Feature Statistics Mixing Regularization for Generative Adversarial Networks

Junho Kim[1] Yunjey Choi[1] Youngjung Uh[2†]

[1]NAVER AI Lab [2]Yonsei University

Abstract

*In generative adversarial networks, improving discriminators is one of the key components for generation performance. As image classifiers are biased toward texture and debiasing improves accuracy, we investigate 1) if the discriminators are biased, and 2) if debiasing the discriminators will improve generation performance. Indeed, we find empirical evidence that the discriminators are sensitive to the style (*e.g.*, texture and color) of images. As a remedy, we propose* feature statistics mixing regularization *(*FSMR*) that encourages the discriminator's prediction to be invariant to the styles of input images. Specifically, we generate a mixed feature of an original and a reference image in the discriminator's feature space and we apply regularization so that the prediction for the mixed feature is consistent with the prediction for the original image. We conduct extensive experiments to demonstrate that our regularization leads to reduced sensitivity to style and consistently improves the performance of various GAN architectures on nine datasets. In addition, adding FSMR to recently-proposed augmentation-based GAN methods further improves image quality. Our code is available at* https://github.com/naver-ai/FSMR.

1. Introduction

Generative adversarial networks (GANs) [8] have achieved significant development over the past several years, enabling many computer vision and graphics applications [4, 5, 13, 21, 22, 24, 29, 40]. On top of the carefully designed architectures [3, 17, 19, 20, 28, 30, 36], GAN-specific data augmentation and regularization techniques have been keys for improvements. Regularization techniques [9, 14–16, 26, 27, 37, 39] stabilize the training dynamics by penalizing steep changes in the discriminator's output within a local region of the input. On the other hand, data augmentation techniques [18, 38] prevent the discriminator from overfitting as commonly adopted in classification domains. Note that both efforts aim to guide the discriminator not to fixate on particular subsets of observations and to generalize over the entire data distribution.

Texture has been shown to provide a strong hint for classifiers [6, 7, 10]. If such a hint is sufficient enough to achieve high accuracy, the models tend not to learn the complexity of the intended task [2]. As the GAN discriminators are inherently classifiers, we presume that they also tend to rely on textures to classify real and fake images. Accordingly, the generators would focus on synthesizing textures which are regarded as real by the biased discriminator. In this paper, we answer the two questions: 1) are discriminators sensitive to style (*e.g.*, texture and color)? and 2) if yes, will debiasing the discriminators improve the generation performance?

To answer the first question, we define style distance as shown in Figure 1a. An ideal discriminator would produce small style distance because the two images have the same content. As we do not have a unit of measurement, we compute relative distance: the style distance divided by the content distance. In other words, we measure the sensitivity to style as multiples of the distance between images with different content. Surprisingly, Figure 1b shows that all baselines have noticeable values in relative distance.

To answer the second question, we debias the discriminators and measure improvements in generative performance. A straightforward approach for debiasing is to suppress the difference in the discriminator's output with respect to the style changes of the input image. Indeed, we observe that imposing a consistency loss [37, 39] on the discriminator between the original image and its stylized version improves the generator as mimicking contents becomes easier than mimicking style to fool the discriminator.

However, this approach leads to other difficulties: the criteria for choosing style images are unclear, and stylizing all training images with various style references requires a huge computational burden and an external style dataset. To efficiently address the style bias issue, we propose *feature statistics mixing regularization* (*FSMR*) which encourages the discriminator's prediction to be invariant to the styles of input images by mixing feature statistics within the discrim-

†Corresponding author.

inator. Specifically, we generate mixed features by combining original and reference features in the discriminator's intermediate layers and impose consistency between the predictions for the original and the mixed features.

In the experiments, we show that FSMR indeed induces the discriminator to have reduced sensitivity to style (Section 4.1). We then provide thorough comparisons to demonstrate that FSMR consistently improves various GAN methods on benchmark datasets (Section 4.2). Our method can be easily applied to any setting without burdensome preparation. Our implementation and models will be publicly available online for the research community. Our contributions can be summarized as follows:

- To the best of our knowledge, our work is the first style bias analysis for the discriminator of GANs.
- We define the relative distance metric to measure the sensitivity to the styles (Section 2).
- We propose feature statistics mixing regularization (FSMR), which makes the discriminator's prediction to be robust to style (Section 3).
- FSMR does not use external style images and outperforms the straightforward solution with external style images (Section 4.1).
- FSMR improves five baselines on all standard and small datasets regarding FID and relative distance (Section 4.2, 4.3).

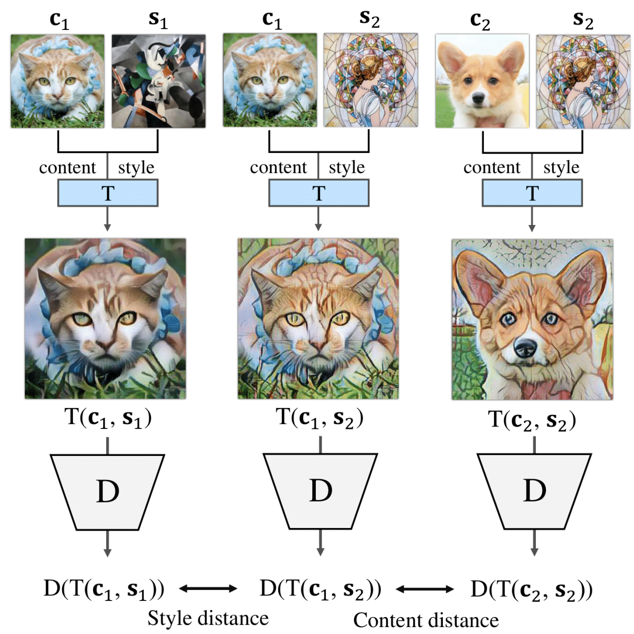
2. Style-bias in GANs

Our work is motivated by the recent finding that CNNs are sensitive to style rather than content, *i.e.*, ImageNet-trained CNNs are likely to make a style-biased decision when the style cue and content cue have conflict [7]. To quantitatively measure how sensitive a discriminator is to style, we compute style distance, content distance, and then relative distance. Afterward, we describe a straightforward baseline solution to reduce the discriminator's distance to style.

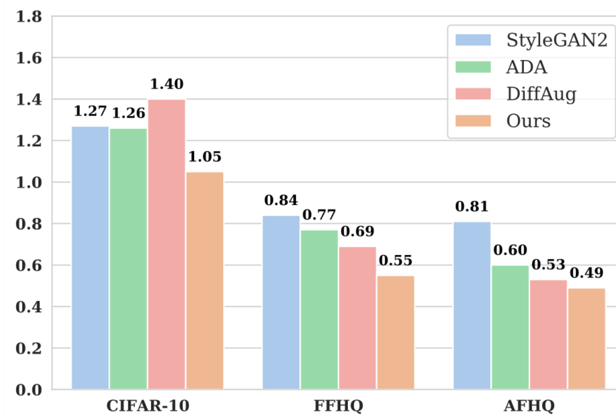
2.1. Style distance and content distance

We define a quantitative measure for how sensitive a discriminator is sensitive to style. First, given a set of training images, we utilize a style transfer method to synthesize differently stylized images of the same content. The styles are randomly chosen from WikiArt [1]. Figure 1a shows some example stylized images from AFHQ [5]. We define style distance d_s between images with different styles and the same content. The content distance d_c is defined vice versa:

$$\underbrace{d_s(\mathbf{c}, \mathbf{s}_1, \mathbf{s}_2)}_{\text{style distance}} = d(T(\mathbf{c}, \mathbf{s}_1), T(\mathbf{c}, \mathbf{s}_2)), \tag{1}$$



(a) Style distance and content distance



(b) Relative distance

Figure 1. (a) The style transfer method $T(\mathbf{c}, \mathbf{s})$ transfers the style of $\mathbf{s}$ on the content of $\mathbf{c}$. We define style distance as the output difference due to style variations. Content distance is defined vice versa. (b) Relative distance across various GAN methods. Relative distance indicates how sensitive a discriminator is to style changes (Eq. 3). See Section 2 for details.

$$\underbrace{d_c(\mathbf{s}, \mathbf{c}_1, \mathbf{c}_2)}_{\text{content distance}} = d(T(\mathbf{c}_1, \mathbf{s}), T(\mathbf{c}_2, \mathbf{s})), \tag{2}$$

where $T(\mathbf{c}, \mathbf{s})$ transfers the style of the reference image $\mathbf{s} \in \mathbb{R}^{C \times H \times W}$ to the content image $\mathbf{c} \in \mathbb{R}^{C \times H \times W}$, and d measures cosine distance in the last feature vectors of the discriminator. In practice, we use adaptive instance normalization (AdaIN) [12] as T. Figure 1 illustrates the process of calculating the content and style distances in Eq. (1) and

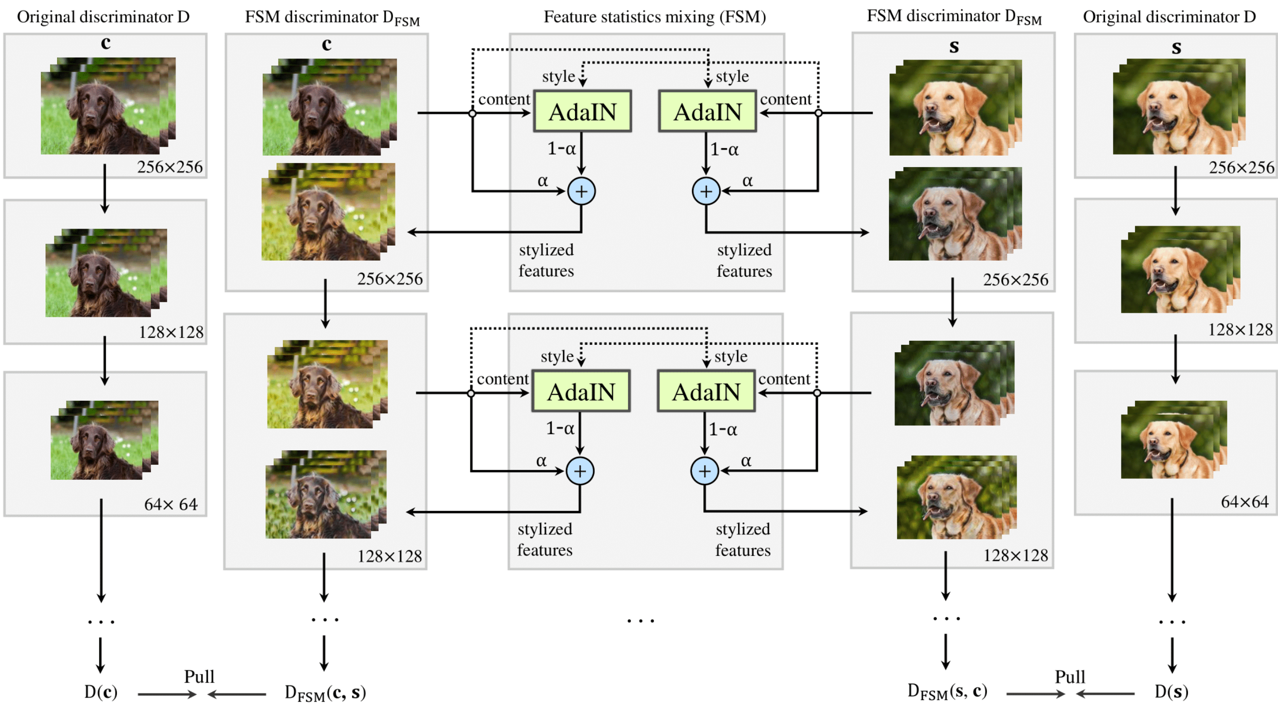


Figure 2. **Overview of feature statistics mixing regularization** (Section 3.2). Within the forward pass in the discriminator, we perturb features by applying AdaIN with a different sample. In deeper layers, the perturbations are applied recursively. A scalar $\alpha \sim \text{Uniform}(0, 1)$ moderates their strength. Then we enforce similarity between the original output and the perturbed one.

(2).

As we do not have a unit of measurement, we compute relative distance ρ, *i.e.*, the style distance divided by the content distance:

$$\underbrace{\rho}_{\text{relative distance}} = \mathop{\mathbb{E}}_{\substack{\mathbf{c}_1,\mathbf{c}_2\in\mathbf{C},\\ \mathbf{s}_1,\mathbf{s}_2\in\mathbf{S}}} \left[\frac{d_s(\mathbf{c}_1,\mathbf{s}_1,\mathbf{s}_2)}{d_c(\mathbf{s}_1,\mathbf{c}_1,\mathbf{c}_2)}\right], \qquad (3)$$

where $\mathbf{C}$ and $\mathbf{S}$ denote the training dataset and an external style dataset, respectively. The larger the ρ value, the more sensitive the discriminator is to style when classifying real and fake images. We will use the relative distance ρ for further analysis from here on. Our goal is to reduce the style distance so that the discriminators consider contents more important and produce richer gradients to the generators.

The relative distances of ImageNet-pretrained ResNet50 and ResNet50 pretrained for classifying Stylized ImageNet [7] supports validity of the metric. As the relative distance of the latter is less than the former and the latter is proven to be less biased toward style, we argue that the discriminators with lower relative distance are less sensitive to style (figures are deferred to Section 4.2).

2.2. Baseline: On-the-fly stylization

A well-known technique for preventing the classifiers from being biased toward styles is to augment the images with their style-transferred versions, especially using the WikiArt dataset [1] as style references [7]. It works because the style transfer does not alter the semantics of the original images or the anticipated output of the network. On the other hand, in GAN training, style transfer drives the images out of the original data distribution, thus it changes the anticipated output of the discriminator [18]. There are two workarounds for such a pitfall: 1) applying stochastic augmentations for both real and fake data [18, 38] and 2) penalizing the output difference caused by the augmentation instead of feeding the augmented images to the discriminator [37, 39]. As our goal is to make the discriminator less sensitive to style changes, we take the second approach as a straightforward baseline, for example, imposing consistency on the discriminator between the original images $\mathbf{c}$ and their randomly stylized images $T(\mathbf{c}, \mathbf{s})$ by

$$L_{\text{consistency}} = \mathbb{E}_{\mathbf{c},\mathbf{s}}\left[(D(\mathbf{c}) - D(T(\mathbf{c}, \mathbf{s})))^2\right], \qquad (4)$$

where $D(\boldsymbol{.})$ denotes the logit from the discriminator. However, it raises other questions and difficulties: the criteria for choosing the style images are unclear, and stylizing each image on-the-fly requires additional costs and an external dataset. Another option is to prepare a stylized dataset instead of on-the-fly stylization but it further requires prohibitively large storage. To combat this, we propose an efficient and generally effective method, feature mixing statistics regularization, whose details are described in the next section.

3. Proposed method

We first describe the traditional style transfer algorithm, AdaIN, as a preliminary. Then, we discuss how our proposed method, feature statistics mixing regularization (FSMR), incorporates AdaIN to induce the discriminator to be less sensitive to style.

3.1. Preliminary: AdaIN

Instance normalization (IN) [33] performs a form of style removal by normalizing feature statistics. Adaptive instance normalization (AdaIN) [12] extends IN to remove the existing style from the content image and transfer a given style. Specifically, AdaIN transforms content feature maps $\mathbf{x}$ into feature maps whose channel-wise mean and variance are the same as those of style feature maps $\mathbf{y}$:

$$\text{AdaIN}(\mathbf{x}, \mathbf{y}) = \sigma(\mathbf{y}) \left(\frac{\mathbf{x} - \mu(\mathbf{x})}{\sigma(\mathbf{x})} \right) + \mu(\mathbf{y}), \tag{5}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{C \times H \times W}$ are features obtained by a pre-trained encoder, and $\mu(\cdot)$ and $\sigma(\cdot)$ denote their mean and standard deviation their spatial dimensions, calculated for each channel, respectively. Then, through a properly trained decoder, the transformed features become a stylized image[1]. Much work has adopted AdaIN within the *generator* for improving the generation performance [5, 13, 19, 21, 22, 24]. On the contrary, our proposed method (FSMR) employs it within the *discriminator* for efficient regularization, as described below.

3.2. Feature statistics mixing regularization

Our goal is to make the discriminator do not heavily rely on the styles of the input images, without suffering from the difficulties of the on-the-fly stylization (Section 2.2). Hence, we propose feature statistics mixing regularization (FSMR), which does not require any external dataset and can be efficiently implemented as per-layer operations in the discriminator. FSMR mixes the mean and standard deviation of the intermediate feature maps in the discriminator using another training sample and penalizes discrepancy between the original output and the mixed one.

Specifically, we define feature statistics mixing (FSM) for feature maps $\mathbf{x}$ with respect to feature maps $\mathbf{y}$ to be AdaIN followed by linear interpolation:

$$\text{FSM}(\mathbf{x}, \mathbf{y}) = \alpha \mathbf{x} + (1 - \alpha) \text{AdaIN}(\mathbf{x}, \mathbf{y}), \tag{6}$$

[1] AdaIN may denote the full stylization process but it denotes the operation on the feature maps (Eq. 5) in this paper.

Algorithm 1 FSM Pseudocode, Tensorflow-like

```
# N: batch size, H: height, W: width, C: channels
def FSM(x, y, eps=1e-5):
   x_mu, x_var = tf.nn.moments(x, axes=[1,2])
   y_mu, y_var = tf.nn.moments(y, axes=[1,2])

   # normalize
   x_norm = (x - mu) / tf.sqrt(var + eps)

   # de-normalize
   x_fsm = x_norm * tf.sqrt(y_var + eps) + y_mu

   # combine
   alpha = tf.random.uniform(shape=[])

   x_mix = alpha * x + (1 - alpha) * x_fsm

   return x_mix # NxHxWxC
```

where $\alpha \sim \text{Uniform}(0, 1)$ controls the intensity of feature perturbation. We suppose that varying α will let the discriminator learn from various strengths of regularization.

Denoting an i-th layer of the discriminator as f_i, a content image as $\mathbf{c}$, and a style reference image as $\mathbf{s}$ which is randomly chosen from the current mini-batch samples, we define the mixed feature maps $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ through feed-forward operations with FSM:

$$\begin{aligned} \tilde{\mathbf{x}}_1 &= \mathbf{x}_1 = f_1(\mathbf{c}), \\ \tilde{\mathbf{y}}_1 &= \mathbf{y}_1 = f_1(\mathbf{s}), \\ \tilde{\mathbf{x}}_{i+1} &= f_{i+1}(\text{FSM}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{y}}_i)), \\ \tilde{\mathbf{y}}_{i+1} &= f_{i+1}(\text{FSM}(\tilde{\mathbf{y}}_i, \tilde{\mathbf{x}}_i)). \end{aligned} \tag{7}$$

Then the final output logit of the mixed feed-forward pass through the discriminator with n convolutional layers becomes:

$$D_{\text{FSM}}(\mathbf{c}, \mathbf{s}) = \text{Linear}(\tilde{\mathbf{x}}_n). \tag{8}$$

Given the original output $D(\mathbf{c})$ and the mixed output $D_{\text{FSM}}(\mathbf{c}, \mathbf{s})$, we penalize their discrepancy with a loss:

$$L_{\text{FSMR}} = \mathbb{E}_{\mathbf{c}, \mathbf{s} \sim p_{\text{data}}} \left[(D(\mathbf{c}) - D_{\text{FSM}}(\mathbf{c}, \mathbf{s}))^2 \right]. \tag{9}$$

Figure 2 illustrates the full diagram of FSMR. This loss is added to the adversarial loss [8] when updating the discriminator parameters. It regularizes the discriminator to produce consistent output under different statistics of the features varying through the layers. Our design of L_{FSMR} is general-purpose and thereby can be combined with other methods [18, 19, 38]. As shown in Algorithm 1, FSM can be implemented with only a few lines of code. Also, we provide the Tensorflow-like pseudo-code of FSMR in Appendix **??**.

3.3. Visualizing the effect of FSM

To visually inspect the effect of FSM in the discriminator, we train a decoder (same architecture as the one for AdaIN [12]) which reconstructs the original image from the 32×32 feature maps of the *original* discriminator.

(a) Style (b) Content (c) Stylization by AdaIN [12] (d) Visualization of FSM

Figure 3. **Visualization of the effect of FSM** (Section 3.3). (a) Example style images. (b) Example content images. (c) AdaIN largely distorts fine details. (d) Reconstruction of FSMed features preserves them.

In Figure 3, the content images go through the discriminator with FSM on all layers with respect to the style images to produce stylized (*i.e.*, *FSMed*) intermediate features. Then the learned decoder synthesizes the result images from the FSMed features.

The FSMed images have similar global styles to the style images but contain semantics of the content images. It has a similar effect to AdaIN but better preserves the fine details of the content. We suggest that it is the key for the discriminator to be able to provide gradients toward more realistic images for the generator leading to higher quality images than the on-the-fly stylization baseline (Section 4.1).

4. Experiments

We conduct extensive experiments on six datasets of CIFAR-10 [25], FFHQ [19], AFHQ [5], CelebA-HQ [17], LSUN Church [34], and MetFaces [18] with five GAN methods such as DCGAN [30], bCRGAN [39], StyleGAN2 [20], DiffAugment [38], and ADA [18]. We choose the datasets and baseline methods following the recent experimental setups [18,38]. We use the relative distance ρ (Eq. 3), Fréchet inception distance (FID) [11], and inception score (IS) [31] as evaluation metrics. When we compute FID, we use all training samples and the same number of fake samples. All the baseline methods are trained using the official implementations provided by the authors. See Appendix **??** for more details. We next conduct thorough experiments to demonstrate the superiority of our method over the straightforward solution and the baselines.

4.1. Comparison with the on-the-fly stylization

In this section, we compare our method with the on-the-fly stylization, *i.e.*, generating stylized images via AdaIN during training and applying consistency regularization (Section 2.2). To perform this, we collect 100 style images from WikiArt [1] and randomly sample one for stylizing each image during training. Note that, unlike the on-the-fly stylization, FSMR does not rely on external style images. We conduct experiments on five benchmark datasets: CIFAR-10, CelebA-HQ, FFHQ, AFHQ, and LSUN Church.

Table 1 compares effect of regularization with on-the-fly stylization and FSMR in FID. While the former improves FID compared to the baselines to some extent, improvements due to FSMR are larger in all cases. For comparison with additional networks and datasets, see Appendix **??**.

To measure the discriminator's sensitivity to style, we compute the relative distance ρ (Eq. 3) for each method. Figure 4 shows the relative distance on CIFAR-10, FFHQ, and AFHQ. As one can easily expect, utilizing the stylized dataset reduces the discriminator's sensitivity toward style. It is worth noting that FSMR not only consistently reduces the sensitivity but also outperforms the competitor in all cases. This is a very meaningful result because FSMR *does not use any external stylized dataset* but it uses only the original images during training. We also observe that the lower relative distances agree with the lower FIDs within the same environment.

We compare the time and memory costs in Table 1. FSMR requires 3.0$\sim$7.4% extra training time, but the on-the-fly method requires 17.2$\sim$26.8% extra training time for additional feedforward passes in image stylization. In addition, the on-the-fly method requires 70.0$\sim$87.5% extra GPU memory to hold pretrained networks and features for image stylization, but FSMR only adds negligible ($\sim$2%) GPU memory. To avoid extra costs for the on-the-fly stylization during training, we can prepare the stylized datasets before training (*i.e.*, different approach but has the same effect as the on-the-fly stylization). However, the one-to-many stylization in advance requires heavy computation and prohibitively large storage as shown in Table 2. For example, to construct the stylized dataset for 1024$\times$1024 FFHQ with 100 style references, we need to process and store more than 7.0M (70k $\times$ 100) images (8.93TB).

As an ablation study, we push toward harsher regularization: using randomly shifted feature maps instead of FSM. We observe that using arbitrary mean and standard deviation in AdaIN (Eq. 5) significantly hampers adversar-

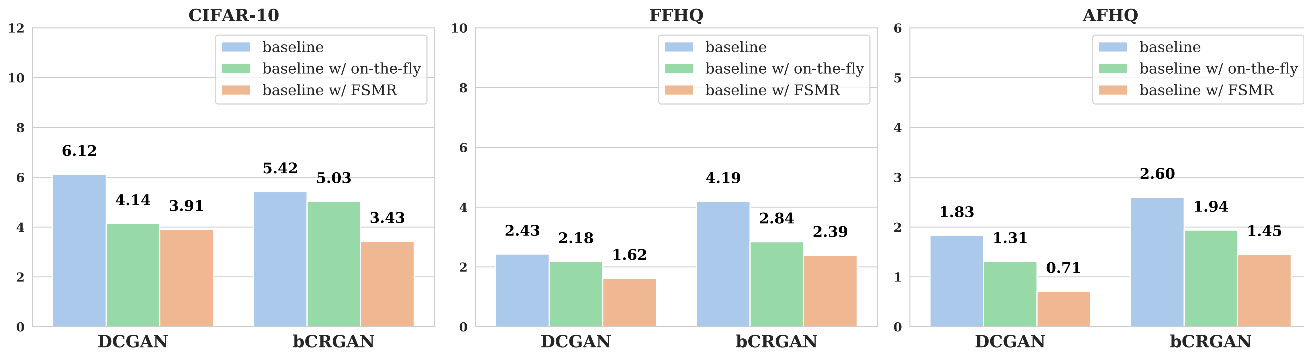


Figure 4. **The relative distance of the discriminators** on CIFAR-10, FFHQ, and AFHQ. We observe a positive correlation with FID in each case. See Appendix **??** for more results on other baselines and datasets.

Method	Standard dataset					Costs	
	CIFAR-10	FFHQ	AFHQ	CelebA-HQ	LSUN Church	Time (Hours)	Memory (GB)
DCGAN	15.89±0.12	7.82±0.10	17.27±0.13	6.71±0.09	17.33±0.11	25.4 (1.5†)	5 (4†)
DCGAN w/ on-the-fly	15.88±0.11	7.33±0.17	14.22±0.15	5.41±0.10	26.05±0.14	31.5 (1.8†)	8.5 (7.5†)
DCGAN w/ FSMR	**14.98±0.09**	**6.76±0.08**	**13.19±0.09**	**5.23±0.10**	**13.84±0.10**	26.2 (1.6†)	5.1 (4†)
bCRGAN	12.46±0.09	6.43±0.08	9.35±0.10	4.31±0.09	13.20±0.10	26.1 (1.6†)	5 (4†)
bCRGAN w/ on-the-fly	12.43±0.10	5.20±0.09	8.63±0.12	3.47±0.09	10.51±0.10	33.1 (1.9†)	8.5 (7.5†)
bCRGAN w/ FSMR	**11.17±0.07**	**4.68±0.08**	**8.33±0.08**	**3.43±0.09**	**9.09±0.07**	27.7 (1.7†)	5.1 (4†)

Table 1. **FID comparison on DCGAN variants** with FSMR and the baseline on-the-fly stylization. The bold numbers indicate the best FID for each baseline. We report the mean FID over 3 training runs together with standard deviations and the additional costs. All image resolutions are set to 128×128 due to the backbone architecture except CIFAR-10 (32×32). Time and memory are measured in 128×128 images, and † indicates what is measured in 32×32 images. Time means a full training time.

	CIFAR-10	CelebA-HQ	FFHQ	AFHQ	LSUN Church
Time	8	10	30	5	40

Table 2. **The time to create the stylized dataset** for each standard dataset, measured in hours.

ial training between a generator and a discriminator, *i.e.*, the training diverges. On the other hand, FSMR using in-domain samples shows the anticipated effect.

4.2. Standard datasets

We evaluate the effectiveness of FSMR on three benchmark datasets, all of which have more than 10k training images: CIFAR-10 (50k), FFHQ (70k), and AFHQ (16k). Table 3 (left) shows that FSMR consistently improves StyleGAN2 even with existing augmentation techniques [18,38]. We emphasize that FSMR enhances baselines by a large gap on AFHQ, in which case the discriminator might be easily biased toward color and texture of the animals.

Figure 5 shows the relative distances on CIFAR-10, FFHQ, and AFHQ for StyleGAN2 variants. FSMR reduces the relative distances in all cases and they agree with the improvements in FID. We also provide the relative distances of ResNet50 networks pretrained on ImageNet and Stylized ImageNet as references in each dataset (Section 2.1). As the lower relative distances agree with the higher classification performances, the lower relative distances of the discriminator agree with the higher generative performances.

In addition, Table 4 demonstrates that applying FSMR on StyleGAN2 variants further improves both FID and IS for both unconditional and class-conditional generation on CIFAR-10. For qualitative results, see Figure 6 and Appendix **??**.

4.3. Small datasets.

GANs are known to be notoriously difficult to train on small datasets due to limited coverage of the data manifold. Being able to train GANs on small datasets would lead to a variety of application domains, making a rich synthesis experience for the users. We tried our method with five small datasets that consist of a limited number of training images such as MetFaces (1k), AFHQ Dog (5k), AFHQ Cat (5k).

Method	Standard dataset			Small dataset			
	CIFAR-10	FFHQ	AFHQ	MetFaces	AFHQ Dog	AFHQ Cat	AFHQ Wild
StyleGAN2	3.89±0.07	5.62±0.10	11.37±0.03	51.88±0.44	19.65±0.07	8.37±0.06	4.17±0.06
+ FSMR	**3.76**±0.03	**3.74**±0.03	**8.59**±0.03	**45.47**±0.42	**18.08**±0.07	**6.69**±0.04	**3.96**±0.03
StyleGAN2-ADA	3.23±0.06	4.05±0.07	7.73±0.11	29.17±0.08	13.56±0.10	6.64±0.09	3.74±0.14
+ FSMR	**2.90**±0.08	**3.91**±0.06	**6.12**±0.10	**27.81**±0.11	**11.76**±0.14	**5.71**±0.10	**3.24**±0.16
StyleGAN2-DiffAug	3.23±0.08	5.35±0.09	7.52±0.08	32.96±0.08	16.92±0.06	6.39±0.05	4.39±0.07
+ FSMR	**2.93**±0.05	**4.99**±0.08	**6.53**±0.05	**29.98**±0.15	**14.55**±0.18	**6.29**±0.07	**4.28**±0.04

Table 3. **FID comparison on StyleGAN2 variants**. The bold numbers indicate the best FID for each baseline. We report the mean FID over 3 training runs together with standard deviations. FSMR improves the baselines in all cases.

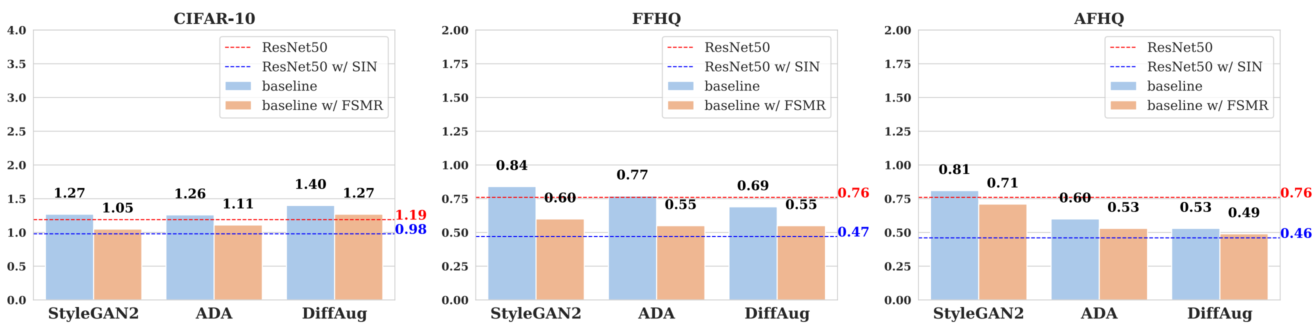


Figure 5. **The relative distance of the discriminators** on CIFAR-10, FFHQ, and AFHQ for StyleGAN2 variants. The higher ρ value, the more sensitive the discriminator is to style when classifying real and fake. We report the reference values for the relative distances using ResNet50 trained on ImageNet (red line) and ResNet50 trained on Stylized ImageNet (blue line) [7]. As the lower relative distances agree with the higher classification performances, the lower relative distances of the discriminator agree with the higher generative performances.

Method	Unconditional		Conditional	
	FID ↓	IS ↑	FID ↓	IS ↑
StyleGAN2	3.89	9.36	3.52	9.77
+ FSMR	**3.76**	**9.58**	**3.35**	**10.05**
StyleGAN2-ADA	3.23	9.47	2.76	9.98
+ FSMR	**2.90**	**9.68**	**2.63**	**10.03**
StyleGAN2-DiffAug	3.23	9.63	3.10	9.84
+ FSMR	**2.93**	**9.81**	**2.87**	**10.02**

Table 4. **FID and inception score comparison** on CIFAR-10 across StyleGAN2 variants. Bold face indicates the best scores for each baseline. We report the mean scores over three training runs.

AFHQ Wild (5k). As shown in Table 3 (right), we can observe that FSMR improves FID stably for all the baseline models, even if the number of data is small. See Figure 6 and Appendix **??** for qualitative results.

5. Related Work

Improving discriminators. While generative adversarial networks [8] have been developing regarding their network architectures [19, 20, 26, 30], regularizing the discriminator has been simultaneously considered as an important technique for stabilizing their adversarial training. Examples include instance noise [14], gradient penalties [9, 26], spectral normalization [27], contrastive learning [15, 16], and consistency regularization [37, 39]. They implicitly or explicitly enforce smooth changes in the outputs within some perturbation of the inputs. Recent methods employ data augmentation techniques to prevent discriminator overfitting [18, 38]. While they explicitly augment the images, our method implicitly augments the feature maps in the discriminator. In addition, while they use standard transformations which are used in training classifiers, our method regularizes the discriminator to produce small changes when the style of the input image is changed and it effectively prevents discriminator from being biased toward style.

Bias toward style. Convolutional neural networks are biased toward style (texture) when they are trained for classi-

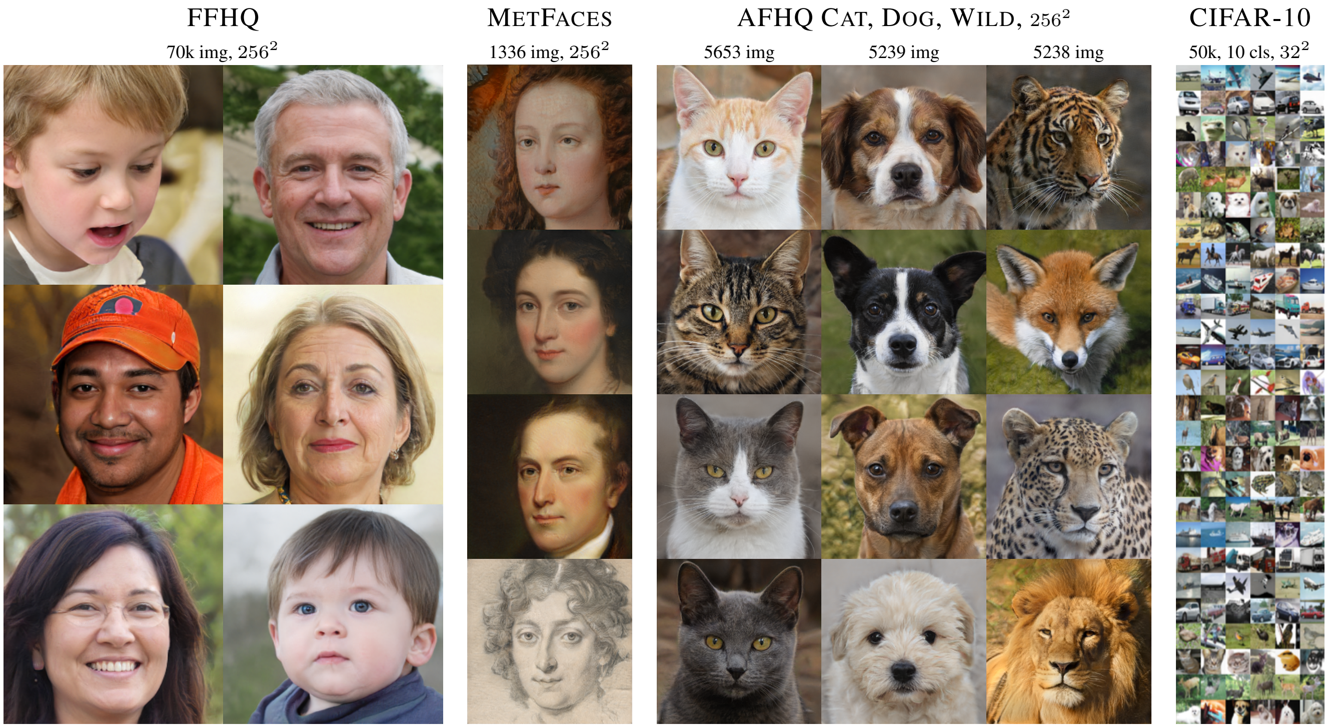


Figure 6. Examples of generated images for several datasets trained using FSMR. Please note that we do not use transfer learning on MetFaces as opposed to ADA. See Appendix **??** for more uncurated results.

fication [6, 7, 10]. The straightforward solution for reducing the bias is randomizing textures of the samples by a style transfer algorithm [7]. It is a kind of data augmentation technique in that the style transfer prevents classifiers from overfitting to styles as geometric or color transformations prevent classifiers from overfitting to certain positions or colors. As simply perturbing the data distribution in GAN training results in perturbed fake distribution [18], we introduce an extra forward pass with an implicitly stylized feature and impose consistency in the output with respect to the original forward pass (Eq. 9). While the linear interpolation of our mixing resembles mixup [35], we do not interpolate target outputs but only soften the changes in feature statistics.

6. Limitation and Discussion

As shown in various experiments, we have found that the discriminators have a bias for style, which enables numerical representation through the relative distance metric. However, we have not found out the optimal value that how much relative distance should be reduced for each model. We observed through the reference value in Figure 5, that even though we could not find the optimal value, the relationship where the relative distance decreases, the less bias to style reduces. We have proposed FSMR, which reduces the bias to style using only internal training datasets, rather than using external datasets, and proved that FSMR is very simple yet effective. In future work, it would be worthwhile to search the optimal value for the relative distances and to unify the relative distances among different models.

7. Conclusion

We observed that the discriminators are biased toward style. To quantitatively measure the amount of bias, we proposed relative distance, *i.e.*, style distance divided by the content distance. While reducing the style bias with a straightforward consistency regularization with style transfer method induces ambiguity and difficulties, our feature statistics mixing regularization (*FSMR*) provides a simple and effective solution. Importantly, FSMR does not explicitly stylize the images but perturbs the intermediate features in the discriminator. We visualize the effect of FSMR and quantitatively analyze its behavior regarding relative sensitivity. The experiments demonstrated that our method consistently improves various network architectures, even in conjunction with the latest techniques.

Acknowledgements The authors thank NAVER AI Lab researchers and Jun-Yan Zhu for constructive discussion. All experiments were conducted on NAVER Smart Machine Learning (NSML) platform [23, 32]. This work was partly supported by an IITP grant (No.2021-0-00155) and an NRF grant (NRF-2021R1G1A1095637). Both grants are funded by the Korean government (MSIT).

References

[1] Wikiart. https://www.kaggle.com/c/painter-by-numbers. 2, 3, 5

[2] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *ICML*, 2020. 1

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 1

[4] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 1

[5] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020. 1, 2, 4, 5

[6] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *NeurIPS*, 2015. 1, 8

[7] 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 *ICLR*, 2019. 1, 2, 3, 7, 8

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NeurIPS*, 2014. 1, 4, 7

[9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017. 1, 7

[10] Katherine L Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. In *NeurIPS*, 2020. 1, 8

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 5

[12] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 2, 4, 5

[13] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 1, 4

[14] Simon Jenni and Paolo Favaro. On stabilizing generative adversarial training with noise. In *CVPR*, 2019. 1, 7

[15] Jongheon Jeong and Jinwoo Shin. Training gans with stronger augmentations via contrastive discriminator. In *ICLR*, 2021. 1, 7

[16] Minguk Kang and Jaesik Park. Contragan: Contrastive learning for conditional image generation. In *NeurIPS*, 2020. 1, 7

[17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 1, 5

[18] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020. 1, 3, 4, 5, 6, 7, 8

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 4, 5, 7

[20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 5, 7

[21] Hyunsu Kim, Yunjey Choi, Junho Kim, Sungjoo Yoo, and Youngjung Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *CVPR*, 2021. 1, 4

[22] Hyunsu Kim, Ho Young Jhoo, Eunhyeok Park, and Sungjoo Yoo. Tag2pix: Line art colorization using text tag with secat and changing loss. In *ICCV*, 2019. 1, 4

[23] Hanjoo Kim, Minkyu Kim, Dongjoo Seo, Jinwoong Kim, Heungseok Park, Soeun Park, Hyunwoo Jo, KyungHyun Kim, Youngil Yang, Youngkwan Kim, et al. Nsml: Meet the mlaas platform with a real-world case study. *arXiv preprint arXiv:1810.09957*, 2018. 8

[24] Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwang Hee Lee. U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. In *ICLR*, 2020. 1, 4

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[26] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, 2018. 1, 7

[27] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 1, 7

[28] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *ICLR*, 2018. 1

[29] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 1

[30] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 1, 5, 7

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016. 5

[32] Nako Sung, Minkyu Kim, Hyunwoo Jo, Youngil Yang, Jingwoong Kim, Leonard Lausen, Youngkwan Kim, Gayoung Lee, Donghyun Kwak, Jung-Woo Ha, et al. Nsml: A machine learning platform that enables you to focus on your models. *arXiv preprint arXiv:1712.05902*, 2017. 8

[33] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 4

[34] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 5

[35] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 8

[36] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019. 1

[37] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. In *ICLR*, 2020. 1, 3, 7

[38] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *NeurIPS*, 2020. 1, 3, 4, 5, 6, 7

[39] Zhengli Zhao, Sameer Singh, Honglak Lee, Zizhao Zhang, Augustus Odena, and Han Zhang. Improved consistency regularization for gans. In *AAAI*, 2021. 1, 3, 5, 7

[40] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2016. 1