Split to Merge: Unifying Separated Modalities for Unsupervised Domain Adaptation

Supplementary Material

Algorithm 1 Training Algorithm of UniMoS

Input: Labeled source data $\{x^s, y^s\}$, unlabeled target data x^t, maximum epoch number max_epoch, pretrained CLIP text encoder g_{txt} and vision encoder g_{vis}.
Output: Trained modality separator G_{txt}, G_{vis}, and trained linear layers Φ_1, Φ_2.

```
 1: Let $epoch = 0$.
 2: while $epoch < max\_epoch$ do
 3:     Obtain teacher output $y^t_{lac}$ via Eq. (1).
 4:     if $epoch \mod 2 == 0$ then
 5:         Take mixed outputs as target vision pseudo label via Eq. (3).
 6:     else
 7:         Obtain target vision pseudo label $y^t_{vac}$ via Eq. (2).
 8:     end if
 9:     Obtain CLIP-extracted vision features $\{f^s_v, f^t_v\} = g_{vis}(x^s, x^t)$ and text features $\mu_i = g_{txt}(t_i)$.
10:     Obtain LAC outputs $\hat{y}_{lac}$ by Eq. (4).
11:     Obtain VAC outputs $\hat{y}_{vac}$ by Eq. (5).
12:     Obtain ensemble outputs $\hat{y}_{ens}$ by Eq. (7).
13:     Obtain modality discrimination outputs $y_{dis}$ by Eq. (6).
14:     Update network parameters via Eq. (8).
15:     Let $epoch = epoch + 1$
16: end while
17: return solution
```

1. Training algorithm

For clarity, we give full training algorithm of UniMoS. We first obtain teacher output from CLIP's zero-shot inference results by Eq. (1):

$$y^t_{lac} = (l_1 - \bar{l}, l_2 - \bar{l}, \cdots, l_k - \bar{l}), \quad l_i = \cos(\mu_i, f^t_v)/T. \tag{1}$$

Then obtain pseudo label by Eq. (2):

$$\hat{y}^t_{vac} = \arg\max_k \cos(f^t_b, \phi_k). \tag{2}$$

To avoid potential error accumulations [2] in pseudo labels, we directly apply mixed outputs from both modalities as pseudo labels in certain epochs:

$$\hat{y}^t_{vac} = \lambda \cdot \hat{y}^t_{vac} + (1-\lambda) \cdot \tilde{y}^t_{lac}, \tag{3}$$

where λ is a fixed mixup ratio 0.3 that combines outputs from both modalities for inference. LAC and VAC outputs are obtained via Eq. (4) and Eq. (5), respectively.

$$\hat{y}_{lac} = (\hat{l}_1, \hat{l}_2, \cdots \hat{l}_k), \quad \hat{l}_i = \cos(\mu_i, f_{lac})/T. \tag{4}$$

$$\hat{y}_{vac} = \Phi_2(f_b), \quad f_b = \Phi_1(f_{vac}). \tag{5}$$

We further train a modality discriminator to align modality features from both domains via Eq. (6):

$$\mathcal{L}_{bce} = -[y_{dis} \log \hat{y}_{dis} + (1 - y_{dis}) \log(1 - \hat{y}_{dis})], \tag{6}$$

We assemble LAC and VAC during training to facilitate modality information exchanges by Eq. (7):

$$\hat{y}^t_{ens} = w \cdot \hat{y}^t_{vac} + (1-w) \cdot \tilde{y}^t_{lac}. \tag{7}$$

Note that the w in Eq. (7) is trainable, which dynamically changes to suit different training phases. While the λ in Eq. (3) is fixed and only used for retrieving pseudo labels and inference, thus does not affect training. Based on the training objectives:

$$\begin{aligned}
\theta_{G_{txt}} &= \arg\min_{\theta_{G_{txt}}} \mathcal{L}_{lac} + \gamma\mathcal{L}_{ortho} + \gamma\mathcal{L}_{bce}, \\
\theta_{G_{vis}} &= \arg\min_{\theta_{G_{vis}}} \mathcal{L}_{vac} + \gamma\mathcal{L}_{ortho} + \gamma\mathcal{L}_{bce}, \\
\theta_W, \theta_{\Phi_1}, \theta_{\Phi_2} &= \arg\min_{\theta_W, \theta_{\Phi_1}, \theta_{\Phi_2}} \mathcal{L}_{vac}, \\
\theta_D &= \arg\min_{\theta_D} \gamma\mathcal{L}_{bce},
\end{aligned} \tag{8}$$

we update the trainable parameters. Training algorithm is given in Algorithm 1. Please refer to main paper for full descriptions of the equations above.

All trainable modules in UniMoS are fully-connected linear layers, which brings very low computation costs. Specifically, the modality separators G_{txt}, G_{vis} are two separate linear layers of shape (d_v, d_v). The bottleneck feature dimension d_b is 256, thus Φ_1 is of shape (d_v,256) with batch normalization, and Φ_2 is of shape (256,K). The modality discriminator D consists of two linear layers (d_v,256) and (256,1) with a ReLU activation layer. The weight generator W consists of two linear layers (d_v,256) and (256,1) with a Sigmoid activation layer.

2. Additional experiments

In this section we provide full results of our experiments.

Mini-DomainNet. We present full results on Mini-DomainNet [8, 13, 21] in Tab. 1 using two backbones. We set $\alpha = 0.1$, $\beta = 1$ and $\gamma = 0.01$ for all tasks. We can observe significant improvements over current SOTA AD-CLIP [14] by 2.8% in average with ResNet50 [4], and 0.4% improvement using ViT [1]. We are unable to compare with

Table 1. Results on Mini-DomainNet. Best results are marked in bold font. Methods with '*' are based on CLIP.

Method	Backbone	clp→pnt	clp→rel	clp→skt	pnt→clp	pnt→rel	pnt→skt	rel→clp	rel→pnt	rel→skt	skt→clp	skt→pnt	skt→rel	Avg.
CLIP* [11]	ResNet50	67.9	84.8	62.9	69.1	84.8	62.9	69.2	67.9	62.9	69.1	67.9	84.8	71.2
DAPrompt* [3]		72.4	87.6	65.9	72.7	87.6	65.6	73.2	72.4	66.2	73.8	72.9	87.8	74.8
ADCLIP* [14]		71.7	88.1	66.0	73.2	86.9	65.2	73.6	73.0	68.4	72.3	74.2	**89.3**	75.2
UniMoS* (ours)		**76.0**	**88.9**	**72.1**	**75.5**	**89.2**	**71.1**	**75.1**	**75.9**	**70.5**	**76.4**	**76.3**	88.9	**78.0**
CLIP* [11]	ViT-B	80.3	90.5	77.8	82.7	90.5	77.8	82.7	80.3	77.8	82.7	80.3	90.5	82.8
DAPrompt* [3]		83.3	92.4	81.1	86.4	92.1	81.0	86.7	83.3	80.8	86.8	83.5	91.9	85.8
ADCLIP* [14]		84.3	**93.7**	82.4	**87.5**	**93.5**	82.4	**87.3**	84.5	81.6	**87.9**	84.8	93.0	86.9
UniMoS* (ours)		**86.2**	93.2	**83.2**	86.9	93.2	**83.2**	86.8	**86.0**	**82.8**	87.0	**86.2**	**93.3**	**87.3**

Table 2. Results on Office-Home. Best results are marked in bold font. Methods with '*' are based on CLIP.

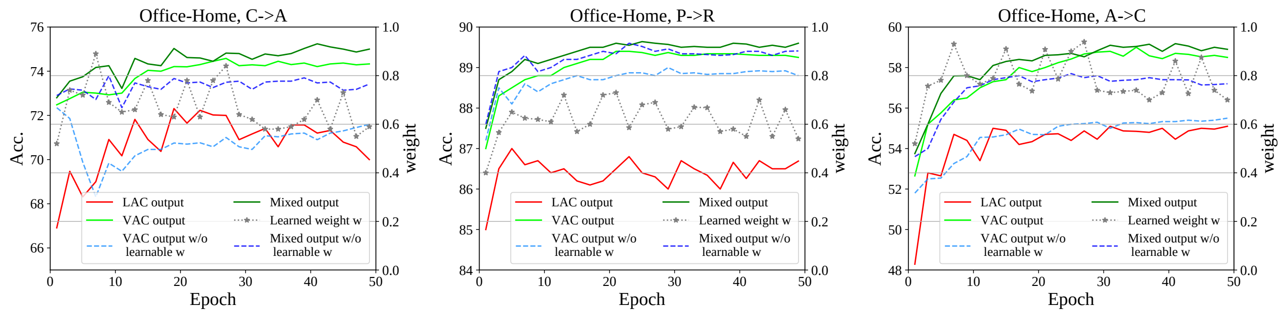
Method	Backbone	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg.
SourceOnly [4]	ResNet50	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
ParetoDA [6]		56.8	75.9	80.5	64.4	73.5	73.7	65.6	55.2	81.3	75.2	61.1	83.9	70.6
CLIP* [11]		51.7	81.5	82.3	71.7	81.5	82.3	71.7	51.7	82.3	71.7	51.7	81.5	71.8
SDAT [12]		58.2	77.1	82.2	66.3	77.6	76.8	63.3	57.0	82.2	74.9	64.7	86.0	72.2
MSGD [18]		58.7	76.9	78.9	70.1	76.2	76.6	69.0	57.2	82.3	74.9	62.7	84.5	72.3
Fixbi [10]		58.1	77.3	80.4	67.7	79.5	78.1	65.8	57.9	81.7	76.4	62.9	86.7	72.7
CST [9]		59.0	79.6	83.4	68.4	77.1	76.7	68.9	56.4	83.0	75.3	62.2	85.1	72.9
ATDOC [7]		60.2	77.8	82.2	68.5	78.6	77.9	68.4	58.4	83.1	74.8	61.5	87.2	73.2
KUDA [15]		58.2	80.0	82.9	71.1	80.3	80.7	71.3	56.8	83.2	75.5	60.3	86.6	73.9
DAPrompt* [3]		54.1	84.3	84.8	74.4	83.7	85.0	74.5	54.6	84.8	75.2	54.7	83.8	74.5
EIDCo [22]		**63.8**	80.8	82.6	71.5	80.1	80.9	72.1	61.3	84.5	**78.6**	65.8	87.1	75.8
ICON [20]		63.3	81.3	84.5	70.3	82.1	81.0	70.3	**61.8**	83.7	75.6	**68.6**	87.3	75.8
ADCLIP* [14]		55.4	85.2	85.6	76.1	85.8	86.2	**76.7**	56.1	85.4	76.8	56.1	85.5	75.9
PADCLIP* [5]		57.5	84.0	83.8	**77.8**	85.5	84.7	76.3	59.2	85.4	78.1	60.2	86.7	76.6
UniMoS* (ours)		59.5	**89.4**	**86.9**	75.2	**89.6**	**86.8**	75.4	58.4	**87.2**	76.9	59.5	**89.7**	**77.9**
SourceOnly [1]	ViT-B	54.7	83.0	87.2	77.3	83.4	85.5	74.4	50.9	87.2	79.6	53.8	88.8	75.5
CLIP* [11]		67.8	89.0	89.8	82.9	89.0	89.8	82.9	67.8	89.8	82.9	67.8	89.0	82.4
TVT [19]		74.9	86.8	89.5	82.8	88.0	88.3	79.8	71.9	90.1	85.5	74.6	90.6	83.6
DAPrompt* [3]		70.6	90.2	91.0	84.9	89.2	90.9	84.8	70.5	90.6	84.8	70.1	90.8	84.0
SSRT [16]		75.2	89.0	91.1	85.1	88.3	89.9	85.0	74.2	91.2	85.7	78.6	91.8	85.4
ADCLIP* [14]		70.9	92.5	92.1	85.4	92.4	**92.5**	**86.7**	74.3	**93.0**	**86.9**	72.6	93.8	86.1
PADCLIP* [5]		**76.4**	90.6	90.8	**86.7**	92.3	92.0	86.0	**74.5**	91.5	**86.9**	**79.1**	93.1	86.7
UniMoS* (ours)		74.9	**94.0**	**92.5**	86.4	**94.3**	**92.5**	86.0	73.9	**93.0**	86.4	74.2	**94.5**	**86.9**
CLIP* [11]	ViT-L	74.2	93.1	93.3	87.3	93.1	93.3	87.3	74.2	93.3	87.3	74.2	93.1	87.0
DAPrompt* [3]		77.3	94.6	94.3	88.6	94.6	94.0	88.8	76.8	94.0	89.0	77.8	94.4	88.7
ADCLIP* [14]		80.3	95.4	**95.7**	**90.9**	95.5	**95.2**	**90.1**	79.6	95.1	**90.8**	81.1	95.9	90.5
UniMoS* (ours)		**80.9**	**96.2**	95.1	90.1	**96.1**	95.1	90.0	**81.4**	**95.2**	89.9	**81.6**	**96.3**	**90.7**

PADCLIP [5] since no public code implementation is available yet.

Office-Home. We provide full results on Office-Home [17] in Tab. 2 using ResNet50 and two variants of ViT as backbone. Utilizing the rich pretrain knowledge in CLIP, CLIP-based methods achieve higher accuracies than single-modality UDA methods. Our method further consistently outperforms all competing methods on all backbones, demonstrating the efficacy of adapting both modalities.

Computation analysis. We provide comprehensive computation analysis in Tab. 3. The GPU memory consumption is computed using consistent batch size of 32. The results on DomainNet is computed on task 'clp→inf'. Note our method requires only one forward through CLIP to obtain pre-extracted features, and we have included the time for pre-extraction in the 'Train time' columns. According to Sec. 1, only a few linear layers is trained in our method, which brings significant computation efficiency, allowing agile applications of our method. We can observe that the trainable parameters of DAPrompt differs on different datasets. The reason is that DAPrompt trains the prompt embedding for each class respectively, therefore the trainable parameters and GPU consumption increase drastically as class number increases from 12 (VisDA) to 345 (DomainNet).

Learnable ensemble weight w. Fig. 1 provides additional examples on the effects of dynamic ensemble weight on three tasks from Office-Home dataset. We can draw the conclusion that the accuracy of VAC drops drastically without learning ensemble weight w, which further affects the over-



(a) Task C→A (b) Task P→R (c) Task A→C

Figure 1. Examples on the effects of ensemble weight w.

Table 3. Computation analysis. Best results are marked in bold font. Methods with '*' are based on CLIP.

Method	Dataset	Backbone	Param.	Throughput (imges/s)	Train time	FLOPs	GPU mem.	Acc.
DAPrompt*	VisDA	ResNet101	1.2M	244	4.3H	11.3G	6.9G	86.9
FixBi			86.1M	102	5.5H	15.73G	17.0G	87.2
CAN			42.5M	31	10.5H	7.9G	11.3G	87.2
PADCLIP*			-	-	23.5H	-	-	**88.5**
UniMoS* (ours)			**0.79M**	**2667**	**0.5H**	**<0.01G**	**1.8G**	88.1
PMTrans	DomainNet	Swin	89.5M	46	30H	15.2G	19.3G	52.4
DAPrompt*		ViT-B	34.5M	31	7.9H	73.9G	22.5G	59.8
UniMoS* (ours)			**0.79M**	**2827**	**0.4H**	**<0.01G**	**2.9G**	**63.6**

all mixed output accuracy. We can observe in Fig. 1b that although accuracy of 'VAC output w/o learnable w' is lower than the full design, the final 'Mixed output w/o learnable w' achieves comparable accuracy with the full design. The reason is that the task P→R is relatively easier, thus complementary knowledge in LAC is able to support VAC for classification.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[2] Yulu Gan, Yan Bai, Yihang Lou, Xianzheng Ma, Renrui Zhang, Nian Shi, and Lin Luo. Decorate the newcomers: Visual domain prompt for continual test time adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7595–7603, 2023.

[3] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji Song, Shuang Li, and Gao Huang. Domain adaptation via prompt learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11, 2023.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[5] Zhengfeng Lai, Noranart Vesdapunt, Ning Zhou, Jun Wu, Cong Phuoc Huynh, Xuelu Li, Kah Kuen Fu, and Chen-Nee Chuah. Padclip: Pseudo-labeling with adaptive debiasing in clip for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16155–16165, 2023.

[6] Jian Liang, Kaixiong Gong, Shuang Li, Chi Harold Liu, Han Li, Di Liu, Guoren Wang, et al. Pareto domain adaptation. *Advances in Neural Information Processing Systems*, 34:12917–12929, 2021.

[7] Jian Liang, Dapeng Hu, and Jiashi Feng. Domain adaptation with auxiliary target domain-oriented classifier. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16632–16642, 2021.

[8] 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*, pages 7640–7650, 2023.

[9] Hong Liu, Jianmin Wang, and Mingsheng Long. Cycle self-training for domain adaptation. *Advances in Neural Information Processing Systems*, 34:22968–22981, 2021.

[10] Jaemin Na, Heechul Jung, Hyung Jin Chang, and Wonjun Hwang. Fixbi: Bridging domain spaces for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1094–1103, 2021.

[11] 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 supervi-

sion. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[12] Harsh Rangwani, Sumukh K Aithal, Mayank Mishra, Arihant Jain, and Venkatesh Babu Radhakrishnan. A closer look at smoothness in domain adversarial training. In *International Conference on Machine Learning*, pages 18378–18399. PMLR, 2022.

[13] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8050–8058, 2019.

[14] Mainak Singha, Harsh Pal, Ankit Jha, and Biplab Banerjee. Ad-clip: Adapting domains in prompt space using clip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4355–4364, 2023.

[15] Tao Sun, Cheng Lu, and Haibin Ling. Prior knowledge guided unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 639–655. Springer, 2022.

[16] Tao Sun, Cheng Lu, Tianshuo Zhang, and Haibin Ling. Safe self-refinement for transformer-based domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7191–7200, 2022.

[17] 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.

[18] Haifeng Xia, Taotao Jing, and Zhengming Ding. Maximum structural generation discrepancy for unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3434–3445, 2022.

[19] Jinyu Yang, Jingjing Liu, Ning Xu, and Junzhou Huang. Tvt: Transferable vision transformer for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 520–530, 2023.

[20] Zhongqi Yue, Hanwang Zhang, and Qianru Sun. Make the u in uda matter: Invariant consistency learning for unsupervised domain adaptation. *arXiv preprint arXiv:2309.12742*, 2023.

[21] Wenyu Zhang, Li Shen, and Chuan-Sheng Foo. Rethinking the role of pre-trained networks in source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18841–18851, 2023.

[22] Yixin Zhang, Zilei Wang, Junjie Li, Jiafan Zhuang, and Zihan Lin. Towards effective instance discrimination contrastive loss for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11388–11399, 2023.