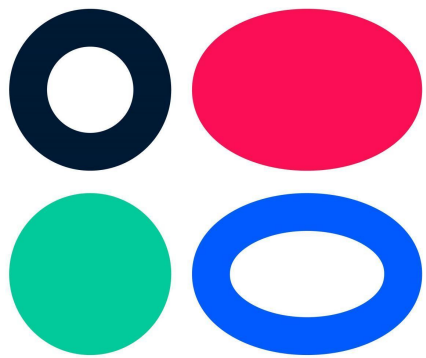
Deep Generative Models
Lecture 11

Roman Isachenko



Ozon Masters

Spring, 2021

Spectral Normalization GAN

How else could we enforce Lipschitzness?

Fact 1

$$\|\mathbf{g}\|_L = \sup_{\mathbf{x}} \sigma(\nabla \mathbf{g}(\mathbf{x}))$$

Here $\sigma(\mathbf{A})$ – spectral norm of matrix $\mathbf{A}$.

$$\sigma(\mathbf{A}) = \max_{\mathbf{h} \neq 0} \frac{\|\mathbf{A}\mathbf{h}\|_2}{\|\mathbf{h}\|_2} = \max_{\|\mathbf{h}\|_2 \leq 1} \|\mathbf{A}\mathbf{h}\|_2 = \lambda_{\max}(\mathbf{A}),$$

where $\lambda_{\max}(\mathbf{A})$ is the largest singular value of $\mathbf{A}$.

Fact 2

$$\|\mathbf{g}_1 \circ \mathbf{g}_2\|_L \leq \|\mathbf{g}_1\|_L \cdot \|\mathbf{g}_2\|_L$$

Miyato T. et al. Spectral Normalization for Generative Adversarial Networks, 2018

Spectral Normalization GAN

Let consider the critic $f(\mathbf{x}, \phi)$ of the following form:

$$f(\mathbf{x}, \phi) = \mathbf{W}_{K+1} a_K(\mathbf{W}_K a_{K-1}(\dots a_1(\mathbf{W}_1 \mathbf{x}) \dots)).$$

This feedforward network is a superposition of simple functions.

- a_k is a pointwise nonlinearities. We assume that $\|a_k\|_L = 1$ (it holds for ReLU).
- $\mathbf{g}(\mathbf{h}) = \mathbf{W}\mathbf{h}$ is a linear transformation.

$$\|\mathbf{g}\|_L = \sup_{\mathbf{x}} \sigma(\nabla \mathbf{g}(\mathbf{x})) = \sigma(\mathbf{W}).$$

Critic spectral norm

$$\|f\|_L \leq \prod_{k=1}^{K+1} \sigma(\mathbf{W}_k).$$

If we replace the weights in the critic by $\mathbf{W}_k^{SN} = \mathbf{W}_k / \sigma(\mathbf{W}_k)$, we will get $\|f\|_L \leq 1$.

Miyato T. et al. Spectral Normalization for Generative Adversarial Networks, 2018

Spectral Normalization GAN

If we apply singular value decomposition to compute the $\sigma(\mathbf{W})$ at each round of the algorithm, the algorithm becomes computationally heavy.

Power iteration

- $\mathbf{u}$ – random vector.
- repeat

$$\mathbf{v} = \frac{\mathbf{W}^T\mathbf{u}}{\|\mathbf{W}^T\mathbf{u}\|}, \quad \mathbf{u} = \frac{\mathbf{W}\mathbf{v}}{\|\mathbf{W}\mathbf{v}\|}$$

- approximate the spectral norm

$$\sigma(\mathbf{W}) \approx \mathbf{u}^T\mathbf{W}\mathbf{v}$$

Miyato T. et al. Spectral Normalization for Generative Adversarial Networks, 2018

Spectral Normalization GAN

Algorithm 1 SGD with spectral normalization

- Initialize $\tilde{\boldsymbol{u}}_l \in \mathcal{R}^{d_l}$ for $l = 1, \ldots, L$ with a random vector (sampled from isotropic distribution).
- For each update and each layer l:
  1. Apply power iteration method to a unnormalized weight W^l:

$$\tilde{\boldsymbol{v}}_l \leftarrow (W^l)^{\mathrm{T}} \tilde{\boldsymbol{u}}_l / \|(W^l)^{\mathrm{T}} \tilde{\boldsymbol{u}}_l\|_2 \tag{20}$$

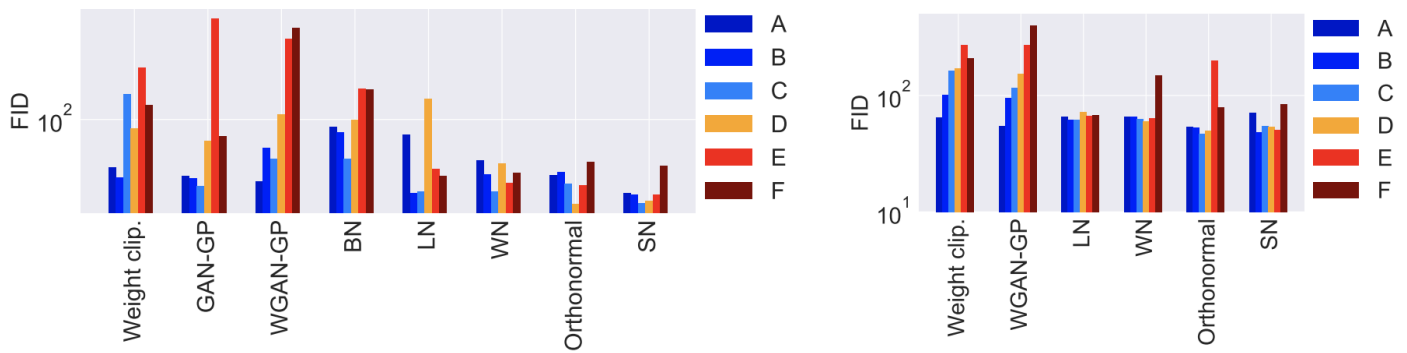
$$\tilde{\boldsymbol{u}}_l \leftarrow W^l \tilde{\boldsymbol{v}}_l / \|W^l \tilde{\boldsymbol{v}}_l\|_2 \tag{21}$$

  2. Calculate $\bar{W}_{\mathrm{SN}}$ with the spectral norm:

$$\bar{W}^l_{\mathrm{SN}}(W^l) = W^l / \sigma(W^l), \text{ where } \sigma(W^l) = \tilde{\boldsymbol{u}}_l^{\mathrm{T}} W^l \tilde{\boldsymbol{v}}_l \tag{22}$$

  3. Update W^l with SGD on mini-batch dataset $\mathcal{D}_M$ with a learning rate α:

$$W^l \leftarrow W^l - \alpha \nabla_{W^l} \ell(\bar{W}^l_{\mathrm{SN}}(W^l), \mathcal{D}_M) \tag{23}$$



(a) CIFAR-10 (b) STL-10

Miyato T. et al. Spectral Normalization for Generative Adversarial Networks, 2018

Divergences

What do we have?

- Forward KL divergence in maximum likelihood estimation
- Reverse KL in variational inference
- JS divergence in vanilla gan
- Wasserstein distance in WGAN

Divergence minimization

$$\min_{p} D(\pi||p)$$

What is a divergence?

Let $\mathcal{S}$ be the set of all possible probability distributions. Then $D : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a divergence if

- $D(\pi||p) \geq 0$ for all $\pi, p \in \mathcal{S}$;
- $D(\pi||p) = 0$ if and only if $\pi \equiv p$.

f-divergence family

f-divergence

$$D_f(\pi||p) = \mathbb{E}_{p(\mathbf{x})} f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) = \int p(\mathbf{x}) f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) d\mathbf{x}.$$

Here $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex, lower-semicontinuous function satisfying $f(1) = 0$.

Fenchel conjugate

$$f^*(t) = \sup_{u \in \text{dom}_f} \left(ut - f(u)\right), \quad f^{**} = f, \quad f(u) = \sup_{t \in \text{dom}_{f^*}} \left(ut - f^*(t)\right)$$

Name	$D_f(P\|Q)$	Generator $f(u)$	$T^*(x)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, \mathrm{d}x$	$u \log u$	$1 + \log \frac{p(x)}{q(x)}$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$	$-\log u$	$-\frac{q(x)}{p(x)}$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} \, \mathrm{d}x$	$(u-1)^2$	$2(\frac{p(x)}{q(x)} - 1)$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 \, \mathrm{d}x$	$\left(\sqrt{u}-1\right)^2$	$(\sqrt{\frac{p(x)}{q(x)}} - 1) \cdot \sqrt{\frac{q(x)}{p(x)}}$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x$	$-(u+1) \log \frac{1+u}{2} + u \log u$	$\log \frac{2p(x)}{p(x)+q(x)}$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x - \log(4)$	$u \log u - (u+1) \log(u+1)$	$\log \frac{p(x)}{p(x)+q(x)}$

Nowozin S., Cseke B., Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, 2016

f-divergence family

Variational divergence estimation

$$
\begin{aligned}
D_f(\pi||p) &= \int p(\mathbf{x}) f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) d\mathbf{x} \\
&= \int p(\mathbf{x}) \sup_{t \in \mathrm{dom}_{f^*}} \left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})} t - f^*(t)\right) d\mathbf{x} \\
&= \int \sup_{t \in \mathrm{dom}_{f^*}} \left(\pi(\mathbf{x}) t - p(\mathbf{x}) f^*(t)\right) d\mathbf{x} \\
&\geq \sup_{T \in \mathcal{T}} \int \left(\pi(\mathbf{x}) T(\mathbf{x}) - p(\mathbf{x}) f^*(T(\mathbf{x}))\right) d\mathbf{x} \\
&= \sup_{T \in \mathcal{T}} \left[\mathbb{E}_\pi T(\mathbf{x}) - \mathbb{E}_p f^*(T(\mathbf{x}))\right]
\end{aligned}
$$

Here $\mathcal{T} : \mathcal{X} \to \mathbb{R}$ is an arbitrary class of functions.

The lower bound is tight for $T^*(\mathbf{x}) = f'\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right)$.

Nowozin S., Cseke B., Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, 2016

f-divergence family

Variational divergence estimation

$$D_f(\pi||p) \geq \sup_{T \in \mathcal{T}} [\mathbb{E}_\pi T(\mathbf{x}) - \mathbb{E}_p f^*(T(\mathbf{x}))]$$

(a) GAN (b) KL (c) Squared Hellinger

Nowozin S., Cseke B., Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, 2016

Evaluation of likelihood-free models

How to evaluate generative models?

Likelihood-based models

- Split data to train/val/test.
- Fit model on the train part.
- Tune hyperparameters on the validation part.
- Evaluate generalization by reporting likelihoods on the test set.

Not all models have tractable likelihoods

- VAE: compare ELBO values.
- GAN: ???

Evaluation of likelihood-free models

Let take some pretrained image classification model to get the conditional label distribution $p(y|\mathbf{x})$ (e.g. ImageNet classifier).

What do we want from samples?

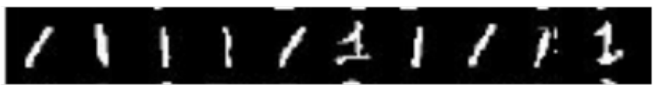
- **Sharpness**

Low sharpness

High sharpness

The conditional distribution $p(y|\mathbf{x})$ should have low entropy (each image $\mathbf{x}$ should have distinctly recognizable object).

- **Diversity**



Low diversity

High diversity

The marginal distribution $p(y) = \int p(y|\mathbf{x})p(\mathbf{x})d\mathbf{x}$ should have high entropy (there should be as many classes generated as possible).

---

image credit: https://deepgenerativemodels.github.io

Evaluation of likelihood-free models

What do we want from samples?

- **Sharpness.** The conditional distribution $p(y|\mathbf{x})$ should have low entropy (each image $\mathbf{x}$ should have distinctly recognizable object).
- **Diversity.** The marginal distribution $p(y) = \int p(y|\mathbf{x})p(\mathbf{x})d\mathbf{x}$ should have high entropy (there should be as many classes generated as possible).

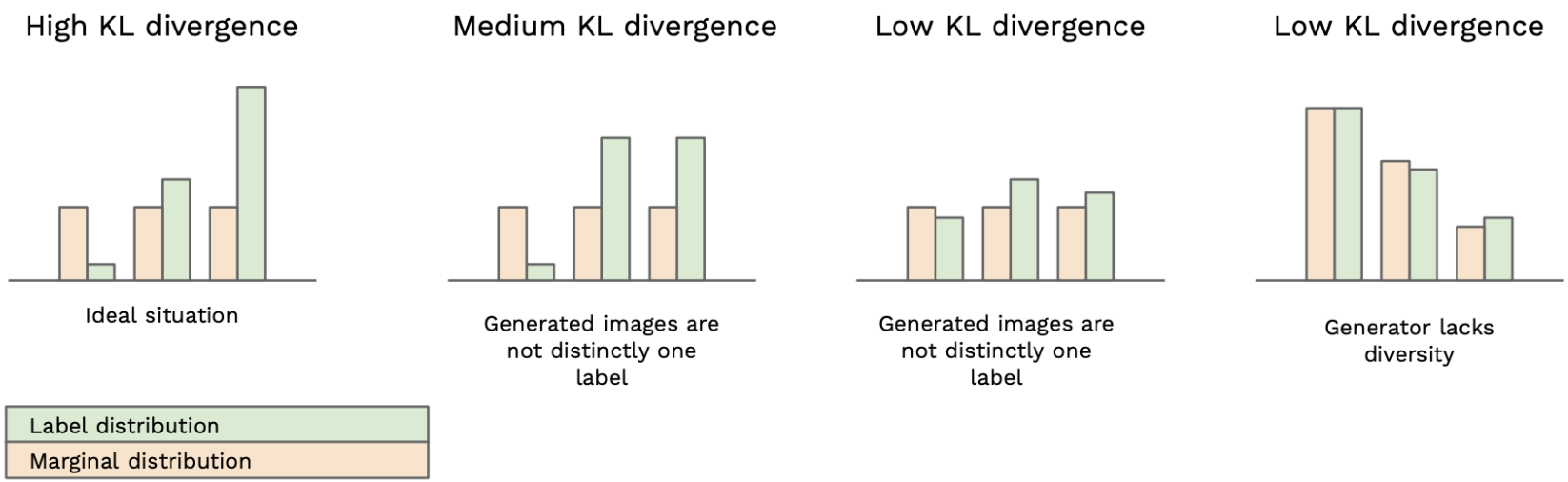


image credit: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

Evaluation of likelihood-free models

What do we want from samples?

- Sharpness $\Rightarrow$ low $H(y|\mathbf{x}) = -\sum_y \int_{\mathbf{x}} p(y, \mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}$.
- Diversity $\Rightarrow$ high $H(y) = -\sum_y p(y) \log p(y)$.

Inception Score

$$
\begin{aligned}
IS &= \exp(H(y) - H(y|\mathbf{x})) \\
&= \exp\left(-\sum_y p(y) \log p(y) + \sum_y \int_{\mathbf{x}} p(y, \mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}\right) \\
&= \exp\left(\sum_y \int_{\mathbf{x}} p(y, \mathbf{x}) \log \frac{p(y|\mathbf{x})}{p(y)} d\mathbf{x}\right) \\
&= \exp\left(\mathbb{E}_{\mathbf{x}} \sum_y p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{p(y)}\right) = \exp\left(\mathbb{E}_{\mathbf{x}} KL(p(y|\mathbf{x})||p(y))\right)
\end{aligned}
$$

Salimans T. et al. Improved Techniques for Training GANs, 2016

Evaluation of likelihood-free models

Inception Score

$$IS = \exp\left(\mathbb{E}_{\mathbf{x}} KL(p(y|\mathbf{x})||p(y))\right)$$

IS limitations

- Inception score depends on the quality of the pretrained classifier $p(y|\mathbf{x})$.
- If generator produces images with a different set of labels from the classifier training set, IS will be low.
- If the generator produces one image per class, the IS will be perfect (there is no measure of intra-class diversity).
- IS only require samples from the generator and do not take into account the desired data distribution $\pi(\mathbf{x})$ directly (only implicitly via a classifier).

Barratt S., Sharma R. A Note on the Inception Score, 2018

Evaluation of likelihood-free models

Theorem

If $\pi(\mathbf{x})$ and $p(\mathbf{x}|\boldsymbol{\theta})$ has moment generation functions then

$$\pi(\mathbf{x}) = p(\mathbf{x}|\boldsymbol{\theta}) \Leftrightarrow \mathbb{E}_{\pi}\mathbf{x}^k = \mathbb{E}_{p}\mathbf{x}^k, \quad \forall k \geq 1.$$

This is intractable to calculate all moments.

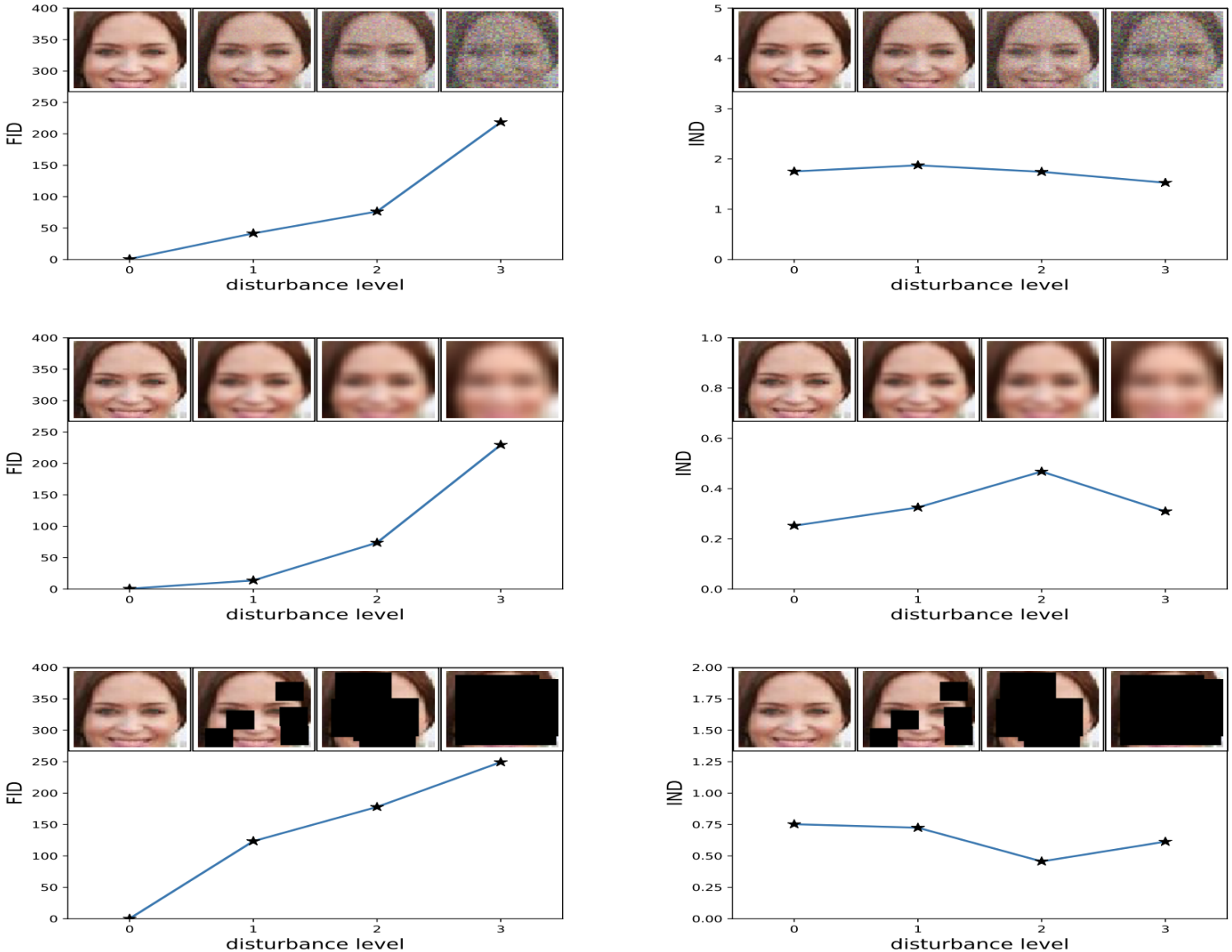
Frechet Inception Distance

$$D^2(\pi, p) = \|\mathbf{m}_{\pi} - \mathbf{m}_{p}\|_2^2 + \mathrm{Tr}\left(\mathbf{C}_{\pi} + \mathbf{C}_{p} - 2\sqrt{\mathbf{C}_{\pi}\mathbf{C}_{p}}\right)$$

- $\mathbf{m}_{\pi}$, $\mathbf{C}_{\pi}$ are mean vector and covariance matrix of feature representations for real samples from $\pi(\mathbf{x})$
- $\mathbf{m}_{p}$, $\mathbf{C}_{p}$ are mean vector and covariance matrix of feature representations for generated samples from $p(\mathbf{x}|\boldsymbol{\theta})$.
- Representations are outputs of intermediate layer from pretrained classification model.

Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017

Evaluation of likelihood-free models



Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017

Evaluation of likelihood-free models

Frechet Inception Distance

$$D^2(\pi, p) = \|\mathbf{m}_\pi - \mathbf{m}_p\|_2^2 + \mathrm{Tr}\left(\mathbf{C}_\pi + \mathbf{C}_p - 2\sqrt{\mathbf{C}_\pi \mathbf{C}_p}\right)$$

FID limitations

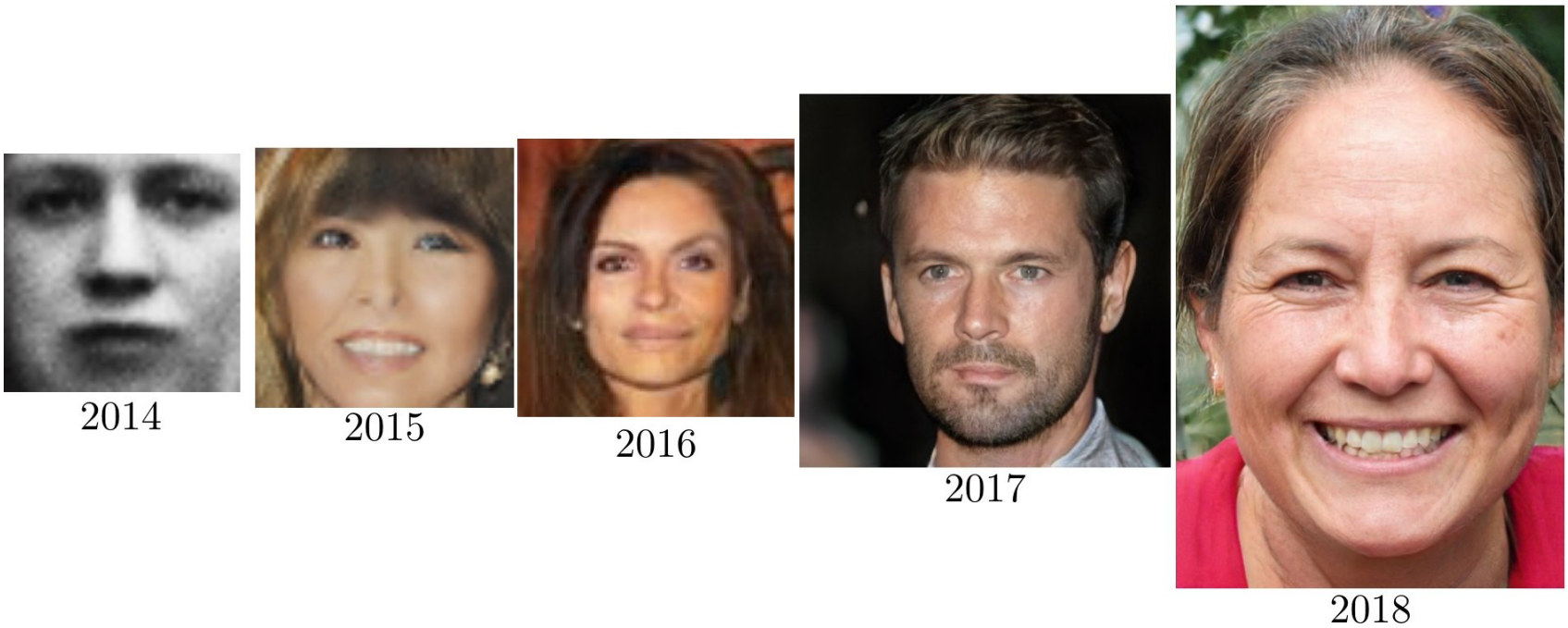
- FID depends on the pretrained classification model.
- FID needs a large samples size for evaluation.
- Calculation of FID is slow.
- FID extimates only two sample moments.

Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017

Summary

- Wasserstein GAN uses Kantorovich-Rubinstein duality to estimate Wasserstein distance.
- Gradient Penalty proposes the regularizer to enforce Lipschitzness.
- Spectral normalization is a weight normalization technique to enforce Lipshitzness.
- f-divergence family is a unified framework for divergence minimization.
- Inception Score and Frechet Inception Distance are the common metrics for GAN evaluation.

Evolution of GANs



- **Vanilla GAN** https://arxiv.org/abs/1406.2661
- **DCGAN** https://arxiv.org/abs/1511.06434
- **CoGAN** https://arxiv.org/abs/1606.07536
- **ProGAN** https://arxiv.org/abs/1710.10196
- **StyleGAN** https://arxiv.org/abs/1812.04948

Self-Attention GAN

- Convolutional layers process the information in a local neighborhood.
- Using convolutional layers alone is computationally inefficient for modeling long-range dependencies in images.

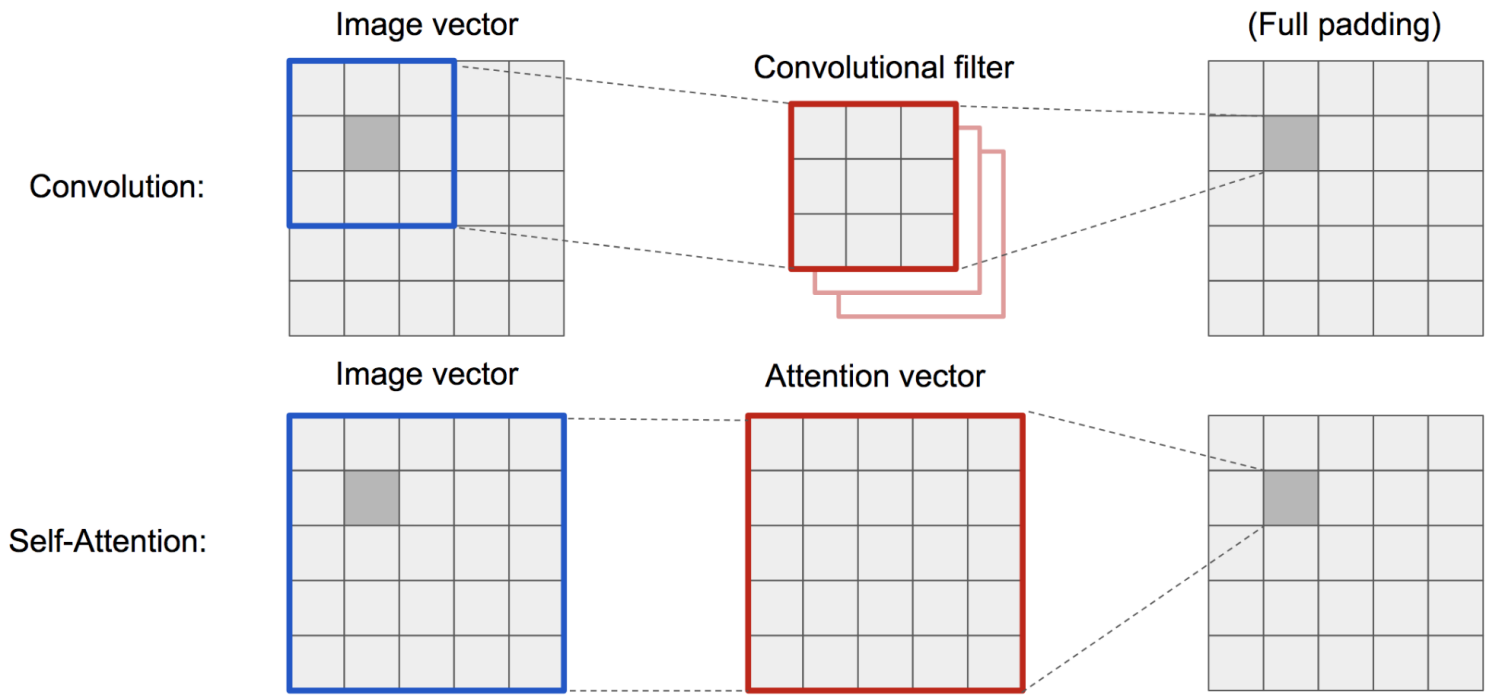
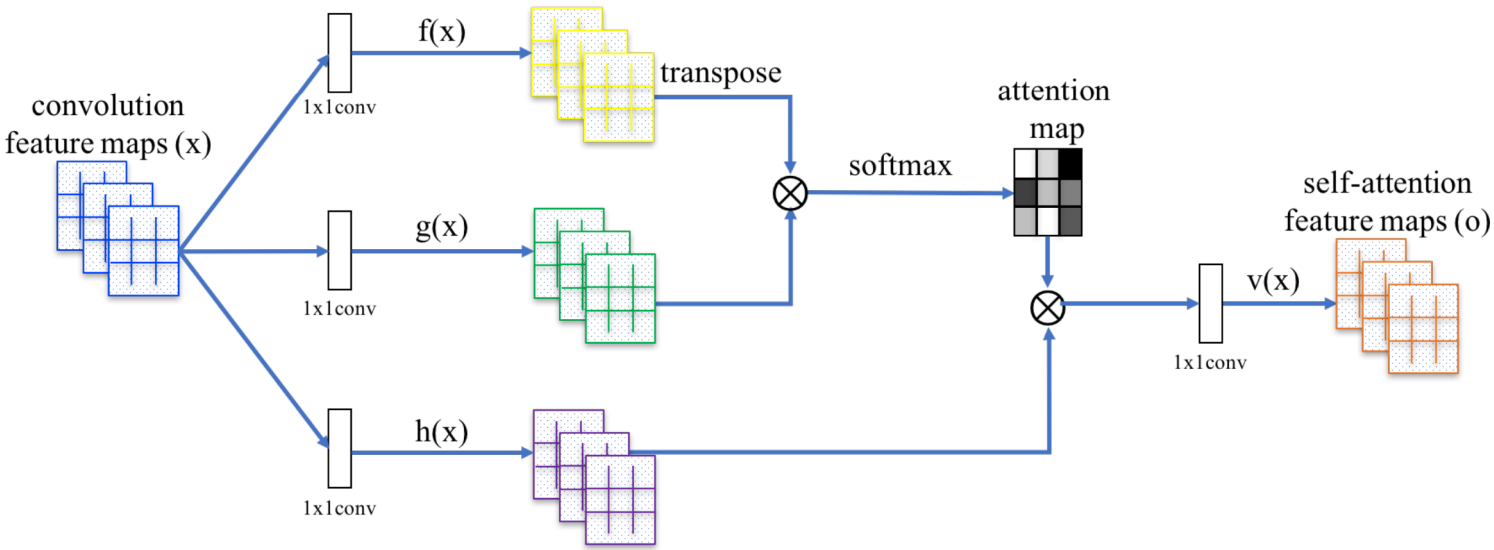


image credit:
https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Self-Attention GAN



- $\mathbf{x}$ – feature vector for one feature location.
- N – number of feature locations.

$$\mathbf{f}(\mathbf{x}) = \mathbf{W}_f(\mathbf{x}), \quad \mathbf{g}(\mathbf{x}) = \mathbf{W}_g(\mathbf{x}), \quad \mathbf{h}(\mathbf{x}) = \mathbf{W}_h(\mathbf{x}), \quad \mathbf{v}(\mathbf{x}) = \mathbf{W}_v(\mathbf{x})$$

$$s_{ij} = \mathbf{f}(\mathbf{x}_i)^T \mathbf{g}(\mathbf{x}_j), \quad a_{ij} = \frac{\exp s_{ij}}{\sum_{i=1}^{N} \exp s_{ij}}, \quad \mathbf{o}_j = \mathbf{v}\left(\sum_{i=1}^{N} a_{ij} \mathbf{h}(\mathbf{x}_i)\right)$$

Zhang H. et al. Self-Attention Generative Adversarial Networks, 2018

Self-Attention GAN

Technical Details

- Hinge loss for training.
- Spectral Normalization in both the generator and the discriminator.
- Separate learning rates for the generator and the discriminator.

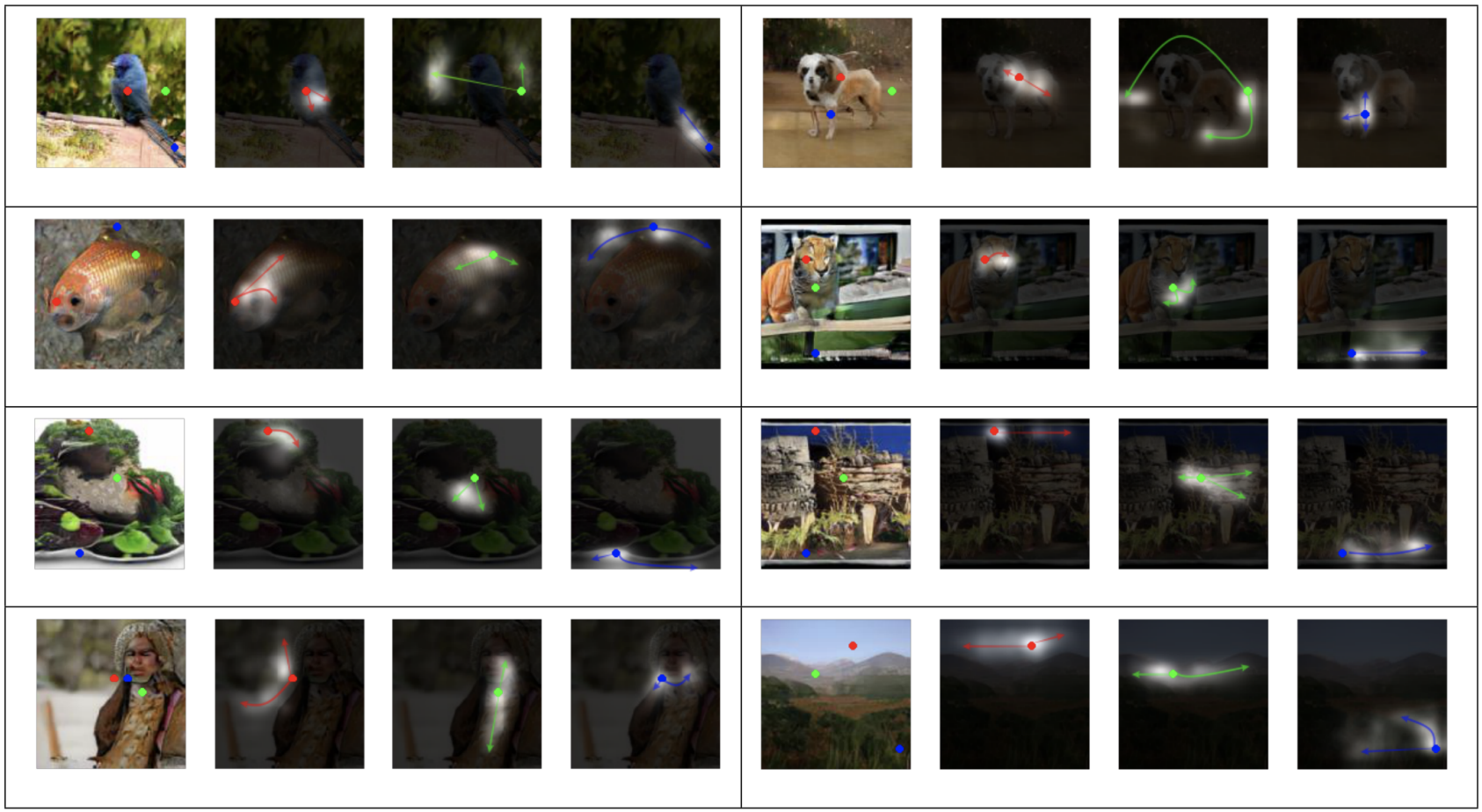
Model	no attention	SAGAN				Residual			
		$feat_8$	$feat_{16}$	$feat_{32}$	$feat_{64}$	$feat_8$	$feat_{16}$	$feat_{32}$	$feat_{64}$
FID	22.96	22.98	22.14	**18.28**	18.65	42.13	22.40	27.33	28.82
IS	42.87	43.15	45.94	51.43	**52.52**	23.17	44.49	38.50	38.96

Model	Inception Score	Intra FID	FID
AC-GAN (Odena et al., 2017)	28.5	260.0	/
SNGAN-projection (Miyato & Koyama, 2018)	36.8	92.4	27.62*
SAGAN	**52.52**	**83.7**	**18.65**

Zhang H. et al. Self-Attention Generative Adversarial Networks, 2018

Self-Attention GAN

Visualization of attention maps



Zhang H. et al. Self-Attention Generative Adversarial Networks, 2018

BigGAN

Model description

- Self-Attention GAN baseline.
- Class-conditional generator.
- Increasing batch size gives tremendous benefit (allows to cover more modes).
- Increasing model size is helpful (wider helps as much as deeper).
- Hinge loss for training.
- Orthogonal regularization for smoothness the generator output.
- Truncation trick for balancing between diversity and fidelity.

Brock A., Donahue J., Simonyan K. Large Scale GAN Training for High Fidelity Natural Image Synthesis, 2018

BigGAN

- **Orthogonal regularization**

$$\|\mathbf{W}^T\mathbf{W} - \mathbf{I}\|^2 \quad \Rightarrow \|\mathbf{W}^T\mathbf{W} - \text{diag}(\mathbf{W}^T\mathbf{W})\|^2$$

- **Truncation trick.** Coordinates of samples $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ which fall outside a predefined range are resampled to fall inside that range.

Batch	Ch.	Param (M)	Shared	Skip-z	Ortho.	Itr $\times 10^3$	FID	IS
256	64	81.5	SA-GAN Baseline			1000	18.65	52.52
512	64	81.5	✗	✗	✗	1000	15.30	58.77($\pm$1.18)
1024	64	81.5	✗	✗	✗	1000	14.88	63.03($\pm$1.42)
2048	64	81.5	✗	✗	✗	732	12.39	76.85($\pm$3.83)
2048	96	173.5	✗	✗	✗	295($\pm$18)	9.54($\pm$0.62)	92.98($\pm$4.27)
2048	96	160.6	✓	✗	✗	185($\pm$11)	9.18($\pm$0.13)	94.94($\pm$1.32)
2048	96	158.3	✓	✓	✗	152($\pm$7)	8.73($\pm$0.45)	98.76($\pm$2.84)
2048	96	158.3	✓	✓	✓	165($\pm$13)	8.51($\pm$0.32)	99.31($\pm$2.10)
2048	64	71.3	✓	✓	✓	371($\pm$7)	10.48($\pm$0.10)	86.90($\pm$0.61)

Brock A., Donahue J., Simonyan K. Large Scale GAN Training for High Fidelity Natural Image Synthesis, 2018

BigGAN

Samples (512x512)

Brock A., Donahue J., Simonyan K. Large Scale GAN Training for High Fidelity Natural Image Synthesis, 2018

BigGAN

Interpolations

Brock A., Donahue J., Simonyan K. Large Scale GAN Training for High Fidelity Natural Image Synthesis, 2018

Progressive Growing GAN

Problems with HR image generation

- Disjoint manifolds $\Rightarrow$ gradient problem.
- Small minibatch $\Rightarrow$ training instability.

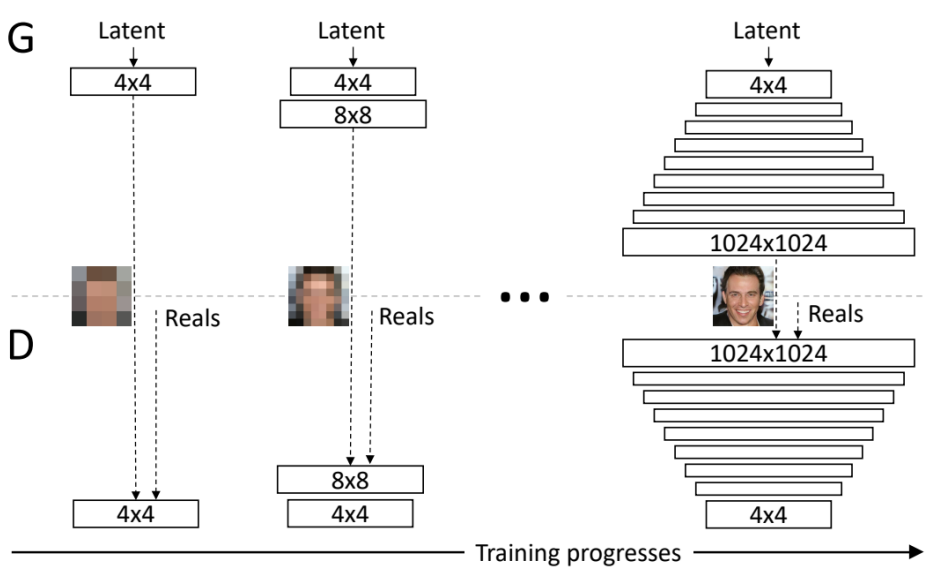
Solution

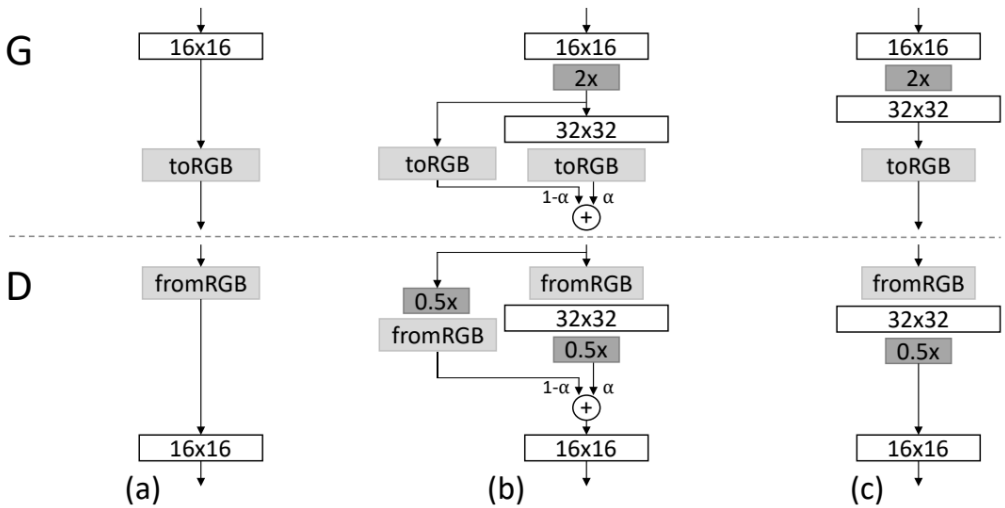
Grow both the generator and discriminator progressively, starting from LR images, and add new layers that introduce higher-resolution details as the training progresses.

- Train GAN which generate 4x4 images (just 2 convolutions for G and D).
- Add upsampling layers to G, downsampling layers to D.
- Train GAN which generate 8x8 images.
- etc.

Karras T. et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017

Progressive Growing GAN





Karras T. et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017

Progressive Growing GAN

Samples (1024x1024)

Karras T. et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017

StyleGAN

- Generating of HR images is hard.
- Progressive growing greatly simplifies the task.
- The ability to control specific features of the generated image is very limited.

Face image features

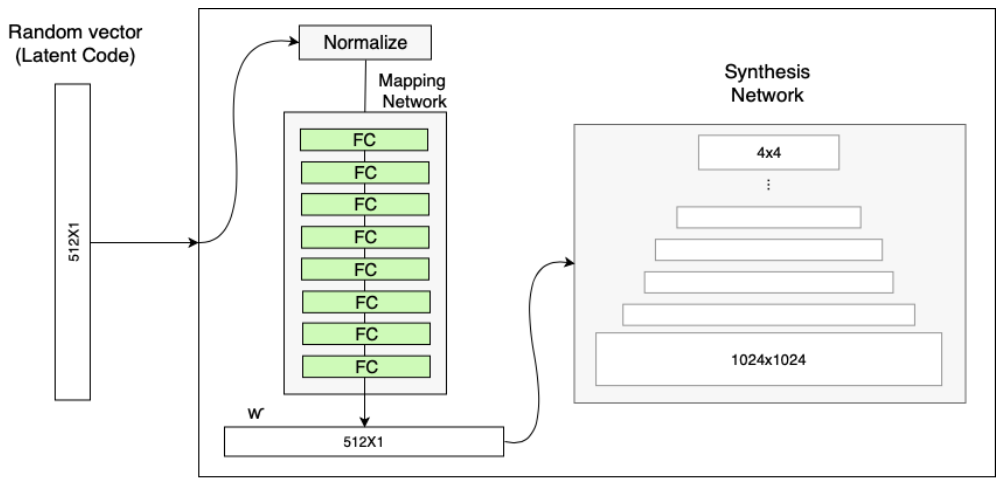
- Coarse (pose, general hair style, face shape). Resolution $4^2 - 8^2$.
- Middle (finer facial features, hair style, eyes open/closed). Resolution $16^2 - 32^2$.
- Fine (color scheme (eye, hair and skin) and micro features). Resolution $64^2 - 1024^2$.

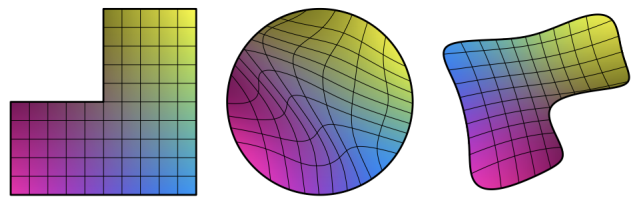
Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Step 1: Mapping Network

- Generator input is likely to be **disentangled**. Each component of input vector **z** should be responsible for one generative factor.
- Mapping network $f : \mathcal{Z} \to \mathcal{W}$ is used to reduce correlations between components of **z**.





(a) Distribution of features in training set

(b) Mapping from $\mathcal{Z}$ to features

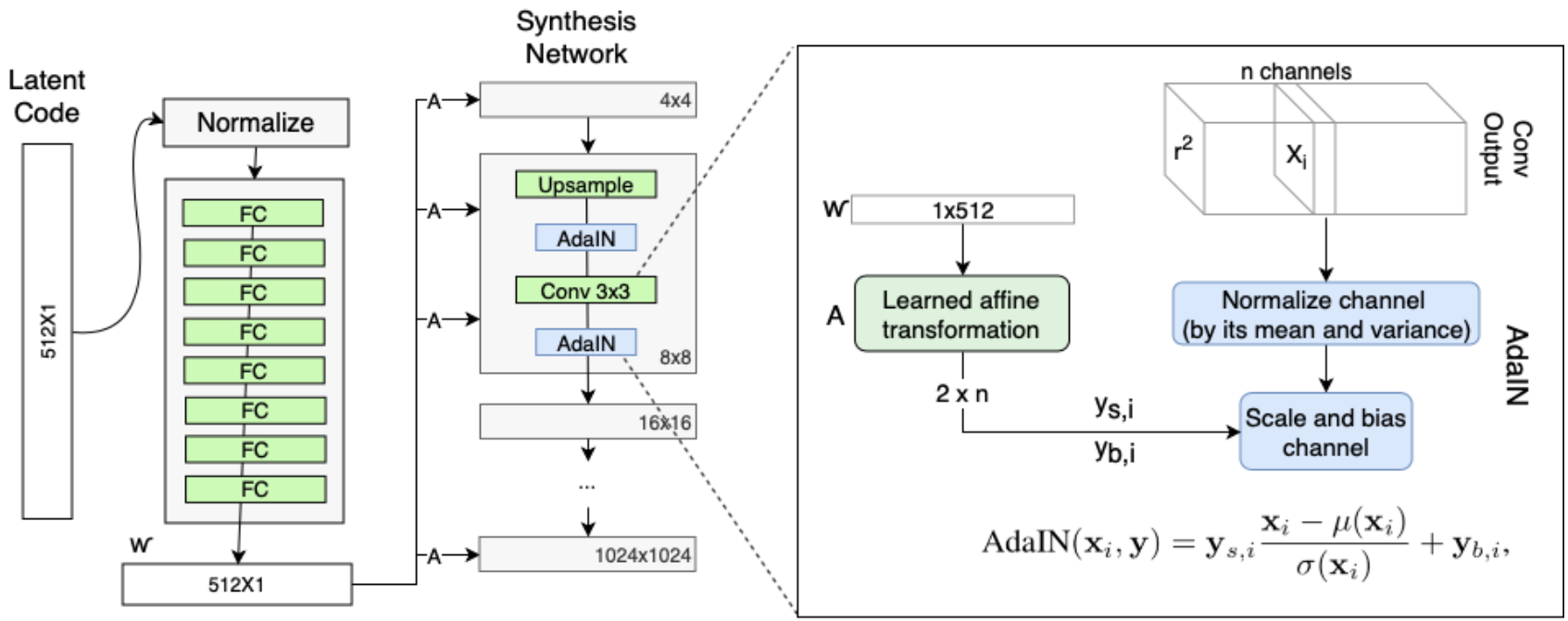
(c) Mapping from $\mathcal{W}$ to features

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Step 2: Style modulation

- Adaptive Instance Normalization transfers the **w** vector to the synthesis Network.
- The module is added to each resolution to define the visual expression of the features.

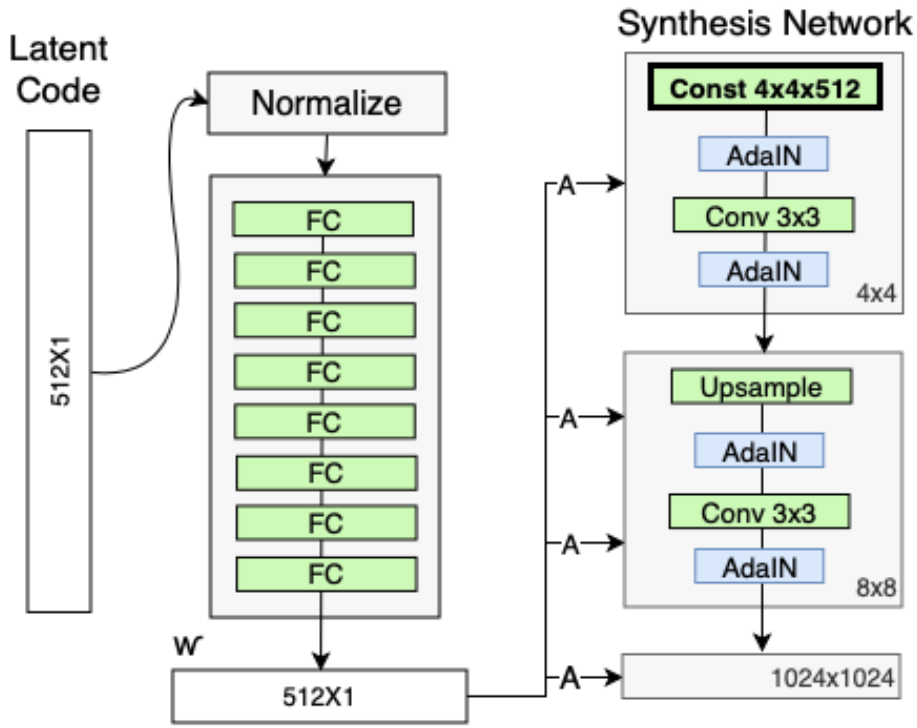


Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Step 3: Remove traditional input

Mapping network provides stochasticity to different stages of the synthesis network. Input of the synthesis network is a trainable vector.

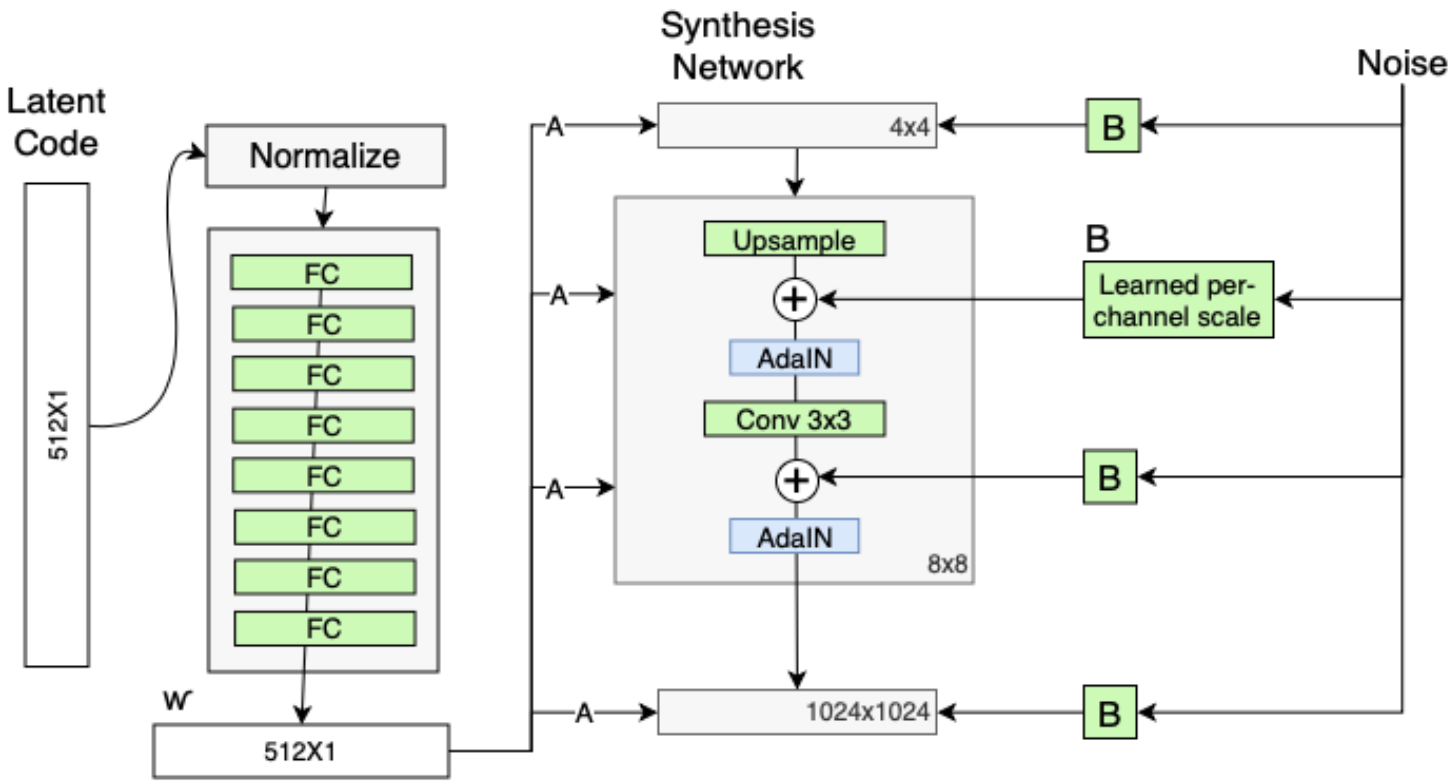


Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Step 4: Stochastic variation

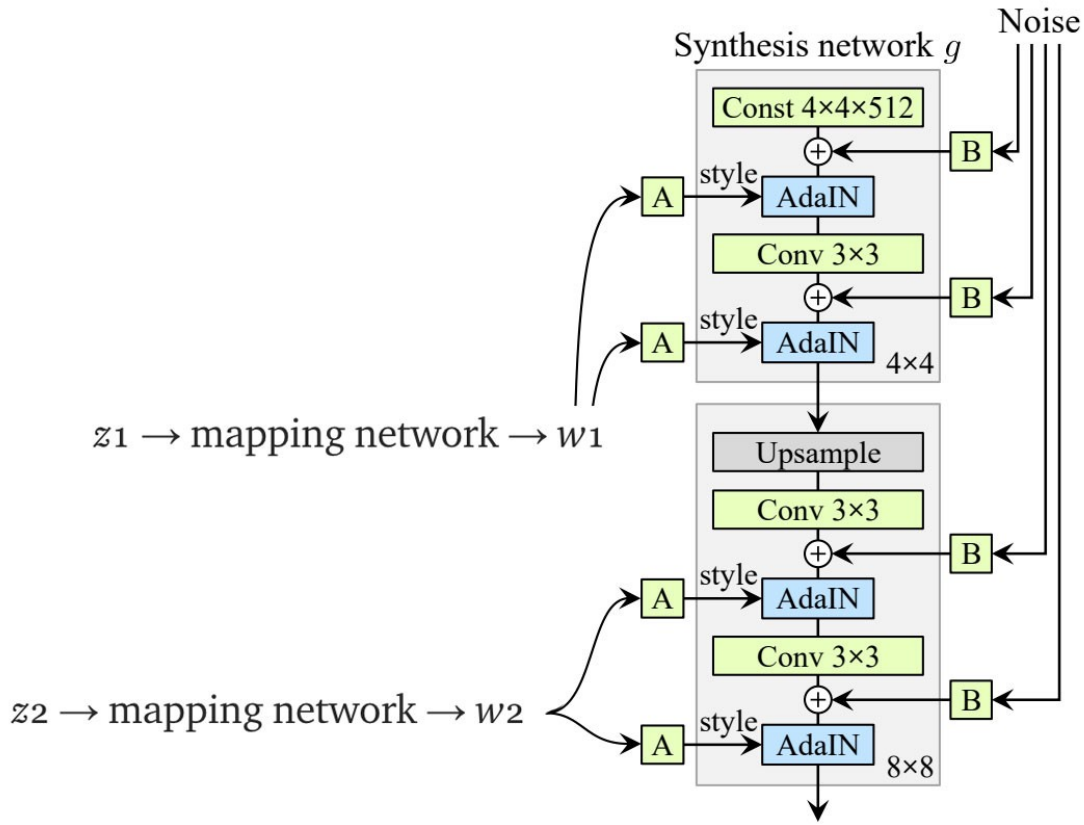
Inject random noise to add small aspects, such as freckles, exact placement of hairs, wrinkles, features which make the image more realistic and increase the variety of outputs.



Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

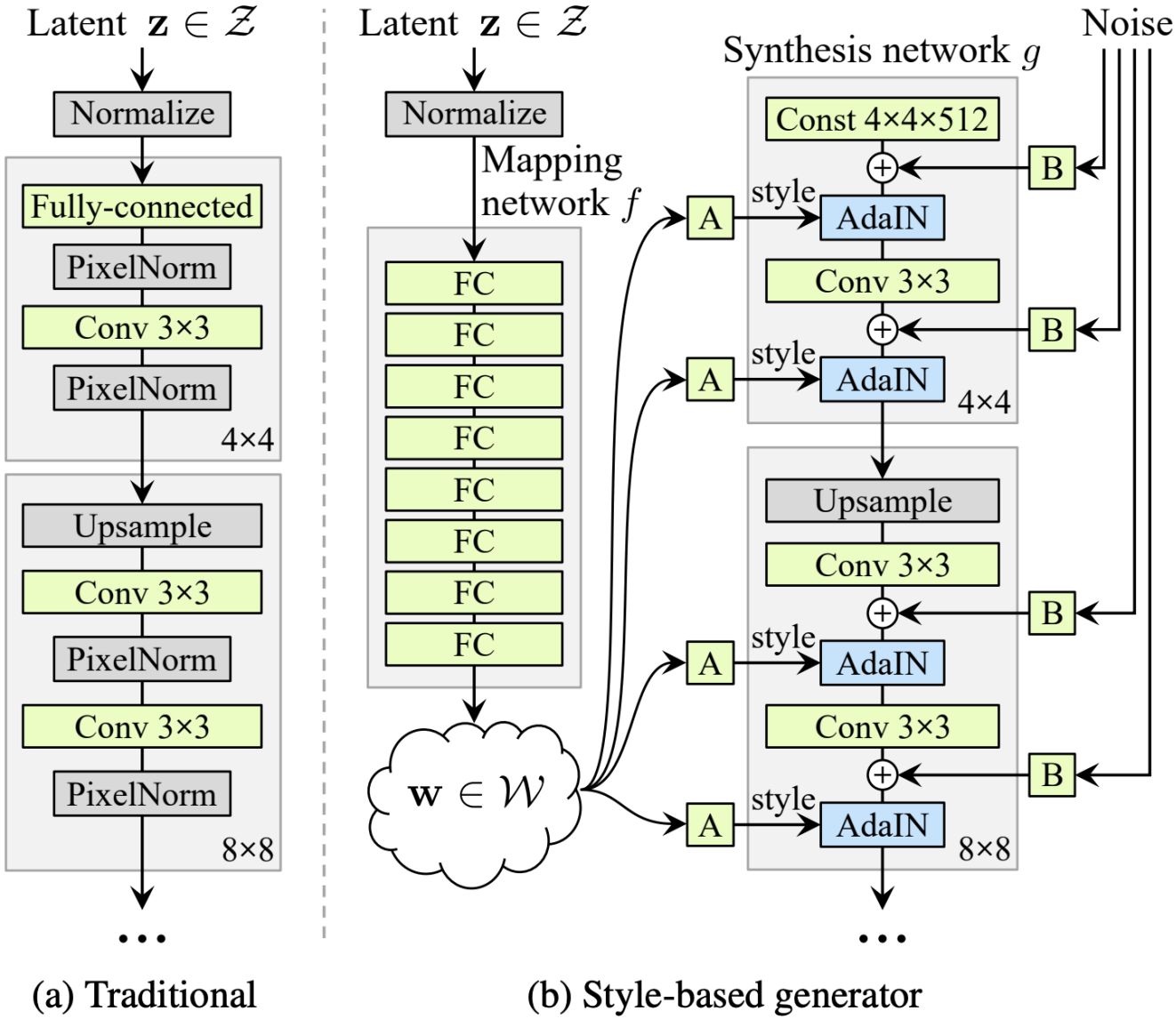
Step 4: Style Mixing



- Makes different levels of synthesis network to be independent.
- Allows to couple diffirent styles.

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN



Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Truncation trick

$$\mathbf{w}' = \hat{\mathbf{w}} - \psi \cdot (\mathbf{w} - \hat{\mathbf{w}}), \quad \hat{\mathbf{w}} = \mathbb{E}_{\mathbf{z}} p(f(\mathbf{z}))$$

- Constant ψ is a tradeoff between diversity and fidelity.
- $\psi = 0.7$ is used for most of the results.
- Truncation is done only at the low-resolution layers.

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

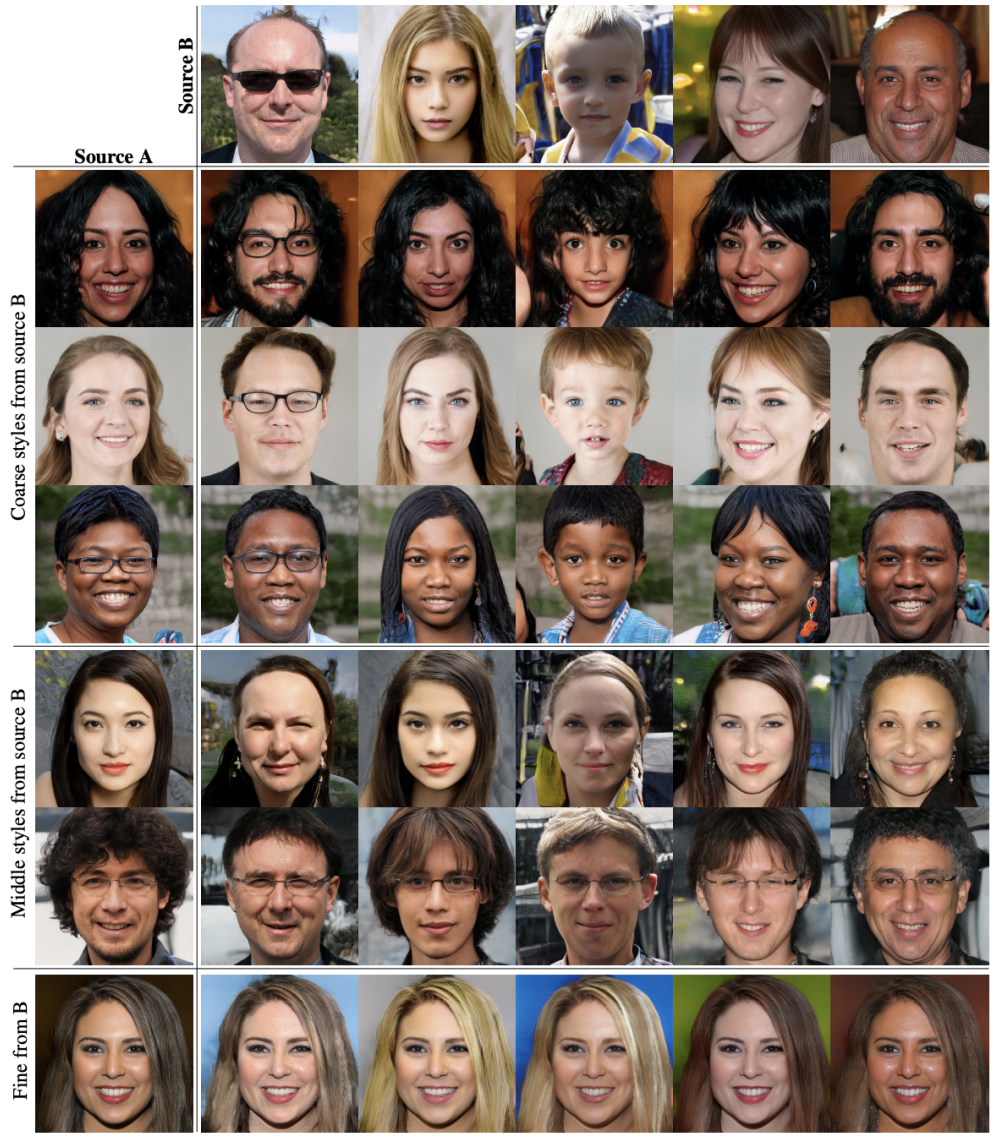
Results

	Method	CelebA-HQ	FFHQ
A	Baseline Progressive GAN [30]	7.79	8.04
B	+ Tuning (incl. bilinear up/down)	6.11	5.25
C	+ Add mapping and styles	5.34	4.85
D	+ Remove traditional input	5.07	4.88
E	+ Add noise inputs	**5.06**	4.42
F	+ Mixing regularization	5.17	**4.40**

Samples (1024x1024)

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN



Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

Summary