Adversarial Style Augmentation for Domain Generalized Urban-Scene Segmentation

Zhun Zhong[1]*† **Yuyang Zhao**[2]* **Gim Hee Lee**[2] **Nicu Sebe**[1]
[1] Department of Information Engineering and Computer Science, University of Trento
[2] Department of Computer Science, National University of Singapore

Abstract

In this paper, we consider the problem of domain generalization in semantic segmentation, which aims to learn a robust model using only labeled synthetic (source) data. The model is expected to perform well on unseen real (target) domains. Our study finds that the image style variation can largely influence the model's performance and the style features can be well represented by the channel-wise mean and standard deviation of images. Inspired by this, we propose a novel adversarial style augmentation (**AdvStyle**) approach, which can dynamically generate hard stylized images during training and thus can effectively prevent the model from overfitting on the source domain. Specifically, AdvStyle regards the style feature as a learnable parameter and updates it by adversarial training. The learned adversarial style feature is used to construct an adversarial image for robust model training. AdvStyle is easy to implement and can be readily applied to different models. Experiments on two synthetic-to-real semantic segmentation benchmarks demonstrate that AdvStyle can significantly improve the model performance on unseen real domains and show that we can achieve the state of the art. Moreover, AdvStyle can be employed to domain generalized image classification and produces a clear improvement on the considered datasets.

1 Introduction

Semantic segmentation plays a critical role in autonomous driving, which has achieved impressive improvements with the recent development of deep networks [23, 5, 1, 48]. However, these achievements have been mostly attributed to large-scale labeled segmentation datasets, in which annotating pixel-wise labels is very expensive and time-consuming. Moreover, the model trained on one dataset commonly produces poor performance on unseen datasets captured in different conditions. This degradation phenomenon is mainly caused by domain shifts [7], including differences in weather, season, light, etc. For instance, the segmentation model trained on the dataset captured in sunny London will have low accuracy when deployed on the streets of Zurich in rainy weather.

To address the cross-domain problem, domain adaptation methods [24, 44] are designed to transfer the knowledge of labeled source data to unlabeled target data. However, one of their main drawbacks is that they require the use of target data during training, which cannot always be accessible in practice. Another promising line is domain generalization (DG), which focuses on learning a generalizable model using only the labeled source domain. To reduce the annotating cost and protect data privacy, the existing DG works [7, 43, 49] in semantic segmentation commonly choose to learn the robust model with synthetic data, *e.g.*, GTAV [32]. In this paper, we focus on this synthetic-to-real DG problem for semantic segmentation.

*Equal contribution. † Corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

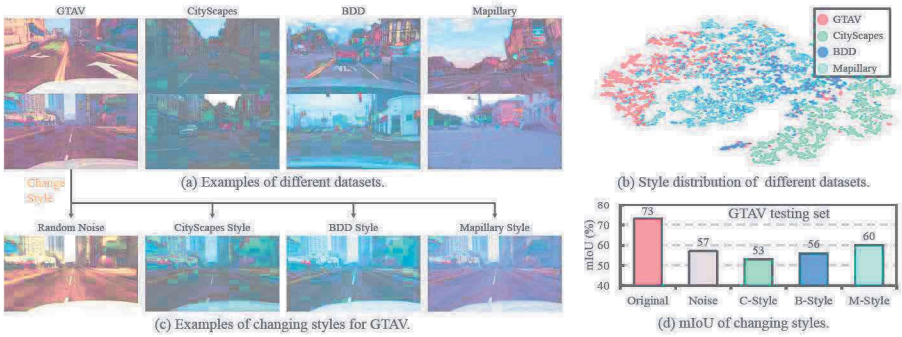

Figure 1: (a) Examples of different datasets. The image styles from different datasets commonly are very different. (b) Style distribution of different datasets. We use image-level mean-variance as the style feature and show that the style distribution gap between different datasets are large. (c) Examples of changing style feature for a GTAV sample, including adding random noise and replacing the style feature with ones of samples from other datasets. (d) mIoU performance of changing styles for GTAV testing set, which is largely reduced after style changing.

To eliminate the impact caused by the large domain gap between synthetic and real data, existing solutions mainly aim at augmenting the synthetic source data with extra real-world samples [43, 15] or learning domain-invariant features with carefully designed modules [28, 7]. The key idea behind them is to avoid the model overfitting on the source domain. This work follows this idea and introduces a new augmenting approach in the perspective of image style for domain generalized semantic segmentation, which is motivated by the observations in Fig. 1. *First*, when visualizing the samples of different datasets in Fig. 1(a) we observe that the image styles are quite different among them, *e.g.*, the road color. The style distribution gap can also be observed in Fig. 1(b). *Second*, the channel-wise mean and standard deviation of an image, which is called style feature in this paper, can well represent the image style. When changing the style feature, the image style of an example varies while the semantic content is well maintained (see Fig. 1(c)). *Third*, changing the style features of testing samples will largely deteriorate the model performance (see Fig. 1(d)). Importantly, when replacing the style features with the ones of CityScapes, which has a large distribution distance to GTAV (see red and green dots in Fig. 1(b)), the performance is lower than other style changing cases. This indicates that the model performance on testing data is highly related to the style distribution gap between training and testing data.

Taking the above observations into consideration, we argue that the image style is an important factor that affects the model performance and propose the adversarial style augmentation (**AdvStyle**) for domain generalized semantic segmentation. Specifically, AdvStyle contains two steps: adversarial style learning and robust model learning. In adversarial style learning, we first decompose the training sample into style feature and normalized image. The style feature is regarded as a learnable parameter, which is used to reconstruct a new training example together with the normalized image. Then, we feed the reconstructed example into the segmentation model and optimize the style feature using the adversarial segmentation loss. The updated style feature is called adversarial style feature and is used to produce hard example in the following step. In robust model training, we first generate an adversarial example by de-normalizing the normalized image with the learned adversarial style feature. The adversarial image and the original image are then used to train a robust model using the segmentation loss. In AdvStyle, the adversarial image is dynamically generated based on the current model. In this way, the model is always encouraged to update with difficult styles and thus will be more robust to style variations in unseen domains. In Fig. 2, we show the comparison between AdvStyle and traditional augmentation methods. Our AdvStyle can significantly improve the model performance on unseen target domains and clearly outperforms the other augmentation methods. To summarize, our contributions are threefold:

- We propose the novel adversarial style augmentation (AdvStyle) for domain generalized semantic segmentation, which can consistently improve the results on unseen real domains. AdvStyle introduces very limited learnable parameters (6-dim feature for each example) and can be easily implemented with different networks and DG methods.

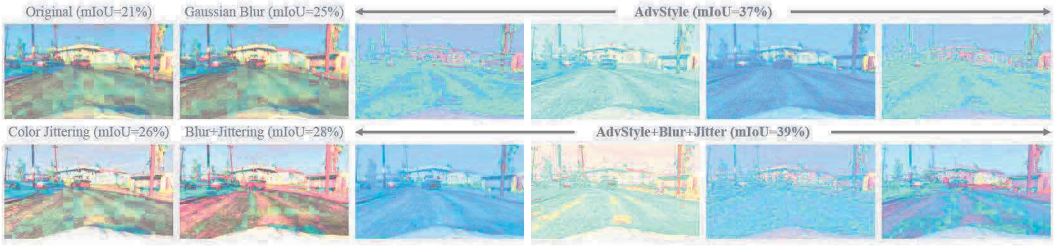

Figure 2: Illustration of different data augmentation methods. We use GTA5 as the source domain and the ResNet-50 as the backbone. The mIoU given in parentheses is evaluated on CityScapes validation set for the model trained with the corresponding augmentation method.

- Experiments on two synthetic-to-real DG benchmarks demonstrate the effectiveness of the proposed AdvStyle and show that we achieve new state-of-the-art DG performance.
- We show that AdvStyle can also be applied to single DG in image classification and can produce state-of-the-art accuracy on two datasets.

2 Related Work

Domain Generalization (DG). To tackle the deficiency of annotated data, DG [43, 15, 7, 36, 49, 4, 46] is introduced to learn a robust model with one or multiple source domains, where the model is expected to perform well on unseen domains. Recent DG works in semantic segmentation mostly use synthetic data as the source domain, which can be automatically generated but have a large distribution gap to real-world datasets. One main stream of DG methods [43, 15] focuses on augmenting training samples with extra real-world data from ImageNet [9]. Learning domain-invariant features [7, 28, 36] is another stream to narrow the domain gap. [28] and [7] leverage instance normalization [37] and whitening transformation to remove the domain-specific information, respectively. [36] exchanges style features of two samples and adjusts style features with the attention mechanism. Different from the above methods, our AdvStyle generates new samples by learning adversarial styles using only the synthetic source data.

Adversarial Training in DG. Adversarial training [13] is initially proposed to learn a robust model that can combat imperceptible perturbations. In recent years, adversarial training is applied to single DG in image classification [39, 31, 30, 11], by regarding adversarial samples as augmented unseen samples. [39] is the first to introduce adversarial samples in DG by max-min iterative training procedure. Later methods form novel domains with the generated adversarial samples and learn a domain-invariant representation by meta-learning [31, 30] or adaptive normalization [11]. Different from them, this paper adopts adversarial training for semantic segmentation and generates adversarial samples in the perspective of image style.

Style Variation. Style features are widely studied in image translation [17, 10]. By varying the style features, the image style can be changed while semantic content will be maintained. Inspired by this, recent works focus on generating data of novel distributions by modifying style features, which are used to train a more robust model. One effective manner is to generate new styles by exchanging [36, 47] or mixing styles [50] between samples. On the other hand, new styles can be generated by learnable modules [40]. Instead, we generate novel styles by adversarial training, which encourages the model to always optimize with hard stylized examples. This work is also closely related to [2], which generates adversarial examples by colorizing. However, it requires a pre-trained colorization model to change the image color, which is much more complex than our AdvStyle. In addition, [2] aims to impair the performance of models by adversarial examples. In contrast, our AdvStyle leverages the adversarial examples to improve the generalization ability of the segmentation model. This work also has a connection with "Learning-to-Simulate" [34]. However, [34] tries to learn good sets of parameters for an image rendering simulator in actual computer vision applications while we attempt to learn a generalized model for semantic segmentation.

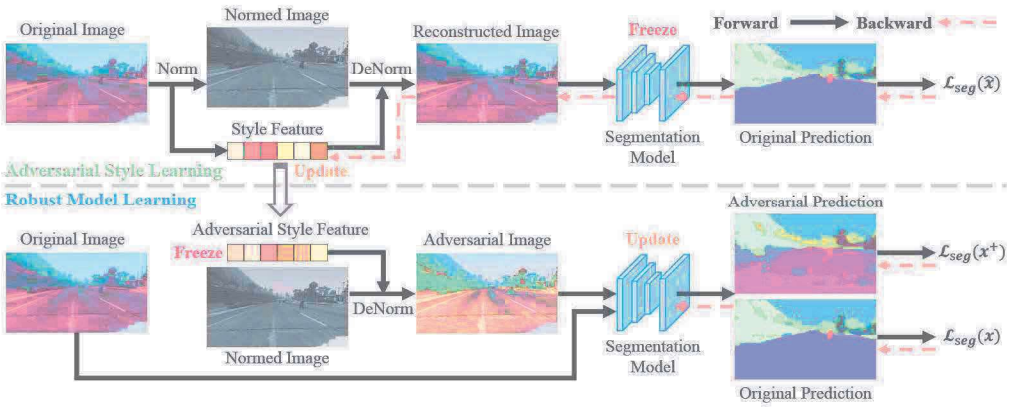

Figure 3: The framework of the proposed adversarial style augmentation.

3 Method

Problem Definition. Synthetic-to-real domain generalization (DG) focuses on training a robust model with *one* labeled synthetic domain $\mathcal{S}$, where the model is expected to perform well on unseen domains $\{\mathcal{T}_1, \mathcal{T}_2, \cdots\}$ of different real-world distributions. As stated by [39], the DG task can be formulated as solving the worst-case problem:

$$\min_{\theta} \sup_{\mathcal{T}: D(\mathcal{S},\mathcal{T}) \leq \rho} \mathbb{E}_{\mathcal{T}} \left[\mathcal{L}_{\text{task}} \left(\theta; \mathcal{T} \right) \right], \tag{1}$$

where θ is the model parameters and $\mathcal{T}$ is the target domains. $\mathcal{L}_{\text{task}}$ denotes the task-specific loss function, which is the pixel-wise cross-entropy loss in this paper. $D(\mathcal{S}, \mathcal{T})$ denotes the distribution distance between the source domain and target domains in semantic space. It is constrained to be lower than ρ for semantic consistency. Inspired by Eq. 1, we propose a novel approach to generate a dynamic source domain $\mathcal{S}^+$, which can help us to reduce the domain shifts between the synthetic domain $\mathcal{S}$ and real-world domains $\mathcal{T}$ during training.

3.1 Overview

In the introduction, we show that the image style is an important factor that influences the DG performance. In addition, the channel-wise mean and standard deviation of an image, which is called style feature, can well represent the image style. Inspired by that, we propose the adversarial style augmentation approach (**AdvStyle**) for domain generalized semantic segmentation. AdvStyle can dynamically generate images with new styles during training and effectively improve the generalization ability of the segmentation model. AdvStyle includes two steps: adversarial style learning and robust model training. In adversarial style learning, we first decompose the image into normalized image and style feature and then update the style feature by adversarial segmentation loss. In robust model training, we compose the adversarial image by the normalized image and learned adversarial style feature. The model is then optimized with both original and adversarial images. These two steps are implemented in each training iteration, enabling us dynamically generate hard stylized samples for the current model. Our overall framework is illustrated in Fig. 3.

3.2 Adversarial Style Learning

Given an training image x at each iteration, we first compute the channel-wise mean μ and standard deviation σ of x and then obtain the normalized image $\bar{x}$ by normalization:

$$\begin{aligned} \mu &= \frac{1}{HW} \sum_{h \in H, w \in W} x_{h,w}, \\ \sigma &= \sqrt{\frac{1}{HW} \sum_{h \in H, w \in W} (x_{h,w} - \mu)^2}, \quad \bar{x} = \frac{x - \mu}{\sigma}, \end{aligned} \tag{2}$$

where H and W denote the spatial size of x.

After that, we initialize the adversarial style feature μ^+ and σ^+ by μ and σ, which are regarded as learnable parameters. Then we reconstruct the image with μ^+, σ^+ and $\bar{x}$ and forward it into the network for loss computation. During the backward, the parameters of the network are fixed and the adversarial style feature is updated by:

$$\begin{aligned} \mu^+ &\leftarrow \mu^+ + \gamma \nabla_{\mu^+} \mathcal{L}_{\text{seg}}\left(\theta; \hat{x}\right), \\ \sigma^+ &\leftarrow \sigma^+ + \gamma \nabla_{\sigma^+} \mathcal{L}_{\text{seg}}\left(\theta; \hat{x}\right), \end{aligned} \tag{3}$$

where γ is the learning rate for adversarial style learning and $\mathcal{L}_{\text{seg}}$ is the cross-entropy loss. $\hat{x} = \bar{x} \cdot \sigma^+ + \mu^+$ is the reconstructed image. Notice that the style feature is optimized by the adversarial gradient of $\mathcal{L}_{\text{seg}}$. Indeed, the adversarial style learning process can be iterated multiple times. In our experiment, we find that one-step adversarial style learning achieves similar results with multi-step ones but is much more efficient. Therefore, we only update the style feature once.

3.3 Robust Model Training

Given the learned adversarial style feature (μ^+ and σ^+), we use it to generate the adversarial sample x^+ with the corresponding normalized image:

$$x^+ = \bar{x} \cdot \sigma^+ + \mu^+. \tag{4}$$

Then the original image x and the generated adversarial image x^+ are forwarded to the model for optimization, which can be formulated by,

$$\min_{\theta} \mathcal{L}_{\text{seg}}(\theta; x) + \mathcal{L}_{\text{seg}}(\theta; x^+). \tag{5}$$

The detailed training procedure and Pytorch-like pseudo-code can be found in the Appendix. G. During testing, we directly input the original samples into the network without implementing AdvStyle.

3.4 Discussion

Adversarial data augmentation have been studied by several works for DG in image classification. Most of them [31, 39] generate pixel-wise perturbations on the training image x, which usually require additional constraint loss to guarantee the semantic consistency in Eq. 1. In addition, these works focus on the image classification task where the recognition result is mostly related to global feature. However, in semantic segmentation, the model needs to produce the per-pixel predictions so that it is more difficult to ensure the pixel-wise semantic consistency during adversarial learning. Instead, our AdvStyle varies the style feature of the image, which will maintain the semantic content of most pixels and thus can well guarantee the pixel-wise semantic consistency for semantic segmentation. We conduct experiments in Table 3, which demonstrate the superiority of the proposed AdvStyle over pixel-wise adversarial learning.

4 Experiments

4.1 Experimental Setup

Datasets and Implementation Details. For the synthetic-to-real domain generalization (DG), we use one of the synthetic datasets (GTAV [32] or SYNTHIA [33]) as the source domain and evaluate the model performance on three real-world datasets (CityScapes [8], BDD-100K [42], and Mapillary [27]). The details of the datasets can be found in the Appendix. A. Following [7], we use DeepLabV3+ [6] as the segmentation model. The segmentation model is constructed by three backbones, including MobileNetV2 [35], ResNet-50 [14] and ResNet-101. We adopt SGD optimizer with an initial learning rate 0.01, momentum 0.9 and weight decay 5×10^{-4} to optimize the model. The polynomial decay [22] with the power of 0.9 is used as the learning rate scheduler. The learning rate of AdvStyle γ is set to 3. All models are trained for 40K iterations with a batch size of 16. Four widely used data augmentation techniques are used during training, including color jittering, Gaussian blur, random cropping and random flipping. The input image is randomly cropped to 768×768 for training. The original image size is used for testing.

Table 1: Evaluation of the proposed AdvStyle on different methods (Baseline, IBN-Net [28] and ISW[7]) and backbones (MobileNetV2 [35], ResNet-50 [14], and ResNet-101). All models are trained on the GTAV training set and tested on CityScapes (C), BDD-100K (B), and Mapillary (M) validation sets.

Methods	MobileNetV2				ResNet-50				ResNet-101			
GTAV→	C	B	M	Mean	C	B	M	Mean	C	B	M	Mean
Baseline	25.92	25.73	26.45	26.03	28.95	25.14	28.18	27.42	32.97	30.77	30.68	31.47
+AdvStyle	**31.81**	**33.01**	**31.50**	**32.11**	**39.62**	**35.54**	**37.00**	**37.39**	**39.52**	**36.39**	**36.10**	**37.34**
IBN-Net	30.14	27.66	27.07	28.29	33.85	32.30	37.75	34.63	37.37	34.21	36.81	36.13
+AdvStyle	**32.45**	**31.55**	**33.09**	**32.36**	**39.32**	**36.42**	**40.82**	**38.85**	**44.04**	**39.96**	**42.67**	**42.22**
ISW	30.86	30.05	30.67	30.53	36.58	35.20	40.33	37.37	37.20	33.36	35.57	35.38
+AdvStyle	**33.23**	**31.84**	**32.00**	**32.36**	**39.60**	**38.59**	**41.89**	**40.03**	**43.44**	**40.32**	**41.96**	**41.91**

Table 2: Results of using SYNTHIA as the source domain. The backbone is ResNet-101.

Methods (SYNTHIA→)	CityScapes	BDD	Mapillary	Mean
Baseline	34.94	21.96	27.94	28.28
Baseline+AdvStyle	**37.59**	**27.45**	**31.76**	**32.27**
IBN-Net	35.83	23.62	28.88	29.44
IBN-Net+AdvStyle	**38.72**	**28.55**	**33.59**	**33.62**
ISW	35.27	23.54	26.72	28.51
ISW+AdvStyle	**39.74**	**28.33**	**32.87**	**33.65**

Evaluation Metric. Following [7], the model obtained by the last training iteration is used to evaluate the mIoU performance on the three real-world validation sets. For each method, we report the result averaged on 3 runs. When using GTAV as the source domain, we use the 19 shared semantic categories for training and evaluation. When using SYNTHIA as the source domain, we use 16 shared categories for training and evaluation, *i.e.*, ignoring the train, truck, and terrain categories.

4.2 Evaluation

Effectiveness of AdvStyle on Different Models. Our AdvStyle is a model-agnostic method, which can be directly applied to different models without modifying the models. To verify the effectiveness of AdvStyle, we apply it to models with different backbones and normalization modules. The backbones include MobileNetV2, ResNet-50 and ResNet-101. The normalization modules include vanilla batch norm (baseline), instance-batch norm (IBN-Net [28]), and instance selective whitening (ISW [7]). In Table 1, we show the results of using GTAV as the source domain. We can make the following conclusions. First, injecting instance-batch norm (IBN-Net) and instance selective whitening (ISW) modules can consistently improve the performance of the baseline. Second, the proposed AdvStyle can significantly enhance the generalization performance of the baseline model for all backbones. Specifically, the average mIoU is increased from 26.03%, 27.42% and 31.47% to 32.11%, 37.39% and 37.34% for MobileNetV2, ResNet-50 and ResNet-101, respectively. In addition, the baseline with AdvStyle produces higher average mIoU than IBN-Net and ISW for all backbones. Third, when adding AdvStyle, the results of IBN-Net and ISW can be further improved for all settings. For example, when using ResNet-101 as the backbone, AdvStyle improves the average mIoU of IBN-Net and ISW by 6.09% and 6.53%, respectively. In Table 2, we show the results of using SYNTHIA as the source domain and also observe clear improvements for AdvStyle. These results verify the prominent advantage of our AdvStyle on different models.

Comparison of Different Augmentation Techniques. We investigate the impact of different augmentation methods, including color jittering, Gaussian blur, AdvPixel [39] and the proposed AdvStyle. AdvPixel is a state-of-the-art method for domain generalized image classification. The main difference between AdvPixel and AdvStyle is that AdvPixel learns pixel-wise adversarial

Table 3: Comparison of different augmentations. Source: GTAV; Backbone: ResNet-50. CJ: Color Jittering, GB: Gaussian Blur, AP: AdvPixel, Ours: AdvStyle.

CJ	GB	AP	Ours	CityScapes	BDD	Mapillary	Mean
-	-	-	-	21.64	22.85	24.22	22.91
✓	-	-	-	26.36	23.82	26.33	25.50
-	✓	-	-	25.77	24.05	26.71	25.51
-	-	✓	-	23.34	28.42	30.64	27.46
-	-	-	✓	**37.51**	**33.74**	**34.73**	**35.32**
✓	✓	-	-	28.95	25.14	28.18	27.42
✓	✓	✓	-	35.42	33.28	33.23	33.97
✓	✓	-	✓	**39.62**	**35.54**	**37.00**	**37.39**

Table 4: Comparison of different style-aware methods. Source: GTAV; Backbone: ResNet-50.

Methods	CityScapes	BDD	Mapillary	Mean
Baseline	28.95	25.14	28.18	27.42
RandStyle	33.40	34.14	31.67	33.07
MixStyle	35.53	32.41	35.87	34.60
CrossStyle	37.26	32.40	34.09	34.58
AdvStyle	**39.62**	**35.54**	**37.00**	**37.39**

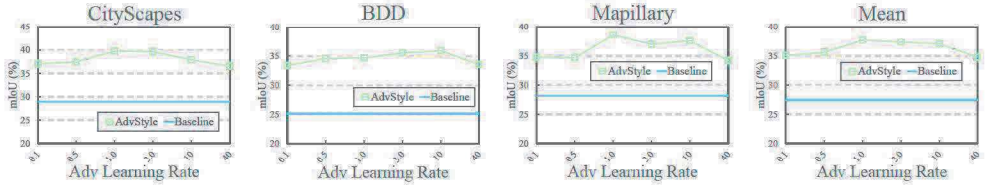

Figure 4: Parameter analysis on the adversarial learning rate.

example while AdvStyle learns style-wise adversarial example. We reproduce AdvPixel in our setting and select the adversarial learning rate (=10) that achieves the best performance. The random cropping and random flipping are used in default.

Results in Table 3 show that all four augmentation methods can improve the generalization performance. Importantly, our AdvStyle produces significant improvement compared to other three methods. Specifically, when using AdvStyle, the average mIoU is increased from 22.91% to 35.32%. This improvement is about 8% higher than the other 3 methods. Moreover, AdvStyle is well complementary to color jittering and Gaussian blur. When combining these three methods, the mIoU is further improved in all target domains. Compared to AdvPixel, our AdvStyle achieves clearly higher performance, no matter using color jittering and Gaussian blur. This demonstrates the advantage of learning adversarial style in domain generalized semantic segmentation.

Comparison of Different Style-Aware Methods. In Table 4, we compare AdvStyle with three style-aware augmentation methods, including MixStyle [50], CrossStyle [36] and RandStyle. MixStyle mixes the styles of two samples with a convex weight while CrossStyle directly swaps the styles of two samples. RandStyle can be regarded a reduction of our AdvStyle, which randomly adds Gaussian noise into the style feature. All style-aware methods are implemented on the image-level for fair comparison. We can find that (1) all style-aware methods can consistently improve the performance on all target domains and (2) AdvStyle achieves the best results. The first finding verifies the effectiveness of augmenting image styles and the second finding shows the benefit of learning adversarial styles over other style-aware methods for domain generalized semantic segmentation.

Parameter Analysis on the Adversarial Learning Rate. The proposed AdvStyle has one important hyperparameter, *i.e.*, the adversarial learning rate γ. To study the impact of γ, we vary it in the range of $[0.1, 40]$. Results in Fig. 4 show that AdvStyle can significantly improve the performance on all target domains even with a small value of γ. The best results are achieved when γ is between 1 and 10. Assigning a too large value to γ (*e.g.*, 40) may produce unrealistic styles and thus hampers the model training.

4.3 Comparison with State-of-The-Art Methods

In Table 5, we compare our method with state-of-the-art DG methods in semantic segmentation, including IBN-Net [28], SW [29], IterNorm [16], ISW [7], DPRC [43] and FSDR [15]. The source

Table 5: Comparison with state-of-the-art domain generalization methods. All models use the GTAV as the source domain. For each backbone, models with the same ID are implemented with the same baseline. Models of "ID=I, II and IV" use the whole set (24,966) for training while models of "ID=III" only use the training set (12,403). The absolute gain of each model is calculated over the corresponding baseline. § denotes extra using the ImageNet images. * indicates using the best trained checkpoints for evaluating each target domain.

Net	ID	Methods (GTAV)	CityScapes		BDD		Mapillary		Mean	
ResNet-50	I	Baseline	22.20	-	N/A		N/A		N/A	
	I	IBN-Net	29.60	7.40↑						
	II	Baseline*	32.45	-	26.73	-	25.66	-	28.28	-
	II	DRPC§*	37.42	4.97↑	32.14	5.41↑	34.12	8.46↑	34.56	6.28↑
	III	Baseline	28.95	-	25.14	-	28.18	-	27.42	-
	III	**Baseline+AdvStyle**	**39.62**	10.67↑	35.54	10.4↑	37.00	8.82↑	37.39	9.97↑
	III	SW	29.91	0.96↑	27.48	2.34↑	29.71	1.53↑	29.03	1.61↑
	III	IterNorm	31.81	2.86↑	32.70	7.56↑	33.88	5.7↑	32.79	5.37↑
	III	IBN-Net	33.85	4.90↑	32.30	7.16↑	37.75	9.57↑	34.63	7.21↑
	III	**IBN-Net+AdvStyle**	39.32	10.37↑	36.42	11.28↑	40.82	12.64↑	38.85	11.43↑
	III	ISW	36.58	7.63↑	35.20	10.06↑	40.33	12.15↑	37.37	9.95↑
	III	**ISW+AdvStyle**	39.60	10.65↑	**38.59**	13.45↑	**41.89**	13.71↑	**40.03**	12.61↑
ResNet-101	I	Baseline*	33.4	-	27.3	-	27.9	-	29.53	-
	I	IBN-Net*	40.3	6.9↑	35.6	8.3↑	35.9	8.0↑	37.26	7.73↑
	I	FSDR§*	44.8	11.4↑	41.2	13.9↑	43.4	15.5↑	43.13	13.6↑
	II	Baseline*	33.56	-	27.76	-	28.33	-	29.88	-
	II	DRPC§*	42.53	8.97↑	38.72	10.96↑	38.05	9.72↑	39.76	9.88↑
	III	Baseline	32.97	-	30.77	-	30.68	-	31.47	-
	III	**Baseline+AdvStyle**	39.52	6.55↑	36.39	5.62↑	36.10	5.42↑	37.34	5.87↑
	III	IBN-Net	37.37	4.40↑	34.21	3.44↑	36.81	6.13↑	36.13	4.66↑
	III	**IBN-Net+AdvStyle**	44.04	11.07↑	39.96	9.19↑	42.67	11.99↑	42.22	10.75↑
	III	ISW	37.20	4.23↑	33.36	2.59↑	35.57	4.89↑	35.38	3.91↑
	III	**ISW+AdvStyle**	43.44	10.47↑	40.32	9.55↑	41.96	11.28↑	41.91	10.44↑
	III	**ISW+AdvStyle***	**45.62**	12.65↑	**41.71**	10.97↑	**46.69**	16.01↑	**44.67**	13.20↑
	IV	ISW	37.51	-	33.54	-	36.12	-	35.72	-
	IV	**ISW+AdvStyle**	**44.51**	-	**39.27**	-	**43.48**	-	**42.42**	-

domain is GTAV and the backbones are ResNet-50 and ResNet-101. Note that, since different methods use different segmentation networks (*e.g.*, DeepLabV2 [5], DeepLabV3+ [6] and FCN [23]), different training sets (*e.g.*, the whole GTAV and the training set of GTAV), different training strategies (*e.g.*, learning rate and optimizer) , different auxiliary data (*e.g.*, ImageNet samples) and different evaluation manners (*e.g.*, the best model and the last model), it is hard to compare them in an absolutely fair way. We show the results of each method as well as the absolute gain against the corresponding baseline.

From Table 5, we can make the following conclusions. **First**, when using the same baseline model, adding AdvStyle can produce better results than IBN-Net, SW, IterNorm and ISW. Moreover, when applying AdvStyle to IBN-Net or ISW, we achieve new state-of-the-art performance for both ResNet-50 and ResNet-101. **Second**, when compared across baselines, "Baseline+AdvStyle" achieves the state-of-the-art mIoU for ResNet-50. On the other hand, when using ResNet-101 as the backbone, "IBN-Net+AdvStyle" produces higher results than DRPC and comparable results with FSDR. Importantly, both FSDR and DRPC use extra ImageNet images and select the best training checkpoints for each target domain. Instead, "IBN-Net+AdvStyle" only utilizes the source data and uses the last training checkpoint to evaluate all target domains. **Third**, when using the best checkpoint for evaluation, we ("ISW+AdvStyle") produce better performance than DRPC and FSDR, leading to the new state-of-the-art results under the "best checkpoint setting". *Even so, we argue that it is more*

Table 6: Results of using GTAV and SYNTHIA as the source domains. The backbone is ResNet-50.

Methods (GTAV+SYNTHIA→)	CityScapes	BDD	Mapillary	Mean
Baseline [7]	35.46	25.09	31.94	30.83
IBN-Net [28]	35.55	32.18	38.09	35.27
ISW [7]	37.69	34.09	38.49	36.75
ISW+AdvStyle	**39.29**	**39.26**	**41.14**	**39.90**

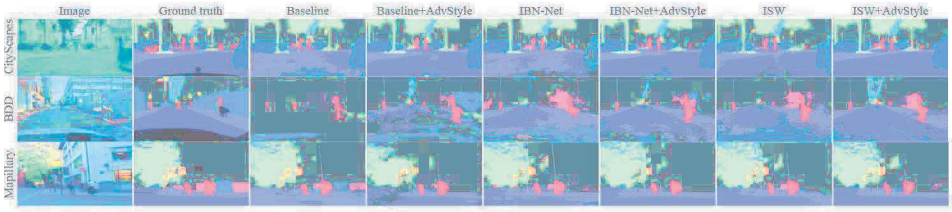

Figure 5: Qualitative comparison of segmentation results. Source: GTAV; Backbone: ResNet-50.

reasonable to use the last checkpoint for evaluating all target domains. This is because we can not always have the right labeled validation sets to select the best model for unseen domains in practice.

To make more fair comparisons with the state-of-the-art methods, we also use the whole set of GTAV for training ISW and "ISW+AdvStyle". The results with ResNet-101 are reported in the last two rows of Table 5. We can find that using the whole set of GTAV can produce higher results on the CityScapes and the Mapillary datasets.

4.4 Performance under Multi-Source Setting

In Table 6, we evaluate the models under the multi-source domain generalization setting, where both GTAV and SYNTHIA are used as the source domains. The compared methods include baseline [7], IBN-Net [28], ISW [7], and our AdvStyle. Clearly, AdvStyle consistently improves the results of ISW, further verifying the effectiveness of the proposed AdvStyle.

4.5 Visualization

Qualitative Comparison of Segmentation Results. In Fig. 5, we compare the segmentation results for different methods on target domains. It is clear that, the proposed AdvStyle can consistently improve the semgentation results for baseline, IBN and ISW models, especially for the easily-confused classes, *e.g.*, road *vs* sidewalk and building *vs* sky.

t-SNE of Styles. In Fig. 6, we visualize the style features generated by AdvStyle during the training phase for the GTAV training set, where ResNet-50 is used as the backbone. We can find that AdvStyle can continuously generate new style features that are different from the original distribution. The new style features have the chance to be located at the distributions of other datasets during the training process. Moreover, AdvStyle will also generate styles that are out of the distributions of the four datasets (G, C, B, M) and may appear in other unseen domains. The visualization results further demonstrate that AdvStyle can encourage the model to meet more diverse and unseen styles during training, leading to a more robust model.

4.6 Evaluation on Image Classification Task

To verify the versatility of the proposed AdvStyle, we evaluate it on single DG in image classification. Experiments are conducted on two popular DG datasets, *i.e.*, Digits and PACS. The details of the datasets and implementation can be found in in the Appendix. B.

Results on Digits. In Table 7, we compare with the baseline (ERM [38]) and 6 state-of-the-art methods, including CCSA [25], d-SNE [41], JiGen [3], ADA [39], M-ADA [31] and ME-ADA [45]. For AdvStyle, we implement it with ERM and ME-ADA. It is clear that, our AdvStyle can significantly

Figure 6: t-SNE visualization of adversarial style features during training.

Table 7: Accuracy of single domain generalization on Digits. MNIST is used as the training set, and the results on different testing domains are reported in different columns.

Method	SVHN	MNIST-M	SYN	USPS	Avg.
ERM	27.8	52.7	39.7	76.9	49.3
CCSA	25.9	49.3	37.3	83.7	49.1
d-SNE	26.2	51.0	37.8	**93.2**	52.1
JiGen	33.8	57.8	43.8	77.2	53.1
ADA	35.5	60.4	45.3	77.3	54.6
M-ADA	42.6	67.9	49.0	78.5	59.5
ME-ADA	42.6	63.3	50.4	81.0	59.3
ERM+AdvStyle	50.4	73.4	58.7	81.6	66.0
ME-ADA+AdvStyle	**55.5**	**74.1**	**59.3**	80.1	**67.3**

Table 8: Accuracy of single domain generalization on PACS. One domain (name in column) is used as the training (source) data and the other domains are used as the testing (target) data.

Method	Art.	Car.	Ske.	Pho.	Avg.
ERM	67.4	74.4	51.4	42.6	58.9
JiGen	69.1	74.6	52.4	41.5	59.4
RSC	68.8	74.5	53.6	41.9	59.7
L2D	74.3	77.5	54.4	45.9	63.0
ERM+AdvStyle	75.8	76.6	58.1	51.1	65.4
RSC+AdvStyle	75.1	78.0	**58.9**	55.5	66.8
L2D+AdvStyle	**80.6**	**78.4**	58.3	**59.7**	**69.3**

improve the accuracy of ERM on all target domains. When applying AdvStyle to the state-of-the-art method (ME-ADA), the performance is improved by a large margin, *i.e.*, 8.0% in average accuracy.

Results on PACS. We compare with the baseline (ERM [38]) and three state-of-the-art DG methods, including JiGen [3], RSC [18] and L2D [40]. We reproduce JiGen, RSC and L2D with their official source codes. All methods use the same baseline (ERM). Results in Table 8 show that JiGen and RSC produce limited improvements. Instead, our AdvStyle can significantly increase the accuracy on all domains for both ERM and RSC. Compared to the recent published work (L2D), our method (ERM+AdvStyle) outperforms it by 2.4% in average accuracy. In addition, AdvStyle can also be applied to L2D and yields an improvement of 6.3% in average accuracy.

The results on Digits and PACS validate that AdvStyle can also be effectively applied to single domain generalized image classification and can achieve state-of-the-art accuracy.

5 Conclusion

In this paper, we propose a novel augmentation approach, called adversarial style augmentation (**AdvStyle**), for domain generalization (DG) in semantic segmentation. AdvStyle dynamically generates hard stylized images by learning adversarial image-level style feature, which can encourage the model learning with more diverse samples. With AdvStyle, the model can refrain from the problem of overfitting on the source domain and thus can be more robust to the style variations of unseen domains. AdvStyle is easy to implement and can be directly integrated with different models without modifying the network structures and learning strategies. Experiments on two synthetic-to-real settings show that AdvStyle can largely improve the generalization performance and achieve state-of-the-art performance. In addition, AdvStyle can be employed to single DG in image classification and obtain significant improvement.

Acknowledgement

This work is supported by the EU H2020 project AI4Media (No. 951911) and the PRIN project PREVUE (Prot. 2017N2RK7K).

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, 2017.

[2] Anand Bhattad, Min Jin Chong, Kaizhao Liang, Bo Li, and DA Forsyth. Unrestricted adversarial examples via semantic manipulation. In *ICLR*, 2020.

[3] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019.

[4] Chen Chen, Zeju Li, Cheng Ouyang, Matt Sinclair, Wenjia Bai, and Daniel Rueckert. Maxstyle: Adversarial style composition for robust medical image segmentation. In *MICCAI*, 2022.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2018.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[7] Sungha Choi, Sanghun Jung, Huiwon Yun, Joanne T Kim, Seungryong Kim, and Jaegul Choo. Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening. In *CVPR*, 2021.

[8] 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 *CVPR*, 2016.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017.

[11] Xinjie Fan, Qifei Wang, Junjie Ke, Feng Yang, Boqing Gong, and Mingyuan Zhou. Adversarially adaptive normalization for single domain generalization. In *CVPR*, 2021.

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[13] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

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

[15] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Fsdr: Frequency space domain randomization for domain generalization. In *CVPR*, 2021.

[16] Lei Huang, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Iterative normalization: Beyond standardization towards efficient whitening. In *CVPR*, 2019.

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

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

[19] Jonathan J. Hull. A database for handwritten text recognition research. *TPAMI*, 1994.

[20] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989.

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

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

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[24] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *CVPR*, 2019.

[25] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017.

[26] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop*, 2011.

[27] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

[28] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *ECCV*, 2018.

[29] Xingang Pan, Xiaohang Zhan, Jianping Shi, Xiaoou Tang, and Ping Luo. Switchable whitening for deep representation learning. In *ICCV*, 2019.

[30] Fengchun Qiao and Xi Peng. Uncertainty-guided model generalization to unseen domains. In *CVPR*, 2021.

[31] Fengchun Qiao, Long Zhao, and Xi Peng. Learning to learn single domain generalization. In *CVPR*, 2020.

[32] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016.

[33] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[34] Nataniel Ruiz, Samuel Schulter, and Manmohan Chandraker. Learning to simulate. In *ICLR*, 2019.

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[36] Zhiqiang Tang, Yunhe Gao, Yi Zhu, Zhi Zhang, Mu Li, and Dimitris Metaxas. Selfnorm and crossnorm for out-of-distribution robustness. *ICCV*, 2021.

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

[38] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[39] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, 2018.

[40] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *ICCV*, 2021.

[41] Xiang Xu, Xiong Zhou, Ragav Venkatesan, Gurumurthy Swaminathan, and Orchid Majumder. d-sne: Domain adaptation using stochastic neighborhood embedding. In *CVPR*, 2019.

[42] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020.

[43] 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 *ICCV*, 2019.

[44] Pan Zhang, Bo Zhang, Ting Zhang, Dong Chen, Yong Wang, and Fang Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In *CVPR*, 2021.

[45] Long Zhao, Ting Liu, Xi Peng, and Dimitris Metaxas. Maximum-entropy adversarial data augmentation for improved generalization and robustness. *NeurIPS*, 2020.

[46] Yuyang Zhao, Zhun Zhong, Fengxiang Yang, Zhiming Luo, Yaojin Lin, Shaozi Li, and Sebe Nicu. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *CVPR*, 2021.

[47] Yuyang Zhao, Zhun Zhong, Zhiming Luo, Gim Hee Lee, and Nicu Sebe. Source-free open compound domain adaptation in semantic segmentation. *TCSVT*, 2022.

[48] Yuyang Zhao, Zhun Zhong, Nicu Sebe, and Gim Hee Lee. Novel class discovery in semantic segmentation. In *CVPR*, 2022.

[49] Yuyang Zhao, Zhun Zhong, Na Zhao, Nicu Sebe, and Gim Hee Lee. Style-hallucinated dual consistency learning for domain generalized semantic segmentation. In *ECCV*, 2022.

[50] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section.I in the Appendix.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Section. G in the Appendix.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Details of Datasets in Domain Generalized Semantic Segmentation

For the synthetic-to-real domain generalization (DG), we use one of the synthetic datasets (GTAV [32] or SYNTHIA [33]) as the source domain and evaluate the model performance on three real-world datasets (CityScapes [8], BDD-100K [42], and Mapillary [27]).

Synthetic datasets. GTAV [32] contains 24,966 images with the size of 1914×1052. It is splited into 12,403, 6,382, and 6,181 images for training, validating, and testing. SYNTHIA [33] contains 9,400 images of 960×720, where 6,580 images are used for training.

Real-world datasets. We use the validation sets of the three real-world datasets for evaluation. CityScapes [8] contains 500 validation images of 2048×1024, collected primarily in Germany. BDD-100K [42] and Mapillary [27] contain 1,000 validation images of 1280×720 and 2,000 validation images of 1920×1080, respectively.

B Details of Single Domain Generalization in Image Classification

Digits includes five domains (MNIST [20], SVHN [26], MNIST-M [12], SYN [12], and USPS [19]) of 10 classes. We use MNIST as the source domain and evaluate the model performance on the other 4 domains. Following ADA [39], we use the ConvNet architecture [20] as the model and use Adam optimizer with learning rate 10^{-4} for optimization. The overall training iteration is set to 10,000 with a batch size of 32. We set the learning rate of AdvStyle to 20,000[2].

PACS [21] contains four domains (Artpaint, Cartoon, Sketch, and Photo) of 7 classes. For evaluation, we select one of them as the source domain and the other domains as the target domains. Following RSC [18], we use the ResNet18 [14] pretrained on ImageNet [9] as the backbone and add a fully-connected layer as the classification head. We train the model by SGD optimizer. The learning rate is initially set to 0.004 and divided by 10 after 24 epochs. The model is trained for 30 epochs in total with a batch size of 128. The learning rate of AdvStyle is set to 3.

Baseline. The baseline model is the vanilla empirical risk minimization (ERM) [38], which directly uses the source domain to train the model with classification loss.

C Position of AdvStyle

We inject AdvStyle at different positions (0-4) to verify the effectiveness of image level augmentation. 0-th indicates the image level. 1st-4th indicate the outputs of 1st-4th layer of ResNet, respectively. Results are shown in the Table below. We can find that injecting AdvStyle at 0th-2nd positions clearly improves the performance and the best result is achieved by applying at 0-th position. Moreover, applying AdvStyle at a deep layer (*e.g.*, 3rd or 4th) fails to improve or even hurts the performance, since more semantic content will be captured instead of styles as the layer deepens.

Table 9: Impact of injecting AdvStyle at different positions.

Position	N/A	0	1	2	3	4
Mean	27.42	**37.39**	34.76	31.15	27.44	26.92

D Comparison of Adversarial Augmentations

AdvPixel [39] is a state-of-the-art method for domain generalized image classification. The main difference between AdvPixel and AdvStyle is that AdvPixel learns pixel-wise adversarial example while AdvStyle learns style-wise adversarial example. The model needs to produce the per-pixel predictions in semantic segmentation. In such a context, AdvPixel may distort the semantic content of original pixels during the pixel-wise adversarial learning. Instead, AdvStyle varies the style feature of the image while retaining the semantic content of most pixels. Therefore, AdvStyle can well guarantee the pixel-wise semantic consistency, making it more suitable for augmenting samples of segmentation. As shown in Table 10, both AdvStyle and AdvPixel improve the performance, while AdvStyle outperforms AdvPixel by 3.42% in mean mIoU. More interestingly, AdvPixel can serve to enhance AdvStyle. We randomly select one adversarial augmentation from AdvPixel and AdvStyle at each iteration. The performance yields an improvement of 0.81% in mean mIoU. The above results verify the effectiveness of adversarial augmentations and the superiority of AdvStyle.

Table 10: Comparison of adversarial augmentations. Source: GTAV; Backbone: ResNet-50. CJ: Color Jittering, GB: Gaussian Blur, AP: AdvPixel, Ours: AdvStyle.

CJ	GB	AP	Ours	CityScapes	BDD	Mapillary	Mean
✓	✓	-	-	28.95	25.14	28.18	27.42
✓	✓	✓	-	35.42	33.28	33.23	33.97
✓	✓	-	✓	39.62	35.54	**37.00**	37.39
✓	✓	✓	✓	**40.65**	**37.16**	36.77	**38.20**

[2]Due to the absent of batch normalization layer, the gradient is very small on the style feature. Therefore, we set a large learning rate for AdvStyle.

Table 11: Results of AdvStyle variants. The backbone is ResNet-50.

Methods (GTAV→)	CityScapes	BDD	Mapillary	Mean
Baseline [7]	28.95	25.14	28.18	27.42
AdvStyle-LAB	37.09	32.89	37.13	35.70
AdvStyle-Patches	39.50	**36.37**	**37.42**	**37.76**
AdvStyle	**39.62**	35.54	37.00	37.39

E Variants of AdvStyle

In this section, we investigate two variants of AdvStyle, which can further demonstrate the versatility of AdvStyle. Also, we hope to provide some inspirations for future work.

AdvStyle in local patches. AdvStyle can be applied to not only the whole image but also the local patches. Specifically, we split each image into 4 patches evenly (top left, top right, bottom left, and bottom right), and regard the channel-wise mean and standard deviation of each patch as learnable parameters (four 6-dim features). Then the model is trained in the same way as AdvStyle. As shown in the Table 11, AdvStyle-Patches can further improve the performance on BDD and Mapillary. However, the mean improvement over all domains is marginal.

AdvStyle in LAB color space. AdvStyle is applied to RGB space in this paper, but it can also be applied to other color space, *e.g.*, LAB color space. To verify this, we first convert the RGB-sample to the counterpart LAB-sample and obtain the learnable mean and standard deviation. Then, we reconvert the LAB-sample to RGB-sample for adversarial learning and model optimization. This manner enables us to implement AdvStyle in the LAB space as well as to use the ImageNet-pretrained parameters. As shown in the Table 11, LAB-based AdvStyle (AdvStyle-LAB) also significantly improves the performance on unseen domains but achieves lower results than RGB-based AdvStyle on two of the three benchmarks. On the other hand, converting between RGB and LAB will increase the training time due to the extra computation costs.

F Quantitative Understanding of AdvStyle

To demonstrate the effectiveness of AdvStyle in narrowing the domain shift, we provide the quantitative analysis on the distribution of different datasets. Specifically, we computed the histograms of pixel values of four datasets (GTAV [32], CityScapes [8], BDD-100K [42], Mapillary [27]) and the AdvStyle-augmented dataset of GTAV which is generated by 4 epochs. The bin size is set to 8. For each dataset, the histograms of RGB channels are normalized by L1-norm and re-scaled ($\times$) by #bins, and then are concatenated as the histogram feature. We estimate the distribution distance between two datasets by computing the KL-distance between their histogram features. Results are reported in Table 12. We can observe that the AdvStyle-augmented dataset has a smaller distance to real datasets, verifying that AdvStyle can narrow the gap between synthetic and real data.

Table 12: Comparison of KL-distance between different datasets.

Source	CityScapes ↓	BDD ↓	Mapillary ↓	Mean ↓
GTAV	0.5867	0.3421	0.3211	0.4166
Adv-GTAV	**0.5587**	**0.3217**	**0.3058**	**0.3954**

G Algorithm and Pytorch-Like Pseudo-Code

The training procedure and Pytorch-like pseudo-code are shown in Alg. 1 and Fig. 7, respectively.

H More Visualizations

Segmentation Results. In Fig. 8, Fig. 9, and Fig. 10, we provide more segmentation results for the baseline and "baseline+AdvStyle".

Examples of AdvStyle. In Fig. 11, we illustrate more examples generated by AdvStyle.

Algorithm 1: The training procedure of AdvStyle.

Inputs: labeled source domain $\mathcal{S}$, segmentation model $\mathcal{F}$ parameterized by θ, batch size N_b, total training iterations max_iter, adversarial learning rate γ, and model learning rate α.
Outputs: Optimized model $\mathcal{F}$ parameterized with θ.

for i in max_iter **do**
  Sample mini-batch $\mathcal{X}$ with N_b images;
  // Stage 1: Adversarial Style Learning.
  Compute channel-wise mean μ, standard deviation σ and normalized images $\bar{\mathcal{X}}$ with Eq. 2;
  Initialize adversarial style feature: $\mu^+ \leftarrow \mu$, $\sigma^+ \leftarrow \sigma$;
  Compute adversarial segmentation loss $-\mathcal{L}_{\text{seg}}$;
  Optimize μ^+ and σ^+ with Eq. 3;
  // Stage 2: Robust Model Training.
  Generate adversarial images $\mathcal{X}^+$ with $\bar{\mathcal{X}}$, μ^+ and σ^+ by Eq. 4;
  Compute the overall training loss $\mathcal{L}_{\text{seg}}(\theta; \mathcal{X}) + \mathcal{L}_{\text{seg}}(\theta; \mathcal{X}^+)$ by Eq. 5;
  Optimize the segmentation model $\mathcal{F}$: $\theta \leftarrow \theta - \alpha \nabla_\theta \left(\mathcal{L}_{\text{seg}}(\theta; \mathcal{X}) + \mathcal{L}_{\text{seg}}(\theta; \mathcal{X}^+)\right)$;
end for
Return $\mathcal{F}$ parameterized with θ.

I Limitations

The main limitation of AdvStyle lies in the increase of training time. The computational cost of AdvStyle is almost double of that of the baseline since one more forward-backward process is required to generate style-adversarial examples. Another limitation is that despite generating hard examples, AdvStyle cannot address severe environmental change in practice, *e.g.*, rainy and snowy weather, since such conditions cannot be represent purely by style features. Those conditions, *e.g.*, rain, snow and fog, can be added to source samples and adversarial-augmented samples manually to alleviate the problem.

```
import torch

def AdvStyle(input, gt, net, optim, adv_lr):
    """
    Args:
        input: source images
        gt: ground-truth labels
        net: segmentation network
        optim: optimizer of net
        adv_lr: learning rate of AdvStyle
    """
    ### Adversarial Style Learning

    # Get style feature and normalized image
    B = input.size(0)
    mu = input.mean(dim=[2, 3], keepdim=True)
    var = input.var(dim=[2, 3], keepdim=True)
    sig = (var + 1e-5).sqrt()
    mu, sig = mu.detach(), sig.detach()
    input_normed = (input - mu) / sig
    input_normed = input_normed.detach().clone()

    # Set learnable style feature and adv optimizer
    adv_mu, adv_sig = mu, sig
    adv_mu.requires_grad_(True)
    adv_sig.requires_grad_(True)
    adv_optim = torch.optim.SGD(params=[adv_mu, adv_sig], lr=adv_lr, momentum=0, weight_decay=0)

    # Optimize adversarial style feature
    adv_optim.zero_grad()
    adv_input = input_normed * adv_sig+ adv_mu
    adv_output = net(adv_input)
    adv_loss = torch.nn.functional.cross_entropy(adv_output, gt)
    (- adv_loss).backward()
    adv_optim.step()

    ### Robust Model Training
    net.train()
    optim.zero_grad()
    adv_input = input_normed * adv_sig + adv_mu
    inputs = torch.cat((input, adv_input), dim=0)
    gt = torch.cat((gt, gt), dim=0)
    outputs = net(inputs)
    loss = F.cross_entropy(outputs, gt)
    loss.backward()
    optim.step()
```

Figure 7: The Pytorch-like pseudo-code of AdvStyle.

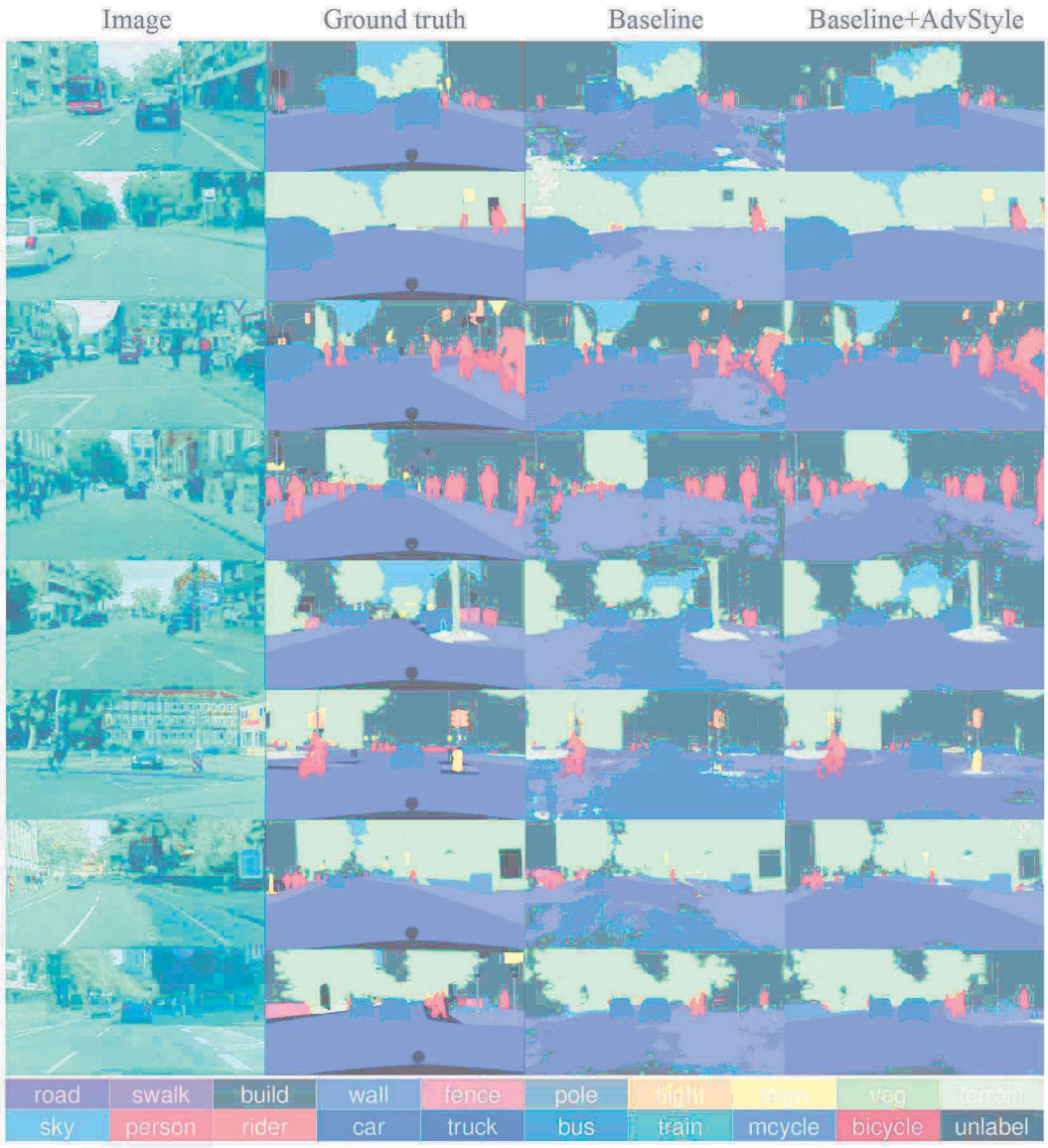

Figure 8: Segmentation results on CityScapes. Source: GTAV; Backbone: ResNet-50.

Figure 9: Segmentation results on BDD-100K. Source: GTAV; Backbone: ResNet-50.

Figure 10: Segmentation results on Mapillary. Source: GTAV; Backbone: ResNet-50.

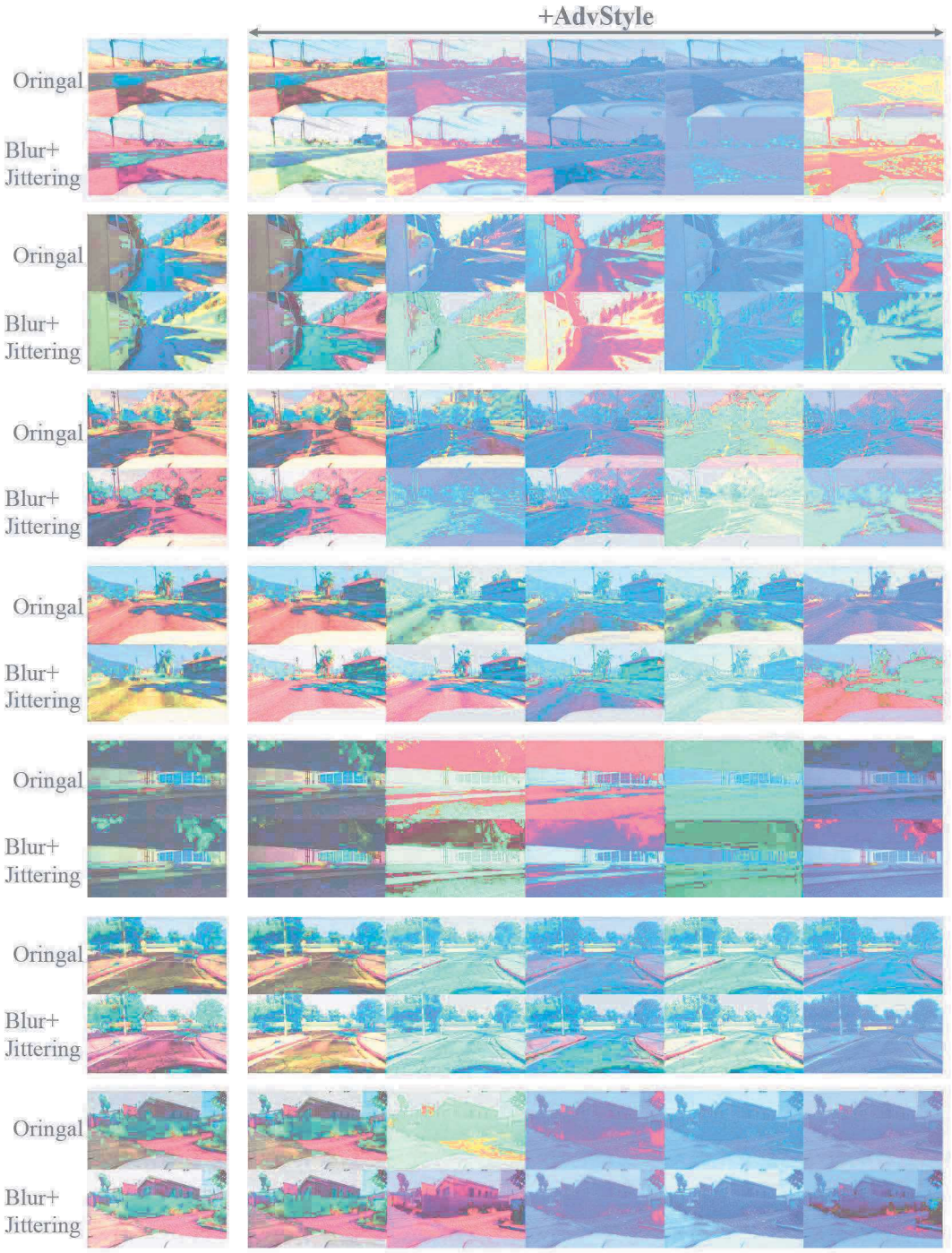

Figure 11: Examples of adversarial style augmentation. Source: GTAV; Backbone: ResNet-50.