arXiv:2102.02811v1 [cs.CV] 4 Feb 2021

SelfNorm and CrossNorm for Out-of-Distribution Robustness

Zhiqiang Tang
Rutgers University
zhiqiang.tang@rutgers.edu

Yunhe Gao
Rutgers University
yunhe.gao@rutgers.edu

Yi Zhu
Amazon Web Services
yzaws@amazon.com

Zhi Zhang
Amazon Web Services
zhiz@amazon.com

Mu Li
Amazon Web Services
mli@amazon.com

Dimitris Metaxas
Rutgers University
dnm@cs.rutgers.edu

Abstract

Normalization techniques are crucial in stabilizing and accelerating the training of deep neural networks. However, they are mainly designed for the independent and identically distributed (IID) data, not satisfying many real-world out-of-distribution (OOD) situations. Unlike most previous works, this paper presents two normalization methods, SelfNorm and CrossNorm, to promote OOD generalization. SelfNorm uses attention to recalibrate statistics (channel-wise mean and variance), while CrossNorm exchanges the statistics between feature maps. SelfNorm and CrossNorm can complement each other in OOD generalization, though exploring different directions in statistics usage. Extensive experiments on different domains (vision and language), tasks (classification and segmentation), and settings (supervised and semi-supervised) show their effectiveness.

1. Introduction

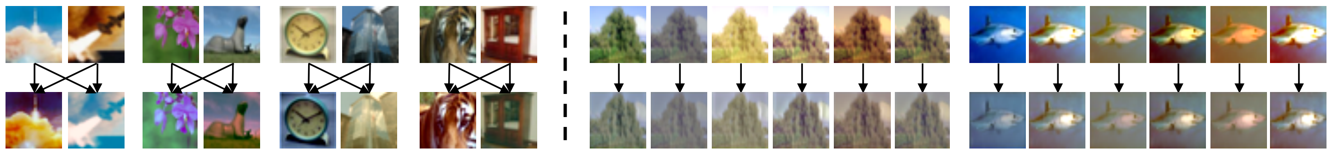
Normalization methods, e.g., Batch Normalization [18], Layer Normalization [1], and Instance Normalization [36], play a pivotal role in training deep neural networks. These methods generally try to make training more stable and convergence faster, assuming that training and test data come from the same distribution. However, few studies investigate normalization in improving OOD generalization in real-world scenarios. For example, image corruptions [11], like snow and blur, can make test data deviate from the clean training distribution. This work aims to encourage the interaction between normalization and OOD generalization. Specifically, we manipulate feature mean and variance to make models generalize better to out-of-distribution data.

Our inspiration comes from the observation that channel-wise mean and variance of feature maps carry some style information. For instance, exchanging the RGB means and variances between two images can transfer style between them, as shown in Figure 1 (a). For many tasks such as CIFAR image classification [20], the style encoded by channel-wise mean and variance is usually less critical in recognizing the object than other information such as object shape. Therefore, we propose CrossNorm which swaps channel-wise mean and variance of feature maps. CrossNorm can augment styles in training, making the model more robust to appearance changes.

Furthermore, given one image in different styles, we can reduce the style discrepancy when adjusting the RGB means and variances properly, as illustrated in Figure 1 (b). Intuitively, the style recalibration can reduce appearance variance, which may be useful in bridging distribution gaps between training and unforeseen testing data. To this end, we propose SelfNorm by using attention [15] to adjust channel-wise mean and variance automatically.

It is interesting to analyze the distinction and connection between CrossNorm and SelfNorm. At first glance, they take opposite actions (style augmentation v.s. style reduction). Even so, they use the same tool: channel-wise statistics and pursue the same goal: OOD robustness. Additionally, CrossNorm can increase the capacity of SelfNorm by style augmentation, making SelfNorm generalize better to OOD data.

Concept and Intuition. The style concept here refers to a family of weak cues associated with the semantic content of interest. For instance, the image style in object recognition can include many appearance-related factors such as color, contrast, and brightness. Style sometimes may help in decision-making, but the model should rely more on vital content cues to become robust. To reduce its bias rather than discarding it, we use CrossNorm with probability in training. The insight beneath CrossNorm is that each instance, or feature map, has its unique style. Further, style cues are not equally important. For example, the yellow color seems more useful than other style cues in recognizing an orange. In light of this, the intuition behind SelfNorm is that at-


(a) Switch RGB mean and variance

(b) Recalibrate RGB mean and variance

Figure 1: CIFAR examples of exchanging (**Left**) and adjusting (**Right**) RGB mean and variance. Swapping the statistics can enrich image styles, whereas recalibrating the statistics properly can encourage style consistency.

tention may help emphasize essential styles and suppress trivial ones. Although we use the channel-wise mean and variance to modify styles, we do not assume that they are sufficient to represent all style cues. Better style representations are available with more complex statistics [23] or even style transfer models [37, 17]. We choose the first and second-order statistics mainly because they are simple, efficient to compute, and can naturally connect normalization to out-of-distribution generalization. In summary, the key contributions are:

- We propose SelfNorm and CrossNorm, two simple yet effective normalization techniques to enhance out-of-distribution generalization.
- SelfNorm and CrossNorm form a unity of opposites in using feature mean and variance for model robustness.
- They are domain agnostic and can advance state-of-the-art robustness performance for different domains (vision or language), settings (fully or semi-supervised), and tasks (classification and segmentation).

2. Related Work

Out-of-distribution generalization. Although the current deep models continue to break records on benchmark IID datasets, they still struggle to generalize to OOD data caused by common corruptions [11] and dataset gaps [30].

To improve the robustness against corruption, Stylized-ImageNet [8] conducts style augmentation to reduce the texture bias of CNNs. Recently, AugMix [13] trains robust models by mixing multiple augmented images based on random image primitives or image-to-image networks [10]. Adversarial noises training (ANT) [31] and unsupervised domain adaptation [32] can also improve the robustness against corruption. CrossNorm is domain agnostic and orthogonal to AugMix and ANT, making it possible for their joint application. Compared to Stylized-ImageNet, CrossNorm has two main advantages. First, CrossNorm is efficient as it transfer styles directly in the feature space of the target CNNs. However, Stylized-ImageNet relies on external style datasets and pre-trained style transfer models. Second, CrossNorm can advance the performance on both clean and corrupted data, while Stylized-ImageNet hurts clean generalization because external styles can result in massive distribution shifts.

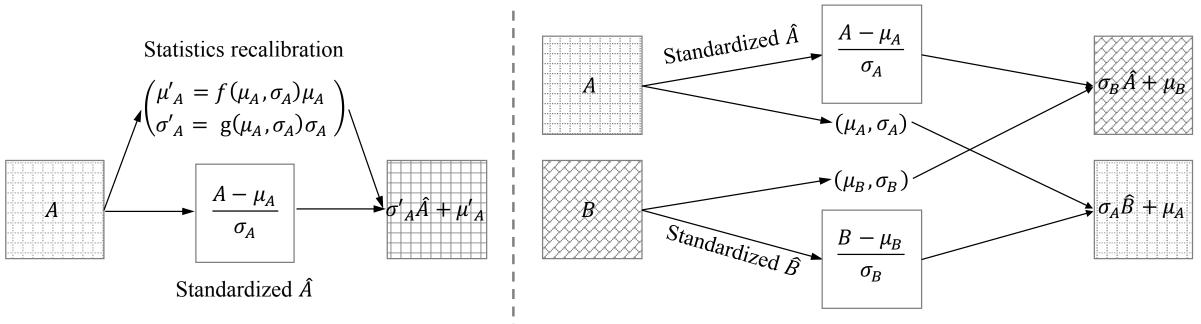
Besides common corruptions, generalization with distribution gaps [30] across different datasets also remains challenging. IBN [28] mixes instance and batch normalization to shrink the domain distances. Domain randomization [40] uses style augmentation for domain generalization on segmentation datasets. It suffers from the same issues of Stylized-ImageNet as it also uses pre-trained style transfer models and additional style datasets. Compared to IBN and the domain randomization, SelfNorm can bridge the domain gaps by style recalibration, and CrossNorm is more efficient and balances better between the source and target domains' performance.

Beyond the vision field, many natural language processing (NLP) applications also face the out-of-distribution generalization challenges [12]. Benefiting from the domain-agnostic property, SelfNorm and CrossNorm can also improve model robustness in the NLP area.

Normalization and attention. Batch Normalization [18] is a milestone technique that inspires many following normalization methods such as Instance Normalization [36], Layer Normalization [1], and Group Normalization [38]. Recently, some works integrate attention [15] into feature normalization. Mode normalization [5] and attentive normalization [22] use attention to weigh a mixture of batch normalizations. Examplar normalization [44] learns to combine multi-type normalizations by attention. By contrast, SelfNorm uses attention with only instance normalization. More importantly, unlike previous normalization approaches, SelfNorm and CrossNorm aim to improve out-of-distribution generalization.

In addition, SelfNorm is different from SE [15], though they use similar attention. First, SelfNorm learns to recalibrate channel-wise mean and variance instead of channel features in SE. Second, SE models the interdependency between channels, while SelfNorm deals with each channel independently. Also, a SelfNorm unit, with $O(n)$, is more lightweight than a SE one, of $O(n^2)$, where n denotes the channel number.

Data augmentation. Data augmentation is an impor-



Statistics recalibration
$\mu'_A = f(\mu_A, \sigma_A)\mu_A$
$\sigma'_A = g(\mu_A, \sigma_A)\sigma_A$
A
$\frac{A-\mu_A}{\sigma_A}$
$\sigma'_A \hat{A} + \mu'_A$
Standardized $\hat{A}$
Standardized $\hat{A}$
$\frac{A-\mu_A}{\sigma_A}$
(μ_A, σ_A)
$\sigma_B \hat{A} + \mu_B$
B
(μ_B, σ_B)
$\sigma_A \hat{B} + \mu_A$
Standardized $\hat{B}$
$\frac{B-\mu_B}{\sigma_B}$

(a) SelfNorm (b) CrossNorm

Figure 2: SelfNorm (**left**) and CrossNorm (**right**). SelfNorm uses attention to recalibrate the mean and variance of a feature map, while CrossNorm swaps the statistics between a pair of feature maps.

tant tool in training deep models. Current popular data augmentation techniques are either label-preserving [3, 24, 14] or label-perturbing [43, 41]. The label-preserving methods usually rely on domain-specific image primitives, e.g., rotation and color, making them inflexible for tasks beyond the vision domain. The label-perturbing techniques mainly work for classification and may have trouble in broader applications, e.g., segmentation. CrossNorm, as a data augmentation method, is readily applicable to diverse domains (vision and language) and tasks (classification and segmentation). The goal of CrossNorm is to boost out-of-distribution generalization, which is also different from many former data augmentation methods.

3. SelfNorm and CrossNorm

Background. Technically, SelfNorm and CrossNorm share the same origin: instance normalization [36]. In 2D CNNs, each instance has C feature maps of size $H \times W$. Given $\boldsymbol{A} \in \mathbb{R}^{H\times W}$, instance normalization first normalizes the feature map and then conducts affine transformation:

$$\gamma \frac{\boldsymbol{A} - \mu_{\boldsymbol{A}}}{\sigma_{\boldsymbol{A}}} + \beta, \tag{1}$$

where $\mu_{\boldsymbol{A}}$ and $\sigma_{\boldsymbol{A}}$ are the mean and standard deviation; γ and β denotes learnable affine parameters. As shown in Figure 1 and also pointed out by the style transfer practices [7, 37, 17], $\mu_{\boldsymbol{A}}$ and $\sigma_{\boldsymbol{A}}$ can encode some style information.

SelfNorm. SelfNorm replaces β and γ with recalibrated mean $\mu'_{\boldsymbol{A}} = f(\mu_{\boldsymbol{A}}, \sigma_{\boldsymbol{A}})\mu_{\boldsymbol{A}}$ and standard deviation $\sigma'_{\boldsymbol{A}} = g(\mu_{\boldsymbol{A}}, \sigma_{\boldsymbol{A}})\sigma_{\boldsymbol{A}}$, as illustrated in Figure 2, where f and g are the attention functions. The adjusted channel becomes:

$$\sigma'_{\boldsymbol{A}} \frac{\boldsymbol{A} - \mu_{\boldsymbol{A}}}{\sigma_{\boldsymbol{A}}} + \mu'_{\boldsymbol{A}}. \tag{2}$$

As f and g learn to scale $\mu_{\boldsymbol{A}}$ and $\sigma_{\boldsymbol{A}}$ based on themselves, *$\boldsymbol{A}$ normalizes itself by self-gating, hence SelfNorm.* SelfNorm is inspired by the fact that attention can help the model emphasize informative features and suppress less useful ones. In terms of recalibrating $\mu_{\boldsymbol{A}}$ and $\sigma_{\boldsymbol{A}}$, SelfNorm expects to highlight the discriminative styles and understate trivial ones. In practice, we use a fully connected (FC) network to wrap attention functions f and g. The architecture is efficient as its input and output are both two scalars. Since each channel has its independent statistics, SelfNorm recalibrates each channel separately using C lightweight FC networks, hence the complexity of $O(C)$.

CrossNorm. CrossNorm exchanges $\mu_{\boldsymbol{A}}$ and $\sigma_{\boldsymbol{A}}$ of channel A with $\mu_{\boldsymbol{B}}$ and $\sigma_{\boldsymbol{B}}$ of channel $\boldsymbol{B}$, i.e., changing β and γ to each other's μ and σ, shown in Figure 2:

$$\sigma_{\boldsymbol{B}} \frac{\boldsymbol{A} - \mu_{\boldsymbol{A}}}{\sigma_{\boldsymbol{A}}} + \mu_{\boldsymbol{B}} \qquad \sigma_{\boldsymbol{A}} \frac{\boldsymbol{B} - \mu_{\boldsymbol{B}}}{\sigma_{\boldsymbol{B}}} + \mu_{\boldsymbol{A}}, \tag{3}$$

where *$\boldsymbol{A}$ and $\boldsymbol{B}$ seem to normalize each other, hence CrossNorm.* CrossNorm is motivated by the key observation that a target dataset, such as a classification dataset, has rich, though subtle, styles. *Specifically, each instance, or even every channel, has its unique style*. Exchanging the statistics can perform efficient style augmentation, reducing the style bias in decision-making. In mini-batch training, we turn on CrossNorm with some probability.

Unity of Opposites. SelfNorm and CrossNorm both start from instance normalization but head in opposite directions. SelfNorm recalibrates statistics to focus on only necessary styles, reducing standardized features (zero-mean and unit-variance) and statistics mixtures' diversity. In contrast, CrossNorm transfers statistics between channels, enriching the combinations of standardized features and statistics. They perform opposite operations mainly because they target different stages. SelfNorm dedicates to style recalibration during testing, while CrossNorm functions only in training. Note that SelfNorm is a learnable module, requiring training to work. Figure 3 shows the flowchart of SelfNorm and CrossNorm. Additionally, SelfNorm helps make the model less sensitive to appearance changes, while CrossNorm aims to lessen the model's style bias. Despite

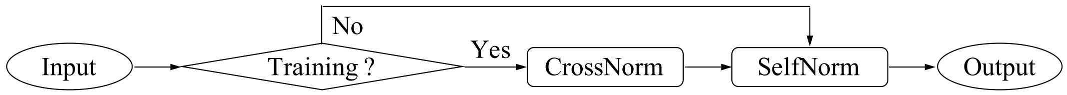


Figure 3: Flowchart for SelfNorm and CrossNorm. SelfNorm learns in training and functions in testing, while CrossNorm works only in training.

these differences, they both can facilitate out-of-distribution generalization. Further, CrossNorm can boost the performance of SelfNorm because its style augmentation can prevent SelfNorm from overfitting to specific styles. Overall, the two seemingly opposed methods form a unity of using normalization statistics to advance out-of-distribution robustness.

3.1. CrossNorm Variants

The core idea of CrossNorm is to swap mean and variance between feature maps. Different choices of feature maps can result in different CrossNorm variants.

1-instance mode. For 2D CNNs, given one instance $\mathbf{X} \in R^{C\times H\times W}$, CrossNorm can exchange statistics between its C channels:

$$\left\{(\boldsymbol{A}, \boldsymbol{B}) \in (\mathbf{X}_{i,:,:}, \mathbf{X}_{j,:,:}) \mid i \neq j, 0 < i, j < C\right\}, \qquad (4)$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ refer to the channel pair in Equation 3.

2-instance mode. If two instances $\mathbf{X}, \mathbf{Y} \in R^{C\times H\times W}$ given, CrossNorm can swap statistics between their corresponding channels, i.e., $\boldsymbol{A}$ and $\boldsymbol{B}$ become:

$$\left\{(\boldsymbol{A}, \boldsymbol{B}) \in (\mathbf{X}_{i,:,:}, \mathbf{Y}_{i,:,:}) \mid 0 < i < C\right\}. \qquad (5)$$

Compared with 1-instance CrossNorm, the 2-instance one tends to consider instance-level style instead of channel-level.

Crop. Moreover, distinct spatial regions probably have different mean and variance statistics. To promote the style diversity, we propose to crop regions for CrossNorm:

$$\{(\boldsymbol{A}, \boldsymbol{B}) \in (\text{crop}(\boldsymbol{A}), \text{crop}(\boldsymbol{B})) \mid r_{\text{crop}} \geq t\} \qquad (6)$$

where the crop function returns a square with area ratio r no less than a threshold $t (0 < t \leq 1)$. The whole channel is a special case in cropping. There are three cropping choices: content only, style only, and both. For content cropping, we crop A only when we use its standardized feature map. In other words, no cropping applies to A when it provides its statistics to B. Cropping both means cropping A and B no matter we employ their standardized feature map or statistics. The cropping strategy can produce diverse styles for both the 1-instance and 2-instance CrossNorms.

3.2. Modular Design

SelfNorm and CrossNorm can naturally work in the feature space, making it flexible to plug them into many network locations. Two questions come: how many units are necessary and where to place them? To simplify the questions, we turn to the modular design by embedding them into a network cell. For example, in ResNet [9], we put them into a residual module. The search space significantly shrinks for the limited positions in a residual module. We will investigate the position choices in experiments. The modular design allows using multiple SelfNorms and CrossNorms in a network. We will show in the ablation study that accumulated style recalibrations are helpful for model robustness. Since excessive style augmentations are harmful [8], we randomly turn on only some CrossNorms in a forward process. Random sampling encourages diverse augmentations even though the same data pass multiple times.

4. Experiment

We evaluate SelfNorm (SN) and CrossNorm (CN) on out-of-distribution data that arise from image corruptions and dataset differences. The evaluation uses not only supervised and semi-supervised settings but also image classification and segmentation tasks. In addition to the vision tasks, we also apply them to a NLP task. Due to limited space, we leave all ablation studies in the appendix.

Image classification datasets. We use benchmark datasets: CIFAR-10 [20], CIFAR-100, and ImageNet[6]. To evaluate the model robustness against corruption, we use the datasets: CIFAR-10-C, CIFAR-100-C, and ImageNet-C [11]. These datasets are the original test data poisoned by 15 everyday image corruptions from 4 general types: noise, blur, weather, and digital. Each noise has 5 intensity levels when injected into images.

Image segmentation datasets. We further validate our method using a domain generalization setting, where the models are trained without any target domain data and tested on the unseen domain. We use the synthetic dataset Grand Theft Auto V (GTA5) [30] as the source domain and generalize to the real-world dataset Cityscapes [2]. GTA5 has the training, validation, and test divisions of 12,403, 6,382, and 6,181, more than those of 2,975, 500, and 1,525 from Cityscapes. Despite the differences, their pixel categories are compatible with each other, allowing to evaluate models' generalization capability from one to another.

Sentiment classification datasets. Besides vision tasks, we demonstrate that our method can also work well on NLP tasks. We use the out-of-distribution (OOD) generaliza-

Table 1: mCE (%) on CIFAR-10-C and CIFAR-100-C. SNCN obtains lower errors than most previous methods with different backbones. Albeit some higher errors than AugMix, it is totally domain agnostic without relying on the image primitives, e.g., rotation, in AugMix. As SNCN and AugMix are orthogonal, their joint usage brings new state-of-the-art results.

CIFAR-10-C	Basic	Cutout	Mixup	CutMix	AutoAug	AdvTr.	AugMix	SN	CN	SNCN	SNCN+AugMix
AllConvNet	30.8	32.9	24.6	31.3	29.2	28.1	15.0	24.0	26.0	17.2	**11.8**
DenseNet	30.7	32.1	24.6	33.5	26.6	27.6	12.7	22.0	24.7	18.5	**10.4**
WideResNet	26.9	26.8	22.3	27.1	23.9	26.2	11.2	20.8	21.6	16.9	**9.9**
ResNeXt	27.5	28.9	22.6	29.5	24.2	27.0	10.9	21.5	22.4	15.7	**9.1**
Mean	29.0	30.2	23.5	30.3	26.0	27.2	12.5	22.1	23.7	17.0	**10.3**
CIFAR-100-C	Basic	Cutout	Mixup	CutMix	AutoAug	AdvTr.	AugMix	SN	CN	SNCN	SNCN+AugMix
AllConvNet	56.4	56.8	53.4	56.0	55.1	56.0	42.7	50.3	52.2	42.8	**36.8**
DenseNet	59.3	59.6	55.4	59.2	53.9	55.2	39.6	53.9	55.4	48.5	**37.0**
WideResNet	53.3	53.5	50.4	52.9	49.6	55.1	35.9	47.4	48.8	43.7	**33.4**
ResNeXt	53.4	54.6	51.4	54.1	51.3	54.4	34.9	47.6	47.0	40.8	**30.8**
Mean	55.6	56.1	52.6	55.5	52.5	55.2	38.3	49.8	50.9	43.5	**34.7**

Table 2: Clean error and mCE (%) of ResNet50 trained 90 epochs on ImageNet. SNCN, using simple domain-agnostic statistics, achieves comparable performance as AugMix. Jointly applying SNCN with AugMix and IBN can produce the lowest clean and corruption errors.

		Noise			Blur				Weather				Digital				
Aug.	Clean	Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	mCE
Standard	23.9	79	80	82	82	90	84	80	86	81	75	65	79	91	77	80	80.6
Patch Uniform	24.5	67	68	70	74	83	81	77	80	74	75	62	77	84	71	71	74.3
Random AA*	23.6	70	71	72	80	86	82	81	81	77	72	61	75	88	73	72	76.1
MaxBlur pool	23.0	73	74	76	74	86	78	77	77	72	63	56	68	86	71	71	73.4
SIN	27.2	69	70	70	77	84	76	82	74	75	69	65	69	80	64	77	73.3
AugMix*	23.4	66	66	66	69	80	65	68	72	72	66	60	63	78	66	71	68.4
CN	23.4	73	75	75	78	89	79	82	79	75	66	61	69	97	69	74	75.3
SN	23.7	69	71	69	77	87	77	80	75	77	70	61	73	83	61	71	73.4
SNCN	23.3	66	67	65	77	89	76	80	72	72	67	59	47	83	62	72	70.4
SNCN+AugMix	**22.3**	61	62	60	70	77	62	68	62	65	63	55	43	73	55	66	**62.8**

tion setting in binary sentiment classification. The model is trained on IMDb dataset [26] and is tested on SST-2 testing dataset [33]. The IMDb dataset collects highly polarized full-length lay movie reviews with 25,000 positive and 25,000 negative reviews. The SST-2 contains 9613 and 1821 reviews for training and testing, which is also a binary sentiment classification dataset but instead contains pithy expert movie reviews.

Metric. For image classification, we use test errors to measure robustness. Given corruption type c and severity s, let E_s^c denote the test error. For CIFAR datasets, we use the average over 15 corruptions and 5 severities: $1/75\sum_{c=1}^{15}\sum_{s=1}^{5}E_{c,s}$. In contrast, for ImageNet, we normalize the corruption errors by those of AlexNet [21]: $1/15\sum_{c=1}^{15}(\sum_{s=1}^{5}E_s^c/\sum_{s=1}^{5}E_{c,s}^{AlexNet})$. The above two metrics follow the convention [13] and are denoted as mean corruption errors (mCE) whether they are normalized or not. Different from classification, segmentation use the mean Intersection over Union (mIoU) over all categories as metric. For sentiment classification, we report accuracy as the metric.

Hyper-parameters. In the experiments, a SN unit uses one fully connected layer, followed by Batch Norm and a sigmoid layer. We put CN ahead of SN, and plug them into every cell in a network, e.g., each residual module in a ResNet. During training, we turn on only some CNs with probability to avoid excessive data augmentation. Unless specified, 2-instance CN is used with cropping. We sample the cropping bounding box ratio uniformly and set the threshold $t = 0.1$. Please refer to the appendix for details of active number and probability.

4.1. Robustness against Unseen Corruptions for Image Classification

Supervised training on CIFAR. Following AugMix [13], we evaluate SN and CN with four different backbones: an All Convolutional Network [35], a DenseNet-BC (k = 12, d = 100) [16], a 40-2 Wide ResNet [42], and a ResNeXt-29

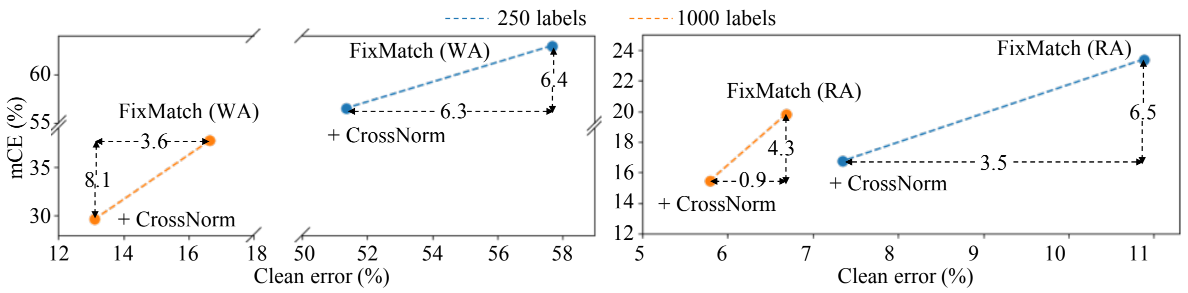


Figure 4: CN for semi-supervised CIFAR-10 classification. We apply CN on top of FixMatch with weak augmentation (WA) (**Left**), or strong RandAugment (RA) (**Right**). For either case, CN can substantially reduce both clean and corruption errors. Compared with RA, CN performs domain agnostic data augmentation, easily applicable to new domains.

(32×4) [39]. We also use the same hyper-parameters in the AugMix Github repository[1].

According to Table 1, individual SN and CN can outperform most previous approaches on robustness against unseen corruptions and combining them can decrease the mean error by ∼12% on both CIFAR-10-C and CIFAR-100-C. One possible explanation is that the corruptions mainly change image textures. SN and CN, through style recalibration and augmentation, may help reduce the texture sensitivity and bias, making the classifiers more robust to unseen corruptions. Also, the domain-agnostic SN and CN are orthogonal to AugMix, which relies on domain-specific operations. Their joint application can continue to lower the mCEs by 2.2% and 3.6% on top of AugMix.

Supervised training on ImageNet. Following the AugMix Github repository, we train a ResNet-50 for 90 epochs with weight decay 1e-4. The learning rate starts from 0.1, divided by 10 at epochs 30 and 60. Note that AugMix reports the results of 180 epochs in their paper. For a fair comparison, we also train it 90 epochs in our experiments. Besides, we also add Instance-batch normalization (IBN) [28] in the final combination with AugMix. It was initially designed for domain generalization but can also boost model robustness against corruption.

Table 2 gives the results on ImageNet. We can observe that both clean and corrupted errors decrease when applying SN and CN separately. Their joint usage can make the clean and corruption errors drop by 10.2% and 0.6% simultaneously, closing the gap with AugMix. Moreover, applying SN and CN on top of AugMix can significantly lower its clean and corruption errors by 1.1% and 5.6%, respectively, achieving state-of-the-art performance. IBN also makes some contributions here since it is complementary to other components.

Semi-supervised training on CIFAR. Apart from supervised training, we also evaluate CN in semi-supervised learning. Following state-of-the-art FixMatch [34] setting, we train a 28-2 Wide ResNet for 1024 epochs on CIFAR-10. The SGD optimizer applies with Nesterov momentum 0.9, learning rate 0.03, and weight decay 5e-4. The probability threshold to generate pseudo-labels is 0.95, and the weight for unlabeled data loss is 1. We sample 250 and 4,000 labeled data with random seed 1, leaving the rest as unlabeled data. In each experiment, we apply CrossNorm to either all data or only unlabeled data and choose the better one. Our experiments use the Pytorch FixMatch implementation [2].

Figure 4 shows the semi-supervised results. We run FixMatch with the strong RandAugment [4] or only weak random flip and crop augmentations. With either FixMatch version, CN can always decrease both the clean and corruption errors, demonstrating its effectiveness in semi-supervised training. Especially with the help of CN, training with 250 labels even has 3% lower corruption error than with 1000 labels, according to the right sub-figure. Additionally, two points are noteworthy here. First, we try FixMatch with only weak augmentations to simulate more general situations. For new domains other than natural images, humans may have the limited domain knowledge to design advanced augmentation operations. Fortunately, CN is domain-agnostic and easily applicable to such situations. Moreover, previous semi-supervised methods mainly focus on in-distribution generalization. Here we introduce out-of-distribution robustness as another metric for more comprehensive evaluation.

4.2. Generalization from Synthetic to realistic data for Image Segmentation

Training setup. We perform domain generalization from GTA5 (synthetic) to Cityscapes (realistic), following the setting of IBN [28]. It uses 1/4 training data in GTA5 to match the data scale of Cityscapes. We train the FCN [25] with ResNet50 backbone in source domain GTA5 for 80 epochs with batch size 16. The network is initialized with ImageNet pre-trained weights. We test the trained model on both the source and target domains. The training uses

[1]https://github.com/google-research/augmix

[2]https://github.com/kekmodel/FixMatch-pytorch

Table 3: Segmentation results (mIoU) on GTAV-Cityscapes domain generalization using a FCN with ResNet50. SN and CN are comparable with IBN and domain randomization (DR) on the target domain. Combining SN and CN can achieve state-of-the-art results.

Methods	Baseline	IBN	DR	SN	CN	SNCN
Source	63.7	64.2	49.0	**64.6**	61.2	63.5
Target	21.4	29.6	32.7	29.9	32.0	**36.5**

random scaling, flip, rotation, and cropping (713×713) for data augmentation. We use the 2-instance CN with style cropping in this setting. Besides, we re-implement the domain randomization [40] and make the training iterations the same as ours. It transfers the synthetic images to 15 auxiliary domains with ImageNet image styles.

Results. Based on Table 3, SN and CN both can substantially increase the segmentation accuracy on the target domain by 8.5% and 10.6%. SN learns to highlight the discriminative styles that are likely to share across domains. CN performs style augmentation to make the model focus more on domain-invariant features. SN and CN get comparable generalization performance as state-of-the-art IBN [28] and domain randomization [40]. However, CN significantly outperforms the domain randomization method by 12.2% on the source accuracy. Because the domain randomization transfers external styles to the source training data, causing dramatic distribution shifts. Moreover, combining SN and CN gives the best generalization performance while still maintaining high source accuracy.

4.3. Out-of-Distribution Generalization for Sentiment Classification

Setup. We also conduct out-of-distribution generalization on the binary sentiment classification task in the NLP field to validate the versatility of SN and CN. The model is trained on the IMDb dataset and then tested on SST-2 dataset. Follow the setting of [12], we use GloVe [29] word embedding and the Convolutional Neural Networks (ConvNets) [19] as the classification model. We use the implementation of ConvNets in this repository[3]. The convolutional layers with three kernel sizes (3,4,5) are used to extract $n - gram$ features within the review texts. SN and CN units are placed between the embedding layer and the convolutional layers. We use the Adam optimizer and train the model for 20 epochs.

Results. From Table 4, we can find that SN improves the performance in both the source and target domains by 2.07% and 0.63%. CN can also increase target accuracy without much degradation in the source domain. Combining them gives a 3.05% accuracy boost. This experiment

[3]https://github.com/bentrevett/pytorch-sentiment-analysis

Table 4: Accuracy (Acc) on OOD generalization for sentiment classification using GloVe embedding and ConvNets model. We train the model on IMDb source dataset and test on SST-2 target dataset.

Methods	Baseline	SN	CN	SNCN
Source	85.67	**86.30**	85.14	85.92
Target	71.86	73.93	73.03	**74.91**

indicates that SN and CN can also work in the NLP area, not limited to the vision tasks. Despite the lack of intuitive explanations as for the image data, the mean and variance statistics in NLP data are also useful in facilitating out-of-distribution generalization.

4.4. Visualization

Apart from the quantitative comparisons, we also provide some visualization results of SN and CN to better understand their effects. It is nontrivial to visualize them directly in feature space. To deal with this, we map the feature changes made by SN and CN back to image space by inverting the feature representations [27]. For detailed experimental settings, refer to the appendix.

To visualize SN at a network location, we first forward an image to obtain the target representation immediately after the SN. Then we turn off the chosen SN and optimize the original image to make its representation fit the target one. In this way, we can examine SN's effect by observing changes in image space. As shown in Figure 6, SN can primarily reduce the contrast and color at the first network block. The effect becomes more subtle as SN goes deeper into the network. One possible explanation is that the high-level representations lose many low-level details, making it difficult to visualize their changes.

In addition to visualizing individual SNs, it is also interesting to see their compound effect. To this end, we reconstruct an image from random noises by matching its representation with a given one. The reconstructed image can show what information is preserved by the feature representation. By comparing two reconstructed images from a network with or without SN, we can observe the joined recalibration effects of SNs before a selected location. From Figure 7, we can find SNs in the first two network blocks can suppress much style information and preserve object shapes. The reconstructions from block 3 do not look visually informative due to the high-level abstraction. Even so, SNs can restrain the high-frequency signals kept in the vanilla network.

In visualizing CN, we pair one content image with multiple style images for better illustration. We first forward them to get their representations at a chosen position. Then we compute standardized features from the content image

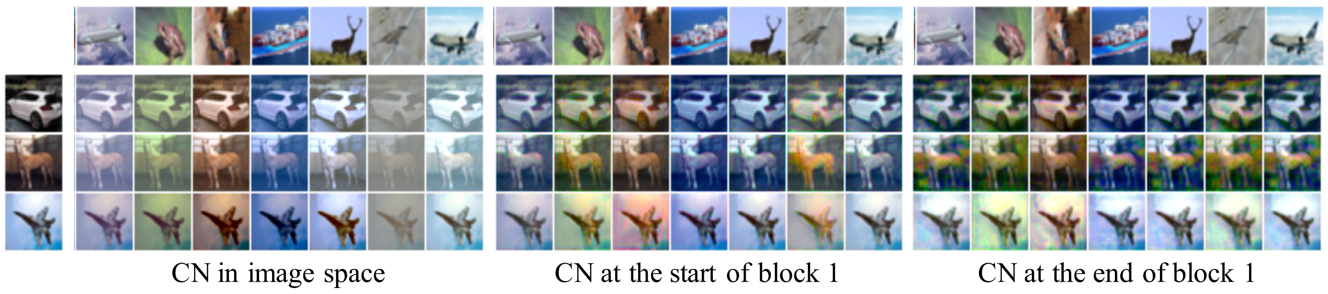


Figure 5: CN visualization at image level (**Left**), the head (**Middle**) and tail (**Right**) of block 1 in a WideResNet-40-2. Both the content (**Row**) and style (**Column**) images are from CIFAR-10. The style rendering changes from global to local as CN gets deeper in the network.

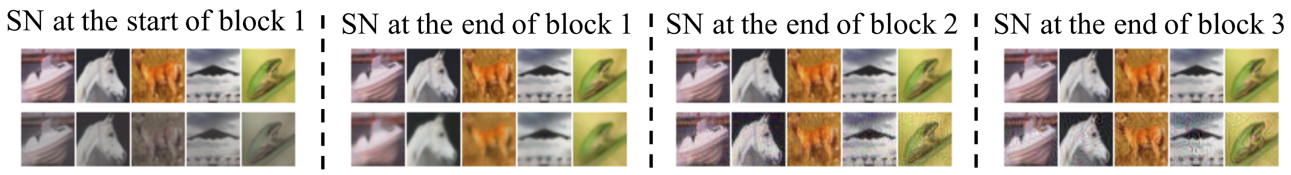


Figure 6: Visualizing 4 single SN units by comparing images before (**Top**) and after (**Bottom**) them. The left two, lying in shallow locations, can adjust styles by suppressing color and adding blur. As SN goes deeper, the recalibration effect is subtle, due to the high-level feature abstraction.

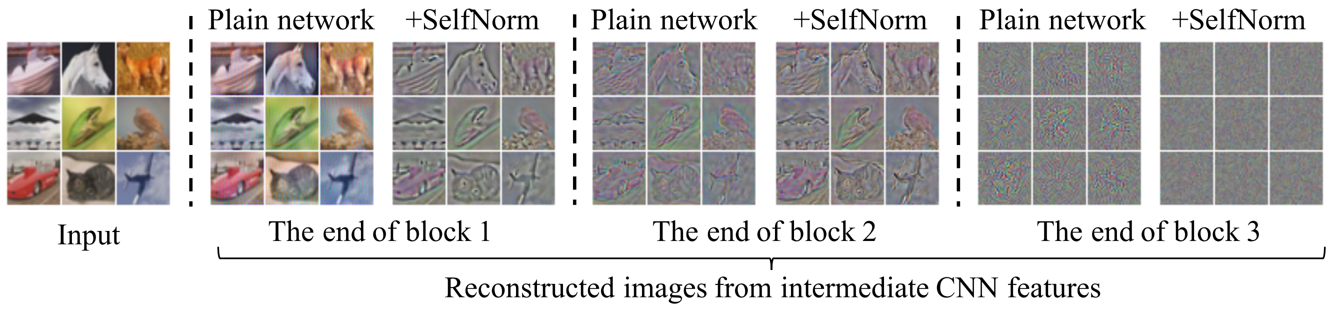


Figure 7: Visualizing accumulated SNs by comparing inverted images. SNs in block 1 can wash away much style information preserved in the vanilla network. Similarly, the plain network's final representation retains some high-frequency signals which are suppressed by SNs.

representation and means and variances of the style image representations. The optimization starts from the content image and tries to fit its representation to the target one mixing the standardized features with different means and variances. Figure 5 shows diverse style changes made by CN. The style changes become more local and subtle as CN moves deeper in the network.

5. Conclusion

In this paper, we have presented SelfNorm and CrossNorm, two simple yet effective normalization techniques to improve OOD robustness. They form a unity of opposites as they confront and conform to each other in terms of approach (statistics usage) and goal (OOD robustness). Beyond their extensive applications, they may also shed light on developing domain agnostic methods applicable to multiple fields such as vision and language, and broad OOD generalization circumstances such as unseen corruptions and distribution gaps across datasets. Given the simplicity of SelfNorm and CrossNorm, we believe there is substantial room for improvement. The current channel-wise mean and variance are not optimal to encode diverse styles. One possible direction is to explore better style representations.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe

Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[3] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, pages 113–123, 2019.

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. *arXiv preprint arXiv:1909.13719*, 2019.

[5] Lucas Deecke, Iain Murray, and Hakan Bilen. Mode normalization. *arXiv preprint arXiv:1810.05466*, 2018.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

[8] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proceedings of the International Conference on Learning Representations*, 2019.

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

[10] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

[11] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[12] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*, 2020.

[13] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *Proceedings of the International Conference on Learning Representations*, 2020.

[14] Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, and Xi Chen. Population based augmentation: Efficient learning of augmentation policy schedules. *arXiv preprint arXiv:1905.05393*, 2019.

[15] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[19] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012.

[22] Xilai Li, Wei Sun, and Tianfu Wu. Attentive normalization. *arXiv preprint arXiv:1908.01259*, 2019.

[23] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Advances in neural information processing systems*, pages 386–396, 2017.

[24] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In *NeurIPS*, pages 6662–6672, 2019.

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[26] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[27] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[28] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 464–479, 2018.

[29] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[30] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European conference on computer vision*, pages 102–118. Springer, 2016.

[31] Evgenia Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. *arXiv preprint arXiv:2001.06057*, 2020.

[32] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift

adaptation. *Advances in Neural Information Processing Systems*, 33, 2020.

[33] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[34] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

[35] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[36] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[37] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017.

[38] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[39] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[40] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2100–2110, 2019.

[41] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, 2019.

[42] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[43] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[44] Ruimao Zhang, Zhanglin Peng, Lingyun Wu, Zhen Li, and Ping Luo. Exemplar normalization for learning deep representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12726–12735, 2020.

6. Appendix

In each experiment, we conduct a grid search on four combinations of active numbers (1, 2) and probability (0.25, 0.5) and report the best result.

6.1. Ablation Study on CIFAR

CN variants. CN can be in 1-instance or 2-instance mode with four cropping options. According to Table 5, the 2-instance mode constantly gets lower errors than the 1-instance. Furthermore, cropping can help decrease the error since it can encourage style augmentation diversity. Note that style cropping may not always be superior. We will give a more detailed study on cropping choices later.

Table 5: 1-instance CN *v.s.* 2-instance CN. We report the mCEs of WideResNet-40-2 on CIFAR-100-C. The 2-instance mode consistently obtains lower errors than the 1-instance one. Moreover, proper cropping can further help decrease the errors.

	1-instance				2-instance			
Crop	Neither	Content	Style	Both	Neither	Content	Style	Both
mCE	50.5	50.6	50.6	49.6	48.8	49.0	**47.9**	48.5

Blocks choices for SN and CN. Given both SN and CN placed at the post-addition location of a residual cell in WideResNet-40-2, we study which blocks in a network should build on the modified cells. No residual cell is required if applying SN and CN to the image space. According to Table 6, They both perform the best on CIFAR-100-C when plugged into all blocks.

Table 6: Exploration of Block choices for SN and CN. We compare the mCEs (%) when applying SN and CN to image space or different blocks in WideResNet-40-2. Using them in all blocks gives the lowest errors on CIFAR-100-C.

Stages	N/A	Image	Block 1	Block 2	Block 3	All
SN	53.3	52.9	48.9	52.2	51.3	**47.4**
CN	53.3	54.3	52.2	51.2	51.5	**48.8**

Order of SN and CN. In this experiment, we study two orders: SN→CN and CN→SN when plugging them into the post-addition place in all residual cells of WideResNet-40-2. Table 7 shows very close mCEs, indicating their order has little influence on the robustness performance.

SN *v.s.* **SE**. Although SN shares a similar attention mechanism with SE, SN obtains much lower corruption error than SE, according to Table 8. SN recalibrates the feature map statistics, suppressing unseen styles in the OOD data, whereas SE, modeling the interdependence between feature channels, may not help OOD robustness.

Table 7: Order study of SN and CN. They both locate at the post-addition position in each residual cell of WideResNet-40-2, and we report the mCE on CIFAR-100-C.

Order	SN→CN	CN→SN
mCE(%)	46.9	46.6

Table 8: Comparison with SE. SN obtains much lower mCE than SE when they are applied to WideResNet-40-2 and CIFAR-100-C. We place SN in the post-addition location.

	Basic	SE	SE(post-addition)	SN
mCE(%)	53.3	52.3	51.0	**47.4**

Modular positions. Here we investigate the positions of SN and CN inside network cells. We give a comprehensive study on different cells and measure the performance on both CIFAR-10-C and CIFAR-100-C. Specifically, we conduct experiments using four backbones: AllConvNet, DenseNet, WIdeResNet and ResNeXt, consisting of three types of cells: naive convolutional cell, dense cell, and residual module, illustrated in Figures 8 and 9. According to Tables 9 and 10, SN's optimal positions are different for CIFAR-10-C and CIFAR-100-C, while CN has stable best positions across the two datasets.

Table 9: Evaluation of **SN** modular positions for AllConvNet, DenseNet, WIdeResNet and ResNeXt. The impacts of different positions are measured by mCE on both CIFAR-10-C (**Top**) and CIFAR-100-C (**Bottom**). Note that the four backbones have three types of cells whose positions are illustrated in Figures 8 and 9.

SN on CIFAR-10-C				
Position	1	2	3	-
AllConvNet	**24.0**	26.4	25.6	-
Position	Pre	Post	-	-
DenseNet	23.4	**22.0**	-	-
Position	Residual	Post	Pre	Identity
WideResNet	22.7	21.3	**20.8**	22.3
Position	Residual	Post	Pre	Identity
ResNeXt	21.9	24.8	**21.5**	22.0
SN on CIFAR-100-C				
Position	1	2	3	-
AllConvNet	**50.3**	51.6	51.0	-
Position	Pre	Post	-	-
DenseNet	**53.9**	54.7	-	-
Position	Residual	Post	Pre	Identity
WideResNet	49.3	**47.4**	49.8	48.4
Position	Residual	Post	Pre	Identity
ResNeXt	**47.6**	49.0	50.9	50.4

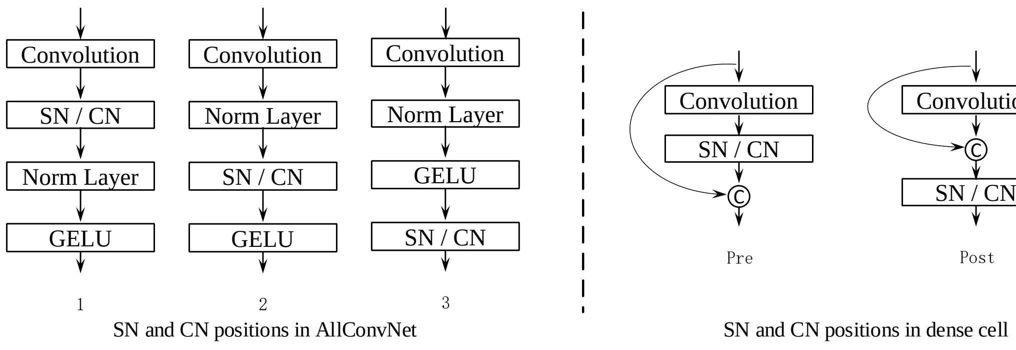


Figure 8: Illustration of SN and CN positions in AllConvNet block, and dense cells in DenseNet. For blocks in AllConvNet, we name the position after convolution layer as *1*, after normalization layer as *2*, and after GELU layer as *3*. For dense cells in DenseNet, we label the position before feature concatenation as *Pre*, and after concatenation as *Post*.

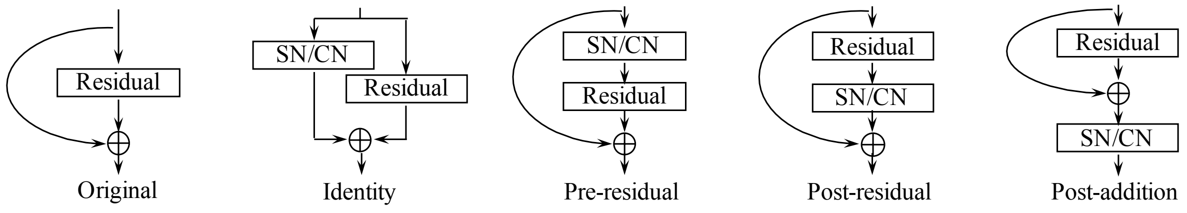


Figure 9: Illustration of SN and CN positions in a residual module. We explore four positions: identity, pre-residual, post-residual, and post-addition.

Table 10: Evaluation of **CN** positions in the cells of four backbones. We measure the position influence by mCE on CIFAR-10-C (**Top**) and CIFAR-100-C (**Bottom**). The position choices in the naive convolution cell, dense cell, and residual module are shown in Figures 8 and 9.

CN on CIFAR-10-C				
Position	1	2	3	-
AllConvNet	**26.0**	26.3	26.8	-
Position	Pre	Post	-	-
DenseNet	**24.7**	29.2	-	-
Position	Residual	Post	Pre	Identity
WideResNet	25.2	**21.6**	24.9	23.3
Position	Residual	Post	Pre	Identity
ResNeXt	26.7	**22.4**	23.8	26.9

CN on CIFAR-100-C				
Position	1	2	3	-
AllConvNet	**52.2**	52.5	52.7	-
Position	Pre	Post	-	-
DenseNet	**55.4**	57.6	-	-
Position	Residual	Post	Pre	Identity
WideResNet	52.1	**48.8**	51.7	50.3
Position	Residual	Post	Pre	Identity
ResNeXt	51.5	**47.0**	47.9	50.2

Table 11: Study of CN cropping choices. We evaluate four cropping choices: neither, content, style, and both when jointly using SN and CN in four backbones. The performance is measured by mCE on both CIFAR-10-C (**Top**) and CIFAR-100-C (**Bottom**). We put the modular positions next to the backbone names.

SNCN with cropping on CIFAR-10-C				
Backbone	Neither	Content	Style	Both
AllConvNet, 1	19.0	20.3	**18.8**	20.3
DenseNet, Conv1 Pre	18.8	**18.2**	18.7	18.8
WideResNet, Post	17.9	18.0	**16.8**	17.5
ResNeXt, Post	**17.7**	18.5	18.4	18.6

SNCN with cropping on CIFAR-100-C				
Backbone	Neither	Content	Style	Both
AllConvNet, 1	44.2	46.9	**43.9**	46.1
DenseNet, Conv1 Pre	51.4	49.4	49.1	**49.0**
WideResNet, Post	46.6	45.1	45.8	**44.5**
ResNeXt, Post	**41.0**	44.9	43.0	46.5

Cropping Choices for CN. Cropping enables diverse statistics transfer between feature maps. Here we study four cropping choices: neither (no cropping), style, content, and both. In Table 11, we can find the best cropping choice may change over backbones and datasets.

Table 12: Incremental ablation study for SN, CN, cropping, consistency regularization and AugMix. We report the mCEs of four backbones on both CIFAR-10-C (**Top**) and CIFAR-100-C (**Bottom**). The modular position and cropping choice are also given in each row.

CIFAR-10-C							
Backbone	Basic	SN	CN	SN+CN +Crop	SN+CN+Crop +Consistency	AugMix	SNCN+Crop +AugMix
AllConvNet, 1, style	30.8	24.0	26.0	18.8	17.2	15.0	**11.8**
DenseNet, Conv1 Pre, both	30.7	22.0	24.7	18.8	18.5	12.7	**10.4**
WideResNet, Post, both	26.9	20.8	21.6	17.5	16.9	11.2	**9.9**
ResNeXt, Post, neither	27.5	21.5	22.4	17.7	15.7	10.9	**9.1**

CIFAR-100-C							
Backbone	Basic	SN	CN	SN+CN +Crop	SN+CN+Crop +Consistency	AugMix	SNCN+Crop +AugMix
AllConvNet, 1, style	56.4	50.3	52.2	43.9	42.8	42.7	**36.8**
DenseNet, Conv1 Pre, both	59.3	53.9	55.4	49.0	48.5	39.6	**37.0**
WideResNet, Post, both	53.30	47.4	48.8	44.5	43.7	35.9	**33.4**
ResNeXt, Post, neither	53.4	47.6	47.0	41.0	40.8	34.9	**30.8**

Incremental ablation study. SN and CN are general and straightforward normalization techniques to improve the OOD robustness. They are orthogonal to each other, and other methods such as the consistency regularization [13] and AugMix [13]. According to Table 12, they can lower the corruption error both separately and jointly. On top of them, proper cropping, consistency regularization, and domain-specific AugMix can further advance the OOD robustness.

6.2. Ablation Study on ImageNet

CN *v.s.* **Stylized-ImageNet**. We also compare CN to Stylized-ImageNet, which transfers styles from external datasets to perform style augmentation. Stylized-ImageNet finetunes a pre-trained ResNet-50 for 45 epochs with double data (stylized and original ImageNets) in each epoch. To compare CrossNorm with Stylized-ImageNet, we perform the finetuning for 90 epochs using only the original ImageNet. In Table 13, although Stylized-ImageNet has 2% lower corruption error than CN, its clean error is 3.8% higher. Because the external styles in Stylized-ImageNet cause large distribution shifts, impairing its clean generalization. In contrast, The more consistent yet diverse internal styles help CN decreases both corruption and clean errors.

SN and CN locations. Moreover, in Table 14, we also investigate the SN and CN locations in a residual module using ImageNet and ResNet50. Similar to the CIFAR results, the post-addition position performs the best for corruption robustness.

Ablation study with IBN. Table 15 reports the results of applying SN or CN with IBN. We can observe that they can cooperate to improve the corruption robustness of ResNet50. Moreover, integrating SN, CN, IBN, and AugMix can bring the lowest corruption error. This shows SN and CN's advantage that they are general and simple to boost other state-of-the-art methods.

Table 13: Comparison of Stylized-ImageNet and CN. Following the Stylized-ImageNet setup, we finetune a pre-trained ResNet50 model 90 epochs on ImageNet. Compared with SIN, CN holds a better balance between clean and corruption errors.

	Basic	Stylized-ImageNet	CN
Clean error (%)	23.9	27.2	**23.4**
mCE(%)	80.6	**73.3**	75.3

Table 14: Investigation of SN (**Top**) and CN (**Bottom**) positions in a residual module of ResNet50 trained 90 epochs on ImageNet.

SN modular positions				
Position	Identity	Pre-Residual	Post-Residual	Post-Addition
Clean error (%)	24.0	**23.0**	23.2	23.7
mCE(%)	75.5	75.8	74.8	**73.4**

CN modular positions				
Position	Identity	Pre-Residual	Post-Residual	Post-Addition
Clean error (%)	25.2	**23.4**	23.5	**23.4**
mCE(%)	78.2	75.8	77.5	**75.3**

Table 15: Ablation study of IBN, SN, CN, consistency regularization(CR), and AugMix(AM) on ImageNet-C with ResNet50. IBN, initially designed for domain generalization, can also decrease mCE. Both SN and CN can further lower the error based on IBN. Combining them with AM gives the best robustness performance.

	ResNet50	ResNet50+IBN(a)				ResNet50+IBN(b)			
	Basic	Basic	+CN	+CN+CR	+CN+AM	Basic	+SN	+SN+AM	+SNCN+AM
Clean err(%)	23.9	23.2	23.1	22.6	22.5	24.0	23.5	**22.3**	**22.3**
mCE(%)	80.6	75.1	73.2	73.6	66.4	74.1	72.6	64.1	**62.8**

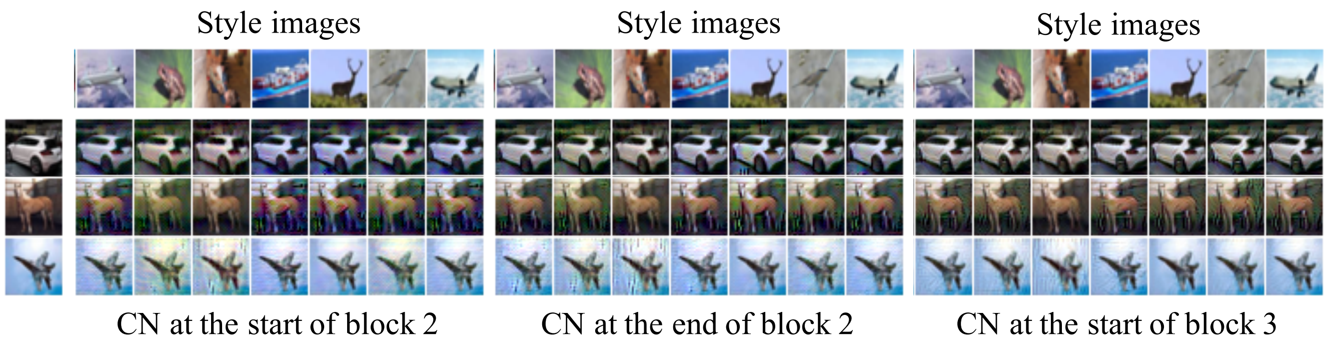


Figure 10: CN visualization at the head (**Left**), the tail of (**Middle**) block 2 and the start of block 3 (**Right**) in a WideResNet-40-2. Both the content (**Row**) and style (**Column**) images are from CIFAR-10. Compared to CNs in block 1, shown in Figure 5, the CNs in blocks 2 and 3 have weaker style transfer effects because the statistics in high-level feature maps may contain less low-level visual information.

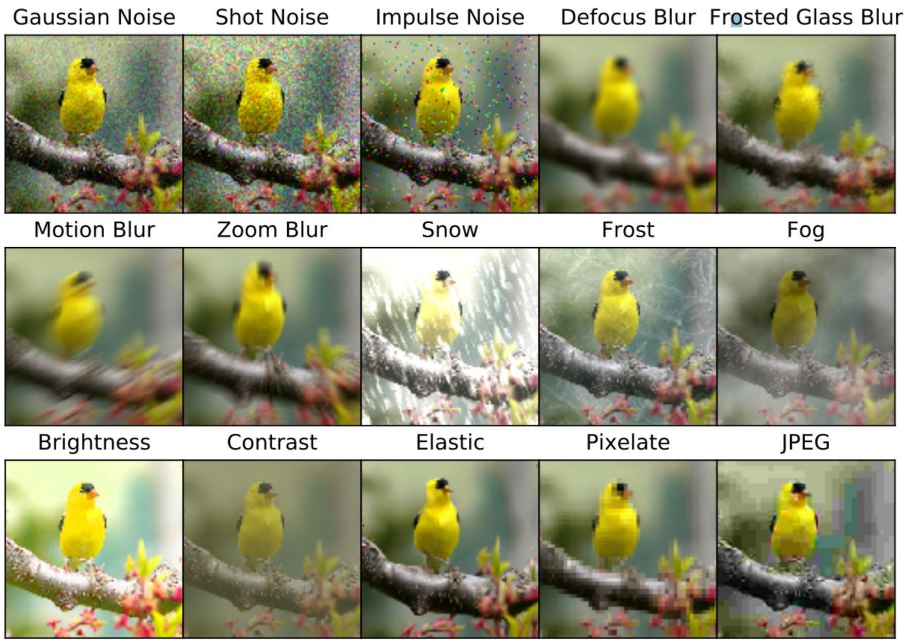


Figure 11: A demostration of corrupted images in ImageNet-C dataset [11]. Fifteen types of algorithmically generated corruptions from noise, blur, weather, and digital categories are applied to images to create the corrupted dataset.

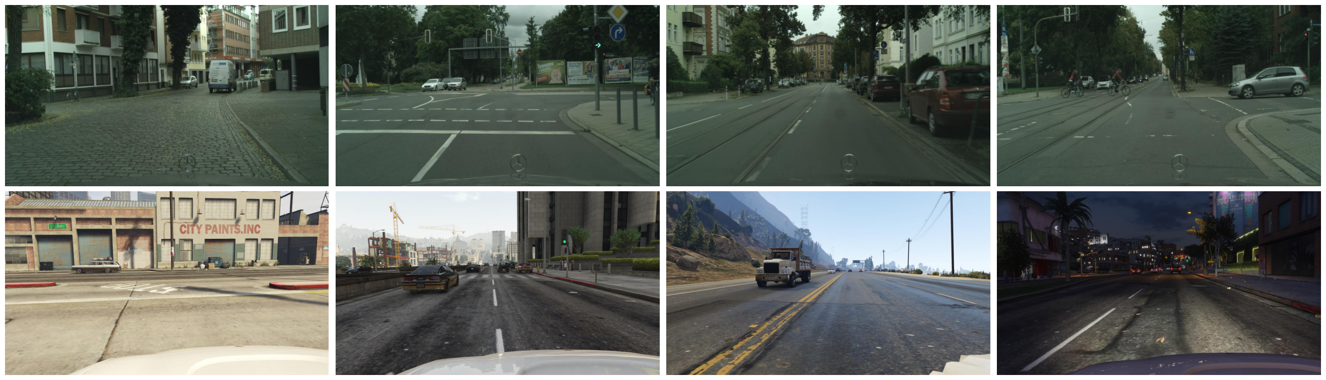


Figure 12: Some examples in segmentation datasets Cityscapes (**Top**) and GTA5 (**Bottom**).

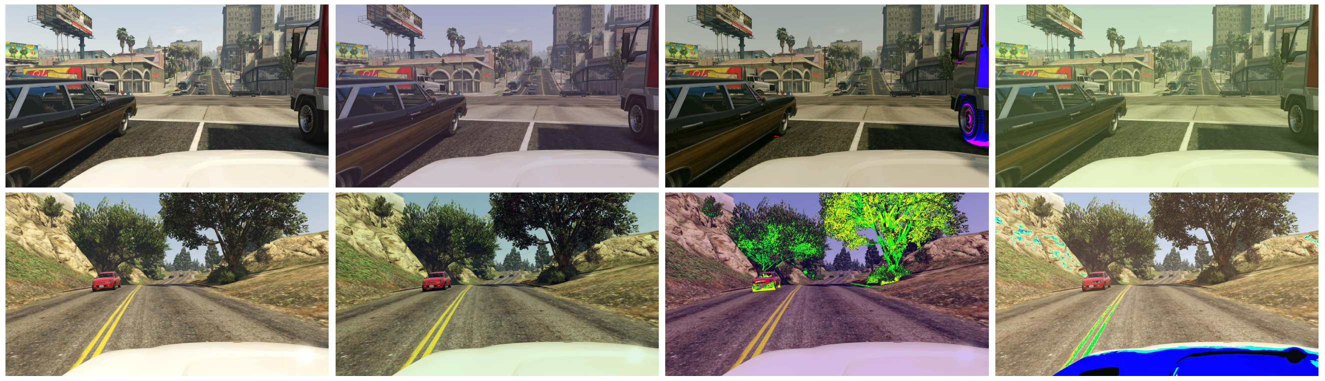


Figure 13: A visualization of CN on image level with style cropping. The two images on the first column are the original images in the GTA5 dataset. We applied CN to these two images with three different style statistics, resulting in images in the last three columns. By calculating style statistics from random cropping, CN can perform a variety of style augmentation. Note that we use CN in both image and feature levels, and we only visualize in the image level here for simplicity.

6.3. Visualization Continued

Visualization setup. Our visualization builds on the technique: understanding deep image representations by inverting them [27]. The goal is to find an image whose feature representation best matches the given one. The search is done automatically by a SGD optimizer with learning rate 1e4, momentum 0.9, and 200 iterations. The learning rate is divided by 10 every 40 iterations. During the optimization, the network is in its evaluation mode with its parameters fixed. In the experiment, we use WideResNet-40-2 and images from CIFAR-10. In visualizing CN, we use the training images and a model trained for 1 epoch. The SN visualization uses test images and a well-trained model. We use different settings for them because CN is for training, while SN works in testing.

More visualization results. Figure 10, extending Figure 5, shows more CN visualizations in deeper network blocks. Figure 11 gives an illustration of 15 corruptions used in robustness evaluation on CIFAR and ImageNet. Moreover, Figure 12 shows some synthetic images from GTA5 and realistic ones from Cityscapes. The visualization of CN applied to the synthetic images is provided in Figure 13.