Supplementary Materials for "Prototype-Guided Continual Adaptation for Class-Incremental Unsupervised Domain Adaptation"

Hongbin Lin[1,3*], Yifan Zhang[2*], Zhen Qiu[1*], Shuaicheng Niu[1], Chuang Gan[4], Yanxia Liu[1†], and Mingkui Tan[1,5†]

[1]South China University of Technology [2] National University of Singapore
[3] Information Technology R&D Innovation Center of Peking University
[4] MIT-IBM Watson AI Lab
[5] Key Laboratory of Big Data and Intelligent Robot, Ministry of Education
{sehongbinlin,seqiuzhen,sensc}@mail.scut.edu.cn,
yifan.zhang@u.nus.edu, ganchuang1990@gmail.com,
{cslyx,mingkuitan}@scut.edu.cn

In this supplementary, we provide more related work and discussions to clarify the differences of ProCA with existing methods. In addition, we also provide more implementation details and more experimental results. We organize the supplementary materials as follows.

1) In Appendix A, we review universal domain adaptation [18] and give more discussions on partial domain adaptation and prototype-based methods.
2) In Appendix B, we present the pseudo-code of our prototype identification scheme.
3) In Appendix C, we provide more construction details of class-incremental domain adaptation data.
4) In Appendix D, we provide more training details of our proposed ProCA.
5) In Appendix E, we provide more results of applying ProCA to improve the PDA method.
6) In Appendix F, we provide more ablation studies, including the influence of hyper-parameters, the number of target prototypes and the number of incremental classes.
7) In Appendix G, we examine the effectiveness of our shared class detection strategy.
8) In Appendix H, we provide more detailed results of *each subtask* in the three datasets in terms of the Step-level Accuracy and the Final S-1 Accuracy.

A More related work and discussions

In this appendix, we first review the literature of universal domain adaptation. After that, to better illustrate our novelty, we discuss the differences between

* Authors contributed equally.
† Corresponding authors.

CI-UDA and two types of relevant methods, *i.e.,* PDA methods and prototype-based methods.

Universal domain adaptation (Uni-DA). Uni-DA [18] assumes that the target label space is not limited to the source label space and may contain target private (unknown) classes. It seeks to classify unlabeled target samples into known classes from the source label space or an additional "unknown" category. With various transferability measures, most existing methods conduct domain alignment by quantifying sample-level transferability [4,18]. In addition, to exploit the structure information, DANCE [17] proposes to learn the structure of the target domain in a self-supervised way, while DCC [10] seeks to better exploit the intrinsic structure of the target domain and discover discriminative clusters. However, existing Uni-DA methods assume all target data are available in advance, making them incapable in CI-UDA.

Table I. The shared class indexes of different detection strategies at each time step on **Office-31-CI**. Note that correct shared classes are in blue while false shared classes are in magenta. Note that the higher SCD Acc. means the strategy detects more shared classes, and the higher TCD Acc. means the strategy detects less false shared classes.

Task	Method	Time Step	Shared Class Index	SCD Acc.	TCD Acc.	Avg.
A→D	HBW [8]	Step 1	[0, 1, 2, 9]	40.0	100.0	70.0
		Step 2	[0, 3, 4, 5, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 30]	100.0	40.0	70.0
		Step 3	[0, 5, 7, 9, 10, 12, 13, 16, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]	100.0	47.6	73.8
	Ours	Step 1	[0, 1, 2, 3, 4, 5, 6, 7, 9, 12]	90.0	90.0	**90.0**
		Step 2	[8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	100.0	90.9	**95.5**
		Step 3	[12, 13, 20, 21, 22, 23, 25, 26, 27, 28, 29]	90.0	81.8	**85.9**
A→W	HBW [8]	Step 1	[0, 1, 2, 6, 9]	50.0	100.0	75.0
		Step 2	[0, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]	100.0	34.5	67.3
		Step 3	[0, 7, 9, 12, 13, 16, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]	100.0	55.6	77.8
	Ours	Step 1	[0, 1, 2, 3, 4, 5, 6, 7, 9, 12]	90.0	90.0	**90.0**
		Step 2	[10, 11, 12, 13, 15, 16, 17, 18, 19, 27]	90.0	90.0	**90.0**
		Step 3	[12, 13, 20, 21, 22, 23, 24, 26, 27, 28, 29]	90.0	81.8	**85.9**
D→A	HBW [8]	Step 1	[0, 1, 6]	30.0	100.0	65.0
		Step 2	[0, 11, 12, 13, 14, 15, 16, 17, 19, 27,29]	80.0	66.7	73.4
		Step 3	[0, 14, 20, 21, 22, 23, 26, 27, 29]	70.0	77.8	**73.9**
	Ours	Step 1	[0, 1, 2, 3, 5, 6, 7, 9, 14]	80.0	88.9	**84.5**
		Step 2	[11, 12, 13, 14, 15, 16, 17, 19, 27]	80.0	88.9	**84.5**
		Step 3	[0, 14, 20, 21, 22, 23, 26, 27, 29]	70.0	77.8	**73.9**
D→W	HBW [8]	Step 1	[0, 2, 6]	30.0	100.0	65.0
		Step 2	[0, 6, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 22, 24, 25]	100.0	66.7	83.4
		Step 3	[0, 12, 15, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]	100.0	71.4	85.7
	Ours	Step 1	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]	100.0	100.0	**100.0**
		Step 2	[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	100.0	100.0	**100.0**
		Step 3	[20, 21, 22, 23, 24, 26, 27, 28, 29, 30]	90.0	90.0	**90.0**
W→A	HBW [8]	Step 1	[0, 1, 2, 5, 6, 9]	60.0	100.0	**80.0**
		Step 2	[0, 2, 4, 5, 7, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 23, 24, 26, 27, 29]	100.0	47.6	73.8
		Step 3	[0, 4, 5, 11, 13, 14, 16, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 29]	90.0	50.0	70.0
	Ours	Step 1	[0, 1, 2, 3, 5, 6, 7, 9, 11, 27, 29]	80.0	72.7	76.4
		Step 2	[10, 11, 12, 14, 15, 16, 17, 18, 19, 27]	90.0	90.0	**90.0**
		Step 3	[11, 14, 16, 18, 20, 21, 22, 23, 24, 26, 27, 29]	80.0	66.7	**73.4**
W→D	HBW [8]	Step 1	[0, 1, 2, 4, 6, 7, 8, 9]	80.0	100.0	90.0
		Step 2	[7, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	100.0	90.9	95.5
		Step 3	[20, 21, 22, 23, 26, 27, 29]	70.0	100.0	85.0
	Ours	Step 1	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]	100.0	100.0	**100.0**
		Step 2	[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	100.0	100.0	**100.0**
		Step 3	[20, 21, 22, 23, 24, 25, 26, 27, 28, 29]	100.0	100.0	**100.0**

Relations to partial domain adaptation methods. PDA [1] assumes that the target label set is a subset of the source label set, and seeks to transfer a model trained from a big labeled source domain to a small unlabeled target domain. To alleviate the negative transfer caused by source private classes, existing PDA methods [1,2,8] decrease the transferability weights of source private classes when aligning the source and target domains. To be specific, ETN [2] quantifies the *instance-level* transferability weights to all source samples. In contrast, PADA [1] and DPDAN [8] assign *class-level* transferability weights to all source classes. However, PADA directly exploits the cumulative probabilities as the transferability weights of all source classes, leading to the presence of the negative impact of source private classes in transfer. To address this, DPDAN [8] proposes the Hard Binary Weights (HBW) strategy which decomposes the source domain into two distributions (*i.e.,* source-positive and source-negative distributions). More specifically, to detect the shared classes, HBW sets the cumulative probabilities threshold by maximizing the variance of these distributions. Similar to HBW, our shared class detection strategy also alleviates the negative transfer at the class level and aims to eliminate the negative impact of source private classes.

However, HBW tends to fail in CI-UDA since the target label space is inconsistent between steps, *i.e.,* the shared classes between the source and the target are inconsistent between different steps. Unfortunately, the target label space inconsistency may bring noise into the optimization of the variance in HBW, resulting in false source-positive and source-negative distributions. In contrast, our strategy sets the pre-defined cumulative probabilities threshold α to detect shared classes, which is more robust to the variation of the target label space. To verify this, we visualize the detected shared classes by HBW and our strategy in different learning steps of CI-UDA. As shown in Table I, compared with our method, the HBW strategy detects more false shared classes (*e.g.,* Step 2 of A$\rightarrow$D) and filters more shared classes out (*e.g.,* Step 1 of D$\rightarrow$W) in CI-UDA. To quantify the results of shared class detection, we use two accuracy measures: 1) ***Shared Class Detection Accuracy (SCD Acc.)***: the truly shared classes divided by the number of ground truths; and 2) ***Total Class Detection Accuracy (TCD Acc.)***: the truly detected shared classes divided by the number of all detected classes. The Experiment shows that our shared class detection strategy outperforms the HBW strategy in CI-UDA with the higher average accuracy in almost all tasks on the Office-31-CI.

Relations to prototype-based methods. Existing prototype-based methods have separately explored prototypes to conduct domain alignment [13] or prevent catastrophic forgetting [15]. However, these methods are different from our ProCA. To be specific, existing prototype-based domain adaptation methods [13,14] conduct domain alignment by aligning source feature prototypes to all target data, while ProCA aligns class-wise source centers and the feature prototypes extracted from the target label prototypes. In addition, even though we select prototypes in the same manner with iCaRL [15] for replaying knowledge, iCaRL constructs the memory bank via images and ignores updating, while ProCA constructs the prototype memory bank based on our target label prototypes and designs a novel way to update this memory bank based on the cumulative probabilities. Note that obtaining image prototypes for knowledge retaining in iCaRL [15] requires data labels but the target domain in CI-UDA is totally unlabeled. Meanwhile, feature prototypes [13,14] for domain adaptation cannot update the feature extractor, so simply detecting them is unable to overcome the knowledge forgetting issue of the feature extractor in CI-UDA. Therefore, a simple combination of existing prototype-based methods is not feasible for CI-UDA while our ProCA provides the first feasible prototype-based solution to CI-UDA (cf. Section 5 in the main paper).

B Pseudo-code of Label Prototype Identification

In this section, we present the pseudo-code of the prototype identification scheme. Specifically, we first detect shared classes in each time step and generate pseudo labels for target data. As shown in Algorithm 1, for each class k in the detected shared class set, we obtain T target label prototypes via a nearest neighbor approach.

Algorithm 1 Label Prototype identification of ProCA

Require: Pseudo-labeled target data $\mathcal{D}_t^k = \{\mathbf{x}_i^k\}_{i=1}^{n_k}$ of class k at the current time; Model G ; Hyper-parameter T.

1: Attain the k-th class feature center: $\mathbf{f}_t^k = \frac{1}{n_k}\sum_{i=1}^{n_k} G(\mathbf{x}_i^k)$;
2: **for** $m = 1 \rightarrow T$ **do**
3: $\quad \mathbf{p}_m^k = \arg\min_{\mathbf{x}^k \in \mathcal{D}_t^k} \left\| \mathbf{f}_t^k - \frac{1}{m}[G(\mathbf{x}^k) + \sum_{i=1}^{m-1} G(\mathbf{p}_i^k)] \right\|_2$;
4: **end for**
5: **return** Label prototypes of class k $\{\mathbf{p}_1^k, ..., \mathbf{p}_T^k\}$.

C Details of Data Construction

In this section, we show the containing classes in each disjoint subset of all the three benchmark datasets (*i.e.,* Office-31-CI, Office-Home-CI and ImageNet-Caltech-CI) in Tables III and II. Specifically, we choose 10 for the number of incremental classes on the three benchmark datasets. For Office-31-CI, we sort the class name in alphabetic order and group every 10 categories into a step. For Office-Home-CI, we randomly group every 10 categories into a step. As a result, each domain of Office-31-CI has 3 disjoint subsets for 3 time steps, while each domain of Office-Home-CI has 6 disjoint subsets for 6 time steps. For ImageNet-Caltech-CI, we adopt the class indexes following [16,6]. As shown in Table III, we also group every 10 categories into a time step based on the sorted class indexes. Thus, each domain of ImageNet-Caltech-CI has 8 disjoint subsets for 8 time steps. We have put the splits of three datasets into the code.

Table II. Class names in each time step on Office-31-CI and Office-Home-CI.

Dataset	Time Step	Class Index	Class Name
Office-31-CI	Step 1	[0,1,2,3,4,5,6,7,8,9]	back pack, bike, bike helmet, bookcase, bottle, calculator, desk chair, desk lamp, desktop computer, file cabinet
	Step 2	[10,11,12,13,14,15,16,17,18,19]	headphones, keyboard, laptop computer, letter tray, mobile phone, monitor, mouse, mug, paper notebook, pen
	Step 3	[20,21,22,23,24,25,26,27,28,29]	phone, printer, projector, punchers, ring binder, ruler, scissors, speaker, stapler, tape dispenser
Office-Home-CI	Step 1	[0,1,2,3,4,5,6,7,8,9]	Drill, Exit Sign, Bottle, Glasses, Computer, File Cabinet, Shelf, Toys, Sink, Laptop
	Step 2	[10,11,12,13,14,15,16,17,18,19]	Kettle, Folder, Keyboard, Flipflops, Pencil, Bed, Hammer, ToothBrush, Couch, Bike
	Step 3	[20,21,22,23,24,25,26,27,28,29]	Postit Notes, Mug, Webcam, Desk Lamp, Telephone, Helmet, Mouse, Pen, Monitor, Mop
	Step 4	[30,31,32,33,34,35,36,37,38,39]	Sneakers, Notebook, Backpack, Alarm Clock, Push Pin, Paper Clip, Batteries, Radio, Fan, Ruler
	Step 5	[40,41,42,43,44,45,46,47,48,49]	Pan, Screwdriver, Trash Can, Printer, Speaker, Eraser, Bucket, Chair, Calendar, Calculator
	Step 6	[50,51,52,53,54,55,56,57,58,59]	Flowers, Lamp Shade, Spoon, Candles, Clipboards Scissors, TV, Curtains, Fork, Soda

Table III. Class indexes in each time step on ImageNet-Caltech-CI.

Task	Time Step	Class Index
I→C	Step 1	[1, 9, 24, 39, 51, 69, 71, 79, 94, 99]
	Step 2	[112, 113, 145, 148, 171, 288, 308, 311, 314, 315]
	Step 3	[327, 334, 340, 354, 355, 361, 366, 367, 413, 414]
	Step 4	[417, 435, 441, 447, 471, 472, 479, 504, 508, 515]
	Step 5	[543, 546, 555, 560, 566, 571, 574, 579, 593, 594]
	Step 6	[604, 605, 620, 621, 637, 651, 664, 671, 713, 745]
	Step 7	[760, 764, 779, 784, 805, 806, 814, 839, 845, 849]
	Step 8	[852, 859, 870, 872, 876, 879, 895, 907, 910, 920]
C→I	Step 1	[0, 2, 7, 9, 11, 27, 28, 29, 30, 33]
	Step 2	[37, 39, 40, 44, 45, 47, 50, 60, 62, 68]
	Step 3	[71, 75, 76, 82, 85, 86, 87, 88, 89, 90]
	Step 4	[92, 94, 96, 97, 106, 107, 108, 109, 110, 112]
	Step 5	[114, 115, 116, 123, 126, 128, 133, 134, 141, 145]
	Step 6	[146, 150, 151, 157, 160, 163, 165, 170, 172, 177]
	Step 7	[178, 179, 181, 185, 188, 192, 193, 196, 198, 200]
	Step 8	[209, 211, 215, 219, 225, 227, 228, 229, 230, 234]

D More Implementation Details

We train ProCA using SGD optimizer with the learning rate, weight decay and momentum set to 1×10^{-3}, 1×10^{-6} and 0.9, respectively. When training in each time step, we update pseudo-labels, label prototypes and source centers every 4, 7 and 5 epochs. Due to the lack of train-validation splits in the three datasets, we report the results at the last epoch for all methods. Note that we do not exploit any additional target augmentation, *e.g.,* [3], for training or evaluation.

E More Results of Enhancing Partial Domain Adaptation

In this section, we apply ProCA to improve ETN [2] to fully investigate the ability of our method to boost existing PDA methods for handling CI-UDA. As shown in Table IV, ProCA could enhance existing partial domain adaptation methods to alleviate catastrophic forgetting and thus overcome CI-UDA.

Table IV. Comparisons of the existing partial domain adaptation methods with and without our label prototype identification strategy on **Office-31-CI**. We show the final accuracy (%) and final S-1 accuracy (%).

Method	Prototypes	Metric	A→D	A→W	D→A	D→W	W→A	W→D	Avg.
ETN	✗	Final Acc. (%)	21.3	82.2	61.7	94.3	64.1	100.0	70.6
	✓		60.4	83.1	65.2	97.9	65.1	100.0	78.6
	✗	Final S-1 Acc. (%)	38.8	95.5	72.2	100.0	67.9	100.0	79.1
	✓		68.3	95.5	75.5	100.0	68.1	100.0	84.6
ProCA (ours)	✓	Final Acc. (%)	81.6	82.6	65.5	99.1	63.9	99.8	82.1
	✓	Final S-1 Acc. (%)	96.7	94.2	74.1	100.0	80.0	100.0	90.8

F More Ablation Studies

In this section, we first study the effect of three hyper-parameters (*i.e.,* λ, η and α) on three datasets. We fix the other hyper-parameters when studying ones. As shown in Table V, ProCA usually achieves the best performance in terms of Final Accuracy when setting $\lambda = 0.1$ and $\eta = 1.0$. Moreover, the results demonstrate that our method is non-sensitive for λ and η. Although ProCA may obtain the best performance in terms of Final Accuracy with a high α, we recommend setting a lower α, *e.g.,* 0.15, since a high threshold possibly filters shared classes out. One may concern false shared classes, but it can be handled by our method in fact since they would be updated by our label prototype identification strategy when a higher cumulative probability comes (c.f. Fig.3 in the main paper).

In addition, we train ProCA with a varying number of prototypes and incremental classes to investigate the effect of the number of target prototypes and that of incremental classes. As shown in Table VI, our method can achieve competitive performance (*i.e.,* 81.4% final Acc.) even with one prototype. With the increase of prototypes, the model retains the previous knowledge better and 20 prototypes in each class are sufficient for our ProCA. As for incremental classes, our method is non-sensitive to the number of incremental classes and performs well on all these settings. More specifically, when adding 10 classes each time step, ProCA achieves the best performance.

Table V. Effect of hyper-parameters λ, η and α on Office-31-CI (A→W, W→A), Office-Home-CI (Pr→Rw, Rw→Pr) and ImageNet-Caltech-CI (C→I). The value of λ is chosen from $[0, 0.05, 0.1, 0.2, 0.5, 1.0]$ and η is chosen from $[0, 0.1, 0.5, 1.0, 1.5, 2.0]$. Moreover, the value of α is chosen from $[0.1, 0.15, 0.2, 0.25, 0.30]$. In each experiment, the rest of hyper-parameters are fixed to the value reported in the main paper.

Dataset	Metric	λ						η						α				
		0	0.05	0.1	0.2	0.5	1.0	0	0.1	0.5	1.0	1.5	2.0	0.1	0.15	0.2	0.25	0.30
Office-31-CI	Final Acc.	84.7	84.9	**85.6**	84.5	84.3	84.1	84.1	84.7	84.6	**85.6**	84.4	84.9	81.6	85.6	85.3	**85.4**	83.5
	Final S-1 Acc.	85.1	85.4	**87.1**	85.8	85.7	84.3	84.8	85.4	87.0	87.1	**87.3**	86.8	86.6	**87.1**	85.9	86.1	85.2
Office-Home-CI	Final Acc.	72.5	73.2	**73.3**	73.0	73.0	73.2	72.5	72.6	72.9	**73.3**	73.2	73.3	73.7	73.3	73.1	**73.4**	73.0
	Final S-1 Acc.	77.5	78.3	**80.6**	78.3	77.1	76.1	76.7	79.0	78.2	**80.6**	77.4	79.0	73.4	**80.6**	79.5	78.9	74.2
ImageNet-Caltech-CI	Final Acc.	82.4	**83.5**	83.1	83.1	83.1	82.3	79.8	82.4	83.0	83.1	**83.5**	83.4	77.2	83.1	84.8	**87.8**	87.0
	Final S-1 Acc.	70.4	73.6	72.0	72.8	**74.2**	69.0	67.8	71.2	71.8	72.0	**73.4**	73.0	69.2	**72.0**	71.6	67.4	68.0

Table VI. Effect of the number of prototypes and incremental classes each time step on **ImageNet-Caltech-CI**. The number of prototypes is chosen from $[1, 5, 10, 20, 40]$ and the number of incremental classes is chosen from $[10, 15, 20, 30, 40]$. Note that we fix the other hyper-parameters when studying ones.

Setting	# Target Label Prototypes					# Incremental classes				
	1	5	10	20	40	10	15	20	30	40
Final Acc.	81.4	82.5	83.1	**83.9**	83.5	**83.1**	80.6	80.0	79.3	80.6
Final S-1 Acc.	65.8	70.6	72.0	73.8	**74.0**	**72.0**	69.0	71.0	68.2	67.8

G Effectiveness of Shared Class Detection

To further investigate the effectiveness of our shared class detection strategy, we compare our method with two variants. The first variant (*i.e.,* Pseudo-labeling) removes the shared class detection strategy and directly clusters target samples for all classes [11]. The second variant (*i.e.,* Pseudo-labeling with HBW) applies the HBW strategy [8] to the clustering method [11] and generates pseudo-labels for target samples. As shown in Table VII, the Final Accuracy of pseudo-labels of [11] yields inferior accuracy (74.1% Avg. Acc.) and even performs worse than source-only (77.5% Avg. Acc., ResNet-50) on Office-31-CI. This is because the pseudo labels generated by clustering may be noisy when facing the label space inconsistency between domains. The second variant also suffers performance degradation (66.6% Avg. Acc.). The reason lies in that the HBW strategy may fail to get the best variance of the source-positive and source-negative distributions in CI-UDA (cf. Appendix A), so it is unable to distinguish the source positive classes and the shared classes well (cf. Table I). In contrast, when using our shared class detection strategy, ProCA detects the shared classes well in various learning steps (cf. Table I) and thus achieves much better performance (cf. Table VII). Such a result demonstrates the superiority of our shared class detection strategy in CI-UDA over existing baselines.

Table VII. Final Accuracy (%) of the pseudo-labels with and without shared class detection (SCD) strategy on **Office-31-CI**.

Method	A→D	A→W	D→A	D→W	W→A	W→D	Avg.
ResNet-50 [7]	74.1	74.4	58.5	96.9	61.2	99.6	77.5
Pseudo-labeling [11]	71.2	73.8	60.3	84.8	63.0	91.5	74.1
Pseudo-labeling [11] with HBW [8]	67.5	71.5	42.7	82.9	44.9	90.2	66.6
Pseudo-labeling with our SCD	**79.7**	**78.3**	**63.5**	**99.0**	**64.9**	**100.0**	**80.9**

H More Experimental Results

To evaluate the ability of our method in sequential learning, we report Step-level Accuracy and the average accuracy of step-1 classes in each time step (S-1 Accuracy) on ImageNet-Caltech-CI (Table VIII), Office-31-CI (Table IX) and Office-Home-CI (Tables X, XI, XII and XIII). The experiments show that: 1) ProCA achieves the best (or at least comparable) performance w.r.t. Step-level Accuracy on all steps of all transfer tasks, which demonstrates the effectiveness of our method. 2) Compared with the other baselines, ProCA shows the least S-1 Accuracy drop on most transfer tasks, which shows that the proposed ProCA is good at alleviating catastrophic forgetting.

Table VIII. Classification accuracies (%) on ImageNet-Caltech-CI. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Avg.
C→I	ResNet-50 [7]	50.6 (50.6)	53.8 (50.6)	64.1 (50.6)	66.4 (50.6)	66.6 (50.6)	69.7 (50.6)	71.3 (50.6)	71.2 (50.6)	61.9 (50.6)
	DANN [5]	47.6 (47.6)	52.4 (55.6)	53.9 (52.2)	53.0 (52.2)	53.0 (52.6)	51.9 (49.4)	58.0 (53.6)	58.8 (54.8)	53.6 (52.3)
	PADA [1]	52.1 (52.1)	38.8 (19.7)	41.1 (21.5)	28.7 (19.8)	34.5 (22.2)	30.4 (34.3)	31.6 (29.5)	37.3 (29.1)	36.8 (28.5)
	ETN [2]	54.6 (54.6)	55.3 (35.6)	25.3 (11.0)	9.4 (0.8)	4.4 (0.0)	3.4 (0.0)	1.8 (0.0)	1.4 (0.0)	19.4 (12.8)
	BA^3US [12]	65.6 (65.6)	52.2 (68.4)	54.1 (65.8)	59.4 (65.6)	59.4 (60.0)	58.5 (58.2)	61.9 (56.2)	60.8 (53.0)	59.0 (61.6)
	CIDA [9]	58.6 (58.6)	61.3 (56.8)	65.4 (58.4)	67.1 (56.4)	65.9 (55.0)	69.4 (55.8)	68.8 (53.8)	69.3 (58.0)	65.7 (56.6)
	ProCA (ours)	**74.4** (74.4)	**74.8** (73.6)	**75.2** (73.4)	**77.5** (73.6)	**78.5** (72.4)	**81.6** (72.0)	**82.1** (71.2)	**83.1** (72.0)	**78.4** (72.8)
I→C	ResNet-50 [7]	81.2 (81.2)	71.8 (81.2)	76.5 (81.2)	73.8 (81.2)	75.2 (81.2)	73.1 (81.2)	71.3 (81.2)	70.7 (81.2)	74.2 (81.2)
	DANN [5]	62.8 (63.4)	53.0 (72.8)	43.3 (66.8)	43.2 (56.5)	36.0 (49.0)	33.9 (43.8)	33.1 (43.9)	31.4 (37.2)	42.1 (54.2)
	PADA [1]	72.4 (72.0)	53.5 (52.7)	50.0 (54.8)	46.1 (51.5)	53.6 (44.4)	40.3 (43.4)	44.5 (42.2)	45.9 (51.0)	50.8 (51.5)
	ETN [2]	75.9 (75.8)	70.5 (78.4)	73.5 (79.4)	69.1 (78.8)	72.2 (79.5)	71.0 (79.2)	48.5 (48.3)	3.1 (0.0)	60.5 (64.9)
	BA^3US [12]	94.0 (94.1)	71.2 (95.4)	82.2 (95.3)	84.5 (93.5)	81.2 (92.3)	77.8 (90.7)	64.0 (79.7)	45.0 (64.7)	75.0 (88.2)
	CIDA [9]	78.2 (78.7)	58.8 (80.5)	61.5 (81.1)	55.9 (78.5)	59.5 (77.8)	58.2 (76.9)	59.1 (77.9)	49.2 (64.6)	60.1 (77.0)
	ProCA (ours)	**97.8** (97.7)	**85.5** (97.5)	**87.6** (96.7)	**85.4** (96.2)	**86.9** (96.3)	**85.3** (95.9)	**84.2** (96.4)	**82.8** (95.0)	**86.9** (96.5)

Table IX. Classification accuracies (%) on Office-31-CI. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Avg.
A→D	ResNet-50 [7]	89.0 (87.8)	75.8 (87.8)	74.1 (87.8)	79.6 (87.8)
	DANN [5]	87.0 (85.6)	77.1 (87.1)	74.9 (85.4)	79.7 (86.0)
	PADA [1]	88.3 (88.7)	63.5 (35.7)	56.9 (35.2)	69.6 (53.2)
	ETN [2]	96.8 (96.2)	63.5 (89.2)	21.3 (38.8)	60.5 (74.7)
	BA^{3}US [12]	89.0 (89.7)	76.8 (89.7)	74.1 (89.7)	80.0 (89.7)
	CIDA [9]	90.3 (89.4)	77.1 (88.5)	70.4 (86.5)	79.3 (88.1)
	ProCA (ours)	**97.4** (97.3)	**84.5** (96.7)	**81.6** (96.7)	**87.8** (96.9)
A→W	ResNet-50 [7]	85.5 (85.3)	75.9 (85.3)	74.4 (85.3)	78.6 (85.3)
	DANN [5]	85.1 (86.1)	75.2 (86.4)	72.5 (85.2)	77.6 (85.9)
	PADA [1]	84.7 (82.9)	72.0 (53.5)	61.5 (49.9)	72.7 (62.1)
	ETN [2]	**97.9** (96.5)	**85.6** (96.5)	82.2 (95.5)	88.6 (96.2)
	BA^{3}US [12]	92.3 (89.1)	84.5 (88.6)	73.3 (89.0)	83.4 (88.9)
	CIDA [9]	82.1 (82.5)	70.6 (84.1)	64.5 (79.8)	72.4 (82.1)
	ProCA (ours)	92.3 (93.7)	83.3 (94.2)	**82.6** (94.2)	**86.1** (94.0)
D→A	ResNet-50 [7]	68.8 (68.6)	68.6 (68.6)	58.5 (68.5)	65.3 (68.5)
	DANN [5]	68.7 (68.2)	64.9 (68.0)	55.7 (67.7)	63.1 (68.0)
	PADA [1]	78.0 (78.2)	57.8 (63.7)	12.5 (17.2)	49.4 (53.0)
	ETN [2]	80.3 (79.5)	73.5 (74.7)	61.7 (72.2)	71.8 (75.5)
	BA^{3}US [12]	**81.3**(81.0)	**78.0** (78.7)	63.3 (76.7)	**74.2** (78.8)
	CIDA [9]	71.0 (71.3)	63.9 (71.5)	48.1 (64.9)	61.0 (69.2)
	ProCA (ours)	78.0 (74.6)	75.0 (73.0)	**65.5** (74.1)	72.8 (73.9)
D→W	ResNet-50 [7]	100.0 (100.0)	99.1 (100.0)	96.9 (100.0)	98.7 (100.0)
	DANN [5]	85.1 (86.1)	75.2 (86.4)	72.5 (85.2)	77.6 (85.9)
	PADA [1]	84.7 (82.9)	72.0 (53.5)	61.5 (49.9)	72.7 (62.1)
	ETN [2]	97.9 (96.5)	85.6 (96.5)	82.2 (95.5)	88.6 (96.2)
	BA^{3}US [12]	100.0 (100.0)	98.1 (100.0)	94.8 (100.0)	97.6 (100.0)
	CIDA [9]	97.4 (97.8)	99.8 (100.0)	95.1 (99.0)	97.4 (99.0)
	ProCA (ours)	**100.0** (100.0)	**100.0** (100.0)	**99.1** (100.0)	**99.7** (100.0)
W→A	ResNet-50 [7]	70.9 (71.4)	71.5 (71.4)	61.2 (71.4)	67.9 (71.4)
	DANN [5]	54.0 (55.1)	62.7 (65.9)	51.4 (65.8)	56.0 (62.3)
	PADA [1]	74.2 (73.9)	60.5 (50.4)	46.7 (39.9)	60.5 (54.7)
	ETN [2]	76.9 (73.5)	74.8 (70.4)	**64.1** (67.9)	71.9 (70.6)
	BA^{3}US [12]	**81.7** (81.3)	**78.3** (79.2)	64.0 (77.3)	**74.7** (79.3)
	CIDA [9]	72.1 (71.8)	65.7 (71.8)	52.7 (70.6)	63.5 (71.4)
	ProCA (ours)	80.0 (82.0)	72.8 (81.3)	63.9 (80.0)	72.2 (81.1)
W→D	ResNet-50 [7]	100.0 (100.0)	99.7 (100.0)	99.6(100.0)	99.8 (100.0)
	DANN [5]	100.0 (100.0)	99.0 (99.2)	97.7 (99.2)	98.9 (99.5)
	PADA [1]	100.0 (100.0)	83.9 (66.2)	84.3 (72.8)	89.4 (79.7)
	ETN [2]	100.0 (100.0)	99.7 (100.0)	**100.0** (100.0)	99.9 (100.0)
	BA^{3}US [12]	100.0 (100.0)	**100.0** (100.0)	100.0 (99.8)	**100.0** (99.9)
	CIDA [9]	100.0 (100.0)	97.7 (100.0)	98.8 (100.0)	98.8 (100.0)
	ProCA (ours)	**100.0** (100.0)	99.7 (100.0)	99.8 (100.0)	99.8 (100.0)

Table X. Classification accuracies (%) on Office-Home-CI with Ar as source domain. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Avg.
Ar→Cl	ResNet-50 [7]	48.9 (51.2)	48.3 (51.2)	45.1 (51.2)	46.1 (51.2)	47.5 (51.2)	47.6 (51.2)	47.2 (51.2)
	DANN [5]	39.6 (43.6)	29.6 (43.9)	29.6 (43.8)	35.2 (48.0)	28.7 (36.6)	33.1 (39.3)	32.6 (42.5)
	PADA [1]	49.9 (52.0)	43.6 (37.0)	29.3 (30.1)	23.9 (23.7)	27.2 (35.0)	24.8 (30.7)	33.1 (34.8)
	ETN [2]	49.2 (51.8)	49.6 (50.4)	42.0 (51.2)	42.2 (50.2)	43.4 (51.3)	42.4 (51.4)	44.8 (51.1)
	BA^3US [12]	53.0 (53.0)	47.0 (54.7)	35.4 (54.9)	33.0 (54.9)	30.7 (52.5)	33.7 (54.6)	38.8 (54.1)
	CIDA [9]	50.5 (54.6)	45.1 (53.0)	40.4 (51.2)	37.3 (52.3)	34.8 (48.7)	32.2 (45.4)	40.1 (50.9)
	ProCA (ours)	**53.3** (57.6)	**54.2** (57.6)	**47.8** (57.3)	**51.4** (58.8)	**52.2** (58.2)	**51.5** (57.1)	**51.7** (57.8)
Ar→Pr	ResNet-50 [7]	66.5 (66.2)	62.9 (66.2)	62.7 (66.2)	64.5 (66.2)	65.6 (66.2)	65.2 (66.2)	64.6 (66.2)
	DANN [5]	51.0 (51.1)	45.3 (53.6)	39.8 (50.4)	40.6 (52.2)	45.9 (52.7)	40.0 (53.9)	43.8 (52.3)
	PADA [1]	61.1 (59.4)	42.7 (27.0)	44.2 (34.5)	41.2 (39.7)	39.9 (35.5)	41.4 (36.4)	45.1 (38.8)
	ETN [2]	65.2 (65.5)	71.3 (64.6)	64.8 (64.9)	64.4 (65.0)	59.4 (60.4)	2.8 (1.1)	54.7 (53.6)
	BA^3US [12]	62.0 (62.7)	50.0 (62.5)	42.3 (61.4)	39.7 (60.4)	41.0 (56.9)	39.7 (54.3)	45.8 (59.7)
	CIDA [9]	66.2 (65.8)	59.6 (64.2)	57.0 (63.8)	52.0 (61.8)	50.8 (61.5)	45.9 (55.9)	55.2 (62.2)
	ProCA (ours)	**86.7** (84.6)	**75.3** (84.1)	**74.0** (83.5)	**73.9** (79.5)	**75.3** (78.2)	**75.1** (77.1)	**76.7** (81.2)
Ar→Rw	ResNet-50 [7]	73.1 (72.3)	70.7 (72.3)	69.8 (72.3)	72.0 (72.3)	72.0 (72.3)	72.7 (72.3)	71.7 (72.3)
	DANN [5]	57.0 (55.8)	51.3 (56.6)	51.3 (60.6)	45.4 (55.3)	42.0 (49.3)	45.8 (54.7)	48.8 (55.4)
	PADA [1]	77.1 (74.9)	60.1 (43.3)	56.9 (42.6)	49.0 (36.9)	56.3 (42.2)	55.1 (43.8)	59.1 (47.3)
	ETN [2]	75.0 (74.6)	73.5 (73.6)	71.9 (72.2)	63.3 (59.2)	29.0 (25.1)	7.4 (0.2)	53.4 (50.8)
	BA^3US [12]	79.4 (78.2)	71.5 (77.6)	64.8 (77.8)	62.9 (77.4)	58.1 (74.3)	63.2 (74.7)	66.7 (76.7)
	CIDA [9]	67.6 (66.7)	60.5 (67.9)	60.2 (69.8)	58.7 (68.2)	58.3 (67.0)	49.1 (54.0)	59.1 (65.6)
	ProCA (ours)	**86.2** (86.5)	**86.2** (84.4)	**83.7** (83.5)	**85.3** (81.5)	**85.4** (82.6)	**85.9** (80.8)	**85.5** (83.2)

Table XI. Classification accuracies (%) on Office-Home-CI with Cl as source domain. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Avg.
Cl→Ar	ResNet-50 [7]	57.7 (58.1)	55.4 (58.1)	52.7 (58.1)	52.7 (58.1)	51.9 (58.1)	54.7 (58.1)	54.2 (58.1)
	DANN [5]	30.9 (30.5)	32.1 (39.0)	34.0 (47.0)	35.4 (45.3)	35.7 (45.7)	36.8 (44.1)	34.1 (41.9)
	PADA [1]	58.7 (52.9)	41.1 (30.3)	31.5 (27.5)	25.4 (19.9)	19.8 (20.3)	18.3 (18.5)	32.5 (28.2)
	ETN [2]	63.3 (63.2)	57.3 (63.0)	54.2 (63.8)	55.5 (59.6)	25.9 (25.5)	4.3 (0.2)	43.4 (45.9)
	BA^3US [12]	**71.4** (68.7)	57.8 (69.3)	48.5 (68.8)	37.5 (64.8)	40.2 (56.4)	36.6 (59.6)	48.7 (64.6)
	CIDA [9]	41.8 (46.0)	41.6 (61.3)	37.1 (55.0)	35.2 (59.3)	35.4 (59.8)	36.5 (57.8)	37.9 (56.5)
	ProCA (ours)	66.1 (63.6)	**64.9** (64.4)	**60.9** (64.7)	**59.7** (62.2)	**58.8** (63.6)	**60.9** (63.5)	**61.9** (63.7)
Cl→Pr	ResNet-50 [7]	67.3 (68.9)	63.8 (68.9)	63.5 (68.9)	60.8 (68.9)	62.6 (68.9)	62.8 (68.9)	63.5 (68.9)
	DANN [5]	40.1 (39.9)	41.3 (40.7)	36.7 (40.1)	39.4 (48.9)	39.4 (45.6)	36.6 (47.3)	38.9 (43.8)
	PADA [1]	75.2 (74.9)	48.0 (31.8)	39.3 (29.8)	36.7 (24.2)	36.6 (23.1)	35.0 (23.6)	45.1 (34.6)
	ETN [2]	67.1 (67.0)	63.6 (69.0)	62.0 (66.1)	63.7 (68.6)	63.7 (68.8)	60.3 (68.3)	63.4 (68.0)
	BA^3US [12]	70.0 (70.6)	63.3 (70.7)	58.4 (72.3)	52.9 (73.0)	46.2 (71.5)	39.1 (64.5)	55.0 (70.4)
	CIDA [9]	60.1 (60.2)	54.2 (65.5)	53.9 (66.3)	50.0 (66.4)	50.0 (66.7)	48.6 (61.6)	52.8 (64.5)
	ProCA (ours)	**71.3** (75.3)	**71.1** (75.4)	**69.8** (75.4)	**65.8** (74.7)	**69.7** (73.9)	**69.7** (74.0)	**69.6** (74.8)
Cl→Rw	ResNet-50 [7]	66.7 (66.0)	65.9 (66.0)	65.6 (66.0)	64.4 (66.0)	65.5 (66.0)	66.1 (66.0)	65.7 (66.0)
	DANN [5]	46.9 (46.9)	46.7 (51.7)	49.1 (54.0)	44.1 (49.1)	44.0 (51.2)	44.1 (53.4)	45.8 (51.1)
	PADA [1]	62.5 (63.3)	49.2 (31.6)	43.5 (36.0)	41.2 (30.2)	34.3 (25.6)	36.3 (30.1)	44.5 (36.1)
	ETN [2]	63.5 (64.0)	65.3 (65.3)	65.2 (63.5)	65.1 (62.9)	39.8 (37.7)	6.3 (0.5)	50.9 (49.0)
	BA^3US [12]	**75.6** (74.7)	68.6 (75.6)	63.1 (74.6)	54.3 (72.5)	46.0 (68.8)	53.7 (72.9)	60.2 (73.2)
	CIDA [9]	64.9 (64.2)	50.4 (63.9)	52.7 (62.8)	52.1 (62.8)	50.8 (62.9)	46.6 (56.0)	52.9 (62.1)
	ProCA (ours)	71.9 (71.6)	**74.6** (69.9)	**74.1** (70.0)	**73.2** (67.4)	**76.0** (67.2)	**75.3** (69.4)	**74.2** (69.3)

Table XII. Classification accuracies (%) on Office-Home-CI with Pr as the source domain. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Avg.
Pr→Ar	ResNet-50 [7]	49.1 (49.5)	50.7 (49.5)	47.7 (49.5)	50.8 (49.5)	50.5 (49.5)	52.4 (49.5)	50.2 (49.5)
	DANN [5]	34.2 (31.2)	25.6 (27.9)	32.3 (35.4)	32.5 (38.4)	30.4 (36.8)	32.0 (35.9)	31.2 (34.3)
	PADA [1]	52.2 (49.0)	36.4 (26.4)	26.2 (17.4)	28.1 (15.9)	25.5 (16.4)	25.9 (12.6)	32.4 (23.0)
	ETN [2]	**63.8** (63.9)	55.8 (60.5)	53.7 (60.7)	53.0 (61.0)	51.5 (58.7)	50.7 (61.6)	54.8 (61.1)
	BA^{3}US [12]	62.5 (59.7)	56.3 (61.0)	47.2 (60.7)	41.7 (63.8)	39.0 (62.5)	36.5 (64.6)	47.2 (62.1)
	CIDA [9]	49.1 (51.9)	48.9 (52.8)	48.5 (52.6)	50.0 (52.0)	50.2 (51.7)	51.6 (52.5)	49.7 (52.2)
	ProCA (ours)	62.0 (60.8)	**66.0** (61.2)	**60.1** (60.0)	**61.3** (57.8)	**60.0** (58.6)	**59.9** (57.7)	**61.6** (59.4)
Pr→Cl	ResNet-50 [7]	49.8 (50.3)	47.3 (50.3)	47.2 (50.3)	46.7 (50.3)	46.3 (50.3)	44.7 (50.3)	47.0 (50.3)
	DANN [5]	37.5 (39.6)	33.4 (42.3)	33.1 (38.6)	30.7 (34.4)	23.3 (30.0)	29.8 (37.4)	31.3 (37.1)
	PADA [1]	49.3 (46.6)	35.0 (35.0)	28.5 (33.2)	24.4 (23.2)	29.0 (37.7)	26.2 (27.5)	32.1 (33.9)
	ETN [2]	47.7 (48.9)	42.4 (48.4)	38.7 (49.3)	36.3 (48.4)	38.2 (47.4)	33.8 (49.0)	39.5 (48.6)
	BA^{3}US [12]	55.4 (57.4)	44.5 (57.6)	38.0 (56.4)	32.3 (54.6)	29.5 (52.1)	24.9 (50.0)	37.4 (54.7)
	CIDA [9]	50.7 (51.6)	42.8 (51.7)	39.4 (51.3)	36.2 (50.2)	36.6 (50.5)	33.5 (49.0)	39.9 (50.7)
	ProCA (ours)	**60.4** (61.6)	**54.6** (59.6)	**54.2** (58.9)	**53.7** (59.0)	**53.2** (57.0)	**50.9** (58.2)	**54.5** (59.1)
Pr→Rw	ResNet-50 [7]	70.1 (69.5)	72.0 (69.5)	72.3 (69.5)	73.1 (69.5)	74.1 (69.5)	74.0 (69.5)	72.6 (69.5)
	DANN [5]	46.3 (46.0)	42.9 (43.6)	48.3 (48.8)	49.7 (50.8)	47.4 (46.4)	49.8 (49.3)	47.4 (47.5)
	PADA [1]	71.6 (71.5)	62.8 (43.4)	46.1 (33.6)	53.8 (43.3)	47.4 (30.8)	53.7 (40.3)	55.9 (43.8)
	ETN [2]	72.8 (73.1)	70.5 (69.6)	71.2 (71.5)	72.4 (71.0)	71.5 (69.5)	70.8 (70.2)	71.5 (70.8)
	BA^{3}US [12]	79.8 (79.3)	73.6 (78.8)	69.3 (77.8)	63.4 (76.8)	59.1 (71.9)	53.4 (65.9)	66.4 (75.1)
	CIDA [9]	65.0 (64.9)	62.3 (68.0)	61.4 (67.1)	59.4 (65.9)	59.7 (65.9)	59.0 (62.3)	61.1 (65.7)
	ProCA (ours)	**81.3** (80.9)	**83.4** (79.6)	**83.2** (78.8)	**85.4** (79.1)	**83.3** (78.7)	**84.7** (79.3)	**83.6** (79.4)

Table XIII. Classification accuracies (%) on Office-Home-CI with Rw as source domain. Note that the results outside the brackets are Step-level Accuracy, while the results in brackets represent the average accuracy of step-1 classes in each time step (S-1 Accuracy).

Task	Method	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Avg.
Rw→Ar	ResNet-50 [7]	63.0 (65.0)	66.0 (65.0)	64.5 (65.0)	67.0 (65.0)	64.4 (65.0)	66.2 (65.0)	65.2 (65.0)
	DANN [5]	37.2 (39.9)	43.9 (47.5)	39.9 (44.6)	41.7 (44.3)	44.3 (47.8)	42.4 (49.0)	41.6 (45.5)
	PADA [1]	70.9 (70.1)	59.3 (37.2)	50.6 (34.0)	40.9 (24.8)	39.5 (17.9)	46.8 (32.8)	51.3 (36.1)
	ETN [2]	68.6 (68.7)	68.1 (67.2)	67.3 (68.2)	66.2 (68.4)	44.4 (51.2)	3.7 (0.9)	53.1 (54.1)
	BA^{3}US [12]	65.1 (68.8)	64.3 (69.0)	58.2 (69.3)	57.4 (68.4)	55.2 (68.8)	52.2 (69.2)	58.7 (68.9)
	CIDA [9]	62.3 (65.1)	63.5 (66.9)	61.9 (67.0)	62.9 (64.7)	62.4 (65.4)	64.0 (69.0)	62.8 (66.3)
	ProCA (ours)	**77.5** (74.0)	**79.3** (74.4)	**74.9** (74.0)	**76.0** (73.9)	**72.7** (72.0)	**75.8** (73.9)	**76.0** (73.7)
Rw→Cl	ResNet-50 [7]	40.1 (43.8)	48.2 (43.8)	48.0 (43.8)	48.6 (43.8)	48.7 (43.8)	47.4 (43.8)	46.8 (43.8)
	DANN [5]	37.8 (42.5)	38.8 (41.5)	39.9 (45.1)	41.1 (43.2)	40.8 (43.7)	40.2 (47.1)	39.7 (43.9)
	PADA [1]	50.2 (52.7)	**52.2** (37.3)	38.8 (27.6)	34.9 (23.8)	38.1 (30.9)	31.4 (33.2)	40.9 (34.3)
	ETN [2]	46.4 (50.6)	50.5 (50.8)	45.0 (50.9)	45.6 (49.9)	45.2 (51.4)	43.5 (51.8)	46.0 (50.9)
	BA^{3}US [12]	**51.6** (55.5)	52.3 (56.6)	45.6 (54.6)	42.5 (56.2)	38.2 (54.7)	35.9 (55.0)	44.3 (55.4)
	CIDA [9]	44.6 (47.3)	45.4 (46.0)	41.3 (46.5)	41.2 (49.6)	40.0 (45.0)	38.0 (46.2)	41.7 (46.8)
	ProCA (ours)	45.4 (47.3)	46.7 (48.4)	**47.9** (48.8)	**50.8** (50.4)	**51.5** (49.4)	**51.0** (47.3)	**48.9** (48.6)
Rw→Pr	ResNet-50 [7]	67.3 (69.2)	74.0 (69.2)	78.1 (69.2)	77.9 (69.2)	78.1 (69.2)	77.4 (69.2)	75.5 (69.2)
	DANN [5]	53.5 (54.7)	55.4 (56.3)	58.7 (60.4)	57.1 (60.6)	55.5 (55.1)	55.2 (57.2)	55.9 (57.4)
	PADA [1]	77.6 (80.6)	61.0 (35.9)	50.5 (22.8)	52.8 (41.3)	55.8 (42.0)	50.0 (47.4)	57.9 (45.0)
	ETN [2]	70.0 (72.3)	75.9 (72.9)	74.4 (71.3)	76.5 (72.8)	76.2 (72.9)	75.1 (73.3)	74.7 (72.6)
	BA^{3}US [12]	73.4 (76.5)	75.4 (76.6)	72.8 (75.7)	71.6 (74.9)	69.0 (76.4)	65.9 (73.2)	71.3 (75.6)
	CIDA [9]	71.6 (73.5)	69.6 (70.4)	68.3 (70.7)	66.4 (68.7)	66.2 (68.0)	65.1 (68.5)	67.9 (69.9)
	ProCA (ours)	**80.6** (85.1)	**85.0** (83.9)	**88.3** (82.9)	**85.4** (82.7)	**86.0** (82.5)	**86.4** (81.8)	**85.3** (83.2)

References

1. Cao, Z., Ma, L., Long, M., et al.: Partial adversarial domain adaptation. In: ECCV (2018)
2. Cao, Z., You, K., Long, M., et al.: Learning to transfer examples for partial domain adaptation. In: CVPR. pp. 2985–2994 (2019)
3. Cubuk, E.D., Zoph, B., et al.: Autoaugment: Learning augmentation policies from data. ArXiv (2018)
4. Fu, B., Cao, Z., Long, M., Wang, J.: Learning to detect open classes for universal domain adaptation. In: ECCV. pp. 567–583 (2020)
5. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML (2015)
6. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset (2007)
7. He, K., Zhang, X., Ren, S., et al.: Deep residual learning for image recognition. In: CVPR (2016)
8. Hu, J., Tuo, H., Wang, C., et al.: Discriminative partial domain adversarial network. In: ECCV. pp. 632–648 (2020)
9. Kundu, J.N., Venkatesh, R.M., Venkat, N., et al.: Class-incremental domain adaptation. In: ECCV. pp. 53–69 (2020)
10. Li, G., Kang, G., Zhu, Y., et al.: Domain consensus clustering for universal domain adaptation. In: CVPR. pp. 9757–9766 (2021)
11. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: ICML (2020)
12. Liang, J., Wang, Y., Hu, D., et al.: A balanced and uncertainty-aware approach for partial domain adaptation. In: ECCV. pp. 123–140 (2020)
13. Pan, Y., Yao, T., Li, Y., et al.: Transferrable prototypical networks for unsupervised domain adaptation. In: CVPR (2019)
14. Qiu, Z., Zhang, Y., Lin, H., et al.: Source-free domain adaptation via avatar prototype generation and adaptation. In: IJCAI (2021)
15. Rebuffi, S.A., Kolesnikov, A., Sperl, G., et al.: icarl: Incremental classifier and representation learning. In: CVPR. pp. 5533–5542 (2017)
16. Russakovsky, O., Deng, J., Su, H., et al.: Imagenet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015)
17. Saito, K., Kim, D., Sclaroff, S., et al.: Universal domain adaptation through self supervision. NeurIPS pp. 16282–16292 (2020)
18. You, K., Long, M., Cao, Z., et al.: Universal domain adaptation. In: CVPR. pp. 2720–2729 (2019)