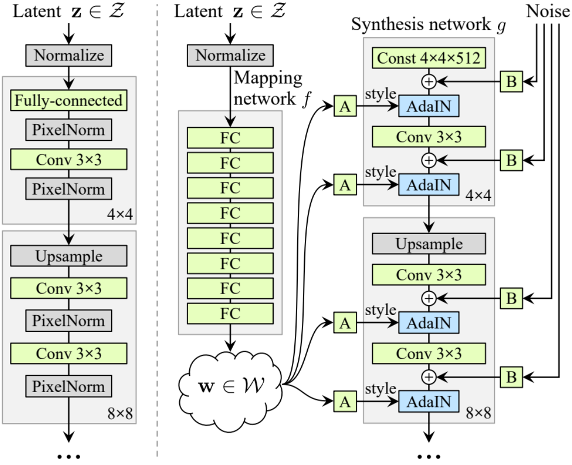
Analyzing and Improving the Image Quality of StyleGAN

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila
NVIDIA

Present by Miaoyun Zhao
Jan 31, 2020

Table of Contents

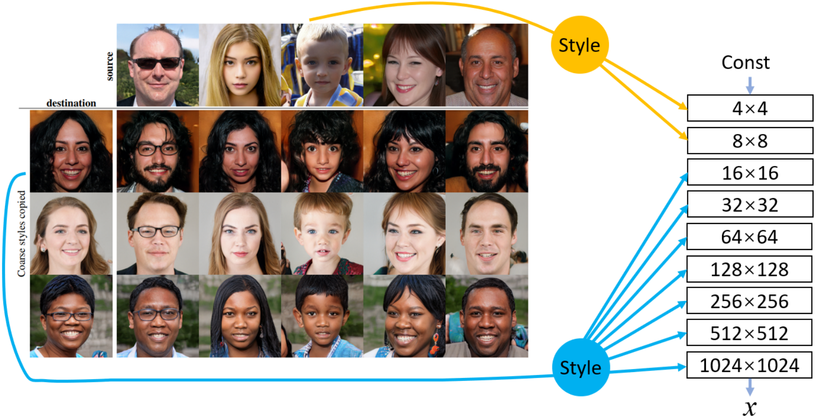
Style-GAN [1] generator architecture



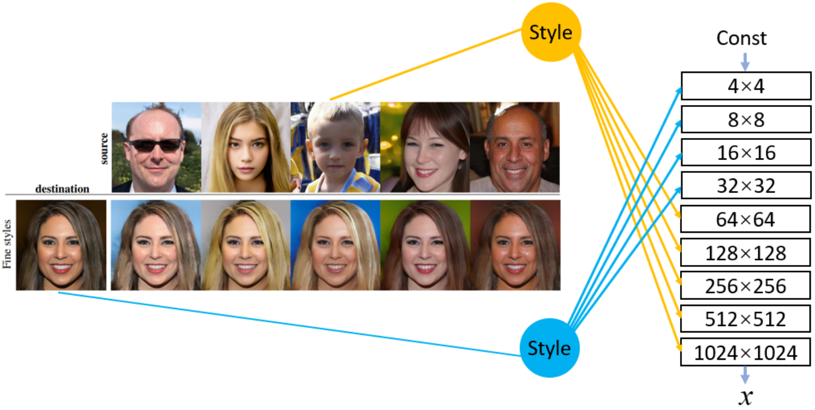
(a) Traditional (b) Style-based generator

[1] A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019.

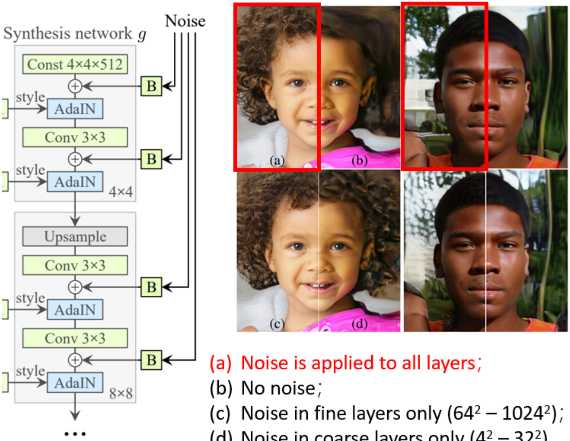
Style-GAN: Latent code controls multi-scale attributes



Style-GAN: Latent code controls multi-scale attributes



Style-GAN: noise input controls finer details



(a) Noise is applied to all layers;
(b) No noise;
(c) Noise in fine layers only ($64^2 - 1024^2$);
(d) Noise in coarse layers only ($4^2 - 32^2$).

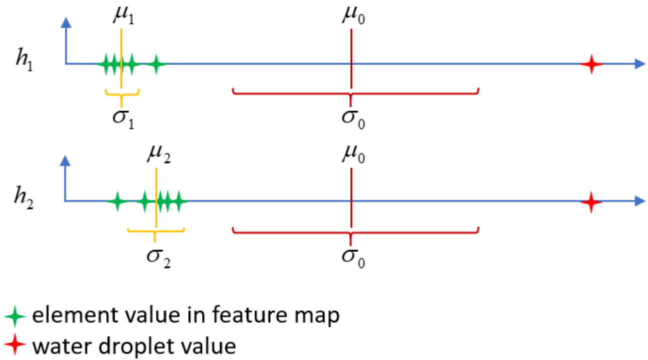
Table of Contents

The generator fully relies on the existence of water droplets

Feature map 64^2 | Feature map 128^2 | Feature map 256^2 | Feature map 512^2 | Generated image

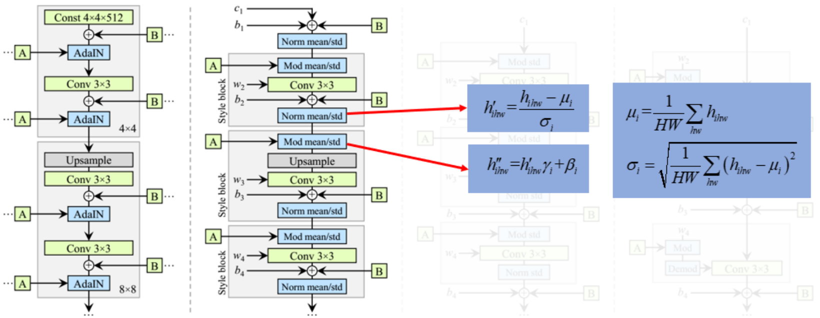
The generator fully relies on the existence of water droplets

Water droplets can protect the relative relationship among feature maps when past instance normalization.



Relax the normalization while retaining the scale-specific effects of the styles

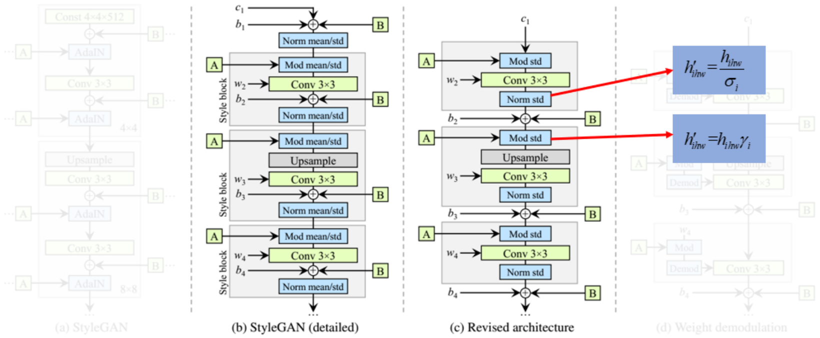
Decompose the AdaIN operation



(a) StyleGAN (b) StyleGAN (detailed) (c) Revised architecture (d) Weight demodulation

Relax the normalization while retaining the scale-specific effects of the styles

Remove the operation on mean



Relax the normalization while retaining the scale-specific effects of the styles

Baked the entire style block as filter adjusting

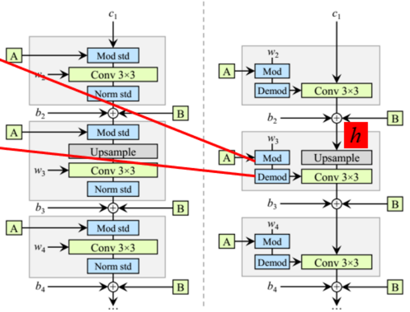
$$h'_{ilhw} = h_{ilhw}\gamma_i \rightarrow \mathbf{W}'_{ijk_1k_2} = s_i \mathbf{W}_{ijk_1k_2}$$

$$h'_{jlhw} = \frac{h_{jlhw}}{\sigma_j} \rightarrow \sigma_j = \sqrt{\sum_{ik_1k_2} {\mathbf{W}'_{ijk_1k_2}}^2}, \quad \mathbf{W}''_{ijk_1k_2} = \frac{\mathbf{W}'_{ijk_1k_2}}{\sigma_j}$$

Suppose $h_{ilhw} \sim \mathbb{N}(0,1)$,

$$h' = \mathrm{cov}(h, \mathbf{W}), h'_{jlhw} = \sum_{ik_1k_2} h_{ik_1k_2} \mathbf{W}_{ijk_1k_2}$$

We have $h'_{jlhw} \sim \mathbb{N}\left(0, \sum_{ik_1k_2} {\mathbf{W}_{ijk_1k_2}}^2\right)$



(c) Revised architecture (d) Weight demodulation

Relax the normalization while retaining the scale-specific effects of the styles

The relaxed normalization can improve the performance.

Configuration	FFHQ, 1024×1024 FID	Path length	Precision	Recall	LSUN Car, 512×384 FID	Path length	Precision	Recall
A Baseline StyleGAN [24]	4.40	195.9	**0.721**	0.399	3.27	1484.5	**0.701**	0.435
B + Weight demodulation	4.39	173.8	0.702	0.425	3.04	862.4	0.685	0.488
C + Lazy regularization	4.38	167.2	0.719	0.427	2.83	981.6	0.688	0.493
D + Path length regularization	4.34	139.2	0.715	0.418	3.43	651.2	0.697	0.452
E + No growing, new G & D arch.	3.31	**116.7**	0.705	0.449	3.19	471.2	0.690	0.454
F + Large networks	**2.84**	129.4	0.689	**0.492**	**2.32**	**415.5**	0.678	**0.514**

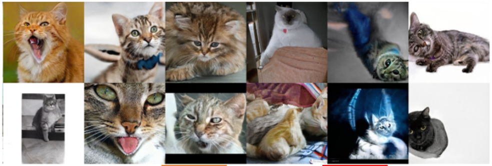
Perceptual path length (PPL), a metric introduced for quantifying the smoothness of the mapping from a latent space to the output image.

Table of Contents

Image quality and generator smoothness

The smoothness of the generator is very important for high quality image generation.

Model 1: FID = 8.53, P = 0.64, R = 0.28, PPL = 924



Model 2: FID = 8.53, P = 0.62, R = 0.29, PPL = 387

Perceptual path length (PPL), a metric introduced for quantifying the smoothness of the mapping from a latent space to the output image. A smooth mapping will have a small PPL.

Smooth the generator via Path length regularization

$$z \longrightarrow \boxed{\text{MLP}} \longrightarrow w \longrightarrow \boxed{g} \longrightarrow x = g(w)$$

$$\mathcal{L}_{pl} = \mathbb{E}_{\boldsymbol{w},\boldsymbol{y}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}(\|\mathbf{J}_{\boldsymbol{w}}^T\boldsymbol{y}\|_2 - a)^2 \tag{1}$$

where

- Intermediate latent code $\boldsymbol{w} \in \mathbb{R}^L, \boldsymbol{w} \sim f(\boldsymbol{z})$;
- $\boldsymbol{y}$ is random image with normally distributed pixel intensities.
- Jacobian matrix $\mathbf{J}_{\boldsymbol{w}} = \frac{\partial g(\boldsymbol{w})}{\partial \boldsymbol{w}}$;
- a is dynamically set as $\mathbb{E}_{\boldsymbol{w},\boldsymbol{y}}(\|\mathbf{J}_{\boldsymbol{w}}^T\boldsymbol{y}\|_2)^2$.

$\mathcal{L}_{pl}$ is minimized when $\mathbf{J}_{\boldsymbol{w}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, $\boldsymbol{\Sigma} = \frac{a}{\sqrt{L}}\mathbf{I}$, which makes the generator an isometrical mapping.
"A consequence of isometry is that straight line segments in the latent space are mapped to geodesics, or shortest paths, on the image manifold."

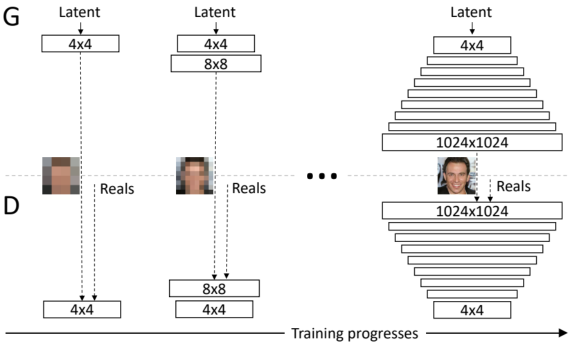
Path length regularization

With path length regularization, the PPL decreased greatly.

Configuration	FFHQ, 1024×1024 FID	Path length	Precision	Recall	LSUN Car, 512×384 FID	Path length	Precision	Recall
A Baseline StyleGAN [24]	4.40	195.9	**0.721**	0.399	3.27	1484.5	**0.701**	0.435
B + Weight demodulation	4.39	173.8	0.702	0.425	3.04	862.4	0.685	0.488
C + Lazy regularization	4.38	167.2	0.719	0.427	2.83	981.6	0.688	0.493
D + Path length regularization	4.34	139.2	0.715	0.418	3.43	651.2	0.697	0.452
E + No growing, new G & D arch.	3.31	**116.7**	0.705	0.449	3.19	471.2	0.690	0.454
F + Large networks	**2.84**	129.4	0.689	**0.492**	**2.32**	**415.5**	0.678	**0.514**

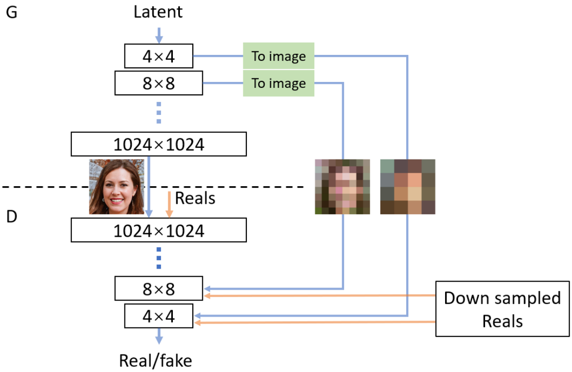
Table of Contents

Progressive GAN based training is complex and slow



Alternative network architectures

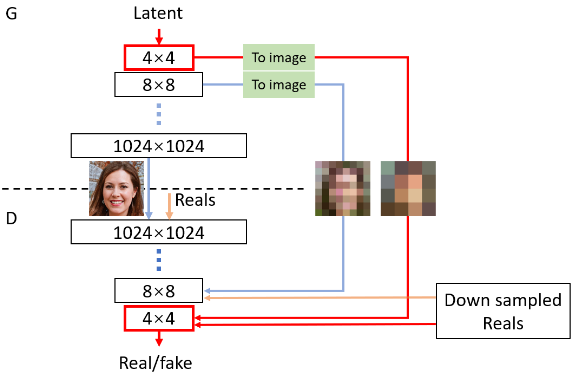
MSG-GAN [2]: Complete the progressive training process Automatically.



[2] MSG-GAN: Multi-Scale Gradients for Generative Adversarial. CVPR, 2019.

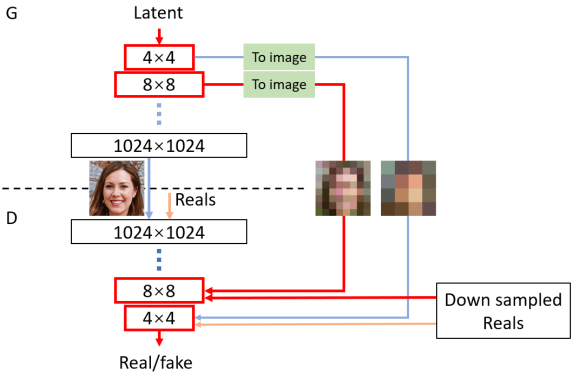
Alternative network architectures

MSG-GAN: Complete the progressive training process Automatically.



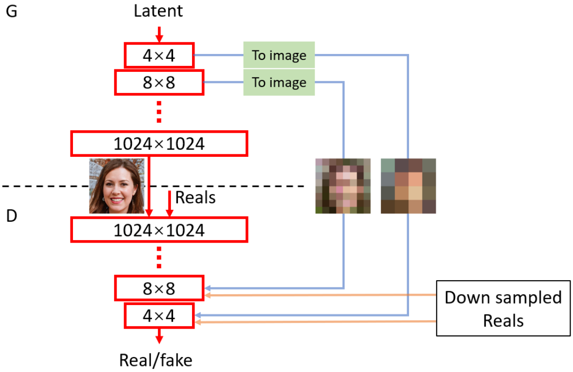
Alternative network architectures

MSG-GAN: Complete the progressive training process Automatically.

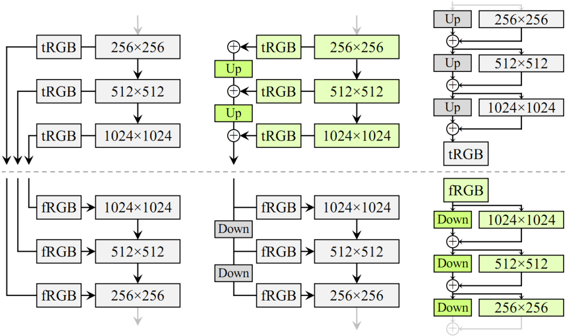


Alternative network architectures

MSG-GAN: Complete the progressive training process Automatically.



Simplify the MSG-GAN



(a) MSG-GAN (b) Input/output skips (c) Residual nets

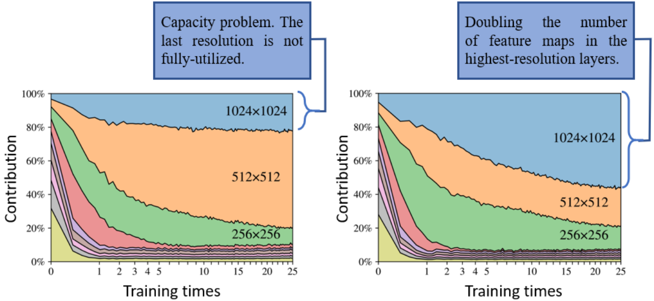
Simplify the MSG-GAN

By training all the resolutions simultaneously, the performance improved obviously.

	Configuration	FFHQ, 1024×1024				LSUN Car, 512×384			
		FID	Path length	Precision	Recall	FID	Path length	Precision	Recall
A	Baseline StyleGAN [24]	4.40	195.9	**0.721**	0.399	3.27	1484.5	**0.701**	0.435
B	+ Weight demodulation	4.39	173.8	0.702	0.425	3.04	862.4	0.685	0.488
C	+ Lazy regularization	4.38	167.2	0.719	0.427	2.83	981.6	0.688	0.493
D	+ Path length regularization	4.34	139.2	0.715	0.418	3.43	651.2	0.697	0.452
E	+ No growing, new G & D arch.	3.31	**116.7**	0.705	0.449	3.19	471.2	0.690	0.454
F	+ Large networks	**2.84**	129.4	0.689	**0.492**	**2.32**	**415.5**	0.678	**0.514**

Simplify the MSG-GAN

Contribution of each resolution to the output of the generator as a function of training time.



- Behaves similar to progressive growing;
- However, the last resolution is not fully utilized.

Simplify the MSG-GAN

Doubling the number of feature maps in the high resolution layers can provide additional benefits.

Configuration	FFHQ, 1024×1024 FID	Path length	Precision	Recall	LSUN Car, 512×384 FID	Path length	Precision	Recall
A Baseline StyleGAN [24]	4.40	195.9	**0.721**	0.399	3.27	1484.5	**0.701**	0.435
B + Weight demodulation	4.39	173.8	0.702	0.425	3.04	862.4	0.685	0.488
C + Lazy regularization	4.38	167.2	0.719	0.427	2.83	981.6	0.688	0.493
D + Path length regularization	4.34	139.2	0.715	0.418	3.43	651.2	0.697	0.452
E + No growing, new G & D arch.	3.31	**116.7**	0.705	0.449	3.19	471.2	0.690	0.454
F + Large networks	**2.84**	129.4	0.689	**0.492**	**2.32**	**415.5**	0.678	**0.514**

Conclusion

- The style control process is actually equal to control the filters in the generator;
- The progressive process can be automatically implemented without changing the model;
- A technique to smooth a network: Path length regularization.

[1] Animesh Karnewar and Raghu Sesha Iyengar.
Msg-gan: Multi-scale gradients gan for more stable and synchronized multi-scale image synthesis.
arXiv preprint arXiv:1903.06048, 2019.

[2] T. Karras, S. Laine, and T. Aila.
A style-based generator architecture for generative adversarial networks.
In *CVPR*, June 2019.