CHAIN: Enhancing Generalization in Data-Efficient GANs via lips*CH*itz continuity constr*AI*ned *N*ormalization

Yao Ni[†] Piotr Koniusz[§,†]

[†]The Australian National University
[§]Data61♥CSIRO

CVPR 2024

Background: Challenges in Data-Efficient GANs and BN

Challenges in Data-Efficient GANs:

- Discriminator overfitting.
- Training instability.

Background: Challenges in Data-Efficient GANs and BN

Challenges in Data-Efficient GANs:

- Discriminator overfitting.
- Training instability.

Advantages of Batch Normalization (BN):

- aids generalization by aligning training and test distributions and reducing sharpness of loss landscape.
- stabilizes training process by mitigating internal covariate shift.

Background: Challenges in Data-Efficient GANs and BN

Challenges in Data-Efficient GANs:

- Discriminator overfitting.
- Training instability.

Advantages of Batch Normalization (BN):

- aids generalization by aligning training and test distributions and reducing sharpness of loss landscape.
- stabilizes training process by mitigating internal covariate shift.

BN appears to benefit the discriminator, but is rarely used as it impairs performance.

Background: Challenges in Data-Efficient GANs and BN

Challenges in Data-Efficient GANs:

- Discriminator overfitting.
- Training instability.

Advantages of Batch Normalization (BN):

- aids generalization by aligning training and test distributions and reducing sharpness of loss landscape.
- stabilizes training process by mitigating internal covariate shift.

BN appears to benefit the discriminator, but is rarely used as it impairs performance.

Goal: Integrate BN into discriminator for improved generalization.

Methods: Generalization error of GANs

Lemma 3.1 (GAN generalization error on function set):

$$\epsilon_{\text{gan}} \leqslant 2d_{\mathcal{H}}(\mu, \hat{\mu}_n) + 2d_{\mathcal{H}}(\nu_n^*, \hat{\nu}_n)$$

$d_{\mathcal{H}}$: discrepancy over discriminator set. $\mu, \hat{\mu}_n$: unseen/seen real data. $\nu_n^*, \hat{\nu}_n$: ideal/seen fake.

Methods: Generalization error of GANs

Lemma 3.1 (GAN generalization error on function set):

$$\epsilon_{\text{gan}} \leqslant 2\, d_{\mathcal{H}}(\mu, \hat{\mu}_n) + 2\, d_{\mathcal{H}}(\nu_n^*, \hat{\nu}_n)$$

$d_{\mathcal{H}}$: discrepancy over discriminator set. $\mu, \hat{\mu}_n$: unseen/seen real data. $\nu_n^*, \hat{\nu}_n$: ideal/seen fake.

$\nu_n^* \approx \hat{\mu}_n \implies$ Lowering real/fake discrepancy aids generalization

Methods: Generalization error of GANs

Lemma 3.1 (GAN generalization error on function set):

$$\epsilon_{\text{gan}} \leqslant 2 d_{\mathcal{H}}(\mu, \hat{\mu}_n) + 2 d_{\mathcal{H}}(\nu_n^*, \hat{\nu}_n)$$

$d_{\mathcal{H}}$: discrepancy over discriminator set. $\mu, \hat{\mu}_n$: unseen/seen real data. $\nu_n^*, \hat{\nu}_n$: ideal/seen fake.

$\nu_n^* \approx \hat{\mu}_n \implies$ Lowering real/fake discrepancy aids generalization

μ is inaccessible. We need further analyze $d_{\mathcal{H}}(\mu, \hat{\mu}_n)$.

Methods: Generalization error of GANs

Lemma 3.2 (GAN generalization error on neural network):

$$\epsilon_{\text{gan}}^{\text{nn}} \leqslant 2\omega\big(\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \|\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}\|_2\big) + 4R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big) + \omega^2\big(|\lambda_{\text{max}}^{\boldsymbol{H}}| + |\lambda_{\text{max}}^{\widetilde{\boldsymbol{H}}}|\big)$$

$\epsilon_{\text{gan}}^{\text{nn}}$: ϵ_{gan} on neural networks, $\omega > 0$. $\boldsymbol{\theta}_d$: D's weights. $\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}, \lambda_{\text{max}}^{\boldsymbol{H}}$: real gradient, top Hessian eigenvalue. $\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}, \lambda_{\text{max}}^{\widetilde{\boldsymbol{H}}}$: fake versions. R: related to $\|\boldsymbol{\theta}_d\|_2^2$ and data size n.

Methods: Generalization error of GANs

Lemma 3.2 (GAN generalization error on neural network):

$$\epsilon_{\text{gan}}^{\text{nn}} \leqslant 2\omega\Big(\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \|\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}\|_2\Big) + 4R\left(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\right) + \omega^2\big(|\lambda_{\max}^{\boldsymbol{H}}| + |\lambda_{\max}^{\widetilde{\boldsymbol{H}}}|\big)$$

$\epsilon_{\text{gan}}^{\text{nn}}$: ϵ_{gan} on neural networks, $\omega > 0$. $\boldsymbol{\theta}_d$: D's weights. $\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}, \lambda_{\max}^{\boldsymbol{H}}$: real gradient, top Hessian eigenvalue. $\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}, \lambda_{\max}^{\widetilde{\boldsymbol{H}}}$: fake versions. R: related to $\|\boldsymbol{\theta}_d\|_2^2$ and data size n.

Reducing weight gradient norms of discriminator aids generalization.

Methods: Motivation for BN

Better generalization on GANs requires:

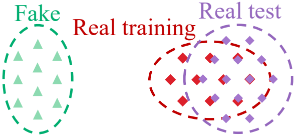
- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Methods: Motivation for BN

Better generalization on GANs requires:

- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Motivation of BN

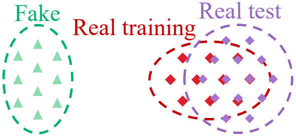


Methods: Motivation for BN

Better generalization on GANs requires:

- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Motivation of BN



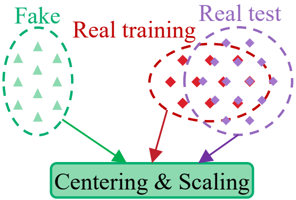
Applying BN **separately** on real/fake batches

Methods: Motivation for BN

Better generalization on GANs requires:

- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Motivation of BN



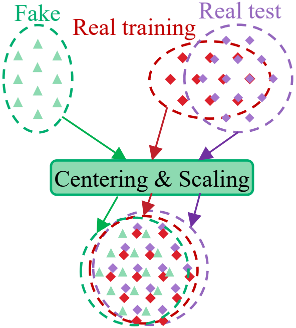
Applying BN **separately** on real/fake batches

Methods: Motivation for BN

Better generalization on GANs requires:

- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Motivation of BN



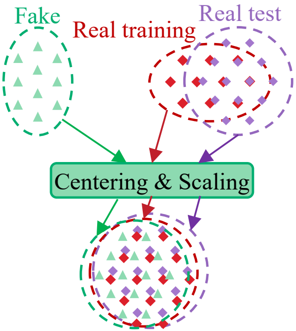
Applying BN **separately** on real/fake batches reduces the fake-real discrepancy via standardization.

Methods: Motivation for BN

Better generalization on GANs requires:

- Lowering real-fake discrepancy
- Reducing weight gradient norms of discriminator

Motivation of BN



Applying BN **separately** on real/fake batches reduces the fake-real discrepancy via standardization.

But incorporating BN risks gradient explosion issues.

Methods: Gradient Issues of BN

Standardization in BN:

$$\text{Linear transformation:} \quad \boldsymbol{Y} = \boldsymbol{A}\boldsymbol{W}$$

$$\text{Centering:} \quad \overset{c}{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{\mu}$$

$$\text{Scaling:} \quad \overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}} / \boldsymbol{\sigma}.$$

Methods: Gradient Issues of BN

Standardization in BN:

Linear transformation: $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{W}$

Centering: $\overset{c}{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{\mu}$

Scaling: $\overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}} / \boldsymbol{\sigma}.$

Theorem 3.1 (Issue in centering, **similarity dropping causes feature divergence**):

$$\mathbb{E}_{\boldsymbol{y}_1, \boldsymbol{y}_2}\big[\cos(\boldsymbol{y}_1, \boldsymbol{y}_2)\big] \geqslant \mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\big[\cos(\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2)\big] = 0$$

$\boldsymbol{y}_1$,$\overset{c}{\boldsymbol{y}}_1$: pre- & post-centering features. Features similar in early layers diverge in later layers.

Methods: Gradient Issues of BN

Standardization in BN:

$$\text{Linear transformation:} \quad \boldsymbol{Y} = \boldsymbol{A}\boldsymbol{W}$$
$$\text{Centering:} \quad \overset{c}{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{\mu}$$
$$\text{Scaling:} \quad \overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}} / \boldsymbol{\sigma}.$$

Theorem 3.1 (Issue in centering, **similarity dropping causes feature divergence**):

$$\mathbb{E}_{\boldsymbol{y}_1, \boldsymbol{y}_2}\big[\cos(\boldsymbol{y}_1, \boldsymbol{y}_2)\big] \geqslant \mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\big[\cos(\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2)\big] = 0$$

$\boldsymbol{y}_1, \overset{c}{\boldsymbol{y}}_1$: pre- & post-centering features. Features similar in early layers diverge in later layers.

Theorem 3.2 (Issue in scaling, **unbounded Lipschitz causes gradient explosion**):

$$\Big\|\operatorname{diag}\Big(\frac{1}{\boldsymbol{\sigma}}\Big)\Big\|_{\text{lc}} = \frac{1}{\sigma_{\min}}$$

Lipschitz constant (lc) is large when $\sigma_{\min} = \min_c \sigma_c$, is small.

Method: CHAIN replaces centering/scaling with 0MR/ARMS

mean $\boldsymbol{\mu}$ and root mean square $\boldsymbol{\psi}$:

$$\mu_c = \frac{1}{B\times H\times W}\sum_{b,h,w} Y_{b,c,h,w}$$

$$\psi_c = \sqrt{\left(\frac{1}{B\times H\times W}\sum_{b,h,w} Y^2_{b,c,h,w}\right) + \epsilon}$$

Method: CHAIN replaces centering/scaling with 0MR/ARMS

mean $\boldsymbol{\mu}$ and root mean square $\boldsymbol{\psi}$:

$$\mu_c = \frac{1}{B\times H\times W}\sum_{b,h,w} Y_{b,c,h,w}$$

$$\psi_c = \sqrt{\left(\frac{1}{B\times H\times W}\sum_{b,h,w} Y_{b,c,h,w}^2\right) + \epsilon}$$

0-mean regularization:

$$\ell^{0\mathrm{MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2$$

$\psi_{\min} = \min_c \psi_c$. $\epsilon = 10^{-5}$. λ: a hyperparameter. $p \in [0,1]$ controls $\ell^{0\mathrm{MR}}$ and Bernoulli mask $\boldsymbol{M} \sim \mathcal{B}(p)$

Pytorch-style pseudo code for $\text{CHAIN}_{\text{batch}}$

```
# Y: B×d×H×W size; lbd:hyperparameter λ
def CHAIN_batch(Y, p, lbd, eps=1e-5):
  reg=Y.mean([0,2,3]).square().sum()*(p*lbd)
```

Method: CHAIN replaces centering/scaling with 0MR/ARMS

mean $\boldsymbol{\mu}$ and root mean square $\boldsymbol{\psi}$:

$$\mu_c = \frac{1}{B\times H\times W}\sum_{b,h,w} Y_{b,c,h,w}$$

$$\psi_c = \sqrt{\left(\frac{1}{B\times H\times W}\sum_{b,h,w} Y^2_{b,c,h,w}\right) + \epsilon}$$

0-mean regularization:

$$\ell^{0\mathrm{MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2$$

Adaptive RMS normalization:

$$\mathrm{ARMS}(\boldsymbol{Y}) = (1-\boldsymbol{M})\odot\boldsymbol{Y} + \boldsymbol{M}\odot\frac{\boldsymbol{Y}}{\boldsymbol{\psi}}\cdot\psi_{\min}$$

$\psi_{\min} = \min_c \psi_c$. $\epsilon = 10^{-5}$. λ: a hyperparameter. $p \in [0,1]$ controls $\ell^{0\mathrm{MR}}$ and Bernoulli mask $\boldsymbol{M} \sim \mathcal{B}(p)$

Pytorch-style pseudo code for $\mathrm{CHAIN}_{\mathrm{batch}}$

```
# Y:B×d×H×W size; lbd:hyperparameter λ
def CHAIN_batch(Y, p, lbd, eps=1e-5):
  reg=Y.mean([0,2,3]).square().sum()*(p*lbd)
  M= (torch.rand(*Y.shape[:2], 1, 1)<p)*1.0
  psi_s=Y.square().mean([0,2,3],keepdim=True)
  psi  = (psi_s + eps).sqrt()
  psi_min = psi.min().detach()
  Y_arms = (1 - M) * Y + M * (Y/psi*psi_min)
  return Y_arms, reg
```

Method: CHAIN reduces feature and weight gradients in D

$$\|\Delta \boldsymbol{y}_c\|_2^2 \leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \Big(\tfrac{(1-p)\psi_c + p\psi_{\min}}{\psi_c}\Big)^2 - \frac{2(1-p)p\psi_{\min}}{B\psi_c}(\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2$$

$$\|\Delta \boldsymbol{w}_c\|_2^2 \leqslant \lambda_{\max}^2 \|\Delta \boldsymbol{y}_c\|_2^2$$

$\Delta \boldsymbol{y}_c, \Delta \dot{\boldsymbol{y}}_c$: c-th column of gradient for CHAIN input/output $\boldsymbol{Y}, \dot{\boldsymbol{Y}}$. $\lambda_{\max}$: top eigenvalue of $\boldsymbol{A}$. $\check{\boldsymbol{y}}_c$: c-th column of $\check{\boldsymbol{Y}} = \boldsymbol{Y}/\psi$. $\Delta \boldsymbol{w}_c$: c-th column of grad for $\boldsymbol{W}$. $p \in [0, 1]$

Method: CHAIN reduces feature and weight gradients in D

$$\|\Delta \boldsymbol{y}_c\|_2^2 \leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \Big(\boxed{\tfrac{(1-p)\psi_c + p\psi_{\min}}{\psi_c}} \Big)^2 - \frac{2(1-p)p\psi_{\min}}{B\psi_c} (\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2$$

$$\|\Delta \boldsymbol{w}_c\|_2^2 \leqslant \lambda_{\max}^2 \|\Delta \boldsymbol{y}_c\|_2^2$$

$\Delta \boldsymbol{y}_c, \Delta \dot{\boldsymbol{y}}_c$: c-th column of gradient for CHAIN input/output $\boldsymbol{Y}, \dot{\boldsymbol{Y}}$. $\lambda_{\max}$: top eigenvalue of $\boldsymbol{A}$. $\check{\boldsymbol{y}}_c$: c-th column of $\check{\boldsymbol{Y}} = \boldsymbol{Y}/\psi$. $\Delta \boldsymbol{w}_c$: c-th column of grad for $\boldsymbol{W}$. $p \in [0, 1]$

$$\tfrac{(1-p)\psi_c + p\psi_{\min}}{\psi_c} \leqslant 1$$

Method: CHAIN reduces feature and weight gradients in D

$$\|\Delta \boldsymbol{y}_c\|_2^2 \leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \Big(\boxed{\tfrac{(1-p)\psi_c + p\psi_{\min}}{\psi_c}}\Big)^2 - \frac{2(1-p)p\psi_{\min}}{B\psi_c} \boxed{(\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2}$$

$$\|\Delta \boldsymbol{w}_c\|_2^2 \leqslant \lambda_{\max}^2 \|\Delta \boldsymbol{y}_c\|_2^2$$

$\Delta \boldsymbol{y}_c, \Delta \dot{\boldsymbol{y}}_c$: c-th column of gradient for CHAIN input/output $\boldsymbol{Y}, \dot{\boldsymbol{Y}}$. $\lambda_{\max}$: top eigenvalue of $\boldsymbol{A}$. $\check{\boldsymbol{y}}_c$: c-th column of $\check{\boldsymbol{Y}} = \boldsymbol{Y}/\psi$. $\Delta \boldsymbol{w}_c$: c-th column of grad for $\boldsymbol{W}$. $p \in [0, 1]$

$\frac{(1-p)\psi_c + p\psi_{\min}}{\psi_c} \leqslant 1 \quad (\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2 \geqslant 0$

Method: CHAIN reduces feature and weight gradients in D

$$\|\Delta \boldsymbol{y}_c\|_2^2 \leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \left(\boxed{\frac{(1-p)\psi_c + p\psi_{\min}}{\psi_c}}\right)^2 - \frac{2(1-p)p\psi_{\min}}{B\psi_c} \boxed{(\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2}$$

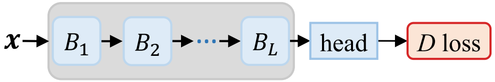
$$\|\Delta \boldsymbol{w}_c\|_2^2 \leqslant \lambda_{\max}^2 \|\Delta \boldsymbol{y}_c\|_2^2$$

$\Delta \boldsymbol{y}_c, \Delta \dot{\boldsymbol{y}}_c$: c-th column of gradient for CHAIN input/output $\boldsymbol{Y}, \dot{\boldsymbol{Y}}$. $\lambda_{\max}$: top eigenvalue of $\boldsymbol{A}$. $\check{\boldsymbol{y}}_c$: c-th column of $\check{\boldsymbol{Y}} = \boldsymbol{Y}/\boldsymbol{\psi}$. $\Delta \boldsymbol{w}_c$: c-th column of grad for $\boldsymbol{W}$. $p \in [0, 1]$

$\frac{(1-p)\psi_c + p\psi_{\min}}{\psi_c} \leqslant 1 \quad (\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2 \geqslant 0 \Longrightarrow$ CHAIN reduces $\|\Delta \boldsymbol{y}_c\|_2$ and $\|\Delta \boldsymbol{w}_c\|_2$.

Pipeline

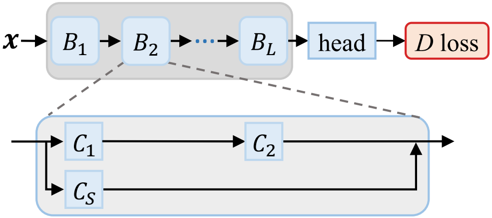
Discriminator with CHAIN



$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator.

Pipeline

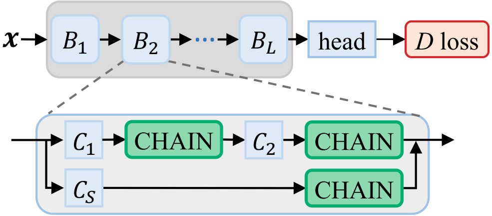
Discriminator with CHAIN



$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator. C_S: Convolution in skip branch.

Pipeline

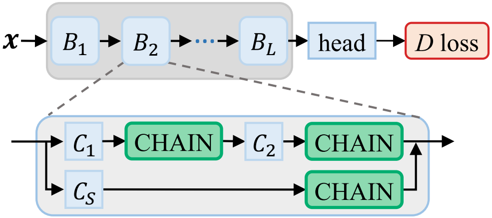
Discriminator with CHAIN



$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator. C_S: Convolution in skip branch.

Pipeline

Discriminator with CHAIN



CHAIN: 0MR & ARMS

0-Mean Regularization loss:

$$\ell^{0\text{MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2$$

Adaptive Root Mean Square normalization:

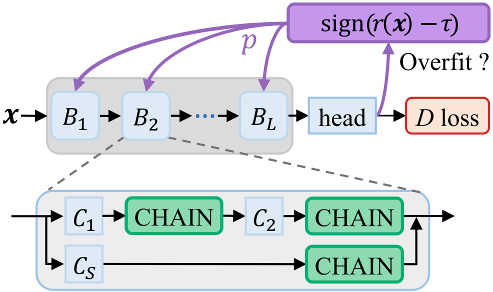
$$\text{ARMS}(\boldsymbol{Y}) = (1 - \boldsymbol{M}) \odot \boldsymbol{Y} + \boldsymbol{M} \odot \frac{\boldsymbol{Y}}{\boldsymbol{\psi}} \cdot \psi_{\text{min}}$$

$\boldsymbol{M} \sim \mathcal{B}(p)$. $\boldsymbol{\mu}, \boldsymbol{\psi}$ are mean and root mean square of feature map $\boldsymbol{Y}$.

$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator. C_S: Convolution in skip branch. $\mathcal{B}$: Bernoulli noise. p: Bernoulli probability and $\ell^{0\text{MR}}$ strength. λ: A hyperparameter.

Pipeline

Discriminator with CHAIN



CHAIN: 0MR & ARMS

0-Mean Regularization loss:

$$\ell^{0\text{MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2$$

Adaptive Root Mean Square normalization:

$$\text{ARMS}(\boldsymbol{Y}) = (1-\boldsymbol{M}) \odot \boldsymbol{Y} + \boldsymbol{M} \odot \frac{\boldsymbol{Y}}{\boldsymbol{\psi}} \cdot \psi_{\min}$$

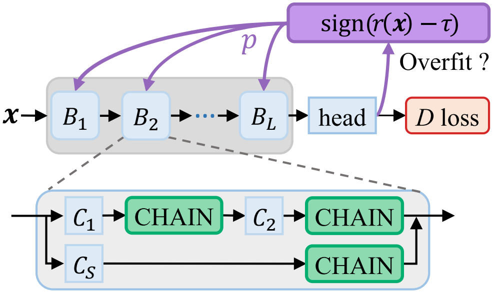
$\boldsymbol{M} \sim \mathcal{B}(p)$. $\boldsymbol{\mu}, \boldsymbol{\psi}$ are mean and root mean square of feature map $\boldsymbol{Y}$.

$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator. C_S: Convolution in skip branch. $\mathcal{B}$: Bernoulli noise. p: Bernoulli probability and $\ell^{0\text{MR}}$ strength. λ: A hyperparameter. τ: A predefined threshold. Δ_p: A small value.

$$\text{Control } p\text{: } r(\boldsymbol{x}) = \mathbb{E}[\text{sign}(D(\boldsymbol{x}))],\ p_{t+1} = p_t + \Delta_p \cdot \text{sign}(r(\boldsymbol{x}) - \tau).$$

Pipeline

Discriminator with CHAIN



CHAIN: 0MR & ARMS

0-Mean Regularization loss:

$$\ell^{0\text{MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2$$

Adaptive Root Mean Square normalization:

$$\text{ARMS}(\boldsymbol{Y}) = (1 - \boldsymbol{M}) \odot \boldsymbol{Y} + \boldsymbol{M} \odot \frac{\boldsymbol{Y}}{\boldsymbol{\psi}} \cdot \psi_{\min}$$

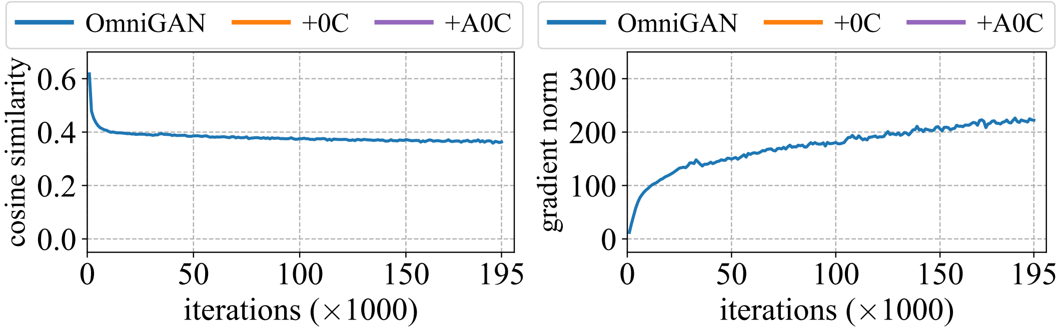
$\boldsymbol{M} \sim \mathcal{B}(p)$. $\boldsymbol{\mu}, \boldsymbol{\psi}$ are mean and root mean square of feature map $\boldsymbol{Y}$.

$\boldsymbol{x}$: Real image. B_l: l-th block. D: Discriminator. C_S: Convolution in skip branch. $\mathcal{B}$: Bernoulli noise. p: Bernoulli probability and $\ell^{0\text{MR}}$ strength. λ: A hyperparameter. τ: A predefined threshold. Δ_p: A small value.

Control p: $r(\boldsymbol{x}) = \mathbb{E}[\text{sign}(D(\boldsymbol{x}))]$, $p_{t+1} = p_t + \Delta_p \cdot \text{sign}(r(\boldsymbol{x}) - \tau)$.

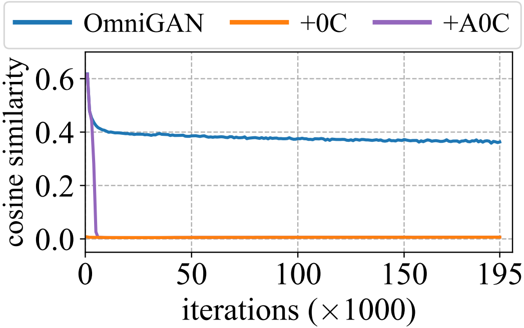
CHAIN is applied separately to real and fake data batches.

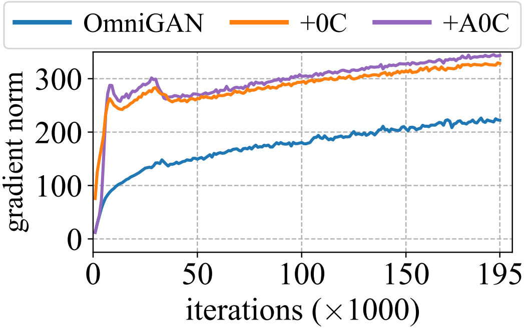
Experiments: Analysis of gradient issue in centering



0C: centering. A0C: adaptive centering. On 10% CIFAR-10.

Experiments: Analysis of gradient issue in centering

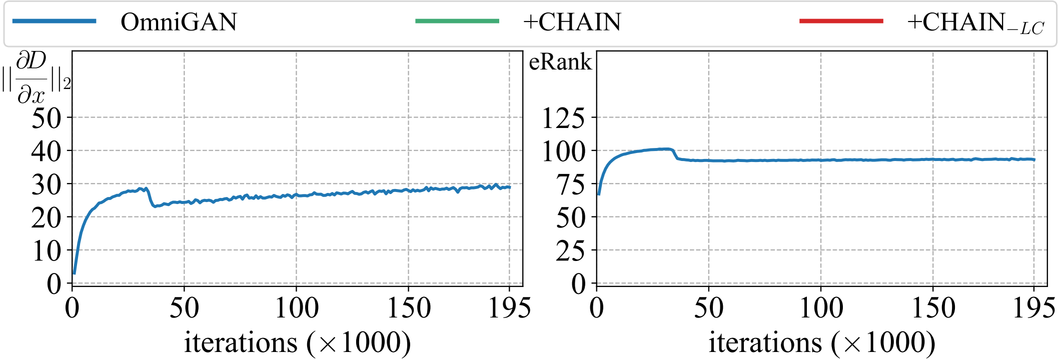




0C: centering. A0C: adaptive centering. On 10% CIFAR-10.

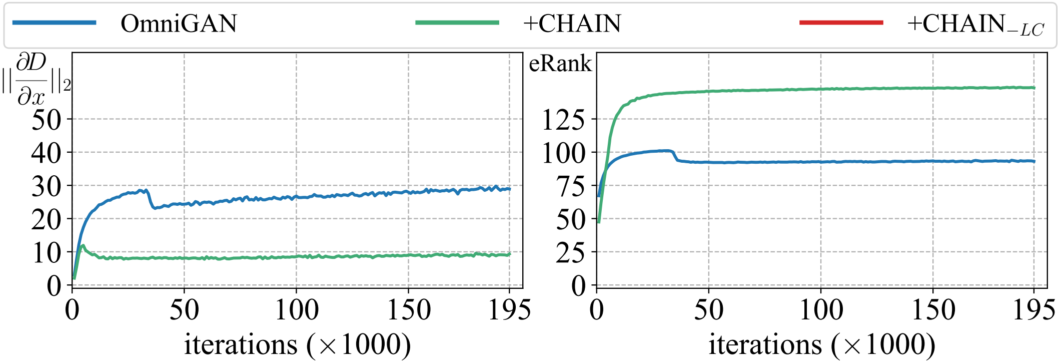
Centering reduces similarity and raises gradient.

Experiments: Analysis of gradient issue in scaling



−LC: without Lipschitz constraint. On 10% CIFAR-10.

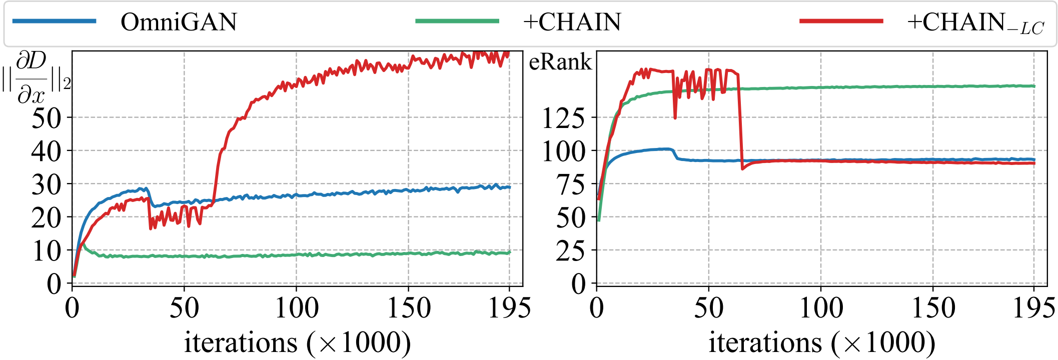
Experiments: Analysis of gradient issue in scaling



−LC: without Lipschitz constraint. On 10% CIFAR-10.

CHAIN reduces latent gradients

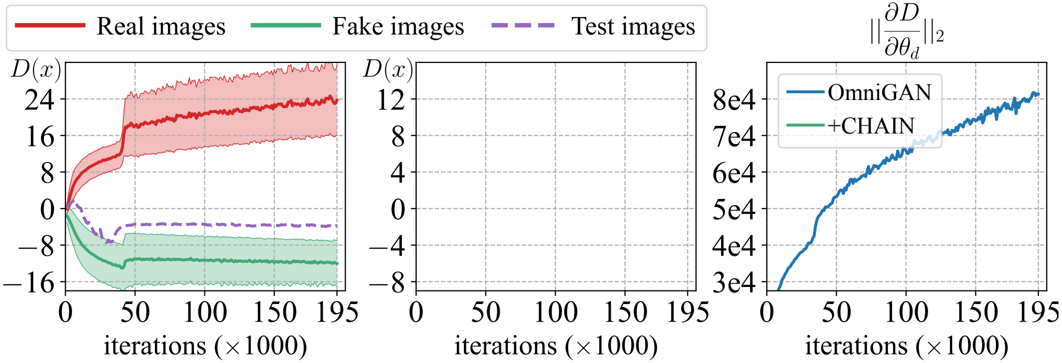
Experiments: Analysis of gradient issue in scaling



$-$LC: without Lipschitz constraint. On 10% CIFAR-10.

CHAIN reduces latent gradients while removing LC raises gradient and impairs feature eRank.

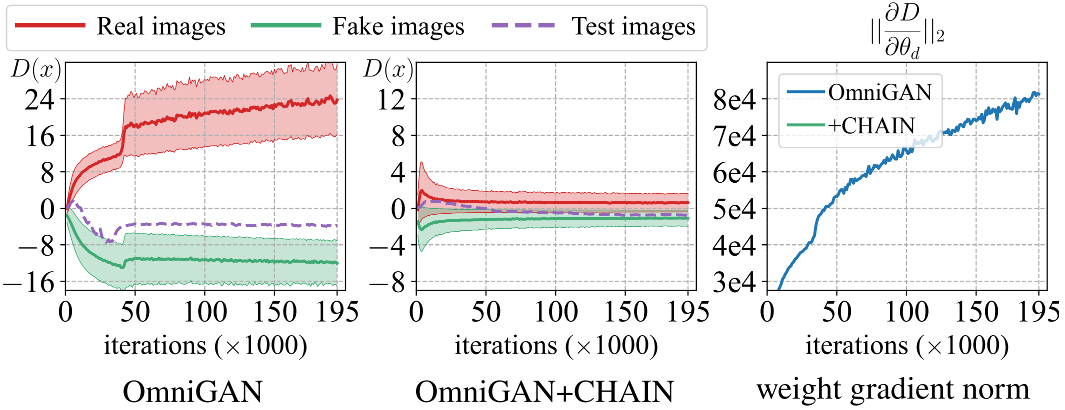
Experiments: Analysis of generalization of CHAIN



OmniGAN | OmniGAN+CHAIN | weight gradient norm

$D(x)$: discriminator output. On 10% CIFAR-10.

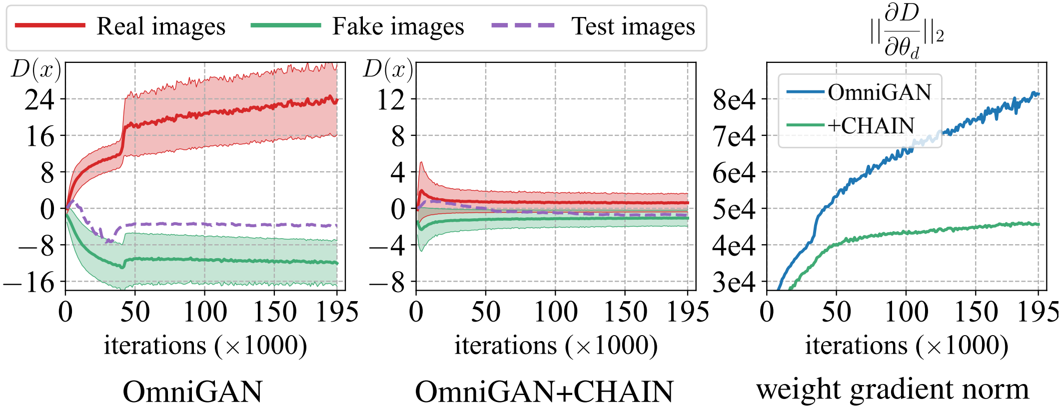
Experiments: Analysis of generalization of CHAIN



$D(x)$: discriminator output. On 10% CIFAR-10.

CHAIN reduces discrepancies among real/fake/testing data

Experiments: Analysis of generalization of CHAIN



$D(x)$: discriminator output. On 10% CIFAR-10.

CHAIN reduces discrepancies among real/fake/testing data and D's weight gradients.

Experiments: Comparison with state of the arts

Method	CIFAR-10						CIFAR-100					
	10% data		20% data		100% data		10% data		20% data		100% data	
	IS↑	tFID↓	IS↑	tFID↓	IS↑	tFID↓	IS↑	tFID↓	IS↑	tFID↓	IS↑	tFID↓
BigGAN	8.24	31.45	8.74	16.20	9.21	5.48	7.58	50.79	9.94	25.83	11.02	7.86
+CHAIN	**8.63**	**12.02**	**8.98**	**8.15**	**9.49**	**4.18**	**10.04**	**13.13**	**10.15**	**11.58**	**11.16**	**6.04**
LeCam+DA	8.81	12.64	9.01	8.53	9.45	4.32	9.17	22.75	10.12	15.96	11.25	6.45
+CHAIN	**8.96**	**8.54**	**9.27**	**5.92**	**9.52**	**3.51**	**10.11**	**12.69**	**10.62**	**9.02**	**11.37**	**5.26**
OmniGAN+ADA	7.86	40.05	9.41	27.04	10.24	4.95	8.95	44.65	12.07	13.54	13.07	6.12
+CHAIN	**10.10**	**6.22**	**10.26**	**3.98**	**10.31**	**2.22**	**12.70**	**9.49**	**12.98**	**7.02**	**13.98**	**4.02**

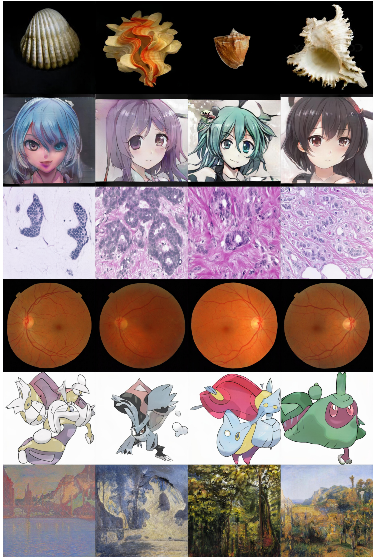
Method (FID↓)	Shells	Skulls	AnimeFace	BreCaHAD	MessidorSet1	Pokemon	ArtPainting
	64 imgs	97 imgs	120 imgs	162 imgs	400 imgs	833 imgs	1000 imgs
FastGAN	138.50	97.87	54.05	63.83	38.33	45.70	43.21
FreGAN	123.75	84.58	49.09	**57.87**	34.61	39.09	43.14
FastGAN−D_{big}	171.35	165.64	76.02	68.63	37.38	53.48	43.04
+CHAIN	**78.62**	**82.47**	**46.27**	58.98	**28.76**	**31.94**	**38.83**

Experiments: Comparison with state of the arts

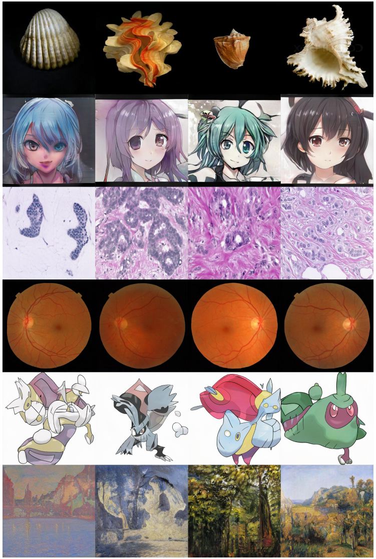
Method	2.5% ImageNet			5% ImageNet			10% ImageNet		
	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓
BigGAN	8.61	101.62	100.09	6.27	90.32	88.01	12.44	50.75	49.84
+CHAIN	**14.68**	**30.66**	**29.32**	**17.34**	**21.13**	**19.95**	**20.45**	**14.70**	**13.84**
ADA	7.93	67.84	66.55	11.56	47.56	46.25	14.82	31.75	30.68
+CHAIN	**16.57**	**23.01**	**21.90**	**19.15**	**16.14**	**15.17**	**22.04**	**12.91**	**12.17**

Method (FID↓)	100-shot			Animal Face	
	Obama	GrumpyCat	Panda	Cat	Dog
StyleGAN2	80.20	48.90	34.27	71.71	131.90
+CHAIN	**28.72**	**27.21**	**9.51**	**38.93**	**53.27**
AdvAug	52.86	31.02	14.75	47.40	68.28
DA	46.87	27.08	12.06	42.44	58.85
InsGen	32.42	22.01	9.85	33.01	44.93
FakeCLR	26.95	19.56	8.42	26.34	42.02
KDDLGAN	29.38	19.65	8.41	31.89	50.22
AugSelfGAN	26.00	19.81	8.36	30.53	48.19
DA+CHAIN	22.87	**17.57**	**6.93**	**19.58**	**30.88**

Generated Images and Conclusions



Generated Images and Conclusions



Conclusions:

- CHAIN reduces real-fake discrepancy and discriminator weight gradients, improving generalization.
- CHAIN lowers latent feature gradients in discriminator, enhancing stability.