Supplementary Material for Order-preserving Consistency Regularization for Domain Adaptation and Generalization

Mengmeng Jing[1,2]*, Xiantong Zhen[2] †, Jingjing Li[1], Cees G. M. Snoek[2]
[1]University of Electronic Science and Technology of China
[2]University of Amsterdam
jingmeng1992@gmail.com zhenxt@gmail.com lijin117@yeah.net
c.g.m.snoek@uva.nl

Feature Visualization. In Fig. 1, the upper row shows the t-SNE [8] visualization of the original representations, we observe that ERM and prediction-based consistency regularization do not distinguish well between categories, while OCR separates different classes more clearly. The bottom row represents the t-SNE of the residual components, we observe an opposite phenomenon to the upper row: The residual components of different classes learned by OCR are mixed together, i.e., less task-related information. As a comparison, for the model trained with ERM, the mixing degree of different categories is not high, which means that the residual components of ERM still retain some task-related information, and the model will be affected by domain-specific attributes. Prediction-based Consistency regularization improves ERM, but it cannot eliminate the domain-specific information well compared with OCR.

Pseudocode for OCR. For a more practical understanding, we summarize the pseudocode of the proposed OCR in **Algorithm 1**.

Detailed Results for Office-Home. In Table 1, we report the detailed results on Office-Home dataset.

Detailed Parameters for Data Augmentations. In Table 2-Table 4, we present the detailed parameters of transformations for the augmented images in four datasets. We describe data augmentation using the PyTorch notations.

Among them, `GaussianNoise` overlaps with the corruption in CIFAR-C. However, Gaussian Noise is only one of the 15 corruptions in CIFAR-C, with minimal impact on the final result. After removing the `GaussianNoise` data augmentation, the test error on GaussianNoise data increases from 36.5 to 37.9, while the average test error of 15 corruptions increases from 31.32 to 31.41. Therefore, our improvements mainly come from the proposed method, the effect of the overlapped data augmentation is minimal.

Implementation Details. In the experiments, we test our method on four different cross-domain tasks: domain adaptation, test-time adaptation, domain generalization classification and domain generalization sementic segmentation.

For the domain adaptation task, the experiment is based on CDAN [6] and SHOT [4], respectively. For the source-free setting of domain adaptation, we use ResNet-50 pre-trained on ImageNet as the backbone. We replace the final FC layer with a bottleneck layer of 256 units and an FC classifier layer. A BN layer and a weight normalization layer are put after the bottleneck layer and classifier layer, respectively. We trained the whole network by back-propagation, with the penultimate bottleneck layer having a learning rate 10 times that of the backbone layers. The weights of the classifier are fixed. We employ mini-batch SGD optimizer with a momentum of 0.9, a weight decay of $1e{-}3$ and learning rate $1e{-}2$ for the bottleneck layers and $1e{-}3$ for the remaining layers. The batch size is 48. For the hyper-parameter, we set $\lambda_0 = 0.7$. The weight for the newly added OCR is 0.2.

For the source-use setting of domain adaptation, We train the model through back propagation. The learning rate of the FC layer is 10 times that of the remaining layers. We employ mini-batch SGD optimizer with momentum 0.9 and learning rate of t-th iteration $\gamma_t = \gamma_0(1 + \alpha t/T)^{-\beta}$, where T is the total iteration number. We set $\lambda_0 = 0.7$ and the weight of OCR loss as 0.2.

For the test-time adaptation task, the implementation is based on [9]. we use ResNeXt-29 pre-trained on the clean CIFAR100 dataset as the backbone. We update the model only one step per iteration. We adopt the Adam optimizer with learning rate 1e-3. Other parameters are set according to [9]. We set $\lambda_0 = 0.8$ and the weight of OCR loss as 0.2.

For the domain generalization classification, the implementation is based on Dassl.pytorch [11]. We use the mini-

*This work was done when Mengmeng Jing was a visiting student at University of Amsterdam.

†Currently with United Imaging Healthcare, Co., Ltd., China.

Algorithm1: Order-preserving Consistency Regularization

```
# g: backbone model
# f: classifier
# Aug: augmentation function
# R: the disentanglement function
# get_lambda: function to compute lambda in Eq. (4)
# H: function to compute entropy
for x in dataloader:        # load a minibatch x
    x2 = Aug(x)        # random augmentation
    z1, z2 = g(x), g(x2)        # representations
    lambda = get_lambda()        # the ratio
    zn = R(z2, z1, lambda)        # domain-variant rep.
    yn = f(zn)        # prediction of domain-variant rep.
    L = H(Softmax(yn))        # OCR loss
    L.backward()        # back-propagate
    update(g, f)        # SGD update
```

batch SGD optimizer with the initial learning rate of 0.002 (except for cartoon with a initial learning rate of 0.02). The learning rate scheduler is cosine. The total optimization epoch is 50. For the hyper-parameters, we set $\lambda_0 = 0.5$. The weight for the newly added OCR loss is 0.2.

For the domain generalization semantic segmentation, the implementation is based on [2]. We use DeepLabV3+ [1] as the backbone model. We employ the mini-batch SGD optimizer with an initial learning rate of 1e-3 and momentum of 0.9. We adopt the polynomial learning rate scheduling [5] whose power is 0.9. The model is trained for 40K iterations. We set the batch size as 2. For the hyper-parameter, we set $\lambda_0 = 0.1$. The weight for the newly added OCR loss is 0.3.

Table 1. Accuracy (%) on Office-Home with ResNet-50 as backbone. "P-Cons. Regular." denotes prediction-based consistency regularization. The best results are highlighted by bold numbers. "SF" denotes source-free.

Method	SF	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.
MCD [7]	×	48.9	68.3	74.6	61.3	67.6	68.8	57.0	47.1	75.1	69.1	52.2	79.6	64.1
w/ OCR [7]	×	51.6	70.7	76.8	64.1	69.7	71.6	59.7	49.6	78.0	71.3	54.7	81.8	66.6
CDAN [6]	×	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
w/ OCR	×	54.0	74.2	78.5	61.1	73.4	71.2	60.7	53.4	80.6	72.2	54.2	82.7	68.0
ResNet-50 [3]	√	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
Source-only	√	44.6	67.3	74.8	52.7	62.7	64.8	53.0	40.6	73.2	65.3	45.4	78.0	60.2
NRC [10]	√	57.7	77.9	81.5	68.3	78.1	78.6	66.2	56.0	83.0	72.4	58.7	84.4	71.9
w/ P-Cons. Regular.	√	57.2	78.1	82.1	68.3	78.4	78.1	66.1	56.6	82.7	74.0	59.0	84.6	72.1
w/ OCR	√	56.9	**79.1**	**82.9**	**69.5**	78.3	79.7	**67.9**	56.4	82.6	74.7	59.0	83.8	72.6
SHOT [4]	√	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
w/ P-Cons. Regular.	√	56.2	77.3	82.6	68.9	78.4	79.5	66.9	53.7	82.8	74.0	58.5	**84.8**	72.0
w/ OCR	√	**58.4**	78.2	81.4	68.2	**78.6**	**80.2**	67.5	**57.7**	**83.1**	**75.2**	**60.0**	84.6	**72.8**

Table 2. Parameters of the transformations for augmented images in PACS dataset. "Prob." means the applying probability.

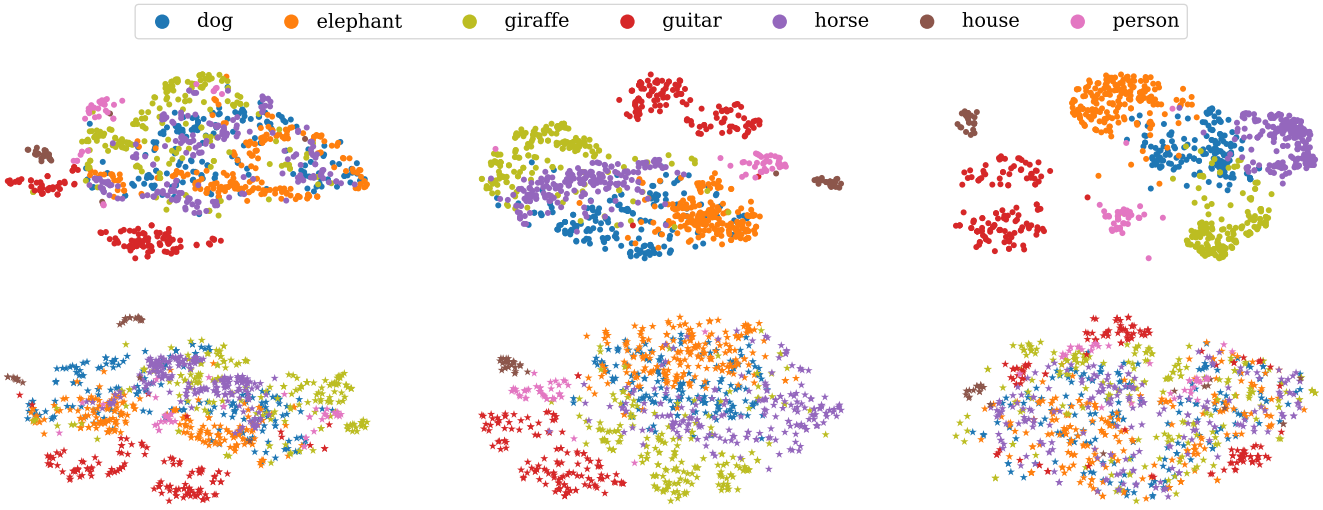
Transformation	Parameter	Prob.
`RandomResize`	height=224, width=224	1.0
`ColorJitter`	brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1	1.0
`RandomGrayscale`		0.2
`GaussianBlur`	kernel_size=21	0.5
`RandomHorizontalFlip`		0.5
`Random2DTranslation`		0.5

Table 3. Parameters of the transformations for augmented images in Office-Home and GTAV. "Prob." means the applying probability.

Transformation	Parameter	Prob.
`RandomResizedCrop`	size=224, scale=(0.2,1.0)	1.0
`ColorJitter`	brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1	0.8
`RandomGrayscale`		0.2
`GaussianBlur`	kernel_size=[0.1, 2.0]	0.5
`RandomHorizontalFlip`		0.5

Table 4. Parameters of the transformations for augmented image in CIFAR100-C dataset. "Prob." means the applying probability.

Transformation	Parameter	Prob.
`ColorJitter`	brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1	0.8
`GaussianBlur`	kernel_size=5,sigma=[0.001,0.5]	0.75
`CenterCrop`	size=32	1.0
`RandomHorizontalFlip`		0.5
`GaussianNoise`	mean=0, std=0.005	0.75



(1) ERM (2)P-Cons. Reg. (3) OCR

Figure 1. t-SNE [8] visualization. The upper row shows the t-SNE visualization of the original representations, we observe that ERM and consistency regularization do not distinguish well between categories, while our method separates different classes more clearly. The bottom row represents the t-SNE visualization of the residual components, the residual components of different classes extracted by OCR are mixed together, i.e., less task-related information.

References

[1] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 2

[2] Sungha Choi, Sanghun Jung, Huiwon Yun, Joanne T Kim, Seungryong Kim, and Jaegul Choo. Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11580–11590, 2021. 2

[3] 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. 3

[4] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020. 1, 3

[5] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015. 2

[6] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in neural information processing systems*, pages 1640–1650, 2018. 1, 3

[7] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018. 3

[8] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(11), 2008. 1, 4

[9] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, 2022. 1

[10] Shiqi Yang, Yaxing Wang, Joost van de Weijer, Luis Herranz, and Shangling Jui. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. In *Advances in Neural Information Processing Systems*, volume 34, 2021. 3

[11] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain adaptive ensemble learning. *IEEE Transactions on Image Processing*, 30:8008–8018, 2021. 1