APPENDIX
Improving Unsupervised Domain Adaptation: A Pseudo-Candidate Set Approach

Aveen Dayal[1], Rishabh Lalla[1], Linga Reddy Cenkeramaddi[2], C Krishna Mohan[1], Abhinav Kumar[1], and Vineeth N Balasubramanian[1]

[1] Indian Institute of Technology Hyderabad, Hyderabad 502285, India
{ai21resch11003@, rishabh.lalla@cse., ckm@cse., abhinavkumar@ee., vineethnb@cse.}iith.ac.in
[2] University of Agder, Grimstad 4879, Norway
linga.cenkeramaddi@uia.no

In this appendix, we provide additional details such as further analysis and implementation details that we could not include in the main paper due to space constraints. Below is a list of the information contained in this appendix.

- **More Results and Analysis**
- **Datasets and Implementation Details**
- **Limitations and Broader Impact**

The code for our work is available at: /qwedaq/UDPCS

A1 More Results and Analysis

Pseudo-candidate set disambiguation: We examine the disambiguation process employed by our **UDPCS** method. In Fig. A1, we present various target samples from the VisDA dataset across 12 classes, along with their ground truth labels and the corresponding predicted labels from the trained UDA model (MDD) before applying our UDPCS refinement strategy. As outlined in the main paper, a crucial step in our UDPCS method involves disambiguating the pseudo-candidate sets by utilizing the label confidence vector $\boldsymbol{w}$, as per Eqn. (6). Fig. A2 demonstrates the evolution of the vector $\boldsymbol{w}$ across different epochs for all classes in the VisDA dataset. In this figure, we depict the distribution of the vector $\boldsymbol{w}$ at intervals of every eight epochs. As observed in Fig. A2, for instances that were incorrectly classified by the trained UDA model, the **UDPCS** method often aligns with the correct label later in the training process. Conversely, for instances that were correctly classified by the trained UDA model, the disambiguation process typically reaches convergence sooner.
Complexity Analysis: The complexity analysis of the UDPCS method, as presented in Table A1, illustrates its efficiency in terms of training time while achieving improvements in average accuracy. For the MDD model, extending the training from 30 to 60 epochs results in no change in average accuracy, with the time required doubling from 200 to 400 minutes. However, when the UDPCS method is applied to the MDD model (MDD+Ours) for 60 epochs, the average accuracy increases to 71.3%, with a total time of 231 minutes.

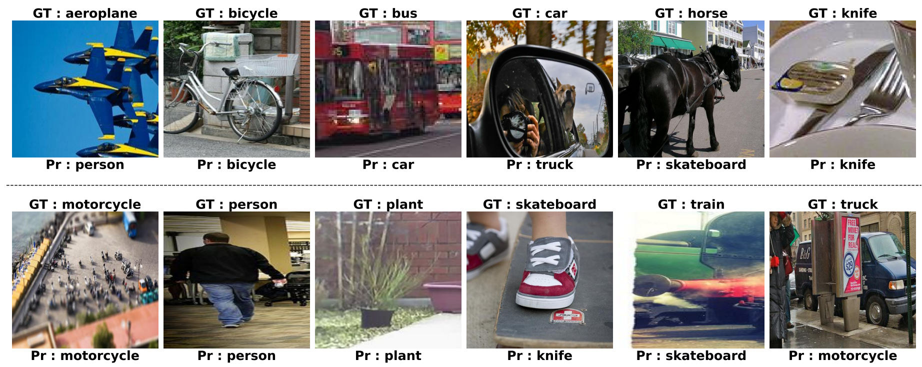

Fig. A1: Predictions of a trained UDA model (MDD) prior to our UDPCS refinement on the target samples of the VisDA dataset. (GT = Ground Truth , Pr = Predictions)

Notably, 200 minutes of this total time are allocated for training the base MDD model for the initial 30 epochs, while the additional 31 minutes are attributed to the UDPCS method's enhancements in the subsequent 30 epochs. This demonstrates that the UDPCS method effectively boosts performance with a relatively minor time overhead. Similarly, for the MCC model, doubling the epochs from 30 to 60 leads to a slight increase in average accuracy from 72.6% to 72.8%, with the time required doubling from 150 to 300 minutes. In contrast, the application of the UDPCS method to the MCC model (MCC+Ours) for 60 epochs elevates the average accuracy to 73.6%, with a total time of 177 minutes. This again highlights the UDPCS method's ability to enhance model performance efficiently.

Table A1: Complexity Analysis of UDPCS method on OfficeHome dataset. (Avg. = Average Accuracy (%), Time in minutes)

Model	Epoch	Time(↓)	Avg. (↑)
MDD	30	**200**	69.5
MDD	60	400	69.5
MDD+Ours	60	231	**71.3**
MCC	30	150	72.6
MCC	60	300	72.8
MCC+Ours	60	177	73.6

Threshold τ: In Table A2, we present the experimental results of our UDPCS method for various values of the hyperparameter τ on the OfficeHome dataset. We evaluate the performance of our technique for two different base UDA methods: MDD and MCC. The choice of τ determines the size of the pseudo-candidate set as per Eqn. (1). As illustrated in Tab. A2, the values for τ were selected based on the magnitude of the predictions made by the trained base UDA model. For the MDD+Ours strategy, we observe that the model achieves the highest average accuracy of 71.3% when $\tau = 0.0009$. Similarly, under the MCC+Ours strategy, the model attains the best average accuracy of 73.6% for both $\tau = 0.0009$ and $\tau = 0.00009$.

No Augmentation: In Table A3, we explore the impact of utilizing weak augmentation $\mathcal{A}_2$ versus no augmentation in the non-candidate set loss L_{nc} on the accuracy of our UDPCS method across various tasks in the OfficeHome dataset. Under the MDD+Ours framework, employing weak augmentation yields

Fig. A2: Pseudo-candidate set disambiguation of the proposed **UDPCS** method on VisDA dataset

Table A2: Accuracy(%) of UDPCS for different τ values on OfficeHome dataset.

τ	A-P	A-C	A-R	C-A	C-P	C-R	P-A	P-C	P-R	R-A	R-C	R-P	**Avg.**
						MDD+Ours							
$\tau = 0.009$	76.8	58.1	80.9	64.3	74.3	74.7	64.5	56.4	81.1	74	61.4	84.3	70.9
$\tau = 0.0009$	77.6	57.8	80.5	65.4	75.1	75.4	65.1	56.5	81.6	74.1	61.3	84.7	**71.3**
$\tau = 0.00009$	77.7	58	80	65	75.2	75.2	63.7	56.7	81.6	73.9	61.2	84.7	71.1
						MCC+Ours							
$\tau = 0.009$	80.8	58.9	83.9	69.6	77.7	79.8	68	56.2	83.2	74.3	62	86.2	73.4
$\tau = 0.0009$	81.1	59.3	84.1	69.2	78	79.6	69.1	56.4	82.8	74.7	62.1	86.4	**73.6**
$\tau = 0.00009$	81.1	59.3	83.1	70.2	77.9	79.1	68.9	57.1	82.5	75.7	62.5	86.1	**73.6**

a slight increase in the overall average accuracy, improving from 71% with no augmentation to 71.3% with weak augmentation. This trend is similarly observed in the MCC+Ours strategy, where the average accuracy enhances marginally from 73.4% without augmentation to 73.6% with weak augmentation.

Table A3: Accuracy(%) of UDPCS for different augmentation used on OfficeHome dataset. (Aug. = Augmentation)

Aug.	A-P	A-C	A-R	C-A	C-P	C-R	P-A	P-C	P-R	R-A	R-C	R-P	**Avg.**
						MDD+Ours							
No aug	77.2	57.9	80.4	65.2	74.5	74.5	64.8	56.4	81.5	74	61.5	84.4	71
Weak aug	77.6	57.8	80.5	65.4	75.1	75.4	65.1	56.5	81.6	74.1	61.3	84.7	**71.3**
						MCC+Ours							
No aug	80.7	58.9	84.2	69.4	77.7	79.6	68.4	56.2	83	74.7	61.9	86.4	73.4
Weak aug	81.1	59.3	84.1	69.2	78	79.6	69.1	56.4	82.8	74.7	62.1	86.4	**73.6**

Table A4: Additional Results.

(a) Avg. Accuracy.

Method	OH	VisDA
HMA	71.7	60.1
+ Ours	**73.3**	**64.9**

(b) Avg. Accuracy.

Method	DN
LeCO	42.1
+ Ours	**47.4**

Additional Results: In Table A4, we discuss results for additional baseline methods, for purposes of completeness. In both these cases, we report the average accuracy results obtained using our implementations of these methods (OH=OfficeHome, DN=DomainNet), due to the inability to reproduce results for LeCO [43] method on DomainNet and HMA [51] method with the provided code bases. (We could not include results of the COT [24] method due to unavailability of code.) We note from Table A4 that adding our method outperforms each of these methods by significant margins on our implementations.

Pseudo-Candidate Set and Domain Gap: The Pseudo-Candidate (PC) Set implicitly captures the information being transferred from the source to the target model, and thus correlates directly with the domain gap. To study this, we measured the domain gap in OfficeHome dataset using Maximum Mean Discrepancy (MMD) and Conditional Maximum Mean Discrepancy (CMMD), against the size of the PC set. The results in Table A5 show a consistent pattern across the experiments: smaller domain gaps correspond to smaller PC sets (shown in bold in table). This is intuitive since smaller gaps imply more similar features, leading to fewer ambiguous labels and a smaller PC set. This trend shows the connection between PC sets and domain gap.

Table A5: Relationship between Pseudo-Candidate Set and Domain Gap. AvgPC= Avg Pseudo-Candidate set size; S-T=source-target domains; A=Art, C=Clipart, P=Product, R=Real-world Domains. Values for CMMD are in e^{-2} scale i.e, read 5.3 as $5.3e^{-2}$, values for MMD are in e^{-3}.

(a) Target Domain is ‘A’

S-T	CMMD	MMD	AvgPC
C-A	5.3	7	4.6
P-A	5.3	6.9	4.5
R-A	**5**	**6.7**	**2.9**

(b) Target Domain is ‘C’

S-T	CMMD	MMD	AvgPC
A-C	5.4	7	4.2
P-C	3.5	4.9	4.1
R-C	**3.3**	**4.8**	**3.9**

(c) Target Domain is ‘P’

OH	CMMD	MMD	AvgPC
A-R	5.1	6.8	4.8
C-R	**3**	**4.8**	**3.7**
P-R	3.3	4.7	4.4

(d) Target Domain is ‘R’

OH	CMMD	MMD	AvgPC
A-P	5.3	6.9	3.2
C-P	3.5	4.9	3.3
R-P	**3.3**	**4.7**	**2.5**

Hyperparameter λ: In Table A6 we show the complete results of the hyperparameter λ, which balances the candidate loss (L_c) and the non-candidate loss (L_{nc}) in the total loss equation ($L_c + \lambda L_{nc}$). We run each experiment for three trials and report the average values. As seen from Table A6, the UDPCS method is robust to different λ values.

Table A6: Accuracy(%) of UDPCS for different λ values on OfficeHome dataset.

λ	A-P	A-C	A-R	C-A	C-P	C-R	P-A	P-C	P-R	R-A	R-C	R-P
						MDD+Ours						
0.5	77.2 ±0.2	58.2 ±0.3	80.6 ±0.2	64.8 ±0.2	75.3 ±0.1	74.9 ±0.4	65.1 ±0.2	56.5 ±0.1	81.0 ±0.7	73.9 ±0.1	61.4 ±0.1	84.5 ±0.3
1	77.5±0.1	58.0±0.3	80.6±0.3	65.3±0.1	75.2 ±0.1	75.1 ±0.3	65.0 ±0.1	56.5 ±0.1	81.6 ±0.2	74.1 ±0.1	61.4 ±0.2	84.5 ±0.2
2	77.3 ±0.1	57.9 ±0.1	80.8 ±0.1	65.1 ±0.2	75.0 ±0.0	75.3 ±0.2	65.1 ±0.2	56.6 ±0.1	81.5 ±0.4	74.0 ±0.2	61.3 ±0.1	84.4 ±0.1
						MCC+Ours						
0.5	80.9 ±0.3	59.1 ±0.2	83.8 ±0.1	69.6 ±0.1	77.8 ±0.2	79.6 ±0.2	68.9 ±0.6	56.6 ±0.3	83.2 ±0.1	74.8 ±0.1	62.1 ±0.2	86.6 ±0.2
1	81.0 ±0.2	59.1 ±0.2	84.3 ±0.4	69.3 ±0.2	77.9 ±0.1	79.6 ±0.0	68.9 ±0.3	56.6 ±0.2	83.2 ±0.4	74.7 ±0.1	62.0 ±0.1	86.5 ±0.1
2	80.9 ±0.1	59.2 ±0.2	84.2 ±0.2	69.2 ±0.4	78.0 ±0.3	79.6 ±0.3	68.9 ±0.3	56.7 ±0.4	82.9 ±0.4	74.8±0.1	62.4 ±0.1	86.6 ±0.1

A2 Datasets and Implementation Details

This section provides detailed information about the datasets, baselines and the implementation of the proposed **UDPCS** method.

Datasets. The proposed method is extensively evaluated on three benchmark UDA datasets defined as follows.

- **VisDA [31]:** The VisDA-2017 dataset consists of the UDA task from a synthetic (source) and real-world (target) domain with a focus on 12 classes. The source domain consists of 152,397 synthetic images and the target domain has 55,388 real-world samples.
- **DomainNet [30]:** This is a large-scale dataset consisting of 0.6 million images across 345 classes, including six domains: clipart, sketch, real, painting, infographics, and quickdraw.
- **OfficeHome [42]:** This is a relatively smaller dataset consisting of 15,500 samples across 65 classes and includes four domains: Art, Clipart, painting, and Real World.

Baselines. As discussed in the main paper, our **UDPCS** method is built on the following diverse UDA methods: MDD [50], MCC [14], LeCo [43], SDAT [35], and NWD [3]. Additionally, we compare **UDPCS** with recent benchmarks, HMA [51] and COT [24]. Note that we focus on unsupervised domain adaptation (UDA) in this work, unlike other related settings such as source-free domain adaptation [52, 54, 56, 57, 60] or few-shot domain adaptation [53, 55, 58, 59, 61]. We hence focus on the most recent state-of-the-art UDA methods in this work for comparisons.

Hyperparameters: Following recent state-of-the-art [24, 51], our **UDPCS** model uses ResNet50 architecture as feature extractor for OfficeHome dataset. It uses ResNet101 architecture as the feature extractor for VisDA and DomainNet datasets. We train our model for 30 epochs using stochastic gradient descent with momentum [34] on an NVIDIA A100 GPU with 40GB RAM. We list the search space for the various hyperparameters used in this work in Tab. A7. We also show the best hyperparameter values used for different datasets in Tab. A8.

Table A7: Search values of all hyperparameters used in the UDPCS method.

Hyperparameter	**Search values**
γ	[0.8,0.85,0.9,1]
τ	[0.009,0.0009,0.00009]
λ	[0.5,1,2]
Learning rate (Lr)	[0.01,0.004,0.005]
Lr decay	[0.75,0.85]
Momentum (M)	[0.85,0.95]

Table A8: Hyperparameter values of UDPCS method.

Dataset	γ	τ	λ	Lr	Lr decay	M	Batch size
			MDD+Ours				
VisDA	0.85	0.0009	1	0.004	0.75	0.9	36
DomainNet	0.85	0.0009	1	0.004	0.75	0.9	32
OfficeHome	0.85	0.0009	1	0.004	0.75	0.9	32
			MCC+Ours				
VisDA	0.85	0.0009	1	0.005	0.75	0.9	36
DomainNet	0.85	0.0009	1	0.005	0.75	0.9	36
OfficeHome	0.85	0.0009	1	0.005	0.75	0.9	36
			LeCo+Ours				
VisDA	0.85	0.0009	1	0.004	0.75	0.9	36
OfficeHome	0.85	0.0009	1	0.04	0.75	0.9	36
			SDAT+Ours				
VisDA	0.85	0.0009	1	0.005	0.75	0.9	32
DomainNet	0.85	0.0009	0.5	0.005	0.75	0.9	32
OfficeHome	0.85	0.0009	1	0.005	0.75	0.9	32
			NWD+Ours				
VisDA	0.85	0.0009	1	0.005	0.75	0.9	36
OfficeHome	0.85	0.0009	1	0.005	0.75	0.9	36

A3 Limitations and Broader Impact

The proposed **UDPCS** technique builds on any off-the-shelf UDA method and refines the model's prediction by leveraging the pseudo-candidate set on target data. Hence, any limitations from the base UDA method may percolate into our method. As the pseudo-candidate sets are dependent on the trained UDA model's predictions, it may be important to have UDA models that are reasonably good in terms of target accuracy (better than random predictions). We however believe that this not a limiting assumption, considering many good UDA methods already exist. This refinement strategy can be easily extended to other settings, such as source-free domain adaptation and test-time adaptation. These are interesting directions of future work.

References

52. Sabbir Ahmed, Abdullah Al Arafat, Mamshad Nayeem Rizve, Rahim Hossain, Zhishan Guo, and Adnan Siraj Rakin. Ssda: Secure source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19180–19190, October 2023. 6
53. Yurong Guo, Ruoyi Du, Yuan Dong, Timothy Hospedales, Yi-Zhe Song, and Zhanyu Ma. Task-aware adaptive learning for cross-domain few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1590–1599, October 2023. 6
54. Mattia Litrico, Alessio Del Bue, and Pietro Morerio. Guiding pseudo-labels with uncertainty estimation for source-free unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7640–7650, June 2023. 6
55. Rundong Luo, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Similarity min-max: Zero-shot day-night domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8104–8114, October 2023. 6
56. Sunandini Sanyal, Ashish Ramayee Asokan, Suvaansh Bhambri, Akshay Kulkarni, Jogendra Nath Kundu, and R Venkatesh Babu. Domain-specificity inducing transformers for source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18928–18937, October 2023. 6
57. Maohao Shen, Yuheng Bu, and Gregory W Wornell. On balancing bias and variance in unsupervised multi-source-free domain adaptation. In *International Conference on Machine Learning*, pages 30976–30991. PMLR, 2023. 6
58. Yizhe Xiong, Hui Chen, Zijia Lin, Sicheng Zhao, and Guiguang Ding. Confidence-based visual dispersal for few-shot unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11621–11631, October 2023. 6
59. Ceyuan Yang, Yujun Shen, Zhiyi Zhang, Yinghao Xu, Jiapeng Zhu, Zhirong Wu, and Bolei Zhou. One-shot generative domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7733–7742, October 2023. 6
60. Yixin Zhang, Zilei Wang, and Weinan He. Class relationship embedded learning for source-free unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7619–7629, June 2023. 6
61. Fei Zhou, Peng Wang, Lei Zhang, Wei Wei, and Yanning Zhang. Revisiting prototypical network for cross domain few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20061–20070, June 2023. 6