Domain-Agnostic Mutual Prompting for Unsupervised Domain Adaptation

Supplementary Material

Appendix Contents

A. More Implementation Details

We implement our method with the Pytorch framework. Our code is built on the `Dassl.pytorch` [1] platForm, which is a principled implementation and evaluation platForm For DA and DG tasks. We train DAMP with a single NVIDIA GeForce RTX 3090 GPU. Details about network architecture, data augmentation and pseudo-labels are described as follows.

Network Architecture. The text encoder f_s and the mutual prompting module G we used are mainly comprised of a TransFormer encoder and a TransFormer decoder, respectively. Specifically, the standard dot-product attention is leveraged. Given a set of queries $Q \in \mathbb{R}^{N_q \times d_k}$, keys $K \in \mathbb{R}^{N_k \times d_k}$ and values $V \in \mathbb{R}^{N_k \times d_v}$, the attentional outputs For all queries can be calculated by:

$$\texttt{Attention}(Q, K, V) = \operatorname{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \in \mathbb{R}^{N_q \times d_v}. \tag{15}$$

For Self-Attention (SA), Q, K, V obtained from the same input sequence $I \in \mathbb{R}^{N \times D}$, through three projection matrixes $M_q \in \mathbb{R}^{D \times d_k}, M_k \in \mathbb{R}^{D \times d_k}, M_v \in \mathbb{R}^{D \times d_v}$,

$$\texttt{SA}(I) = \texttt{Attention}(IM_q, IM_k, IM_v) \in \mathbb{R}^{N \times d_v}. \tag{16}$$

Multi-Head Self-Attention (MHSA) extends SA by using multiple attention heads,

$$\texttt{MHSA}(I) = \operatorname{Concat}\left(\mathrm{head}_1, \ldots, \mathrm{head}_\mathrm{h}\right) M_O, \tag{17}$$

where $\mathrm{head}_i = \texttt{Attention}\left(IM_q^i, IM_k^i, IM_v^i\right)$, h is the head number, and $M_O \in \mathbb{R}^{hd_v \times D}$ maps the intermediate embeddings to match the input dimension.

For f_s, each encoder layer $\texttt{Enc}_j$ comprised of a MHSA block and a feed-Forward block $\texttt{FD}(\cdot)$, with $h = 8$, $d_k = d_v = D = 512$. A residual connection and LayerNorm operation $\texttt{LN}(\cdot)$ is employed after each of them, i.e.,

$$\begin{aligned} &\texttt{Enc}_j(I) = \texttt{LN}(\texttt{FD}(I') + I'), \\ &I' = \texttt{LN}(\texttt{MHSA}(I) + I) \end{aligned} \tag{18}$$

For G, a Masked MHSA (M-MHSA) block and Multi-Head Cross-Attention (MHCA) block is used in each $\texttt{Dec}_l$. Specifically, the M-MHSA alters MHSA by imposing a mask on the attention scores, i.e.,

$$\begin{aligned} &\texttt{M-MHSA}(I) = \operatorname{Concat}\left(\mathrm{head}_1, \ldots, \mathrm{head}_\mathrm{h}\right) M_O, \\ &\mathrm{head}_i = \texttt{MaskedAttention}\left(IM_q^i, IM_k^i, IM_v^i\right), \\ &\texttt{MaskedAttention}(Q, K, V) = \operatorname{softmax}\left(\frac{S \odot QK^T}{\sqrt{d_k}}\right), \end{aligned} \tag{19}$$

where $S \in \mathbb{R}^{N_q \times N_k}$ is the mask. The MHCA block is a variation of MHSA by using different source of Q and K, V, i.e.,

$$\begin{aligned} &\texttt{MHCA}(I_1, I_2) = \operatorname{Concat}\left(\mathrm{head}_1, \ldots, \mathrm{head}_\mathrm{h}\right) M_O, \\ &\mathrm{head}_i = \texttt{Attention}\left(I_1 M_q^i, I_2 M_k^i, I_2 M_v^i\right). \end{aligned} \tag{20}$$

Each decoder layer $\texttt{Dec}_l$ can be Formulated as,

$$\begin{aligned} &\texttt{Dec}_l(I_1, I_2) = \texttt{LN}(\texttt{FD}(I_1'') + I_1''), \\ &I_1'' = \texttt{LN}(\texttt{MHCA}(I_1, I_2) + I_1), \\ &I_1' = \texttt{LN}(\texttt{M-MHSA}(I_1) + I_1). \end{aligned} \tag{21}$$

We use $d_k = d_v = D = 256$ and $h = 4$ For the decoder. The `InProj` and `OutProj` are two linear layers with LayerNorm operations. In this work, we use the same G For language-guided visual prompting and vision-guided language prompting since the two modalities are aligned in the CLIP embedding space. Therefore, they can attend to each other by freely change the position in $\texttt{Dec}_l$.

Data augmentations. In this work, we use random flip as a simple weak augmentation operation. For strong augmentation, we select two random operations For each sample from the RandAugment library [3], which includes invert, rotation, color enhancing, auto contrast and other transFormations. Randomly selecting and combining these strong augmentations is intended to simulate the diverse domain shifts that can occur in real-world data.

Pseudo-label. For previous methods that incorporate domain-specific tokens in the textual prompt, obtaining pseudo-labels for the target domain is challenging. On one hand, the domain-specific prompts are not transferable, i.e., we cannot obtain high-quality target pseudo-labels with the learned source-specific prompts. On the other hand, without high-quality pseudo-labels, learning target-specific prompts becomes an ill-posed problem. To circumvent this problem, DAPrompt only [7] uses a naive textual prompt, i.e., “a photo of a [CLS]. a [Domain] image.” to obtain pseudo-labels, where [CLS] and [Domain] are the name of each

[1] https://github.com/KaiyangZhou/Dassl.pytorch

Algorithm 1 Training Procedure of DAMP For UDA.

```
Require: Labeled source dataset $\mathcal{D}_s$, unlabeled target dataset $\mathcal{D}_t$, total training epochs $E$, iteration number per epoch $N_e$.
Ensure: Optimal $\boldsymbol{p}_{1:N}$, $\gamma_v$, $\gamma_s$ and $G$.
Initialize parameters for $\boldsymbol{p}_{1:N}$, $\gamma_v$, $\gamma_s$ and $G$.
for $t = 1$ to $E$ do
    for $i = 1$ to $N_e$ do
        Sample a source batch $\mathcal{B}_s \sim \mathcal{D}_s$ and a target batch $\mathcal{B}_t \sim \mathcal{D}_t$
        obtain $\{\boldsymbol{s}'_k\}_{k=1}^{K}$ and $\boldsymbol{v}'$ for each $\boldsymbol{x} \in \mathcal{B}_s \cup \mathcal{B}_t$ according to Eq. (6) and (4).
        Calculate $\mathcal{L}_{sup}^{s}$ and $\mathcal{L}_{sup}^{t}$ according to Eq. (12) and (13).
        Calculate $\mathcal{L}_{sc}^{s}$ and $\mathcal{L}_{sc}^{t}$ according to Eq. (9) and (10).
        Calculate $\mathcal{L}_{idc}$ within $\mathcal{B}_s$ and $\mathcal{B}_t$ respectively according to Eq. (8) and sum them up.
        Calculate $\mathcal{L}_{im}$ according to Eq. (11).
        Update parameters via optimizing $\mathcal{L}_{all}$.
    end for
end for
Return final model parameters $\boldsymbol{p}_{1:N}$, $\gamma_v$, $\gamma_s$ and $G$.
```

class and each domain, respectively. This resuls in knowledge isolation between the two domains.

In this work, we learn shared prompts For both domains. This allows us to leverage the rich source domain knowledge to pseudo-label the target domain. However, we found that in the early stages of training, the source domain model is not yet well-trained, resulting in low-quality pseudo-labels. To address this, we propose combining prior knowledge with the source knowledge to obtain better pseudo-labels. Specifically, we first generate a naive textual prompt to produce naive soft pseudo-labels $\tilde{y}_i^t$ for each target sample y_i^t. We also generate source-enabled soft pseudo-labels $\dot{y}_i^t$ from the model outputs according to Eq. (7). The final pseudo label is an ensemble of both:

$$\hat{y}_i^t = (1-\alpha)\tilde{y}_i^t + \alpha\dot{y}_i^t. \tag{22}$$

The weight α is gradually increased from 0 to 1 during training. Weighting the naive and source-enabled pseudo-labels via the α enables a smooth transition from relying more on the prior knowledge to relying more on the source-knowledge as training progresses.

B. Algorithm

To better understand our method, we summarize the training procedure of DAMP for UDA in Algorithm 1.

C. Experiments on Multi-Source UDA

To evaluate the versatility of our method in various domain adaptation scenarios, we extend our method to the multi-source domain adaptation (MSDA) setting.

Datasets. We evaluate our method on two widely used MSDA datasets. Specifically, we reuse the **Office-Home** [18] dataset for MUDA by combining arbitrary 3 domains as source domains and regard the rest domain as the target domain, which forms 4 adaptation tasks (→Ar, →Cl, →Pr, →Rw). **DomainNet** [12] is the lagest and the most challenging dataset for domain adaptation. It consists of 6 diverse domains, including Clipart, Painting, Real, Sketch, Quickdraw and Infograph. These domains encompass a wide range of visual styles, making the dataset challenging for domain adaptation tasks. Similar to Office-Home, 6 tasks are constructed for MSDA.

Experimental Setup. It is a natural extension to apply our method to the MSDA setting, where the goal is to learn a shared set of prompts across all source domains and the target domain. Specifically, we extend $\mathcal{L}_{sup}$ and $\mathcal{L}_{sc}$ to include losses on all source domains. We treat each domain with equal importance. For $\mathcal{L}_{idc}$, we obtain a batch of samples from each domain in each iteration, and compute $\mathcal{L}_{idc}$ within each domain batch. The utilization of domain labels in this process distinguishes our method from other single-source domain adaptation methods which simply mix the source domains. For convenient comparison with previous methods, we use the ResNet-50 backbone on Office-Home and ResNet-101 on DomainNet. Other training configurations remain consistent with those employed in single-source UDA, as detailed in Sec. 4.

Experimental Results. The results on **DomainNet** are reported in Table 6. DAMP achieves the best average accuracy of 57.8%, outperforming the previous state-of-the-art MPA by 3.7%. This demonstrates the effectiveness of DAMP on multi-source domain adaptation. Compared to single-source methods like DANN, MCD and DAPrompt, DAMP brings substantial gains, improving over DAPrompt by 5.8%. This shows the benefits of domain-agnostic prompts and exploiting multiple source domains in our method. Besides, DAMP also surpasses other multi-source domain adaptation methods such as M^{3}SDA-β, SImpA, LtC-MSDA and T-SVDNet by a large margin. Notably, DAMP achieves the best performance on 5 out of 6 tasks. The consistent improvements over competitive baselines validate the robustness of DAMP.

On **Office-Home** (Table 7), we can observe that DAMP again achieves state-of-the-art accuracy (79.2%), outperforming the closest competitor MPA by 3.8%. Compared to single-source methods, DAMP brings significant gains over 6.4% over DAPrompt, showing the benefit of learning domain-agnostic prompts in the multi-source setting. DAMP surpasses other multi-source domain adapta-

Table 6. Classification accuracies (%) on **DomainNet** for MSDA with ResNet-101. * Prompt learning-based methods.

Method	→Clipart	→Infograph	→Painting	→Quickdraw	→Real	→Sketch	Avg.
Zero-Shot							
CLIP [13]	61.3	42.0	56.1	10.3	79.3	54.1	50.5
Source Combined							
DANN [6]	45.5	13.1	37.0	13.2	48.9	31.8	32.6
MCD [15]	54.3	22.1	45.7	7.6	58.4	43.5	38.5
DAPrompt * [7]	62.4	43.8	59.3	10.6	81.5	54.6	52.0
CoOp * [22]	63.1	41.2	57.7	10.0	75.8	55.8	50.6
Multi-Source							
M^{3}SDA-β [12]	58.6	26.0	52.3	6.3	62.7	49.5	42.6
SImpAI$_{101}$ [17]	66.4	26.5	56.6	18.9	68.0	55.5	48.6
LtC-MSDA [19]	63.1	28.7	56.1	16.3	66.1	53.8	47.4
T-SVDNet [10]	66.1	25.0	54.3	16.5	65.4	54.6	47.0
PFSA [5]	64.5	29.2	57.6	**17.2**	67.2	55.1	48.5
PTMDA [14]	66.0	28.5	58.4	13.0	63.0	54.1	47.2
MPA * [2]	65.2	47.3	62.0	10.2	82.0	57.9	54.1
DAMP * (Ours)	**69.7**	**51.0**	**67.5**	14.7	**82.5**	**61.5**	**57.8**

Table 7. Classification accuracies (%) on **Office-Home** for MSDA with ResNet-50. * Prompt learning-based methods.

Method	→Ar	→Cl	→Pr	→Rw	Avg.
Zero-Shot					
CLIP [13]	71.5	50.2	81.3	82.4	71.4
Source Combined					
DAN [11]	68.5	59.4	79.0	82.5	72.4
DANN [6]	68.4	59.1	79.5	82.7	72.4
CORAL [16]	68.1	58.6	79.5	82.7	72.2
DAPrompt * [7]	72.8	51.9	82.6	83.7	72.8
CoOp * [22]	70.7	52.9	82.9	83.9	72.4
Multi-Source					
MDDA [21]	66.7	**62.3**	79.5	79.6	71.0
SImpAI$_{50}$ [17]	70.8	56.3	80.2	81.5	72.2
MFSAN [23]	72.1	62.0	80.3	81.8	74.1
MPA * [2]	74.8	54.9	86.2	85.7	75.4
DAMP * (Ours)	**77.7**	61.2	**90.1**	**87.7**	**79.2**

tion methods like MDDA, SImpA and MFSAN by solid margins. Furthermore, by comparing with the results for single-source UDA (Table 1), we can observe that for any target domain, using multiple source domain data is better than using only one source domain. This validates that our method indeed utilizes source knowledge effectively.

D. Experiments on Doamin Generalization

We show that with only minor changes, our method can also be used for domain generalization tasks.

Datasets. We use four popular DG datasets in this experiment, namely, VLCS [4], PACS [9], Office-Home [18] and TerraIncognita [1]. **VLCS** contains images from PASCAL VOC 2007 (V), LabelMe (L), Caltech (C) and SUN (S). There are 5 object categories shared by all domains: bird, car, chair, dog and person. **PACS** collects totally 9,991 images from Photo (P), Art painting (A), Cartoon (C) and Sketch (S) with 7 common categories. **Office-Home** is originally used in domain adaptation, which contains images from Art (Ar), Clipart (Cl), Product (Pr) and Real-World (Rw) across 65 categories. **TerraIncognita** comprises a collection of wildlife photographs captured by cameras at various locations. We follow [8] to use 4 locations, i.e., {L38, L43, L46, L100}, for the DG task, which have totally 24,788 samples of 5 classes.

Experimental Setup. For the DG task, we implement our method on DomainBed [2], a standard DG benchmark in the community. We strictly follow [8] to split each domain into 80% training data and 20% validation data, and use standard training-domain validation for model selection. The results are obtained by three trials with seed=$\{1,2,3\}$.

Different from domain adaptation, in DG we cannot access any target sample during training. Therefore, we remove $\mathcal{L}_{im}$ and target-related terms in $\mathcal{L}_{sup}$, $\mathcal{L}_{sc}$ and $\mathcal{L}_{idc}$ for optimization. In this scenario, our method aims to elicit domain-invariant visual embeddings from multiple source domains and instance-compatible text embeddings for classification. Due to the absence of the target domain, large-scale pre-trained knowledge becomes more important in DG. Therefore, CLIP-based methods will have significantly better results than traditional ones. For fair comparison, all compared baselines are built on CLIP, and we use the ViT-B/16 vision backbone for all datasets following [20].

Specifically, there are three categories of baselines for comparison. The first category of methods fine-tune the image encoder of CLIP using common DG algorithms (e.g., like ERM, DANN) and freeze the text encoder for classification. The second category directly use the zero-shot ability of CLIP and prompt the text encoder with manually designed prompts (‘a photo of [CLS]’) for classification.

[2] https://github.com/facebookresearch/DomainBed

Table 8. Classification accuracies (%) on VLCS, PACS, Office-Home, and TerraIncognita for domain generalization. The best results are highlighted in bold. All compared methods are implemented based on CLIP with ViT-B/16 backbone.

Method	VLCS	PACS	Office-Home	TerraInc	Avg
Fine-tuning (CLIP)					
ERM	82.7 ± 0.3	92.9 ± 1.9	78.1 ± 2.1	50.2 ± 1.7	75.9
CORAL [16]	82.0 ± 0.2	93.2 ± 1.1	78.9 ± 1.9	**53.5** ± **0.7**	76.9
DANN [6]	83.2 ± 1.2	93.8 ± 1.3	78.8 ± 1.1	52.2 ± 2.0	77.0
Zero-shot					
CLIP [13]	82.3 ± 0.1	96.1 ± 0.1	82.3 ± 0.2	34.1 ± 0.1	73.7
Prompt Learning					
DPL [20]	84.3 ± 0.4	97.3 ± 0.2	84.2 ± 0.2	52.6 ± 0.6	79.6
DAMP (Ours)	**84.5** ± **0.3**	**97.4** ± **0.2**	**85.0** ± **0.4**	**53.7** ± **0.2**	**80.2**

The third category resorts to learnable prompts for adapt the pre-trained CLIP to specific domains. For our method, the hyperparameter configuration is consistent with the ones in UDA and MSDA.

Experimental Results. We report the mean accuracies as well as standard derivation on four datasets in Table 8. Our DAMP achieves the best average accuracy of 80.2% across all datasets. Compared to fine-tuning-based methods like ERM, CORAL and DANN, DAMP brings significant improvements of 4.3%, 3.3% and 3.2% respectively in average accuracy. We conjecture the reason is that large-scale pre-training is a very effective approach to bridge the domain gap. However, fine-tuning the image encoder is prone to destroy the pre-trained knowledge encoded in CLIP. This demonstrates the superiority of adapting pre-trained models via prompt learning over fine-tuning for domain generalization. On the other hand, DAMP also surpasses the vanilla zero-shot CLIP model by 6.5%, showing the benefits of learning adaptive prompts compared to relying solely on pre-trained knowledge. Compared with DPL that only prompts the text encoder, our method prompts both vision and textual modalities for generalizing both visual images and textual semantics to unseen domains. Besides, two regularizations (i.e., $\mathcal{L}_{idc}$ and $\mathcal{L}_{sc}$) encourage the embeddings to be more domain-agnostic, thus outperforming DPL on all datasets. Even though the margins appear small, i.e., 0.6% over DPL, it is well-recognized that further advancing the state-of-the-art on DG benchmarks is extremely challenging. Even slight gains of 1% are considered significant and difficult in the DG community, which typically indicate non-trivial improvements in the robustness and generalization abilities of the model across diverse domains.

E. Additional Analytical Experiments for UDA

E.1. Confusion Matrix Visualization

To illustrate how our method benefits UDA, we visualize the confusion matrixes obtained by different methods in Fig. 6. We can observe that directly using the zero-shot classification capability of CLIP can easily lead to confusion between categories. For instance, the model may easily predict ”car” as ”bus” or ”truck,” or predict ”bicycle” as ”motorcycle”, because these categories are conceptually similar. In contrast, DAPrompt adjusts the textual semantics of categories specifically for each domain (dataset), making it clearer to distinguish the semantic differences between categories and to some extent alleviating the confusion problem. However, DAPrompt does not perform any adaptation in the visual modality, making it susceptible to domain shifts in the visual modality. Additionally, DAPrompt uses class-level semantic embeddings for classification, which does not take into account variations within categories. Imagine if the visual embedding of a truck in the multimodal space is closer to the semantic representation of ”bus” than ”truck.” In this case, DAPrompt would have no way of classifying it as a truck. In contrast, our method allows the semantic embeddings to dynamically adjust their positions based on visual cues for each sample, providing a customized set of semantic embeddings for classification, and therefore performs better in disambiguation compared to CLIP and DAPrompt.

E.2. Effectiveness of the Pseudo-label Strategy

To evaluate the effectiveness of our ensemble-based pseudo-label strategy described in Appendix. A, we conduct experiments to compare with other two pseudo-label strategies, i.e., using the naive prompts (‘a photo of a [CLS]’) for zero-shot prediction and using the learned prompts $\boldsymbol{p}_{1:N}$ with *post-model* multual prompting. The former strategy is used in DAPrompt [7] and the latter is the outputs of our model. As shown in Fig. 7, we can observe a common phenomenon across all tasks. Initially, due to the zero-shot capability of CLIP, it can provide high-quality pseudo-labels for the target domain samples, resulting in high accuracy achieved in the first epoch for learned prompts. However, as the proportion of zero-shot pseudo-labels is still high at this point, the accuracy of the ensem-

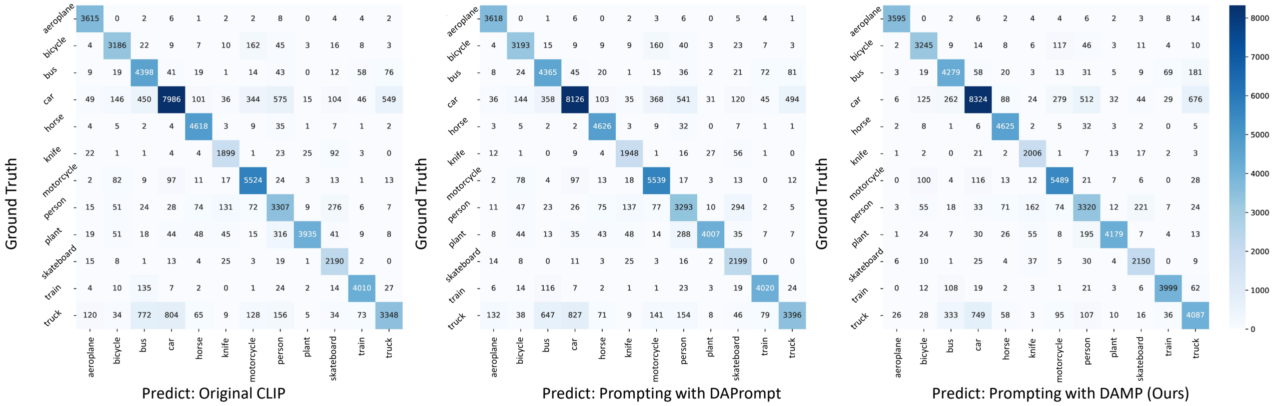
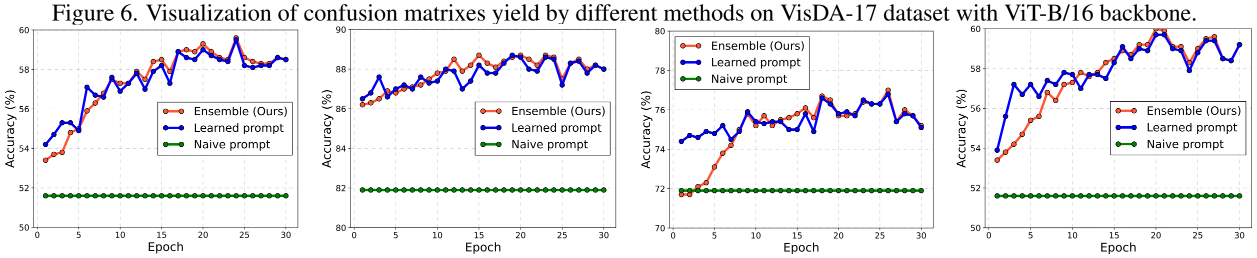


Figure 6. Visualization of confusion matrixes yield by different methods on VisDA-17 dataset with ViT-B/16 backbone.



(a) Ar→Cl (b) Ar→Pr (c) Pr→Ar (d) Pr→Cl

Figure 7. Pseudo-label accuracies of different strategies. We choose four tasks on Office-Home as examples (ResNet-50).

ble pseudo-labels remains lower than that of the learned prompt-based pseudo-labels. As training progresses (after 10 epochs), the ensemble pseudo-labels outperforms the other two strategies. We conjecture the reason is that the incorporation of certain prior knowledge (naive prompts) helps alleviate the risk of overfitting the learned prompts to the source domain. This enables the model to achieve accuracy that cannot be attained by relying solely on learned prompts for pseudo-labeling. Finally, the accuracy of the ensemble gradually converges towards the accuracy of the learned prompts.

E.3. Effectiveness of Parameter-Sharing Strategy

Table 9 shows the impact of parameter-sharing G on the performance of our method. It turns out that the parameter-sharing strategy (w/ PS) slightly outperforms the version without parameter-sharing (w/o PS) on both vision backbones. Parameter-sharing enables a single prompting module G to transform both visual and textual embeddings bidirectionally. This allows richer cross-modal interactions and fusion between the modalities, eliciting better domain- and modality-shared representations. In contrast, without parameter-sharing, the promptings of vision of vision and text modalities will be more independent, and more tunable parameters make it difficult to train and less effective. The consistently positive gains across various backbones indicate that parameter-sharing is an effective and generalizable design choice for mutual prompting in DAMP. It enables a single compact module to prompt both modalities flexibly for domain adaptation.

References

[1] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, pages 456–473, 2018. 3

[2] Haoran Chen, Zuxuan Wu, Xintong Han, and Yu-Gang Jiang. Multi-prompt alignment for multi-source unsupervised domain adaptation. *arXiv preprint arXiv:2209.15210*, 2022. 3

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR workshops*, pages 702–703, 2020. 1

[4] Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *ICCV*, pages 1657–1664, 2013. 3

[5] Yangye Fu, Ming Zhang, Xing Xu, Zuo Cao, Chao Ma, Yanli Ji, Kai Zuo, and Huimin Lu. Partial feature selection and alignment for multi-source domain adaptation. In *CVPR*, pages 16654–16663, 2021. 3

[6] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189. PMLR, 2015. 3, 4

[7] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji

Table 9. Classification accuracies of our method with (w/) and without (w/o) Parameter-Sharing (PS) strategy in the mutual prompting module. The results are obtained on **Office-Home** dataset.

Method	f_v	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.**
DAMP w/o PS	ResNet-50	59.5	88.6	86.4	76.5	89.0	86.8	76.6	59.8	86.9	77.1	60.4	89.5	78.1
DAMP w/ PS		59.7	88.5	86.8	76.6	88.9	87.0	76.3	59.6	87.1	77.0	61.0	89.9	78.2
DAMP w/o PS	ViT-B/16	75.9	93.7	92.2	86.1	94.3	91.7	85.7	76.1	92.0	85.5	76.1	93.6	86.9
DAMP w/ PS		75.7	94.2	92.0	86.3	94.2	91.9	86.2	76.3	92.4	86.1	75.6	94.0	87.1

Song, Shuang Li, and Gao Huang. Domain adaptation via prompt learning. *TNNLS*, pages 1–11, 2023. 1, 3, 4

[8] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2020. 3

[9] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, pages 5542–5550, 2017. 3

[10] Ruihuang Li, Xu Jia, Jianzhong He, Shuaijun Chen, and Qinghua Hu. T-svdnet: Exploring high-order prototypical correlations for multi-source domain adaptation. In *ICCV*, pages 9991–10000, 2021. 3

[11] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105. PMLR, 2015. 3

[12] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, pages 1406–1415, 2019. 2, 3

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 3, 4

[14] Chuan-Xian Ren, Yong-Hui Liu, Xi-Wen Zhang, and Ke-Kun Huang. Multi-source unsupervised domain adaptation via pseudo target domain. *TIP*, 31:2122–2135, 2022. 3

[15] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, pages 3723–3732, 2018. 3

[16] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, pages 443–450. Springer, 2016. 3, 4

[17] Naveen Venkat, Jogendra Nath Kundu, Durgesh Singh, Ambareesh Revanur, et al. Your classifier can secretly suffice multi-source domain adaptation. pages 4647–4659, 2020. 3

[18] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5018–5027, 2017. 2, 3

[19] Hang Wang, Minghao Xu, Bingbing Ni, and Wenjun Zhang. Learning to combine: Knowledge aggregation for multi-source domain adaptation. In *ECCV*, pages 727–744. Springer, 2020. 3

[20] Xin Zhang, Shixiang Shane Gu, Yutaka Matsuo, and Yusuke Iwasawa. Domain prompt learning for efficiently adapting clip to unseen domains. *TJSAI*, 38(6):B–MC2_1, 2023. 3, 4

[21] Sicheng Zhao, Guangzhi Wang, Shanghang Zhang, Yang Gu, Yaxian Li, Zhichao Song, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source distilling domain adaptation. In *AAAI*, pages 12975–12983, 2020. 3

[22] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 130(9):2337–2348, 2022. 3

[23] Yongchun Zhu, Fuzhen Zhuang, and Deqing Wang. Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources. In *AAAI*, pages 5989–5996, 2019. 3