Improved Self-Training for Test-Time Adaptation

Supplementary Material

In Appendix, we provide additional experimental results at Section 1, 2, and 3, and discuss the limitation and social impact of our work at Section 4.

1. More Experimental Results

1.1. Prompt Tuning for CLIP

Some recent works [5, 7, 10, 11] have shown that prompt tuning can improve the robustness of CLIP [5]. CoOp [11] and CoCoOp [10] propose to use a few labeled training samples to tune the prompt and achieve better robustness. TPT [7] optimizes the prompt to encourage consistent predictions across augmented views by minimizing the marginal entropy. In contrast to CoOp and CoCoOp, TPT does not require any labeled data and can be applied to any downstream tasks at test time in an online manner.

We show that our method can be employed to prompt tuning for CLIP. We follow TPT and adopt episodic test-time optimization to tune the prompt on the base model for every adaptation step. *We modify the source code of TPT, which differ from the experimental implementation described in the main paper and lead to a slight performance variation.* The results are shown in Table 1. Our method achieves the best average performance on ImageNet and its variants.

1.2. More Comparison

In Table 2, we conduct comparisons with EATA [4] and TTAC [8], as well as the offline performance of our method on ImageNet-C and CIFAR10-C. **a)** EATA [4] enhances its robustness by applying a reliable sample selection criterion to corrupted images with low source accuracy. However, fully utilizing limited data in online learning can also significantly improve the model's adaptive performance. **b)** The pseudo-label refinement proposed by TTAC [8], which uses source features as additional supervision, indeed reduces more noise in pseudo-labels.

2. Implementation Details

2.1. Object Detection

For object detection experiments, we utilize YOLOv3 [6] as the backbone, and pre-train it on the KITTI-Clear dataset using an SGD optimizer with a learning rate of $1e-4$ and a batch size of 30. At test time, SimpAug is applied to an input image and produces 32 augmented views. The detection boxes are mapped to the original image according to the transformation parameters (i, j, h, w, hf) of SimpAug for each view. Then, boxes are filtered by a confidence threshold of 0.001. We store these boxes with their features in the memory queue with a capacity of $10K$. Because the features lie in three different scales, we run PLCA on each scale separately. Non-maximum suppression (NMS) is applied to the corrected boxes with a confidence threshold of 0.25 and an IoU threshold of 0.45. The final boxes are considered as targets to compute the loss and update the model.

2.2. Semantic Segmentation

For semantic segmentation experiments, we adapt a DeepLab-V2 [1] with a ResNet-101 backbone pre-trained

Method	Supervised	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
Zero-shot [5]	✗	58.16	21.83	51.41	56.15	33.37	44.18	40.69
Ensemble [5]	✗	59.81	23.24	52.91	**60.72**	**35.48**	46.43	43.09
CoOp [11]	✓	**63.33**	23.06	55.40	56.60	34.67	46.61	42.43
CoCoOp [10]	✓	62.81	23.32	**55.72**	57.74	34.48	46.81	42.82
TPT [7]	✗	60.74	**26.67**	54.70	59.11	35.09	47.26	43.89
Ours (Online)	✗	61.20	26.48	54.94	60.53	35.35	**47.70**	**44.33**
Zero-shot [5]	✗	66.73	47.87	60.86	73.98	46.09	59.11	57.20
Ensemble [5]	✗	68.34	49.89	61.88	77.65	48.24	61.20	59.42
CoOp [11]	✓	**71.51**	49.71	**64.20**	75.21	47.99	61.72	59.28
CoCoOp [10]	✓	71.02	50.63	64.07	76.18	**48.75**	62.13	59.91
TPT [7]	✗	68.98	**54.77**	63.45	77.06	47.94	62.44	60.81
Ours (Online)	✗	69.63	54.50	63.78	**78.35**	48.56	**62.96**	**61.30**

Table 1. Prompt tuning for CLIP-RN50 (top) and CLIP-ViT-B/16 (bottom) on ImageNet and its variants at online test time. CoOp and CoCoOp are tuned on ImageNet using 16-shot training samples per class, and others require no labeled data. "Average" denotes the average accuracy on all five datasets, and "OOD Average" on out-of-distribution datasets, including ImageNet-A, -V2, -R, and -Sketch.

Methods	SF	Gauss	Shot	Impul	Defcs	Gls	Mtn	Zm	Snw	Frst	Fg	Brt	Cnt	Els	Px	Jpg	Mean
Source	✓	28.8	22.9	26.2	9.5	20.6	10.6	9.3	14.2	15.3	17.5	7.6	20.9	14.7	41.3	14.7	18.3±0.00
TTT++ (Offline) [3]	✓	12.8	11.1	11.2	7.3	17.1	8.2	6.5	9.4	9.9	7.9	5.0	5.1	13.7	8.8	10.6	9.6±0.00
Ours (Offline)	✓	11.8	10.4	12.9	6.3	14.4	7.2	6.6	8.1	8.0	8.3	5.4	6.7	11.4	8.6	11.7	9.2±0.03
TTT++ (Online) [3]	✓	15.5	14.1	23.6	9.1	25.1	11.4	8.1	13.2	13.1	13.4	6.6	6.9	17.6	12.5	13.6	13.6±0.03
EATA (Online) [4]	✓	18.7	16.6	22.6	9.5	22.6	10.7	10.2	13.4	13.7	15.4	7.9	12.4	16.8	15.6	18.0	14.9±0.06
Ours (Online)	✓	12.8	11.4	14.9	6.7	15.8	7.7	6.9	8.9	8.6	10.1	5.6	8.0	11.9	10.5	12.8	10.2±0.02
Source	✓	98.4	97.7	98.4	90.6	92.5	89.8	81.8	89.5	85.0	86.3	51.1	97.2	85.3	76.9	71.7	86.2±0.00
SHOT (Offline) [2]	✓	73.8	70.5	72.2	79.2	80.6	58.5	54.0	53.6	63.0	47.3	39.2	97.7	48.7	46.1	53.0	62.5±0.00
Ours (Offline)	✓	70.1	66.3	67.5	73.6	75.1	62.5	53.9	54.4	60.4	46.8	38.6	81.5	48.3	45.6	48.2	59.5±0.11
TTAC (Online) [8]	✗	72.1	69	71.7	77.4	76.3	66.9	56.1	58.9	61.6	48.6	38.4	78.9	49.2	47.9	50.7	61.6±0.05
SHOT (Online) [2]	✓	83.9	82.3	83.7	83.9	83.8	72.6	61.9	65.7	68.6	54.8	39.4	85.9	58.1	53.1	62.3	69.3±0.03
EATA (Online) [4]	✓	74.1	72.4	74.1	78.8	78.3	68.8	57.9	59.7	64.0	49.8	42.3	80.7	51.8	49.3	53.0	63.7±0.06
Ours (Online)	✓	72.9	70.8	73.1	80.7	79.7	69.6	57.4	59.8	63.1	50.0	39.3	83.9	51.8	48.5	50.8	63.4±0.03

Table 2. Top-1 Classification Error (%) for all corruptions on CIFAR-10C (level 5) (top) and ImageNet-C (level 5) (bottom). Lower is Better. SF denotes source free.

Methods	FB	Gauss	Shot	Impul	Defcs	Gls	Mtn	Zm	Snw	Frst	Fg	Brt	Cnt	Els	Px	Jpg	Mean
Source	-	28.8	22.9	26.2	9.5	20.6	10.6	9.3	14.2	15.3	17.5	7.6	20.9	14.7	41.3	14.7	18.3±0.00
Ours (Online)	✓	16.7	13.9	18.8	7.2	16.6	8.3	7.6	11.3	10.7	11.4	6.5	12.4	11.9	13.3	12.3	11.9±0.03
Ours (Online)	✗	12.8	11.4	14.9	6.7	15.8	7.7	6.9	8.9	8.6	10.1	5.6	8.0	11.9	10.5	12.8	10.2±0.04
Source	-	98.4	97.7	98.4	90.6	92.5	89.8	81.8	89.5	85.0	86.3	51.1	97.2	85.3	76.9	71.7	86.2±0.00
Ours (Online)	✓	75.9	99.8	80.6	75.8	76.0	70.5	60.7	61.2	64.1	53.3	40.7	99.7	54.1	48.8	52.7	67.6±0.09
Ours (Online)	✗	72.9	70.8	73.1	80.7	79.7	69.6	57.4	59.8	63.1	50.0	39.3	83.9	51.8	48.5	50.8	63.4±0.03

Table 3. Top-1 Classification Error (%) for all corruptions on CIFAR-10C (level 5) (top) and ImageNet-C (level 5) (bottom). Lower is Better. FB denotes freeze BatchNorm layers of the backbone.

on the clear split of CarlaTTA. The model is trained with an SGD optimizer using a learning of $2.5e-4$ and a batch size of 16. At test time, SimpAug produces 32 augmented views for each input image, and the segmentation masks are mapped to the original image according to the transformation parameters (i, j, h, w, hf) of SimpAug for each view. Since runnig PLCA for every pixel and its features is computationally expensive, we extract objects by separating the masks into sub-masks according to the predictive classes, and average the 3D feature maps of each sub-mask to obtain 1D vectors. Then, we run PLCA on these objects and their features to obtain corrected pseudo-labels, which are assigned to all pixels in the corresponding sub-mask. Non-maximum suppression (NMS) is applied for every pixel prediction in the original image. The final pixel-level pseudo-labels are considered as targets for model updating.

3. Ablation Study

3.1. Detailed Study for α and "Maxiter" in PLCA

We perform ablation studies on these two parameters in Table 4 and Table 5. The α determines the relative amount of information a sample receives from its neighbors and its initial label information in a single iteration of Eqn.4. α is an update step for PLCA; if it is too small, it leads to oscillations during iterations, and if it is too large, the convergence speed is slow. "Maxiter" controls the maximum number of iterations for solving the linear system of Eqn.7 using the conjugate gradient method, which directly impacts the computational cost of PLCA.

α (maxiter=20)		0.999	0.99	0.9	0.5	0.1	0.0	Source
RN50	CIFAR10	76.9	**78.9**	76.6	72.0	70.7	70.4	68.7
ViT-B/32	Food101	85.4	**86.1**	85.0	83.2	82.6	82.5	80.7

Table 4. Ablation study of α in PLCA.

maxiter (α=0.99)		100	50	20	10	5	1	Source
usage time (seconds)		8.74	6.30	3.72	2.13	1.79	1.25	N/A
RN50	CIFAR10	**79.0**	79.0	78.9	78.5	76.9	69.0	68.7
ViT-B/32	Food101	**86.1**	86.1	86.1	85.8	85.1	82.5	80.7

Table 5. Ablation study of "maxiter" in PLCA.

3.2. Distance Metrics for Constructing Graphs

We try to replace Euclidean distance in Eqn.3 with Cosine distance in Table 6. The impact of using Cosine distance

Distance Metrics		Euclidean	Cosine	Source
RN50	CIFAR10	**78.9**	73.2	68.7
ViT-B/32	Food101	**86.1**	84.7	80.7

Table 6. Study of distance metrics for constructing graphs.

on ResNet with BatchNorm is more significant than on ViT with InstanceNorm.

3.3. Computational cost of IST

Compared to related TTA methods, the primary computational overhead of IST originates from PLCA, while test-time augmentation and optimization are common practices in existing SOTA methods such as MEMO [9] and TTAC [8]. We utilize Faiss, an efficient search library, and sparse matrices to construct the graph structure. The average time taken for this process is 0.38 seconds, which accounts for 10.2% of the total runtime of PLCA, amounting to 3.72 seconds.

3.4. Study of Freeze BN layers

For test time adaptation with CLIP model, we freeze batch normalization (BN) layers for stable optimization. Since BN is sensitive to the distribution shift of input images, freezing BN layers can prevent the model from rapidly deteriorating, which leads to serious hard-to-recover errors in pseudo-labels. However, fine-tuning BN layers can be of great help in resolving the distribution shift caused by image corruptions. Under the condition of a batch size of 128, we conduct online test-time adaptation experiments on CIFAR-10C and ImageNet-C with and without freezing BN layers. The results listed in Table 3 demonstrate that fine-tuning BN layers can improve the performance of our method on both datasets. Nevertheless, it is not reliable to estimate the mean and variance of the input distribution with a small batch size, *e.g.* when the batch size is 1. In order to ensure the performance of our method for single sample test-time adaptation, we adopt the strategy of freezing BN layers in the main paper.

4. Limitation and Social Impact

In this work, we propose PLCA to rectify the bias in model predictions. However, it requires a substantial number of samples for support. To this end, SimpAug is capable of providing sufficient data when the testing batch size is small. On the other hand, direct application of PLCA on models with extremely poor calibration to the target distribution often fails to improve the accuracy of pseudo-labels. This necessitates test-time re-training of the model to facilitate self-calibration and enhance the performance of PLCA.

CLIP, as a vision-language foundation model with robust zero-shot capabilities, is highly suitable for exploring the boundaries of unsupervised test-time adaptation (TTA) methods in practical scenarios. Through our study, we aim to advance the research and social awareness towards the problem of TTA with foundation models.

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 1

[2] 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. 2

[3] Yuejiang Liu, Parth Kothari, Bastien Van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34: 21808–21820, 2021. 2

[4] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pages 16888–16905. PMLR, 2022. 1, 2

[5] 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 *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[6] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1

[7] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289, 2022. 1

[8] Yongyi Su, Xun Xu, and Kui Jia. Revisiting realistic test-time training: Sequential inference and adaptation by anchored clustering. *Advances in Neural Information Processing Systems*, 35:17543–17555, 2022. 1, 2, 3

[9] Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in Neural Information Processing Systems*, 35: 38629–38642, 2022. 3

[10] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825, 2022. 1

[11] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 1