Supplementary of Improved Test-Time Adapting for Domain Generalization

Liang Chen[1] Yong Zhang[2*] Yibing Song[3] Ying Shan[2] Lingqiao Liu[1*]
[1] The University of Adelaide [2] Tencent AI Lab [3] AI[3] Institute, Fudan University
{liangchen527, zhangyong201303, yibingsong.cv}@gmail.com
yingsshan@tencent.com lingqiao.liu@adelaide.edu.au

In this supplementary material, we provide,
1. Resource usage for ITTA in Section 1.
2. Grad-CAM visualizations of different loss terms in Section 2.
3. Parameter analysis of ITTA in Section 3;
4. Using a different augmentation skill for ITTA in Section 4.
5. Using different updating steps or a strategy for ITTA during the test phase in Section 5.
6. Using different network structures for the learnable consistency loss and adaptive parameters in Section 6.
7. Comparisons with other related methods in Section 7.
8. Detailed experimental results in the DomainBed benchmark in Section 8.

1. Resource Usage Comparisons Between ITTA and the Baseline Model

Requiring extra resources for our ITTA is a common limitation for existing test-time-based arts. To further evaluate our method, in this section, we compare FLOPS, model size, and inference time in Table 1. We compare only with ERM as most existing methods utilize the same network during inferences. We note that compare to the baseline model, ITTA requires extra Flops and processing time, this is because the adaptation process uses extra forward and backward steps during the test phase. While the parameters between the two models are similar because the newly included adaptive blocks are much smaller in size compared to the original model.

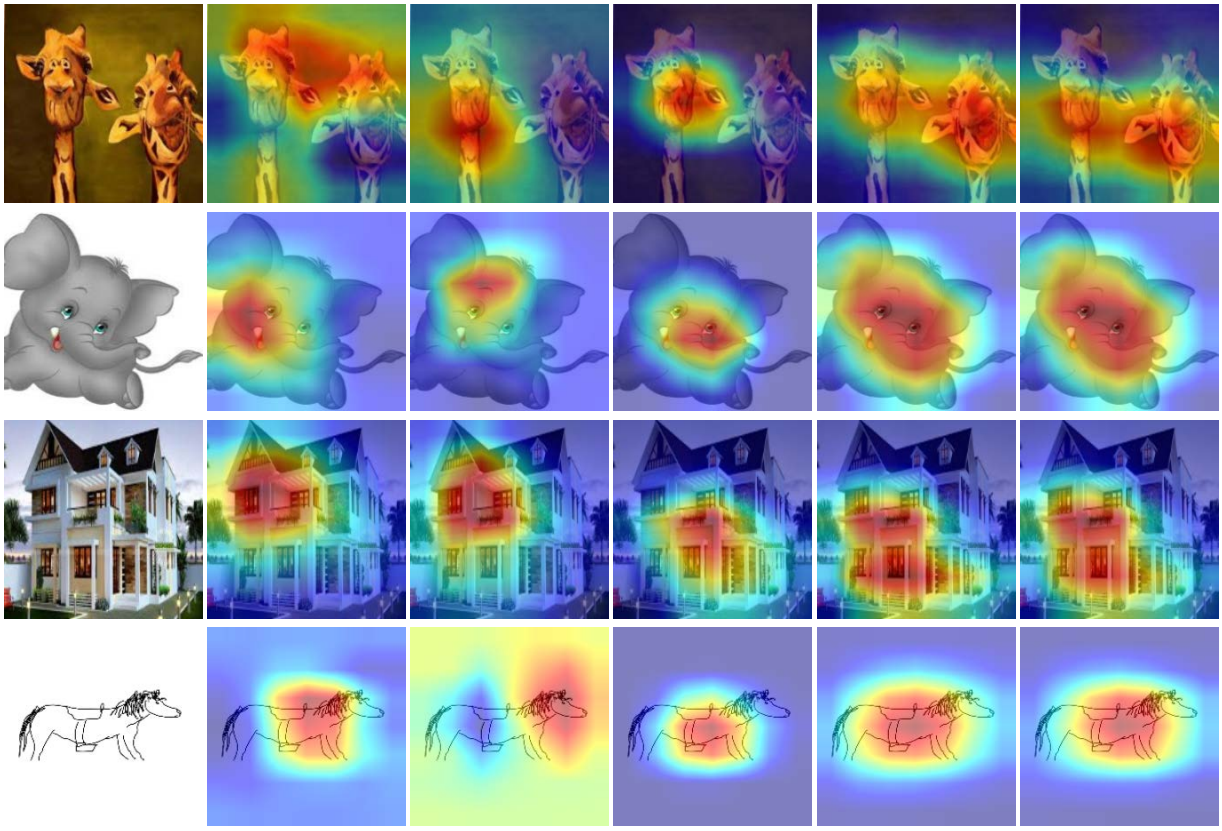
Table 1. Resource comparisons during testing. Here inc. and exc. columns in ITTA indicate to include and exclude the TTA phase.

Model	Flops (G)	Params (M)	Time (s)
Baseline	1.82	11.18	0.004
ITTA (inc. | exc.)	6.12 | 1.83	14.95 | 14.94	0.021 | 0.005

2. Grad-CAM Visualizations of Different Self-Supervised Objectives

In Section 5 of the manuscript, we provide Grad-CAM [26] visualizations of our learnable consistency and the main losses to illustrate their alignment. To further show the differences between several TTT tasks [29, 32], we present more visual examples in this section. Results are shown in Figure 1. We observe that the entropy minimization [32] and rotation estimation [29] objectives do not activate the same regions as the main loss. As shown in the first row, for the class label of giraffe, both the main loss and our learned loss can correctly locate the two giraffes in the image, while the rotation estimation task can only locate one target, the same observation can be found when the learned weights are disabled in our loss term. Meanwhile, although the two objects can be found for the entropy minimization task, the corresponding hot region does not align with that of the main loss. Similar phenomena can be observed in other samples. These visual examples demonstrate that our learned objective can better align with the main task than the TTT tasks adopted in previous works [29, 32], explaining why using the proposed learnable consistency loss can better improve TTT.

*Corresponding authors. This work is done when L. Chen is an intern in Tencent AI Lab.



(a) Input (b) Entropy (c) Rotation (d) Ours w/o f_w (e) Ours (f) Main

Figure 1. Grad-CAM [26] visualizations from different loss terms. We use images with varying class labels (*i.e.* giraffe, elephant, house, and horse from top to bottom) from the four target domains of PACS [13] as inputs (*i.e.* art, cartoon, photo, and sketch domains from top to bottom). "Entropy" and "Rotation" here denote the entropy minimization and rotation estimation tasks in [32] and [29]. Ours w/o f_w is the learnable consistency loss in Eq. (1) in the manuscript (*i.e.* $\|f_w(z - z')\|$) when f_w is disabled. The proposed learnable consistency loss can align well with the main classification task.

3. Parameter Analysis

In this section, we analyze the hyper-parameter used in ITTA. We use the weight parameter α to balance the contributions from the main loss and weighted consistency loss (*i.e.* $\mathcal{L}_{main} + \alpha\mathcal{L}_{wcont}$ in Eq. (2) of our manuscript). To analyze the sensitivity of ITTA regarding different values of α, we conduct ablation studies in the PACS benchmark [13]. Results are listed in Table 2. We observe that the proposed ITTA can obtain favorable performances when α is in the range of 0.1 to 10, and it performs the best on average when setting as 1. We thus fix the parameter as 1 in all experiments.

Table 2. Sensitivity analysis of ITTA regarding different values of α in the unseen domain from PACS [13]. The reported accuracies (%) and standard deviations are computed from 60 trials in each target domain.

Values	Target domain				Avg.
	Art	Cartoon	Photo	Sketch	
$\alpha = 0.1$	83.9 ± 0.7	76.2 ± 1.1	94.8 ± 0.2	78.8 ± 0.8	83.4 ± 0.2
$\alpha = 1$ (Ours)	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3
$\alpha = 10$	83.9 ± 0.5	77.4 ± 0.6	94.2 ± 0.7	77.3 ± 0.8	83.2 ± 0.3
$\alpha = 100$	81.5 ± 1.2	77.0 ± 0.6	92.6 ± 0.7	78.9 ± 2.1	82.5 ± 0.9

4. A Different Augmentation Skill for ITTA

In our manuscript, we use the existing augmentation strategy from [35] to obtain the augmented feature. In this section, we replace this implementation with that from [16] to further verify if our ITTA can still thrive with another augmentation skill.

Table 3. Performances of our method with another augmentation strategy from [16] in the unseen domain from PACS [13]. The reported accuracies (%) and standard deviations are computed from 60 trials in each target domain.

Model	Target domain				Avg.
	Art	Cartoon	Photo	Sketch	
ERM	78.0 ± 1.3	73.4 ± 0.8	94.1 ± 0.4	73.6 ± 2.2	79.8 ± 0.4
Ours w/o f_w & TTT	74.9 ± 0.4	74.1 ± 0.8	90.6 ± 0.3	79.7 ± 0.7	79.8 ± 0.4
Ours w/o f_w	77.1 ± 1.0	73.6 ± 1.1	89.9 ± 0.4	78.4 ± 0.8	79.7 ± 0.2
Ours w/o TTT	77.5 ± 0.3	73.2 ± 0.6	92.4 ± 0.4	78.0 ± 1.0	80.3 ± 0.3
Ours (w/ f_w & TTT)	79.2 ± 0.8	74.9 ± 1.1	92.2 ± 0.3	76.9 ± 0.7	80.8 ± 0.4

Different from [35] that mixes the statics of the feature to synthesize new information, [16] uses an affine transformation to create new features, where the weight for the transformation is sampled from a normal distribution with the mean value of one and standard value of zero, and the bias for the transformation is sampled from a normal distribution with the mean and standard values both zero. Experiments are conducted on the PACS benchmark [13] with the leave-one-out strategy.

We compare ITTA with several different variants. (1) Ours w/o f_w & TTT: this variant is the baseline model which uses the naive consistency loss for training and does not include TTT during the test phase. (2) Ours w/o f_w: we disable the f_w in our consistency loss, which uses the naive consistency loss for the test-time updating. (3) Ours w/o TTT: we do not update any parameters during the test phase. This variant is used to verify whether TTT can improve the pretrained model when replacing the augmentation strategy. We also compare these variants with the ERM method to show their effectivenesses.

Results are listed in Table 3. We observe that ERM performs favorably against the baseline model, indicating that this augmentation strategy may not be beneficial for the training process. Meanwhile, we observe that when f_w is disabled, the performances seem to decrease in 3 out of 4 target domains, and the average accuracy is also inferior to the baseline (*i.e.* Ours w/o f_w & TTT). This result is in line with the finding in [19] that an inappropriate TTT task may deteriorate the performance. In comparison, we note that the performances are both improved when f_w is enabled (*i.e.* Ours w/o TTT and Ours), which once again demonstrates that the proposed learnable consistency loss can improve the trained model. Moreover, we can also observe that when combining f_w and TTT, our model is superior to other variants and the ERM method. These results demonstrate that the proposed two strategies can improve the current TTT framework despite a less effective augmentation strategy.

5. Different Updating Steps or Strategies for ITTA

In the manuscript, we use one TTT step for ITTA before during the testing step. In this section, we conduct experiments to evaluate the performances of ITTA with different TTT steps. Experiments are conducted on the PACS benchmark [13] with the leave-one-out strategy, and each target domain is examined with 60 sets of random seeds and hyper-parameter settings. Results are listed in Table 4. We observe that the average accuracies of using more TTT steps are not improved greatly while the computational times are proportional to the TTT steps. To this end, we use one TTT step for ITTA as a compromise between accuracy and efficiency.

We use the online setting from TTT [29] for all arts, which assumes test samples arrive sequentially and updates the adaptive blocks based on the states optimized from a previous sample. In this section, we also test ITTA in an episodic manner (*i.e.* Epi) [5]. Results in Table 4 suggest that while the episodic updating strategy performs slightly worse than the current scheme, and it still outperforms the baseline.

Table 4. Evaluations of ITTA in the unseen domain from PACS [13] with different TTT steps and updating strategies during the testing phase. The reported accuracies (%) and standard deviations are computed from 60 trials in each target domain. The time consumption (TC) is computed using one image with the size of 224×224. Epi. denotes updating ITTA in an episodic manner.

Steps	Target domain				Avg.	TC
	Art	Cartoon	Photo	Sketch		
1 step (Ours)	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3	2.4 ms
2 step	84.2 ± 0.9	77.5 ± 0.6	94.4 ± 0.4	79.1 ± 1.0	83.8 ± 0.1	4.2 ms
3 step	84.5 ± 1.2	77.6 ± 0.6	94.0 ± 0.6	79.3 ± 0.1	83.9 ± 0.3	6.1 ms
Epi.	83.6 ± 0.7	77.9 ± 0.5	95.2 ± 0.1	76.6 ± 0.5	83.3 ± 0.4	

6. Different Network Structures for the Learnable Consistency Loss and Adaptive Parameters

In our implementation, we use 10 layers of learnable parameters for f_w, and we use 5 layers of learnable parameters for f_Θ after each block. In this section, we evaluate our ITTA with different network structures for these two modules. Specifically, we compare the original implementation with the variants that use 1, 5, and 15 layers for f_w and 1, 10, and 15 layers for f_Θ to evaluate the performances of different structures. Similarly, we conduct experiments on the PACS benchmark [13] with the leave-one-out strategy, and each target domain is examined with 60 sets of random seeds and hyper-parameter settings. Evaluation results are listed in Table 5. We observe that their differences in the average accuracy are rather subtle on account of the variances. To this end, we use the original implementation with 10 layers of learnable parameters for f_w and 5 layers of learnable parameters for f_Θ, which performs relatively better than other variants.

Since the adaptive blocks f_Θ are attached after each layer of the network, one may wonder how the varying locations of the adaptive blocks affect the performance of ITTA. To answer this question, we further conduct experiments by adding the adaptive blocks after different layers of the original network. Denoting as Loc = la_n given the n layers in the original network, we note that the model performs less effectively when the adaptive block is placed after the 1st layer of the network, and using all four adaptive blocks (*i.e.* ours) is more effective than other alternatives.

7. Comparisons with Other Related Methods

Apart from the proposed ITTA, some other works also propose to include learnable parameters in their auxiliary losses. Examples include MetaReg [2] and Feature-Critic [18] which both suggest using meta-learning to produce more general models. The main difference between these arts and ITTA is that parameters in the auxiliary loss from [2, 18] are gradually refined by episode training, and they are updated via a gradient alignment step in ITTA (see Sec. 3.1 in the manuscript), which is much simpler. In this section, we compare ITTA with these two arts in the PACS dataset [13] using the same settings aforementioned. Because MetaReg [2] does not release codes, we thus directly cite the data from their paper in the comparison. Different from others, the results in [2] are averaged by 5 trials according to their paper, which is much less than our experimental settings. Meanwhile, we also compare with TTT++ [19] which suggests storing the momentum of the features from the source domain and enforcing the similarity between momentums of features from the source and target domains. We use the same setting in Section 5.1 from the manuscript to evaluate TTT++. Results are listed in Table 6. We observe that our method consistently outperforms that from [2, 18, 19] for both the cases with and without TTT, indicating that the proposed learnable consistency loss and updating method is not only simpler but also more effective than the losses in [2, 18].

Table 5. Performances of our method with different network structures for the consistency loss (*i.e.* f_w) and adaptive parameters (*i.e.* f_Θ) in the unseen domain from PACS [13]. Here 'Loc=la_n' locates the adaptive block after the n-th layer of the model ('la_4' is the last layer). The reported accuracies (%) and standard deviations are computed from 60 trials in each target domain.

	Structures	Target domain				Avg.
		Art	Cartoon	Photo	Sketch	
Structures of f_w	1 layer	83.5 ± 1.2	76.0 ± 1.0	95.3 ± 0.2	78.7 ± 1.5	83.4 ± 0.4
	5 layers	83.7 ± 0.6	76.8 ± 0.9	94.6 ± 0.3	78.8 ± 0.3	83.5 ± 0.3
	10 layers (Ours)	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3
	15 layers	84.1 ± 0.4	75.8 ± 0.2	94.3 ± 0.3	79.5 ± 0.4	83.4 ± 0.2
Structures of f_Θ	1 layer	84.0 ± 0.6	77.4 ± 0.5	94.4 ± 0.5	78.3 ± 0.4	83.5 ± 0.3
	5 layers (Ours)	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3
	10 layers	84.8 ± 0.3	76.0 ± 0.6	94.1 ± 0.5	78.3 ± 0.1	83.3 ± 0.3
	15 layers	83.9 ± 0.8	76.0 ± 0.5	93.8 ± 0.4	78.7 ± 1.4	83.1 ± 0.6
Locations of f_Θ	Loc=la_1	83.4 ± 0.7	76.8 ± 0.3	94.4 ± 0.3	77.8 ± 0.3	83.1 ± 0.3
	Loc=la_2	83.4 ± 0.6	77.7 ± 0.6	94.2 ± 0.5	78.0 ± 0.5	83.3 ± 0.3
	Loc=la_3	84.0 ± 0.4	77.5 ± 0.3	94.4 ± 0.1	77.8 ± 0.1	83.4 ± 0.2
	Loc=la_4	84.1 ± 0.7	77.8 ± 0.5	94.8 ± 0.2	76.9 ± 1.5	83.4 ± 0.4

Table 6. Compare with learnable losses in [2, 18] in the unseen domain from PACS [13]. The reported accuracies (%) and standard deviations are computed from 60 trials in each target domain except for [2] where the numbers are directly cited from their paper.

Model	Target domain				Avg.
	Art	Cartoon	Photo	Sketch	
MetaReg [2]	83.7 ± 0.2	77.2 ± 0.3	95.5 ± 0.2	70.3 ± 0.3	81.7
Feture-Critic [18]	78.4 ± 1.6	75.4 ± 1.2	92.6 ± 0.5	73.3 ± 1.4	80.0 ± 0.3
TTT++ [19]	84.3 ± 0.1	78.4 ± 0.5	93.8 ± 1.3	73.2 ± 3.2	82.4 ± 1.1
Ours w/o TTT	83.3 ± 0.5	76.0 ± 0.5	94.4 ± 0.5	76.7 ± 1.4	82.8 ± 0.3
Ours	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3

8. Detailed Results in the DomainBed Benchmark [8]

this section presents the average accuracy in each domain from different datasets. As shown in Table 7, 8, 9, 10, and 11, these results are detailed illustrations of the results in Table 2 in our manuscript. For all the experiments, we use the "training-domain validate set" as the model selection method. A total of 22 methods are examined for 60 trials in each unseen domain, and all methods are trained with the leave-one-out strategy using the ResNet18 [9] backbones.

Table 7. Average accuracies on the PACS [13] datasets using the default hyper-parameter settings in DomainBed [8].

	art	cartoon	photo	sketch	Average
ERM [30]	78.0 ± 1.3	73.4 ± 0.8	94.1 ± 0.4	73.6 ± 2.2	79.8 ± 0.4
IRM [1]	76.9 ± 2.6	75.1 ± 0.7	94.3 ± 0.4	77.4 ± 0.4	80.9 ± 0.5
GroupGRO [25]	77.7 ± 2.6	76.4 ± 0.3	94.0 ± 0.3	74.8 ± 1.3	80.7 ± 0.4
Mixup [33]	79.3 ± 1.1	74.2 ± 0.3	94.9 ± 0.3	68.3 ± 2.7	79.2 ± 0.9
MLDG [14]	78.4 ± 0.7	75.1 ± 0.5	94.8 ± 0.4	76.7 ± 0.8	81.3 ± 0.2
CORAL [28]	81.5 ± 0.5	75.4 ± 0.7	95.2 ± 0.5	74.8 ± 0.4	81.7 ± 0.0
MMD [15]	81.3 ± 0.6	75.5 ± 1.0	94.0 ± 0.5	74.3 ± 1.5	81.3 ± 0.8
DANN [7]	79.0 ± 0.6	72.5 ± 0.7	94.4 ± 0.5	70.8 ± 3.0	79.2 ± 0.3
CDANN [17]	80.4 ± 0.8	73.7 ± 0.3	93.1 ± 0.6	74.2 ± 1.7	80.3 ± 0.5
MTL [4]	78.7 ± 0.6	73.4 ± 1.0	94.1 ± 0.6	74.4 ± 3.0	80.1 ± 0.8
SagNet [20]	82.9 ± 0.4	73.2 ± 1.1	94.6 ± 0.5	76.1 ± 1.8	81.7 ± 0.6
ARM [34]	79.4 ± 0.6	75.0 ± 0.7	94.3 ± 0.6	73.8 ± 0.6	80.6 ± 0.5
VREx [12]	74.4 ± 0.7	75.0 ± 0.4	93.3 ± 0.3	78.1 ± 0.9	80.2 ± 0.5
RSC [10]	78.5 ± 1.1	73.3 ± 0.9	93.6 ± 0.6	76.5 ± 1.4	80.5 ± 0.2
SelfReg [11]	82.5 ± 0.8	74.4 ± 1.5	95.4 ± 0.5	74.9 ± 1.3	81.8 ± 0.3
MixStyle [35]	82.6 ± 1.2	76.3 ± 0.4	94.2 ± 0.3	77.5 ± 1.3	82.6 ± 0.4
Fish [27]	80.9 ± 1.0	75.9 ± 0.4	95.0 ± 0.4	76.2 ± 1.0	82.0 ± 0.3
SD [22]	83.2 ± 0.6	74.6 ± 0.3	94.6 ± 0.1	75.1 ± 1.6	81.9 ± 0.3
CAD [24]	83.9 ± 0.8	74.2 ± 0.4	94.6 ± 0.4	75.0 ± 1.2	81.9 ± 0.3
CondCAD [24]	79.7 ± 1.0	74.2 ± 0.9	94.6 ± 0.4	74.8 ± 1.4	80.8 ± 0.5
Fishr [23]	81.2 ± 0.4	75.8 ± 0.8	94.3 ± 0.3	73.8 ± 0.6	81.3 ± 0.3
Ours	84.7 ± 0.4	78.0 ± 0.4	94.5 ± 0.4	78.2 ± 0.3	83.8 ± 0.3

Table 8. Average accuracies on the VLCS [6] datasets using the default hyper-parameter settings in DomainBed [8].

	Caltech	LabelMe	Sun	VOC	Average
ERM [30]	97.7 ± 0.3	62.1 ± 0.9	70.3 ± 0.9	73.2 ± 0.7	75.8 ± 0.2
IRM [1]	96.1 ± 0.8	62.5 ± 0.3	69.9 ± 0.7	72.0 ± 1.4	75.1 ± 0.1
GroupGRO [25]	96.7 ± 0.6	61.7 ± 1.5	70.2 ± 1.8	72.9 ± 0.6	75.4 ± 1.0
Mixup [33]	95.6 ± 1.5	62.7 ± 0.4	71.3 ± 0.3	75.4 ± 0.2	76.2 ± 0.3
MLDG [14]	95.8 ± 0.5	63.3 ± 0.8	68.5 ± 0.5	73.1 ± 0.8	75.2 ± 0.3
CORAL [28]	96.5 ± 0.3	62.8 ± 0.1	69.1 ± 0.6	73.8 ± 1.0	75.5 ± 0.4
MMD [15]	96.0 ± 0.8	64.3 ± 0.6	68.5 ± 0.6	70.8 ± 0.1	74.9 ± 0.5
DANN [7]	97.2 ± 0.1	63.3 ± 0.6	70.2 ± 0.9	74.4 ± 0.2	76.3 ± 0.2
CDANN [17]	95.4 ± 1.2	62.6 ± 0.6	69.9 ± 1.3	76.2 ± 0.5	76.0 ± 0.5
MTL [4]	94.4 ± 2.3	65.0 ± 0.6	69.6 ± 0.6	71.7 ± 1.3	75.2 ± 0.3
SagNet [20]	94.9 ± 0.7	61.9 ± 0.7	69.6 ± 1.3	75.2 ± 0.6	75.4 ± 0.8
ARM [34]	96.9 ± 0.5	61.9 ± 0.4	71.6 ± 0.1	73.3 ± 0.4	75.9 ± 0.3
VREx [12]	96.2 ± 0.0	62.5 ± 1.3	69.3 ± 0.9	73.1 ± 1.2	75.3 ± 0.6
RSC [10]	96.2 ± 0.0	63.6 ± 1.3	69.8 ± 1.0	72.0 ± 0.4	75.4 ± 0.3
SelfReg [11]	95.8 ± 0.6	63.4 ± 1.1	71.1 ± 0.6	75.3 ± 0.6	76.4 ± 0.7
MixStyle [35]	97.3 ± 0.3	61.6 ± 0.1	70.4 ± 0.7	71.3 ± 1.9	75.2 ± 0.7
Fish [27]	97.4 ± 0.2	63.4 ± 0.1	71.5 ± 0.4	75.2 ± 0.7	76.9 ± 0.2
SD [22]	96.5 ± 0.4	62.2 ± 0.0	69.7 ± 0.9	73.6 ± 0.4	75.5 ± 0.4
CAD [24]	94.5 ± 0.9	63.5 ± 0.6	70.4 ± 1.2	72.4 ± 1.3	75.2 ± 0.6
CondCAD [24]	96.5 ± 0.8	62.6 ± 0.4	69.1 ± 0.2	76.0 ± 0.2	76.1 ± 0.3
Fishr [23]	97.2 ± 0.6	63.3 ± 0.7	70.4 ± 0.6	74.0 ± 0.8	76.2 ± 0.3
Ours	96.9 ± 1.2	63.7 ± 1.1	72.0 ± 0.3	74.9 ± 0.8	76.9 ± 0.6

Table 9. Average accuracies on the OfficeHome [31] datasets using the default hyper-parameter settings in DomainBed [8].

	art	clipart	product	real	Average
ERM [30]	52.2 ± 0.2	48.7 ± 0.5	69.9 ± 0.5	71.7 ± 0.5	60.6 ± 0.2
IRM [1]	49.7 ± 0.2	46.8 ± 0.5	67.5 ± 0.4	68.1 ± 0.6	58.0 ± 0.1
GroupGRO [25]	52.6 ± 1.1	48.2 ± 0.9	69.9 ± 0.4	71.5 ± 0.8	60.6 ± 0.3
Mixup [33]	54.0 ± 0.7	49.3 ± 0.7	70.7 ± 0.7	72.6 ± 0.3	61.7 ± 0.5
MLDG [14]	53.1 ± 0.3	48.4 ± 0.3	70.5 ± 0.7	71.7 ± 0.4	60.9 ± 0.2
CORAL [28]	55.1 ± 0.7	49.7 ± 0.9	71.8 ± 0.2	73.1 ± 0.5	62.4 ± 0.4
MMD [15]	50.9 ± 1.0	48.7 ± 0.3	69.3 ± 0.7	70.7 ± 1.3	59.9 ± 0.4
DANN [7]	51.8 ± 0.5	47.1 ± 0.1	69.1 ± 0.7	70.2 ± 0.7	59.5 ± 0.5
CDANN [17]	51.4 ± 0.5	46.9 ± 0.6	68.4 ± 0.5	70.4 ± 0.4	59.3 ± 0.4
MTL [4]	51.6 ± 1.5	47.7 ± 0.5	69.1 ± 0.3	71.0 ± 0.6	59.9 ± 0.5
SagNet [20]	55.3 ± 0.4	49.6 ± 0.2	72.1 ± 0.4	73.2 ± 0.4	62.5 ± 0.3
ARM [34]	51.3 ± 0.9	48.5 ± 0.4	68.0 ± 0.3	70.6 ± 0.1	59.6 ± 0.3
VREx [12]	51.1 ± 0.3	47.4 ± 0.6	69.0 ± 0.4	70.5 ± 0.4	59.5 ± 0.1
RSC [10]	49.0 ± 0.1	46.2 ± 1.5	67.8 ± 0.7	70.6 ± 0.3	58.4 ± 0.6
SelfReg [11]	55.1 ± 0.8	49.2 ± 0.6	72.2 ± 0.3	73.0 ± 0.3	62.4 ± 0.1
MixStyle [35]	50.8 ± 0.6	51.4 ± 1.1	67.6 ± 1.3	68.8 ± 0.5	59.6 ± 0.8
Fish [27]	54.6 ± 1.0	49.6 ± 1.0	71.3 ± 0.6	72.4 ± 0.2	62.0 ± 0.6
SD [22]	55.0 ± 0.4	51.3 ± 0.5	72.5 ± 0.2	72.7 ± 0.3	62.9 ± 0.2
CAD [24]	52.1 ± 0.6	48.3 ± 0.5	69.7 ± 0.3	71.9 ± 0.4	60.5 ± 0.3
CondCAD [24]	53.3 ± 0.6	48.4 ± 0.2	69.8 ± 0.9	72.6 ± 0.1	61.0 ± 0.4
Fishr [23]	52.6 ± 0.9	48.6 ± 0.3	69.9 ± 0.6	72.4 ± 0.4	60.9 ± 0.3
Ours	54.4 ± 0.2	52.3 ± 0.8	69.5 ± 0.3	71.7 ± 0.2	62.0 ± 0.2

Table 10. Average accuracies on the TerraInc [3] datasets using the default hyper-parameter settings in DomainBed [8].

	L100	L38	L43	L46	Average
ERM [30]	42.1 ± 2.5	30.1 ± 1.2	48.9 ± 0.6	34.0 ± 1.1	38.8 ± 1.0
IRM [1]	41.8 ± 1.8	29.0 ± 3.6	49.6 ± 2.1	33.1 ± 1.5	38.4 ± 0.9
GroupGRO [25]	45.3 ± 4.6	36.1 ± 4.4	51.0 ± 0.8	33.7 ± 0.9	41.5 ± 2.0
Mixup [33]	49.4 ± 2.0	35.9 ± 1.8	53.0 ± 0.7	30.0 ± 0.9	42.1 ± 0.7
MLDG [14]	39.6 ± 2.3	33.2 ± 2.7	52.4 ± 0.5	35.1 ± 1.5	40.1 ± 0.9
CORAL [28]	46.7 ± 3.2	36.9 ± 4.3	49.5 ± 1.9	32.5 ± 0.7	41.4 ± 1.8
MMD [15]	49.1 ± 1.2	36.4 ± 4.8	50.4 ± 2.1	32.3 ± 1.5	42.0 ± 1.0
DANN [7]	44.3 ± 3.6	28.0 ± 1.5	47.9 ± 1.0	31.3 ± 0.6	37.9 ± 0.9
CDANN [17]	36.9 ± 6.4	32.7 ± 6.2	51.1 ± 1.3	33.5 ± 0.5	38.6 ± 2.3
MTL [4]	45.2 ± 2.6	31.0 ± 1.6	50.6 ± 1.1	34.9 ± 0.4	40.4 ± 1.0
SagNet [20]	36.3 ± 4.7	40.3 ± 2.0	52.5 ± 0.6	33.3 ± 1.3	40.6 ± 1.5
ARM [34]	41.5 ± 4.5	27.7 ± 2.4	50.9 ± 1.0	29.6 ± 1.5	37.4 ± 1.9
VREx [12]	48.0 ± 1.7	41.1 ± 1.5	51.8 ± 1.5	32.0 ± 1.2	43.2 ± 0.3
RSC [10]	42.8 ± 2.4	32.2 ± 3.8	49.6 ± 0.9	32.9 ± 1.2	39.4 ± 1.3
SelfReg [11]	46.1 ± 1.5	34.5 ± 1.6	49.8 ± 0.3	34.7 ± 1.5	41.3 ± 0.3
MixStyle [35]	50.6 ± 1.9	28.0 ± 4.5	52.1 ± 0.7	33.0 ± 0.2	40.9 ± 1.1
Fish [27]	46.3 ± 3.0	29.0 ± 1.1	52.7 ± 1.2	32.8 ± 1.0	40.2 ± 0.6
SD [22]	45.5 ± 1.9	33.2 ± 3.1	52.9 ± 0.7	36.4 ± 0.8	42.0 ± 1.0
CAD [24]	43.1 ± 2.6	31.1 ± 1.9	53.1 ± 1.6	34.7 ± 1.3	40.5 ± 0.4
CondCAD [24]	44.4 ± 2.9	32.9 ± 2.5	50.5 ± 1.3	30.8 ± 0.5	39.7 ± 0.4
Fishr [23]	49.9 ± 3.3	36.6 ± 0.9	49.8 ± 0.2	34.2 ± 1.3	42.6 ± 1.0
Ours	51.7 ± 2.4	37.6 ± 0.6	49.9 ± 0.6	33.6 ± 0.6	43.2 ± 0.5

Table 11. Average accuracies on the DomainNet [21] datasets using the default hyper-parameter settings in DomainBed [8].

	clip	info	paint	quick	real	sketch	Average
ERM [30]	50.4 ± 0.2	14.0 ± 0.2	40.3 ± 0.5	11.7 ± 0.2	52.0 ± 0.2	43.2 ± 0.3	35.3 ± 0.1
IRM [1]	43.2 ± 0.9	12.6 ± 0.3	35.0 ± 1.4	9.9 ± 0.4	43.4 ± 3.0	38.4 ± 0.4	30.4 ± 1.0
GroupGRO [25]	38.2 ± 0.5	13.0 ± 0.3	28.7 ± 0.3	8.2 ± 0.1	43.4 ± 0.5	33.7 ± 0.0	27.5 ± 0.1
Mixup [33]	48.9 ± 0.3	13.6 ± 0.3	39.5 ± 0.5	10.9 ± 0.4	49.9 ± 0.2	41.2 ± 0.2	34.0 ± 0.0
MLDG [14]	51.1 ± 0.3	14.1 ± 0.3	40.7 ± 0.3	11.7 ± 0.1	52.3 ± 0.3	42.7 ± 0.2	35.4 ± 0.0
CORAL [28]	51.2 ± 0.2	15.4 ± 0.2	42.0 ± 0.2	12.7 ± 0.1	52.0 ± 0.3	43.4 ± 0.0	36.1 ± 0.2
MMD [15]	16.6 ± 13.3	0.3 ± 0.0	12.8 ± 10.4	0.3 ± 0.0	17.1 ± 13.7	0.4 ± 0.0	7.9 ± 6.2
DANN [7]	45.0 ± 0.2	12.8 ± 0.2	36.0 ± 0.2	10.4 ± 0.3	46.7 ± 0.3	38.0 ± 0.3	31.5 ± 0.1
CDANN [17]	45.3 ± 0.2	12.6 ± 0.2	36.6 ± 0.2	10.3 ± 0.4	47.5 ± 0.1	38.9 ± 0.4	31.8 ± 0.2
MTL [4]	50.6 ± 0.2	14.0 ± 0.4	39.6 ± 0.3	12.0 ± 0.3	52.1 ± 0.1	41.5 ± 0.0	35.0 ± 0.0
SagNet [20]	51.0 ± 0.1	14.6 ± 0.1	40.2 ± 0.2	12.1 ± 0.2	51.5 ± 0.3	42.4 ± 0.1	35.3 ± 0.1
ARM [34]	43.0 ± 0.2	11.7 ± 0.2	34.6 ± 0.1	9.8 ± 0.4	43.2 ± 0.3	37.0 ± 0.3	29.9 ± 0.1
VREx [12]	39.2 ± 1.6	11.9 ± 0.4	31.2 ± 1.3	10.2 ± 0.4	41.5 ± 1.8	34.8 ± 0.8	28.1 ± 1.0
RSC [10]	39.5 ± 3.7	11.4 ± 0.8	30.5 ± 3.1	10.2 ± 0.8	41.0 ± 1.4	34.7 ± 2.6	27.9 ± 2.0
SelfReg [11]	47.9 ± 0.3	15.1 ± 0.3	41.2 ± 0.2	11.7 ± 0.3	48.8 ± 0.0	43.8 ± 0.3	34.7 ± 0.2
MixStyle [35]	49.1 ± 0.4	13.4 ± 0.0	39.3 ± 0.0	11.4 ± 0.4	47.7 ± 0.3	42.7 ± 0.1	33.9 ± 0.1
Fish [27]	51.5 ± 0.3	14.5 ± 0.2	40.4 ± 0.3	11.7 ± 0.5	52.6 ± 0.2	42.1 ± 0.1	35.5 ± 0.0
SD [22]	51.3 ± 0.3	15.5 ± 0.1	41.5 ± 0.3	12.6 ± 0.2	52.9 ± 0.2	44.0 ± 0.4	36.3 ± 0.2
CAD [24]	45.4 ± 1.0	12.1 ± 0.5	34.9 ± 1.1	10.2 ± 0.6	45.1 ± 1.6	38.5 ± 0.6	31.0 ± 0.8
CondCAD [24]	46.1 ± 1.0	13.3 ± 0.4	36.1 ± 1.4	10.7 ± 0.2	46.8 ± 1.3	38.7 ± 0.7	31.9 ± 0.7
Fishr [23]	47.8 ± 0.7	14.6 ± 0.2	40.0 ± 0.3	11.9 ± 0.2	49.2 ± 0.7	41.7 ± 0.1	34.2 ± 0.3
Ours	50.7 ± 0.7	13.9 ± 0.4	39.4 ± 0.5	11.9 ± 0.2	50.2 ± 0.3	43.5 ± 0.1	34.9 ± 0.1

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 5, 6, 7

[2] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018. 4, 5

[3] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, 2018. 7

[4] Gilles Blanchard, Aniket Anand Deshmukh, Urun Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *arXiv preprint arXiv:1711.07910*, 2017. 5, 6, 7

[5] Liang Chen, Yong Zhang, Yibing Song, Jue Wang, and Lingqiao Liu. Ost: Improving generalization of deepfake detection via one-shot test-time training. In *NeurIPS*, 2022. 3

[6] 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*, 2013. 6

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016. 5, 6, 7

[8] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021. 5, 6, 7

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[10] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *ECCV*, 2020. 5, 6, 7

[11] Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *ICCV*, 2021. 5, 6, 7

[12] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *ICML*, 2021. 5, 6, 7

[13] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. 2, 3, 4, 5

[14] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018. 5, 6, 7

[15] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018. 5, 6, 7

[16] Pan Li, Da Li, Wei Li, Shaogang Gong, Yanwei Fu, and Timothy M Hospedales. A simple feature augmentation for domain generalization. In *ICCV*, 2021. 2, 3

[17] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*, 2018. 5, 6, 7

[18] Yiying Li, Yongxin Yang, Wei Zhou, and Timothy Hospedales. Feature-critic networks for heterogeneous domain generalization. In *ICML*, 2019. 4, 5

[19] 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? In *NeurIPS*, 2021. 3, 4, 5

[20] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *CVPR*, 2021. 5, 6, 7

[21] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019. 7

[22] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. In *NeurIPS*, 2021. 5, 6, 7

[23] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. In *ICML*, 2022. 5, 6, 7

[24] Yangjun Ruan, Yann Dubois, and Chris J Maddison. Optimal representations for covariate shift. In *ICLR*, 2022. 5, 6, 7

[25] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *ICLR*, 2020. 5, 6, 7

[26] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 1, 2

[27] Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *ICLR*, 2021. 5, 6, 7

[28] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, 2016. 5, 6, 7

[29] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020. 1, 2, 3

[30] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999. 5, 6, 7

[31] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 6

[32] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021. 1, 2

[33] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020. 5, 6, 7

[34] Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. Adaptive risk minimization: A meta-learning approach for tackling group distribution shift. *arXiv preprint arXiv:2007.02931*, 2020. 5, 6, 7

[35] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*, 2021. 2, 3, 5, 6, 7