Appendices for "Adversarially Adaptive Normalization for Single Domain Generalization"

A. Additional Experimental Results

On the Effect of Normalization. In Table 9, we study the impact on the generalization ability by using different normalization techniques with RSDA [53] for domain augmentation on the Digits benchmark. It reports average accuracies, standard deviations and p-values for each domain in the Digits benchmark. We observe that adding either batch normalization (BN) or BN-test to the ConvNet architecture makes the performance worse than the baseline without any normalization layer. Instance normalization shows small improvement over the baseline but still underperforms ASR-Norm. ASR-Norm outperforms all methods on average and achieves significant improvement for challenging domains, including SVHN and SYN. On the easier domains like MNIST-M and USPS, ASR performs on a par with the baseline (RSDA).

Method	SVHN	MNIST-M	SYN	USPS	Avg.
RSDA+BN[22]	39.4±5.2	76.6±2.2	60.5±1.5	84.2±2.1	65.2
RSDA+BN-Test[37]	45.7±2.8	80.3±1.2	59.7±1.4	81.8±1.1	66.9
RSDA+IN[50]	47.1±3.4	80.6±0.9	61.9±1.5	85.4±1.4	68.8
RSDA+SN[32]	37.7±3.8	77.1±1.4	60.5±1.8	86.1±1.7	65.4
RSDA	47.4±4.8	81.5±1.6	62.0±1.2	83.1±1.2	68.5
RSDA+AR	47.8 ±3.2	80.0±1.0	64.0±0.9	**86.7±1.5**	69.6
RSDA+AS	49.4±2.3	81.4±0.7	63.5±1.2	81.4±1.1	69.3
RSDA+ASR (Ours)	**52.8±3.8**	80.8±0.6	**64.5±1.1**	82.4±1.4	**70.1**
p-value: Ours vs. RSDA	**0.036**	0.193	**0.003**	0.197	-
p-value: Ours vs. AS	**0.050**	0.088	0.115	0.108	-
p-value: Ours vs. AR	**0.020**	0.080	0.214	$<$ **1e-3**	-

Table 9: Single domain generalization accuracies with different normalization on Digits. MNIST is used as the training set, and the results on different testing domains are reported in different columns.

Statistical significance of results on CIFAR-10-C. In Table 10 reports the standard deviations and p-values for the one-sided two-sample t-test on the accuracies for CIFAR-10-C in addition to Table 5. The results show consistently statistical significance of ASR-norm's improvements over M-ADA, SN, AR, and AS in different corruption levels.

Analysis of Residual Learning. Fig. 7a shows the evolution of the adaptive weights $\lambda_{\boldsymbol{\mu}}$ and $\lambda_{\boldsymbol{\sigma}}$ in the residual terms of standardization statistics along the training process of the PACS benchmark. The weights for learned statistics are initialized close to 0 and learn to increase gradually, meaning that the model favors the learned statistics increasingly along the training process. That verifies the learned statistics are indeed favored the model for domain generalization. We note that the increasing speed of the residual weights for PACS is not as fast as that for CIFAR-10-C.

Method	Level 1	Level 2	Level 3	Level 4	Level 5	Avg.
ERM+BN	87.8±0.1	81.5±0.2	75.5±0.4	68.2±0.6	56.1±0.8	73.8
ERM+ASR (ASR alone)	89.4±0.2	86.1±0.2	82.9±0.3	78.6±0.6	72.9±1.0	82.0
M-ADA	90.5±0.3	86.8±0.4	82.5±0.6	76.4±0.9	65.6±1.2	80.4
ADA+SN	**91.5±0.2**	88.4±0.6	85.5±0.5	81.2±0 7	75.3±0.8	84.4
ADA+AR	90.4±0.1	87.7±0.3	85.1±0.6	81.1±0.7	76.6±1.0	84.2
ADA+AS	91.4±0.1	88.9±0.2	86.3±0.4	82.8±0.5	77.3±0.7	85.4
ADA+ASR (Ours)	**91.5±0.2**	**89.3±0.6**	**86.9±0.5**	**83.7±0.7**	**78.4±0.8**	**86.0**
p-value: Ours vs. ERM/ERM+ASR/M-ADA	$<$ **1e-3**	$<$ **1e-3**	$<$ **1e-3**	$<$ **1e-3**	$<$ **1e-3**	-
p-value: Ours vs. ADA+SN	0.5	**0.025**	**0.006**	**0.001**	$<$ **1e-3**	-
p-value: Ours vs. ADA+AR	**0.001**	**0.003**	**0.005**	**0.003**	**0.050**	-
p-value: Ours vs. ADA+AS	0.199	0.121	**0.049**	**0.035**	**0.036**	-

Table 10: Single domain generalization accuracies and p-values on CIFAR-10-C with different corruption levels. Significant results are highlighted (p-value ≤ 0.05).

The reason for that could be we used a pretrained model for PACS, which already learned some useful statistics. Fig. 7b shows the evolution of the adaptive weights λ_β and λ_γ in the residual terms of rescaling statistics, where we have the similar observations.

Visualization of Learned Statistics. Figure 8 visualizes the learned standardization statistics $\boldsymbol{\mu}_{\text{stan}}$ and $\boldsymbol{\sigma}_{\text{stan}}$ using t-SNE [33] for different domains in PACS. We notice that the learned statistics show clustering structures for each domain, meaning that ASR-Norm learns different patterns of standardization statistics for each domain. This finding resembles previous papers on using domain-specific statistics for multi-domain data [45]. However, our method learns the soft clustered embeddings in an automatic way without the hard domain label on each sample.

CIFAR-10-C Results for Different Corruption Types. CIFAR-10-C contains 19 corruption types including, brightness, gaussian noise, saturate, contrast, glass blur, shot noise, defocus blur, impulse noise, snow, elastic transform, jpeg compression, spatter, fog, speckle noise, frost, motion blur, zoom blur, gaussian blur, and pixelate. These 19 corruption types can be categoried into 4 categories including, noise, blur, weather, and digital categories [14]. Figure 6 shows the average accuracies for each corruption type across five intensity levels. We observe that ASR-Norm makes consistent improvements over BN in most corruption types, except for brightness.

B. Detailed Formulation of Adversarial Domain Augmentation

Adversarial domain augmentation (ADA) [53] approximately optimizes the robust objective $\mathcal{L}_{RL}$ in Eq 2 by expanding the training set with synthesized adversarial examples along the training process. Specifically, we define the distance D between two distributions P and Q by the Wasserstein distance as [53]:

$$D_\theta(P, Q) := \inf_{M \in \Pi(P,Q)} \mathbb{E}_M[c_\theta((X, Y), (X', Y'))], \quad (10)$$

where c_θ is a learned distance measure over the space $\mathcal{X} \times \mathcal{Y}$. In ADA, c_θ is measured with the semantic features

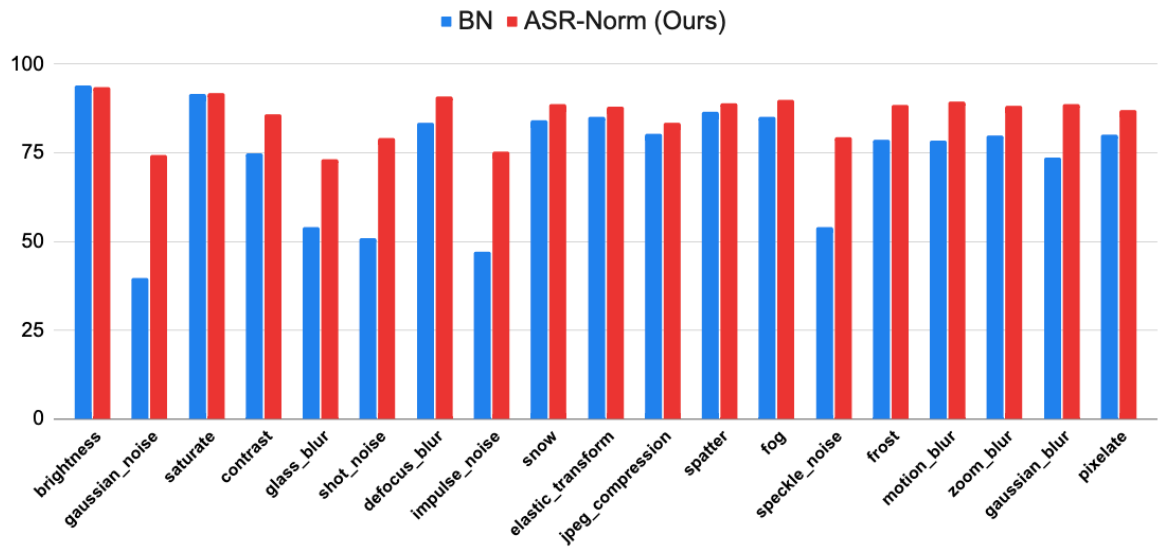
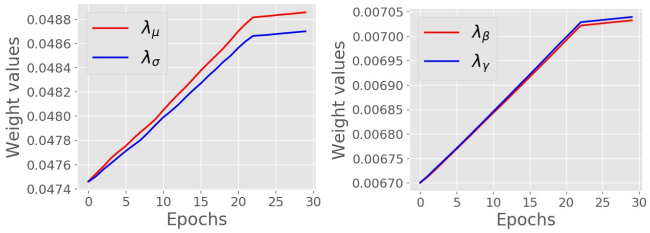
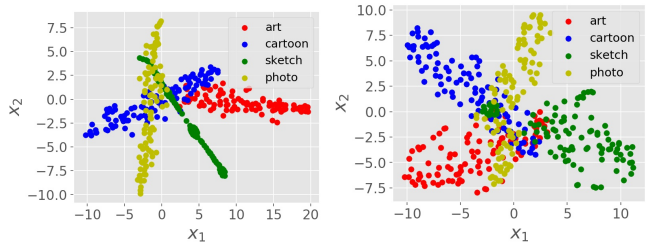


Figure 6: Single domain generalization results on CIFAR-10-C for each corruption type.



(a) Weights $\lambda_{\boldsymbol{\mu}}, \lambda_{\boldsymbol{\sigma}}$. (b) Weights $\lambda_\beta, \lambda_\gamma$.

Figure 7: Weights learn to increase the contribution from learned statistics along the training process on the PACS benchmark.



(a) Visualization of $\boldsymbol{\sigma}_{\text{stan}}$. (b) Visualization of $\boldsymbol{\mu}_{\text{stan}}$.

Figure 8: Visualization of learned standardization statistics for different domains on PACS benchmark.

learned by the neural networks:

$$c_\theta((x,y),(x',y')) := c((F_\theta(x),y),(F_\theta(x'),y')), \quad (11)$$

where F_θ is a feature extractor outputting intermediate activations in the neural networks, and

$$c((z,y),(z',y')) := \frac{1}{2}\|z-z'\|_2^2 + \infty \cdot 1_{\{y\neq y'\}}. \quad (12)$$

Then, the key observation is that optimizing $\mathcal{L}_R$ can be solved by optimizing the Lagrangian relaxation with penalty parameter η:

$$\mathcal{L}_{RL} := \sup_P\{\mathbb{E}_P[l(\theta;(X,Y))] - \eta D_\theta(P,P_s)\}. \quad (13)$$

The gradient of $\mathcal{L}_{RL}$, under a suitable condition, can be rewritten as [3, 53],

$$\nabla_\theta L_{RL} = \mathbb{E}_{(X,Y)\sim P_s}[\nabla_\theta l(\theta;(x_\eta^*,Y))], \quad (14)$$

where

$$x_\eta^* = \underset{x\in\mathcal{X}}{\text{argmax}}\{l(\theta;(x,Y)) - \eta c_\theta((x,Y)),(X,Y))\}. \quad (15)$$

A min-max algorithm is used to estimate the gradients approximately as discussed in Sec. 3.1.2.

C. Additional Experimental Settings

In Figure 9, we show some visual examples from the Digits benchmark. SVHN, MNIST-M and SYN are more challenging domains that have larger distributional shift from MNIST than USPS.

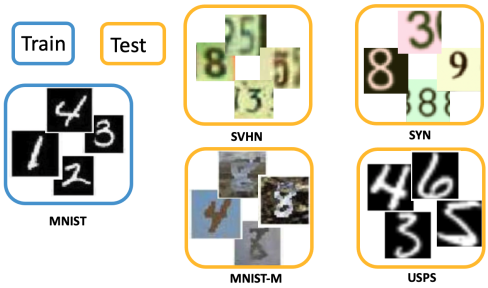


Figure 9: Single domain generalization with Digits benchmark. Only MNIST is used for training and the goal is to learn a model that generalizes well to other digits domains, including, SVHN, MNIST-M, SYN, USPS.