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

Supplementary Material

1. 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:

$$\max \mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))]. \tag{1}$$

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

$$\begin{aligned}
&\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))] && (2)\\
=&\mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x})\log p(y_i|\mathbf{x})\right] \\
&\quad - \sum_{i=1}^{L} \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]\log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]] && (3)\\
=&\mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x})\log p(y_i|\mathbf{x})\right] \\
&\quad - \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x})\log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]\right] && (4)\\
=&\mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x})\log \frac{p(y_i|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]}\right] && (5)\\
=&\mathbb{E}_{\mathbf{x}}\left[\int p(y|\mathbf{x})\log \frac{p(y|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x})]}\,\mathrm{d}y\right] && (6)\\
=&\int p(\mathbf{x})\,\mathrm{d}\mathbf{x}\int p(y|\mathbf{x})\log \frac{p(y|\mathbf{x})}{\int p(\mathbf{x})p(y|\mathbf{x})\,\mathrm{d}\mathbf{x}}\,\mathrm{d}y && (7)\\
=&\int p(\mathbf{x})\,\mathrm{d}\mathbf{x}\int p(y|\mathbf{x})\log \frac{p(y|\mathbf{x})}{p(y)}\,\mathrm{d}y && (8)\\
=&\iint p(y,\mathbf{x})\log \frac{p(y,\mathbf{x})}{p(y)p(\mathbf{x})}\,\mathrm{d}y\,\mathrm{d}\mathbf{x} \;=\; \mathcal{I}(y;\mathbf{x}) && (9)
\end{aligned}$$

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 $p(y|\mathbf{x};\theta)$.

2. Additional Datasets Details

Overall statistics of the datasets and the number of labeled source examples used in our experiments can be found in Table 1. For Office [10], Office-Home [12] and VisDA [9], we follow the same setting in [3], randomly sampling labeled images from the source domain and ensure that each class has at least one labeled example. For DomainNet [8], we use the same split files as [11] and further select 1-shot and 3-shots labeled samples in the training set for each class.

3. Additional Implementation Details

We implemented our model in PyTorch [7]. 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 [4]. 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.

Table 1: Dataset statistics and labeled source used

<table>
<tr><th>Dataset</th><th>Domain</th><th># total image</th><th># labeled images</th><th># classes</th></tr>
<tr><td rowspan="3">Office [10]</td><td>Amazon (A)</td><td>2817</td><td rowspan="3">1-shot and 3-shots labeled source</td><td rowspan="3">31</td></tr>
<tr><td>DSLR (D)</td><td>498</td></tr>
<tr><td>Webcam (W)</td><td>795</td></tr>
<tr><td rowspan="4">Office-Home [12]</td><td>Art (Ar)</td><td>2427</td><td rowspan="4">3% and 6% labeled source</td><td rowspan="4">65</td></tr>
<tr><td>Clipart (Cl)</td><td>4365</td></tr>
<tr><td>Product (Pr)</td><td>4439</td></tr>
<tr><td>Real (Rw)</td><td>4357</td></tr>
<tr><td rowspan="2">VisDA [9]</td><td>Synthetic (Syn)</td><td>152K</td><td rowspan="2">0.1% and 1% labeled source</td><td rowspan="2">12</td></tr>
<tr><td>Real (Rw)</td><td>55K</td></tr>
<tr><td rowspan="4">DomainNet [8]</td><td>Clipart (C)</td><td>18703</td><td rowspan="4">1-shot and 3-shots labeled source</td><td rowspan="4">126</td></tr>
<tr><td>Painting (P)</td><td>31502</td></tr>
<tr><td>Real (R)</td><td>70358</td></tr>
<tr><td>Sketch (S)</td><td>24582</td></tr>
</table>

Table 2: Accuracy of cross-domain weighted kNN with different SSL methods.

Method	D→A	Rw→Cl
ImageNet pre-train	62.5	40.6
ID [13]	70.3	51.9
CDS [3]	72.5	53.7
protoNCE [4]	72.3	49.3
$\mathcal{L}_{\text{InSelf}} + \mathcal{L}_{\text{CrossSelf}}$	**75.5**	**55.3**

Table 3: Accuracy of cross-domain weighted kNN with different FUDA methods.

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

4. Quantitative Feature Analysis

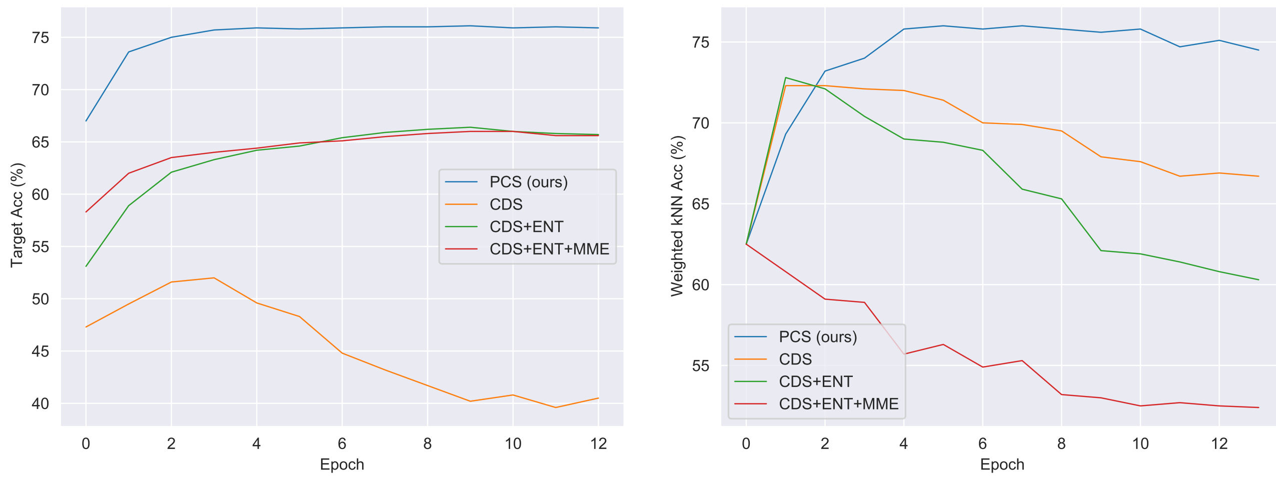
To quantitatively compare the quality of learned features with different approaches, we perform classification with weighted k-nearest neighbor (kNN) classifier proposed by Wu *et al*. [13] in a cross-domain manner. Specifically, given a test image $\mathbf{x}^t$, we first compute its normalized feature $\mathbf{f}^t = F(\mathbf{x}^t)$, 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 [13].

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 2 and Table 3, 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.

5. Stability Analysis of PCS

To show the performance stability of PCS, we conduct multiple runs with three different random seeds. Table 4 reports the averaged accuracy and standard deviation of the three runs on the 1-shot and 3-shots settings of Office.

Figure 1 shows adaptation accuracy vs. training epochs using cosine classifier (Figure 1a) and weighted kNN classifier (Figure 1b). From the plots, we have the following observations. (1) The target accuracy of PCS increases more steadily and robustly compared to other methods. In Figure 1a, CDS starts decreasing at Epoch 3. In Figure 1b, CDS and CDS+ENT starts decreasing at Epoch 1; while CDS+ENT+MME decreases from the beginning of training. In contrast, the performance of PCS increases smoothly until the end of training. (2) PCS converges much faster than other methods. We can see in Figure 1a that PCS plateaus at around Epoch 3, while CDS+ENT and



(a) Target Acc. with cosine classifier vs. training epochs

(b) Target Acc. with Weighted kNN vs. training epochs

Figure 1: Stability of Target Accuracy during training procedure.

Table 4: 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 5: 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 [3]	0.43	-0.71
PCS w/o APCU	-53.3	-22.8
PCS (Ours)	**-58.4**	**-26.5**

CDS+ENT+MME reaches the best performance at Epoch 9 and 10.

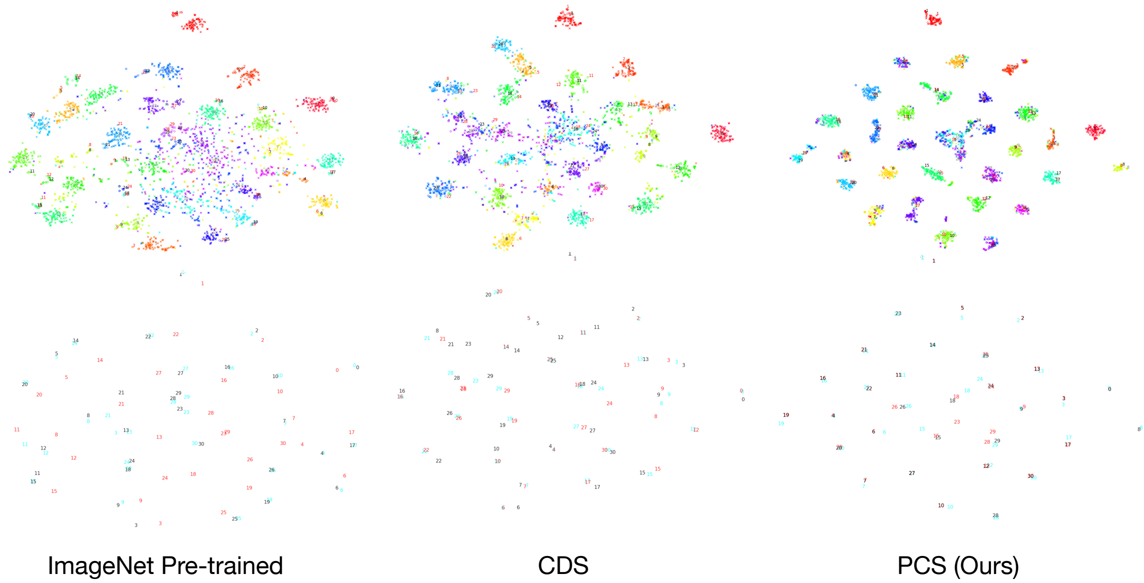
6. Prototype Quality Comparison

To further compare how well source and target are aligned, we provide more t-SNE [6] visualizations on Office (D→A) and Office-Home (Rw→Cl) in Figure 2a and 2b, comparing ImageNet Pre-training, CDS [3] 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, best view after zooming in). 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 2, for each class, the prototypes of source, target and the weight vector of classifier get more aggregated with PCS than other methods, which demonstrates that PCS could better align source and target representations for each category.

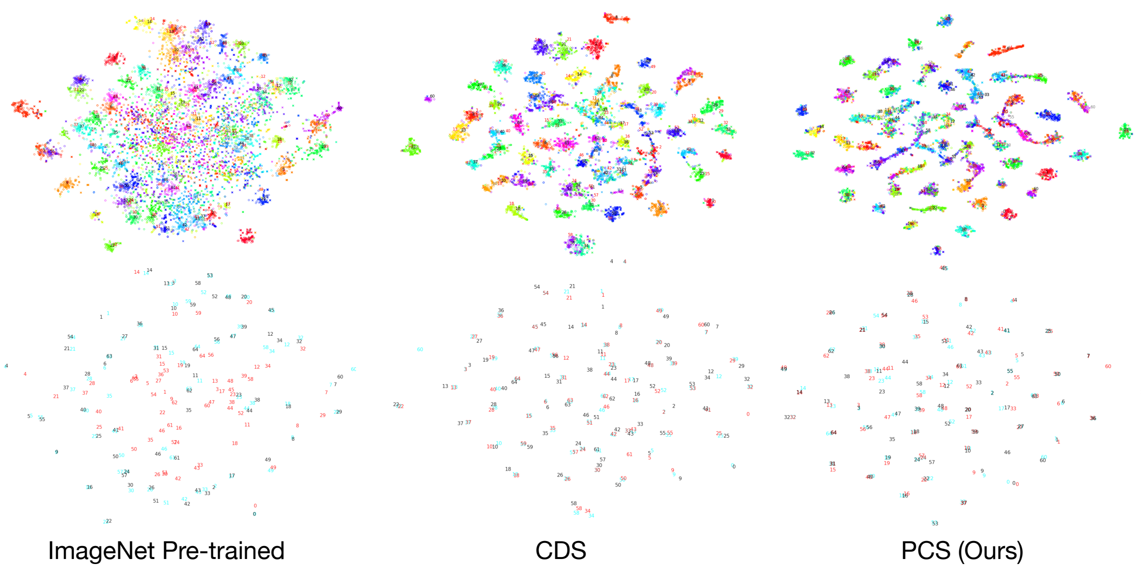
In a well-learned feature embedding space, prototypes of different classes should be far / different from each other. To quantitatively measure the similarity of the learned prototypes, we compute the sum of cosine similarities between all pairs of prototypes. From the results shown in Table 5, we can see that the prototypes learned with PCS have the least similarities, indicating that PCS learns an embedding space with better semantic structure.

7. Image Retrieval Results

We present cross-domain image retrieval results in Figure 3. 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 3. As shown in Figure 3, features



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



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

Figure 2: 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).

from model trained with CDS are biased to some wrong attributes, *e.g.* color, texture and other visual clues; and quantitatively similar features do not correspond to semantically similar images in different domains. In contrast, we can see that PCS could extract features that are more discriminative and semantically meaningful across domains.

8. 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 com-

Figure 3: Image retrieval examples of the closest cross-domain neighbors using CDS (a) and PCS (b) in Office-Home (Target: Real, Source: Art).

Table 6: 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 [1]	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 [5]	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 [2]	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 [11]	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 [3]	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 7: 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 [1]	79.7	82	68.2	96.9	67.4	99.1	82.2
CDAN [5]	92.9	94.1	71	98.6	69.3	100	87.7
MMDIA [2]	92.1	90.3	75.3	98.7	74.9	99.8	88.8
MME [11]	88.8	87.3	69.2	98.7	65.6	100	84.9
CDS + MME [3]	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**

parison with other UDA methods on Office and Office-Home are presented in Table 7 and Table 6, respectively. We can see that PCS achieves the best results even with fully-labeled source, which demonstrates that the proposed PCS could potentially be applied to a wider range of domain adaptation settings.

9. More Ablation Study Results

In this section, we provide more ablation study results. Ablation experiments similar to Table 2 in the main paper are performed on Office-Home, with results shown in Table 8. As we can see in the table, adding each component contributes to the final adaptation accuracy without any performance degradation, which demonstrates the effectiveness of all components in our PCS framework.

Table 8: 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**

References

[1] 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. 5

[2] 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. 5

[3] 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. 1, 2, 3, 5

[4] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020. 1, 2

[5] 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. 5

[6] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 3

[7] 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. 1

[8] 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. 1, 2

[9] 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. 1, 2

[10] 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. 1, 2

[11] 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. 1, 5

[12] 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. 1, 2

[13] 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