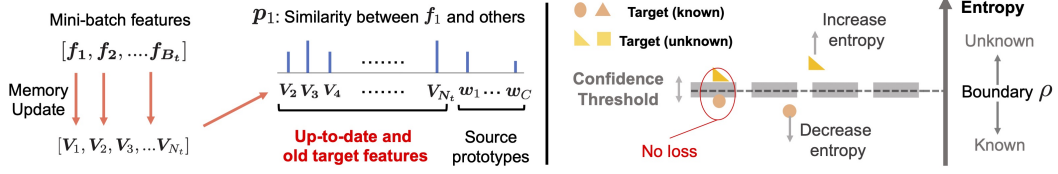
Figure 2: **Left**: Similarity distribution calculation in neighborhood clustering (best viewed in color). We minimize the entropy of the similarity distribution between each point (as shown for f_1), the prototypes, and the other target samples. Since most target samples are absent in the mini-batch, we store the their features in a memory bank, updating it with each batch. **Right**: An overview of the entropy separation loss (best viewed in color). We further decrease small entropy to move the sample to a "known"-class prototype, and increase large entropy to move it farther away. Since distinguishing "known" vs "unknown" samples near the boundary is hard, we introduce a confidence threshold that ignores such ambiguous samples.

prototypes. To minimize the entropy, the point will move closer to a nearby point (we assume a neighbor exists) or to a prototype. This approach is illustrated in Fig. 2.

Specifically, we calculate the similarity to all target samples and prototypes for each mini-batch of target features. Let $V \in R^{N_t \times d}$ denotes a memory bank which stores all target features and $F \in R^{(N_t+K) \times d}$ denotes the target features in the memory bank and the prototype weight vectors, where d is the feature dimension in the last linear layer:

$$V = [V_1, V_2, ... V_{N_t}], \tag{1}$$

$$F = [V_1, V_2, ... V_{N_t}, \mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_K], \tag{2}$$

where the V_i and $\mathbf{w}$ are L2-normalized. To consider target samples absent in the mini-batch, we employ a memory bank to store and use the features to calculate the similarity as done in [42]. In every iteration, V is updated with the mini-batch features. Let f_i denote features in the mini-batch and B_t denote sets of target samples' indices in the mini-batch. For all $i \in B_t$, we set

$$V_i = f_i. \tag{3}$$

Therefore, the memory bank V contains both updated target features from the current mini-batch and the older target features absent in the mini-batch. Unlike [42], we update the memory so that it simply stores features, without considering the momentum of features in previous epochs. Let F_j denote the j-th item in F, then the probability that the feature f_i is a neighbor of the feature or prototype F_j is, for $i \neq j$ and $i \in B_t$,

$$p_{i,j} = \frac{\exp(F_j^\top f_i / \tau)}{Z_i}, \tag{4}$$

where

$$Z_i = \sum_{j=1, j \neq i}^{N_t+K} \exp(F_j^\top f_i / \tau), \tag{5}$$

and the temperature parameter τ controls the distribution concentration degree [18]. Therefore, τ controls the number of neighbors for each sample, which implicitly affects the number of clusters. We provide the analysis on the parameter in supplemental material. Then, the entropy is calculated as

$$\mathcal{L}_{\text{nc}} = -\frac{1}{|B_t|} \sum_{i \in B_t} \sum_{j=1, j \neq i}^{N_t+K} p_{i,j} \log(p_{i,j}). \tag{6}$$

We minimize the above loss to align each target sample to either a target neighbor or a prototype, whichever is closer.

3.3   Entropy Separation loss (ES)
The neighborhood clustering loss encourages the target samples to become well-clustered, but we still need to align some of them with "known" source categories while keeping the "unknown" target samples far from the source. In addition to the domain-specific batch normalization (see Sec. 3.4), which can work as a form of weak domain alignment, we need an explicit objective to encourage alignment or rejection of target samples. As pointed out in [44], "unknown" target samples are likely

Table 1: Summary of the Universal comparisons. Each dataset (Office, OC, OH, VisDA) has multiple domains and adaptation scenarios and we provide the average accuracy over all scenarios. Our DANCE method substantially improves performance compared to the source-only model in all settings and the average rank of DANCE is significantly higher than all other baselines.

Method	Closed DA			Partial DA			Open set DA			Open-Partial DA			Avg	
	Office	OH	VD	OC	OH	VD	Office	OH	VD	Office	OH	VD	Acc	Rank
Source Only	76.5	54.6	46.3	75.9	57.0	49.9	89.1	69.6	43.2	86.4	71.0	38.8	61.7	4.8± 1.2
DANN [12]	**85.9**	62.7	69.1	42.2	40.9	38.7	88.7	72.8	48.2	88.7	76.7	50.6	65.7	3.5± 1.7
ETN [3]	85.2	64.0	64.1	**92.8**	69.4	59.8	88.2	71.9	51.7	88.3	72.6	66.6	70.5	2.9± 1.6
STA [24]	73.6	44.7	48.1	69.8	47.9	48.2	89.9	69.3	48.8	89.8	72.6	47.4	61.2	4.5± 1.3
UAN [44]	84.4	58.8	66.4	52.9	34.2	39.7	91.0	74.6	50.0	84.1	75.0	47.3	62.0	4.1± 1.3
DANCE (ours)	85.5	**69.1**	**70.2**	84.7	**71.1**	**73.7**	**94.1**	**78.1**	**65.3**	93.9	**80.4**	**69.2**	**77.3**	**1.2± 0.4**

to have a larger entropy of the source classifier's output than "known" target samples. This is because "unknown" target samples do not share common features with "known" source classes.

Inspired by this, we propose to draw a boundary between "known" and "unknown" points using the entropy of a classifier's output. We visually introduce the idea in Fig. 2. The distance between the entropy and threshold boundary, ρ, is defined as $|H(\boldsymbol{p}) - \rho|$, where $\boldsymbol{p}$ is the classification output for a target sample. By maximizing the distance, we can make $H(\boldsymbol{p})$ far from ρ. We expect that the entropy of "unknown" target samples will be larger than ρ whereas for the "known" ones it will be smaller. Tuning the parameter ρ based on each adaptation setting requires a validation set. Instead, we define $\rho = \frac{\log(K)}{2}$, where K is the number of source classes. Since $\log(K)$ is the maximum value of $H(\boldsymbol{p})$, we assume ρ depends on it, and confirm that the defined value empirically works well. We perform an analysis of ρ in the supplemental material. The above formulation assumes that "known" and "unknown" target samples can be separated with ρ. However, in many cases, the threshold can be ambiguous and can change due to domain shift. Therefore, we propose to introduce a confidence threshold parameter m such that the final form of the loss is

$$\mathcal{L}_{es} = \frac{1}{|B_t|} \sum_{i \in B_t} \mathcal{L}_{es}(\boldsymbol{p}_i), \quad \mathcal{L}_{es}(\boldsymbol{p}_i) = \begin{cases} -|H(\boldsymbol{p}_i) - \rho| & (|H(\boldsymbol{p}_i) - \rho| > m), \\ 0 & otherwise. \end{cases} \tag{7}$$

The introduction of the confidence threshold m allows us to give the separation loss only to confident samples. When $|H(\boldsymbol{p}_i) - \rho|$ is sufficiently large, the network is confident about a decision of "known" or "unknown". Thus, we train the network to make the sample far from the value ρ.

3.4 Training with Domain Specific Batch Normalization
To enhance alignment between source and target domain, we propose to utilize domain-specific batch normalization [7, 23, 34]. The batch normalization layer whitens the feature activations, which contributes to a performance gain. As reported in [34], simply splitting source and target samples into different mini-batches and forwarding them separately helps alignment. This kind of weak alignment matches our goal because strongly aligning feature distributions can harm the performance on non-closed set domain adaptation.

Final Objective. The final objective is

$$\mathcal{L} = \mathcal{L}_{cls} + \lambda(\mathcal{L}_{nc} + \mathcal{L}_{es}), \tag{8}$$

where $\mathcal{L}_{cls}$ denotes the cross-entropy loss on source samples. The loss on source and target is calculated in a different mini-batch to achieve domain-specific batch normalization. To reduce the number of hyper-parameters, we used the same weighting hyper-parameter λ for $\mathcal{L}_{nc}$ and $\mathcal{L}_{es}$.

4 Experiments

4.1 Experimental Settings
The goal of the experiments is to compare DANCE with the baselines across all sub-cases of Universal DA (i.e., CDA, PDA, ODA, and OPDA) under the four object classification datasets and four settings for each dataset. We follow the settings of [26] for closed (CDA), [2] for partial (PDA), [24] for open-set (ODA), and [44] for open-partial domain adaptation (OPDA) in our experiments.

Datasets. As the most prevalent benchmark dataset, we use **Office** [33], which has three domains (Amazon (A), DSLR (D), Webcam (W)) and 31 classes. The second benchmark dataset **OfficeHome**

Table 2: Results on closed domain adaptation including SAFN [43], CDAN [26] and MDD [46].

Method	Office (31 / 0 / 0)							Office-Home (65 / 0 / 0)													VisDA 12 / 0 / 0
	A2W	D2W	W2D	A2D	D2A	W2A	Avg	A2C	A2P	A2R	C2A	C2P	C2R	P2A	P2C	P2R	R2A	R2C	R2P	Avg	
SO	74.1	95.3	99.0	80.1	54.0	56.3	76.5	37.0	62.2	70.7	46.6	55.1	60.3	46.1	32.0	68.7	61.8	39.2	75.4	54.6	46.3
DANN [12]	86.7	97.2	99.8	86.1	**72.5**	**72.8**	**85.9**	46.8	68.4	76.6	54.7	63.9	69.7	57.1	44.7	75.7	64.9	51.3	78.7	62.7	69.1
ETN [3]	87.9	**99.2**	**100**	88.4	68.7	66.8	85.2	46.7	69.5	74.8	62.1	66.9	71.9	56.7	44.1	77.0	70.6	50.4	77.9	64.0	64.1
STA [24]	77.1	90.7	98.1	75.5	51.4	48.9	73.6	30.4	46.8	55.9	33.6	46.2	51.1	35.0	28.3	58.2	51.3	33.1	66.5	44.7	48.1
UAN [44]	86.5	97.0	**100**	84.5	69.6	68.7	84.4	45.0	63.6	71.2	51.4	58.2	63.2	52.6	40.9	71.0	63.3	48.2	75.4	58.7	66.4
DANCE	**88.6**	97.5	**100**	**89.4**	69.5	68.2	85.5	**54.3**	**75.9**	**78.4**	**64.8**	**72.1**	**73.4**	**63.2**	**53.0**	**79.4**	**73.0**	**58.2**	**82.9**	**69.1**	**70.2**
Methods tailored for Closed Domain Adaptation																					
SAFN [43]	88.8	98.4	99.8	87.7	69.8	69.7	85.7	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3	NA
CDAN [26]	93.1	98.2	100	89.8	70.1	68.0	86.6	49.0	69.3	74.5	54.4	66	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8	70.0
MDD [46]	94.5	98.4	100	93.5	74.6	72.2	88.9	54.9	73.7	77.8	60	71.4	71.8	61.2	53.6	78.1	72.5	60.2	82.3	68.1	74.6

Table 3: Results on partial domain adaptation including SAN [2] and IAFN [43].

Method	Office-Caltech(10 / 21 / 0)						Office-Home(25 / 40 / 0)														VisDA (6 / 6 / 0)
	A2C	W2C	D2C	D2A	W2A	Avg	A2C	A2P	A2R	C2A	C2P	C2R	P2A	P2C	P2R	R2A	R2C	R2P	Avg		
SO	75.4	70.7	68.5	80.4	84.6	75.9	37.1	64.5	77.1	52.0	51.3	62.4	52.0	31.3	71.6	66.6	42.6	75.1	57.0		46.3
DANN [12]	41.9	42.7	43.4	41.5	41.5	42.2	35.5	48.2	51.6	35.2	35.4	41.4	34.8	31.7	46.2	47.5	34.7	49.0	40.9		38.7
ETN [3]	**88.9**	**92.3**	**92.9**	**95.4**	**94.3**	**92.8**	52.1	**74.5**	83.1	69.8	65.2	76.5	69.1	50.6	82.5	76.3	53.8	79.1	69.4		59.8
STA [24]	75.7	72.4	62.8	70.5	67.7	69.8	35.0	55.2	59.7	37.5	48.4	53.5	36.0	32.2	59.9	54.3	38.5	64.6	47.9		48.2
UAN [44]	47.1	49.7	50.6	55.5	61.6	52.9	24.5	35.0	41.5	34.7	32.3	32.7	32.7	21.1	43.0	39.7	26.6	46.0	34.2		39.7
DANCE	88.8	79.2	79.4	83.7	92.6	84.8	**53.6**	73.2	**84.9**	70.8	67.3	82.6	70.0	50.9	84.8	77.0	55.9	81.8	**71.1**		**73.7**
Methods tailored for Partial Domain Adaptation																					
SAN [2]	NA	NA	NA	87.2	91.9	NA	44.4	68.7	74.6	67.5	65.0	77.8	59.8	44.7	80.1	72.2	50.2	78.7	65.3		NA
ETN [3]	89.5	92.6	93.5	95.9	92.3	92.7	59.2	77.0	79.5	62.9	65.7	75.0	68.3	55.4	84.4	75.7	57.7	84.5	70.5		66.0
IAFN [43]	NA	NA	NA	NA	NA	NA	58.9	76.3	81.4	70.4	73.0	77.8	72.4	55.3	80.4	75.8	60.4	79.9	71.8		67.7

(OH) [41] contains four domains and 65 classes. The third dataset **VisDA (VD)** [31] contains 12 classes from two domains: synthetic and real images. We provide an analysis of varying the number of classes using Caltech [14] and ImageNet [8] because these datasets contain a large number of classes. Let L_s denotes a set of classes present in the source, L_t denotes a set of classes present in the target. The class split in each setting ($|L_s \cap L_t|/|L_s - L_t|/|L_t - L_s|$) is shown in each table. We follow the experimental settings of [26, 2, 24, 44] for each split (see suppl. material for details).

Evaluation. We use the same evaluation metrics as previous works. In CDA and PDA, we simply calculate the accuracy over all target samples. In ODA and OPDA, we average the accuracy over classes including "unknown". For example, an average over 11 classes is reported in the Office ODA setting. We run each experiment three times and report the average result. For all settings, we activate "unknown" sample rejection method for all methods since we assume that we do not know the kind of a category shift.

Implementation. All experiments are implemented in Pytorch [30]. We employ ResNet50 [17] pre-trained on ImageNet [8] as the feature extractor in all experiments. We remove the last linear layer of the network and add a new weight matrix to construct **W**. For baselines, we use their implementation. Hyper-parameters for each method are tuned on the "Amazon to DSLR" OPDA setting. We set λ in Eq. 9 as 0.05 and m in Eq. 7 as 0.5 for our method. For all comparisons, we use the same hyper-parameters, batch-size, learning rate, and checkpoint. The analysis of sensitivity to hyper-parameters is discussed in the supplementary.

Comparisons. We show two kinds of comparisons to provide better empirical insights. The first is the universal comparison to the 5 baselines including state-of-the-art methods on CDA, PDA, ODA, and OPDA. As we assume that we do not have prior knowledge of the category shift in the target domain, so all methods use fixed hyper-parameters, which are tuned on the "Amazon to DSLR" OPDA setting. This means that even for CDA and PDA, all methods use "unknown" example rejection during testing. Since methods for CDA and PDA are not designed with "unknown" example rejection, we reject samples using the entropy of classifier's output. The second is the comparison with methods tailored for each setting. In addition to the 5 baselines, we report published state-of-the-art results on each setting if available. Note that the universal results should not be directly compared with the methods tailored for each setting, as they are optimized for each setting with prior knowledge and do not have "unknown" example rejection in CDA and PDA.

Universal comparison baselines: Source-only (SO). The model is trained with source examples without using target samples. By comparing to it, we can see how much gain we can obtain by performing adaptation. **Closed-set DA (CDA)**. Since this is the most popular setting of domain adaptation, we employ DANN [11], a standard approach of feature distribution matching between

Table 4: Results on open-set domain adaptation.

| | Office(10 / 0 / 11) | | | | | | | Office-Home(15 / 0 / 50) | | | | | | | | | | | | | VisDA |
Method	A2W	D2W	W2D	A2D	D2A	W2A	Avg	A2C	A2P	A2R	C2A	C2P	C2R	P2A	P2C	P2R	R2A	R2C	R2P	Avg	(6 / 0 / 6)
SO	83.8	95.3	95.3	89.6	85.6	84.9	89.1	55.1	79.8	87.2	61.8	66.2	76.6	63.9	48.5	82.4	75.5	53.7	84.2	69.6	43.3
DANN [12]	87.6	90.5	91.2	88.7	87.4	87.0	88.7	62.1	78.0	86.4	75.5	72.0	79.3	68.8	52.5	82.7	76.1	58.0	82.7	72.8	48.2
ETN [3]	86.7	90.0	90.1	89.1	86.7	86.6	88.2	58.2	79.9	85.5	67.7	70.9	79.6	66.2	54.8	81.2	76.8	60.7	81.7	71.9	51.7
STA [24]	91.7	94.4	94.8	90.9	87.3	80.6	89.9	56.6	74.7	86.5	65.7	69.7	77.3	63.4	47.8	81.0	73.6	57.1	78.8	69.3	51.7
UAN [44]	88.0	95.8	94.8	88.1	89.9	89.4	91.0	63.3	82.4	86.3	75.3	76.2	82.0	69.4	58.2	83.4	76.1	60.5	81.9	74.6	50.0
DANCE	93.6	97.0	97.1	95.7	91.0	90.3	94.1	64.1	84.1	88.3	76.7	80.7	84.9	77.6	62.7	85.4	80.8	65.1	87.1	78.1	65.3

(a) Source only   (b) DANN   (c) DANCE w/o NC   (d) DANCE
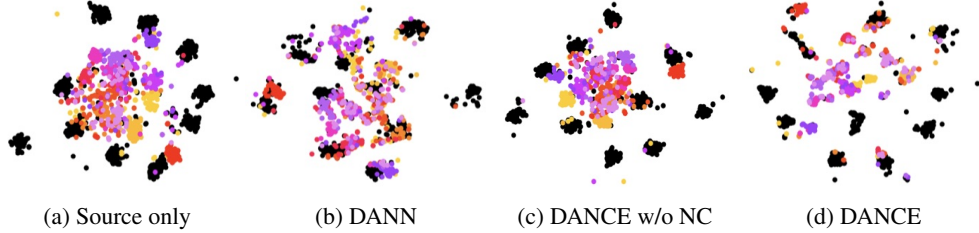
Figure 3: t-SNE [27] plots of target examples (Best viewed in color). Black dots are target "known" examples while other colors are "unknown" examples. The colors indicate different classes. DANCE extracts discriminative features for "known" examples while keeping "unknown" examples far from "known" examples. Although we do not give supervision on "unknown" classes, some of them are clustered correctly.

domains. **Partial DA (PDA)**. ENT [3] is the state-of-the-art method in PDA. This method utilizes the importance weighting on source samples with adversarial learning. **Open-set DA (ODA)**. STA [24] tries to align target "known" examples as well as rejecting "unknown" samples. This method assumes that there is a particular number of "unknown" samples and rejects them as "unknown". **Open-Partial DA (OPDA)**. UAN [44] tries to incorporate the value of entropy to reject "unknown" examples.

4.2 Results

Overview (Table 1). As seen in Table 1, which summarizes the universal comparison, DANCE is the only method which improves the performance compared to SO, a model trained without adaptation, in all settings. In addition, DANCE performs the best on open set and open-partial adaptation in all settings and the partial and closed domain adaptation setting for OfficeHome and VisDA. Our average performance is much better than other baselines with respect to both accuracy and rank. Due to a limited space, we put the result of OPDA in supplemental material. **CDA (Table 2)**. DANCE significantly improves performance compared to the source-only model (SO), and shows comparable performance to some of the baseline methods. Even compared to methods specialized in this setting, DANCE shows superior performance in OfficeHome. Some baselines show better performance in Office. However, such methods designed for CDA fail in adaptation when there are "unknown" examples. **PDA (Table 3)**. DANCE significantly improves accuracy compared to SO and achieves a comparable performance to ETN, which is one of the state-of-the-art methods in PDA. Although ETN in the universal comparison shows better performance than DANCE in Office, it does not perform well on ODA and OPDA. In the case of VisDA and OfficeHome, DANCE outperforms all baselines. **ODA (Table 4)**. DANCE outperforms all the other baselines including ones tailored for ODA. STA and UAN are designed for the ODA and OPDA achieve decent performance on these settings but show poor performance on some settings in CDA and PDA. One reason is that their method assumes that there there is a particular number of "unknown" examples in the target domain and reject them as "unknown".

Feature Visualization. Fig. 3 shows the target feature visualization with t-SNE [27]. We use the ODA setting of "DSLR to Amazon" on Office. The target "known" features (black plots) are well clustered with DANCE. In addition, most of the "unknown" features (the other colors) are kept far from "known" features and "unknown" examples in the same class are clustered together. Although we do not give supervision on the "unknown" classes, similar examples are clustered together. The visualization supports the results of the clustering performance (see below).

Ablation by clustering "unknown" examples. Here, we evaluate how well the learned features can cluster samples from both "known" and "unknown" classes in the ODA setting. The goal of the ODA is to classify samples from "unknown" classes into a single class, i.e. "unknown". Here, we evaluate the ability to cluster "unknown" classes into their original classes. Then, we train a new linear classifier on top of the fixed learned feature extractor. We use one labeled example per

Table 5: Linear classification accuracy given 1 labeled target sample per class in open-set setting (Known Accuracy/ Novel Accuracy). NC provides better clustered features for both known and novel classes. Adding entropy separation loss (DANCE) further improves the performance.

Method	R2A known / novel	R2C known / novel	R2P known / novel	P2A known / novel	P2C known / novel	P2R known / novel
ImgNet	37.5 / 31.0	35.3 / 36.4	64.8 / 56.9	36.9 / 31.0	36.3 / 36.0	66.3 / 45.5
SO	42.4 / 30.7	43.4 / 33.8	69.9 / 53.8	38.6 / 30.1	37.0 / 32.2	65.1 / 39.1
DANN	41.3 / 30.2	42.4 / 33.4	62.8 / 50.7	41.6 / 28.9	40.1 / 31.6	67.2 / 38.8
NC	48.4 / **33.9**	47.8 / **36.6**	**74.9** / 56.6	45.6 / 33.3	42.5 / 37.5	**74.7 / 45.6**
DANCE	**49.1** / 33.8	**48.7** / 36.5	**74.9 / 57.9**	**46.4 / 35.2**	**43.0 / 38.1**	74.1 / 45.2

(a) Learning curve (b) Accuracy on all (c) AUC of "unknown" (d) Accuracy on all
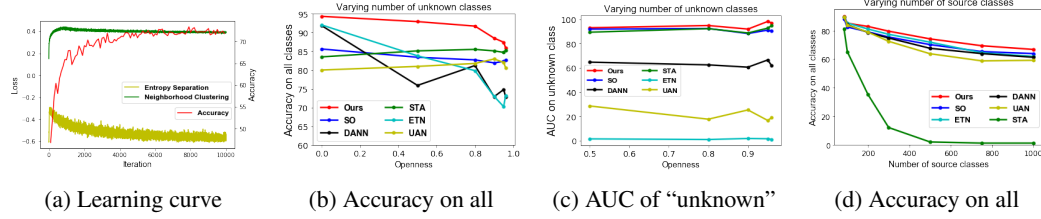
Figure 4: (a): Learning curve in the VisDA PDA setting. The accuracy improves as the two losses decrease. (b)(c): Increasing number of "unknown" classes. (d): Increasing number of source classes. DANCE outperforms than other methods even when there are many outlier source classes or many outlier target classes.

category for training. Then, we evaluate the classification accuracy for both "known" and "unknown". Since the feature extractor is fixed, we can evaluate its own ability to cluster the samples. In this experiment, we use OfficeHome (15 "known" and 50 "unknown" classes). As we can see in Table 5, DANCE maintain or improve performance for "unknown" classes compared to ImageNet model while the baseline methods much worsen performance to classify "unknown" classes. NC provides well-clustered features for both types of classes and adding entropy separation further improves the performance. The result also shows the effectiveness of our method for class-incremental domain adaptation [22]. This result and the feature visualization indicate that the features learned by DANCE are better for clustering samples from "unknown" classes.

The number of "unknown" classes. We analyze the behavior of DANCE under the different the number of "unknown" classes. In this analysis, we use open set adaptation from Amazon in Office to Caltech, where there are 10 shared classes and many unshared classes. Openness is defined as $1 - \frac{|L_s \cap L_t|}{|L_t - L_s|}$. $L_s \cap L_t$ corresponds to the shared 10 categories. We increased the number of "unknown" categories, i.e. $|L_t - L_s|$. Fig. 4b shows the accuracy of all classes whereas Fig. 4c shows area under the receiver operating characteristic curve on "unknown" classes. As we add more "unknown" classes, the performance of all methods decreases. However, DANCE consistently performs better than other methods and is robust to the number of "unknown" classes.

The number of source private classes. We analyze the behavior under the different the number of source private classes in the OPDA setting. We vary the number of classes present only in the source (i.e., $|L_s - L_t|$). To conduct an extensive analysis, we use ImageNet-1K [8] as the source domain and Caltech-256 as the target domain. They have 84 shared classes. We use all of the unshared classes of Caltech as "unknown" target while we increase the number of the classes of ImageNet (i.e., $|L_s - L_t|$). The result is shown in Fig. 4d. As we have more unshared source classes, the performance degrades as seen in Fig. 4d. DANCE consistently shows better performance. Since STA just tries to classify almost all target examples as "unknown," the performance is significantly worse.

5 Conclusion

In this paper, we introduce Domain Adaptive Neighborhood Clustering via Entropy optimization (DANCE) which performs well on universal domain adaptation. We propose two novel self-supervision based components: neighborhood clustering and entropy separation which can handle arbitrary category shift. DANCE is the only model which outperforms the source-only model in all settings and the state-of-the-art baselines in many settings. In addition, we show that DANCE extracts discriminative feature representations for "unknown" class examples without any supervision on the target domain.

Broader Impact

Our work is applicable to training deep neural networks with less supervision via knowledge transfer from auxiliary datasets. Modern deep networks outperform humans on many datasets given a lot of annotated data, such as in ImageNet. Our proposed method can help reduce the burden of collecting large-scale supervised data in many applications where large related datasets are available. The positive impact of our work is to reduce the data gathering effort for data-expensive applications. This can make the technology more accessible for institutions and individuals that do not have rich resources. It can also help applications where data is protected by privacy laws and is therefore difficult to gather, or in sim2real applications where simulated data is easy to create but real data is difficult to collect. The negative impacts could be to make these systems more accessible to companies, governments or individuals that attempt to use them for criminal activities such as fraud. Furthermore, As with all current deep learning systems, ours is susceptible to adversarial attacks and lack of interpretability. Finally, while we show improved performance relative to state-of-the-art, negative transfer could still occur, therefore our approach should not be used in mission-critical applications or to make important decisions without human oversight.

6   Acknowledgement

References

[1] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[2] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2018.

[3] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] Fabio Maria Cariucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[5] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision (ECCV)*, 2018.

[7] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

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

[10] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.

[11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning (ICML)*, 2014.

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

[13] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2005.

[14] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. *California Institute of Technology*, 2007.

[15] Philip Haeusser, Thomas Frerix, Alexander Mordvintsev, and Daniel Cremers. Associative domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

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

[19] Jiabo Huang, Qi Dong, Shaogang Gong, and Xiatian Zhu. Unsupervised deep learning by neighbourhood discovery. In *International Conference on Machine Learning (ICML)*, 2019.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[21] Jogendra Nath Kundu, Naveen Venkat, and R Venkatesh Babu. Universal source-free domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[22] Jogendra Nath Kundu, Rahul Mysore Venkatesh, Naveen Venkat, Ambareesh Revanur, and R Venkatesh Babu. Class-incremental domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2020.

[23] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[24] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[25] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning (ICML)*, 2015.

[26] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

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

[28] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[29] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[30] 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. *Openreview*, 2017.

[31] 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.

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[33] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision (ECCV)*, 2010.

[34] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[35] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, 2017.

[36] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *European Conference on Computer Vision (ECCV)*, 2018.

[37] Ozan Sener, Hyun Oh Song, Ashutosh Saxena, and Silvio Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

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

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

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

[41] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[42] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[43] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[44] Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Universal domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[45] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[46] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael I Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning (ICML)*, 2019.

[47] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[48] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *European Conference on Computer Vision (ECCV)*, 2018.

A Dataset Detail

In PDA, 10 classes in Caltech-256 are used as shared classes $(L_s \cap L_t)$. The other 21 classes are used as source private classes $(L_s - L_t)$. Since DSLR and Webcam do not have many examples, we conduct experiments on D to A, W to A, A to C (Caltech), D to C, and W to C shifts. In OSDA, the same 10 classes are used as shared classes $(L_s \cap L_t)$ and the selected 11 classes are used as unknown classes $(L_t - L_s)$. The setting is the same as [36]. In OPDA, the same 10 class are used as shared classes $(L_s \cap L_t)$ and then, in alphabetical order, the next 10 classes are used as source private classes $(L_s - L_t)$, and the remaining 11 classes are used as unknown classes $(L_t - L_s)$. The second benchmark dataset is **OfficeHome (OH)** [41], which contains four domains and 65 classes. In PDA, in alphabetical order, the first 25 classes are selected as shared classes $(L_s \cap L_t)$ and the rest classes are source private classes $(L_s - L_t)$. In OSDA, the first 15 classes are used as shared classes $(L_s \cap L_t)$ and the rest classes are used as unknown classes $(L_t - L_s)$. In OPDA, the first 10 classes are used as shared classes $(L_s \cap L_t)$, the next 5 classes are source private classes $(L_s - L_t)$ and the rest are unknown classes $(L_t - L_s)$. The third dataset is **VisDA** [31], which contains 12 classes from the two domains, synthetic and real images. The synthetic domain consists of 152,397 synthetic 2D renderings of 3D objects and the real domain consists of 55,388 real images. In PDA, the first 6 classes are used as shared classes $(L_s \cap L_t)$ and the rest are source private classes $(L_s - L_t)$. In OSDA, we follow [36] and use the 6 classes as shared classes $|L_s \cap L_t|$ and the rest as unknown classes $(L_t - L_t)$. In OPDA, the first 6 classes are shared classes $(L_s \cap L_t)$, the next 3 are source private classes $(L_s - L_t)$ and the other 3 classes are unknown classes $(L_t - L_s)$. We mainly perform experiments on these three datasets with four settings because it enables direct comparison with many state-of-the-art results. We provide an analysis of varying the number of classes using Caltech [14] and ImageNet [8] because these datasets contain a large number of classes.

B Implementation Detail

We list the implementation details which are excluded from the main paper due to a limit of space. We used TITAN X (Pascal) with 12GB. One GPU is used for each experiment and each experiment takes about 2 hours.

DANCE (universal comparison). The batch-size is set as 36. The temperature parameter in Eq. 5 is set as 0.05 by following [34]. We train a model for 10,000 iterations with nestrov momentum SGD and report the performance at the end of the iterations. The initial learning rate is set as 0.01, which is decayed with the factor of $(1 + \gamma \frac{i}{10,000})^{-p}$, where i denotes the number of iterations and we set $\gamma = 10$ and $p = 0.75$. The learning rate of pre-trained layers is multiplied by 0.1. We follow [34] for this scheduling method.

Baselines (universal comparison). We use the following released codes for ETN [3](https://github.com/thuml/ETN), UAN [44](https://github.com/thuml/Universal-Domain-Adaptation), and STA [24](https://github.com/thuml/Separate_to_Adapt). We tune the hyper-parameter of these methods by validating the performance on OPDA, Amazon to DSLR, Office. Since we could not see improvements by changing the hyper-parameters from their codes, we employed the hyper-parameters provided in their codes. For ETN, we use the hyper-parameters for Office-Home. For UAN and STA, we use the hyper-parameters for Office. We implement DANN by ourselves and tuned the hyper-parameters by the performance on OPDA, Amazon to DSLR, Office. For all of these methods, we report the performance at the end of training for comparison. We observe that there is a gap in the performance between the best checkpoint and the final checkpoint. This can explain the gap between the reported performance in their paper and the performance in our universal comparisons.

Baselines tailored for each category shift. We run experiments for ETN (A2C, W2C, D2C, PDA) since the results are not available in their papers. For ETN, we report the performance which employs the same hyper-parameters as the universal comparison but does not use "unknown" sample rejection. For the methods tailored for each setting, we show the performance of the results reported in their papers. "NA" indicates the results are not available in their paper. We observe the performance gap in our universal comparison and the reported performance in each paper. For example, the performance of UAN in OPDA has a big gap between the universal comparison and the reported accuracy although we use the same hyper-parameters. We could obtain similar performance to the reported number if

Table A: Results on open-partial domain adaptation. USFDA [21] focuses on open-partial domain adaptation without access to source samples in adapting a model to a target domain. The number of UAN [44] in a lower row is taken from their paper.

	Universal comparison																				
Method	Office(10 / 10 / 11)							Office-Home(10 / 5 / 50)													VisDA
	A2W	D2W	W2D	A2D	D2A	W2A	Avg	A2C	A2P	A2R	C2A	C2P	C2R	P2A	P2C	P2R	R2A	R2C	R2P	Avg	(6 / 3 / 3)
SO	75.7	95.4	95.2	83.4	84.1	84.8	86.4	50.4	79.4	90.8	64.9	66.1	79.9	71.6	48.5	87.6	77.8	52.1	82.8	71.0	38.8
DANN [12]	87.6	90.5	91.2	88.7	87.4	87.0	88.7	59.9	80.6	89.8	77.5	73.3	86.4	78.5	61.5	88.5	80.3	62.1	82.4	76.7	50.6
ETN [3]	89.1	90.6	90.9	86.3	86.4	86.5	88.3	58.2	78.5	89.1	77.2	69.3	87.5	77.0	56.0	88.2	77.5	58.4	83.0	75.0	66.6
STA [24]	85.2	96.3	95.1	88.1	87.9	86.0	89.8	54.8	76.6	**91.2**	71.5	71.8	82.0	70.7	50.1	88.2	74.1	60.0	80.5	72.6	47.4
UAN [44]	76.2	82.0	80.4	80.0	93.8	92.2	84.1	60.8	79.1	87.8	72.4	73.5	83.2	78.6	56.4	87.4	79.9	61.1	79.8	75.0	47.3
DANCE	**92.8**	**97.8**	**97.7**	**91.6**	**92.2**	**91.4**	**93.9**	**64.1**	**84.3**	91.2	**84.3**	**78.3**	**89.4**	**83.4**	**63.6**	**91.4**	**83.3**	**63.9**	**86.9**	**80.4**	**69.2**
	Methods tailored for Open-Partial Domain Adaptation																				
UAN [44]	85.6	94.8	98.0	86.5	85.5	85.1	89.2	63.0	82.8	87.9	76.9	78.7	85.4	78.2	58.6	86.8	83.4	63.2	79.4	77.0	60.8
USFDA [21]	85.6	95.2	97.8	88.5	87.5	86.6	90.2	63.4	83.3	89.4	71.0	72.3	86.1	78.5	60.2	87.4	81.6	63.2	88.2	77.0	63.9

Table B: Evaluation on two metrics on open-set and open partial domain adaptation. OS is average of all classes. OS* is the average of known classes.

	Open Set											
Method	A to W		D to W		W to D		A to D		D to A		W to A	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
SO	83.8	87.7	95.3	99.0	95.3	100.0	89.6	93.6	85.6	86.3	84.9	88.2
DANN	87.6	95.7	90.5	99.3	91.2	100.0	88.7	96.9	87.4	**95.4**	87.0	95.2
ETN	86.7	95.4	90.0	99.0	90.1	99.1	89.1	98.0	86.7	95.3	86.6	95.3
STA	91.7	94.6	94.4	98.1	94.8	100.0	90.9	94.2	87.3	88.8	80.6	82.4
UAN	86.2	94.6	89.5	98.5	90.2	99.2	89.8	98.7	85.8	94.4	84.2	92.7
DANCE	**93.6**	**97.2**	**97.0**	**100.0**	**97.1**	**100.0**	**95.7**	**98.4**	**91.0**	94.9	**90.3**	**95.6**
	Open Partial											
Method	A to W		D to W		W to D		A to D		D to A		W to A	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
SO	75.7	79.2	95.4	98.1	95.2	100.0	83.4	88.3	84.1	84.9	84.8	85.7
DANN	83.0	90.0	89.3	97.1	89.5	96.9	81.9	88.8	80.2	86.8	78.2	77.8
ETN	89.1	**98.0**	90.6	**99.7**	90.9	**100.0**	86.3	**94.9**	86.4	**95.0**	86.5	**95.1**
STA	85.2	87.8	96.3	99.1	95.1	100.0	88.1	90.6	87.9	88.7	86.0	87.1
UAN	78.8	86.7	83.5	91.9	84.9	93.4	77.5	85.3	75.7	83.3	75.6	83.1
DANCE	**92.8**	96.4	**97.8**	99.1	**97.7**	99.6	**91.6**	94.0	**92.2**	94.5	**91.4**	94.7

we pick up the best checkpoint for each setting. But, we report the performance of fixed iterations' checkpoints for a fair comparison, which can explain the gap.

Table C: Standard deviation of DANCE in experiments on Office and VisDA. The deviation is calculated by three runs. DANCE shows descent deviations.

Setting	A2W	D2W	W2D	A2D	D2A	W2A	Avg	VisDA
CDA	88.6±0.4	97.5±0.4	100±0.0	89.4±1.3	69.5±1.5	68.2±0.0	85.5±0.2	70.2±0.3
ODA	93.6±2.3	97.0±0.2	97.1±0.5	95.7±0.3	91.0±0.8	90.3±0.2	94.1±2.5	65.3±2.3
OPDA	92.8±0.2	97.8±0.6	97.7±0.5	91.6±1.9	92.2±0.1	91.4±0.4	93.9±0.3	69.2±0.6

| Setting	A2C	W2C	D2C	D2A	W2A	Avg	VisDA
PDA	88.8±0.4	79.2±0.3	79.4±0.3	83.7±3.3	92.6±0.5	84.8±1.5	73.7±2.9

C Supplemental Results

Detailed results of ODA and OPDA. Table A and Table B shows the detailed results of ODA and OPDA. OS* shows the averaged accuracy over known classes while OS shows the averaged accuracy including unknown class. DANCE shows good performance on both metrics. ETN shows better results on OS* than DANCE in several scenarios. In ETN results, OS* shows much better results on OS, which means that ETN is not good at recognizing unknown samples as unknown. This is clearly shown in Fig. 4 (c) in our main paper.

Comparison with Jigsaw [5]. Table D shows the comparison with jigsaw puzzle based self-supervised learning. To consider the self-supervised learning part of DANCE, we replaced neigh-

Table D: Comparison between jigsaw [5, 28] and DANCE on the Office dataset. For a fair comparison, we replace the loss of entropy similarity with jigsaw puzzle loss.
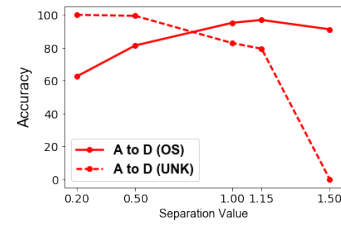
Setting	Method	A2W	D2W	W2D	A2D	D2A	W2A	Avg
Closed	Jigsaw	87.7	**98.7**	**100.0**	84.5	61.7	62.5	82.5
	DANCE	**88.6**	97.5	**100.0**	**89.4**	**69.5**	**68.2**	**85.5**
Open	Jigsaw	89.4	95.5	93.6	93.8	90.3	89.3	92.0
	DANCE	**93.6**	**97.0**	**97.1**	**95.7**	**91.0**	**90.3**	**94.1**
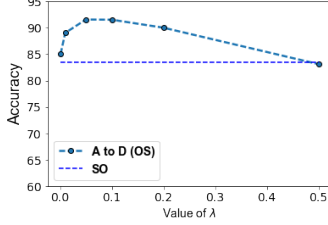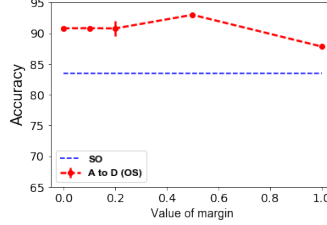
(a) Value of λ in Eq. 9. (b) Value of m in Eq. 7. (c) Value of ρ in Eq. 7.

Figure A: (a): Varying the value of λ in Eq. 9. (b): Varying the value of margin in Eq. 7. (c): Varying the value of ρ in Eq. 7, which is determined based on the number of known classes.

borhood clustering loss with the jigsaw puzzle loss on the target domain. The jigsaw puzzle loss is calculated on target samples. We can see that DANCE performed better in almost all settings and confirm the effectiveness of clustering based self-supervision for this task.

Results with standard deviations. Table C show results of DANCE with standard deviations. We show only the averaged accuracy over three runs in the main paper due to a limit of space. We show the standard deviation. We can observe that DANCE shows decent standard deviations.

Sensitivity to hyper-parameters. In Fig. A, we show the sensitivity to hyper-parameters on OPDA setting of Amazon to DSLR, which we used to tune the hyper-parameters. Although ρ in Eq. 5 is decided based on the number of source classes, we show the behavior of our method when changing it in Fig. A(c). When we increase the value, more examples will be decided as known, then the performance on unknown examples decreases.