CISC6000 Deep Learning
Generative Adversarial Networks (GANs)

Dr. Yijun Zhao

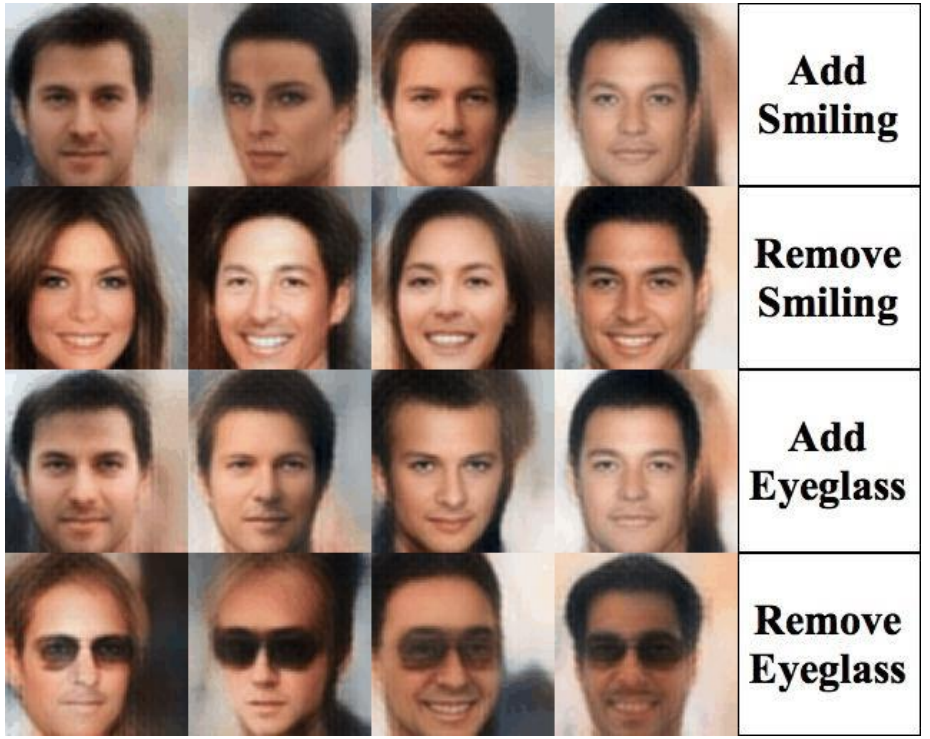
Fordham University

Generative Models

Two generative models facing neck to neck

- VAE: rooted in Bayesian inference
- **GAN: rooted in game theory**

Variational Autoencoder



https://houxianxu.github.io/assets/project/dfcvae

Magic of GANs

Training examples

Model samples

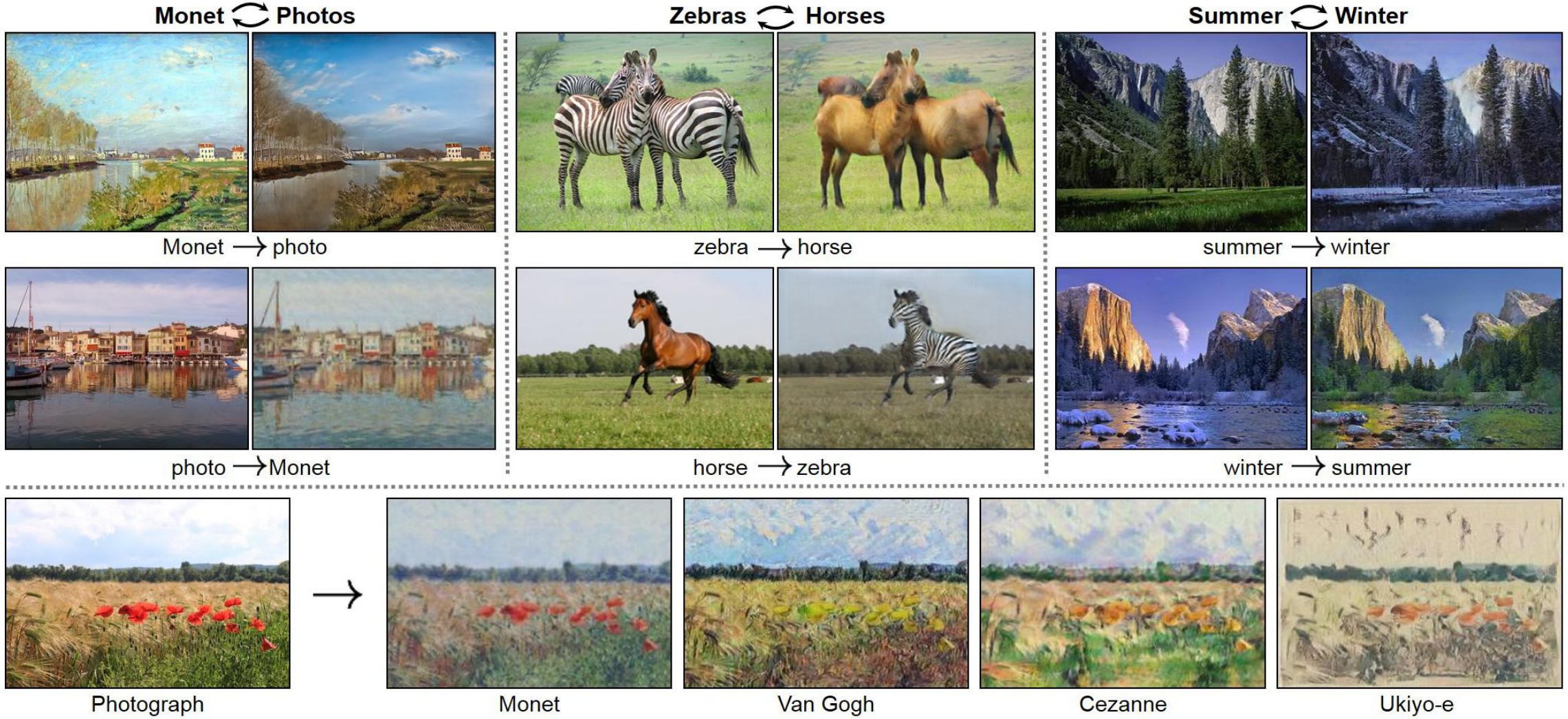
Goodfellow NIPS tutorial and accompanying paper (arXiv:1701.00160v4 [cs.LG]) provided some figures

Magic of GANs

Which one is Computer generated?

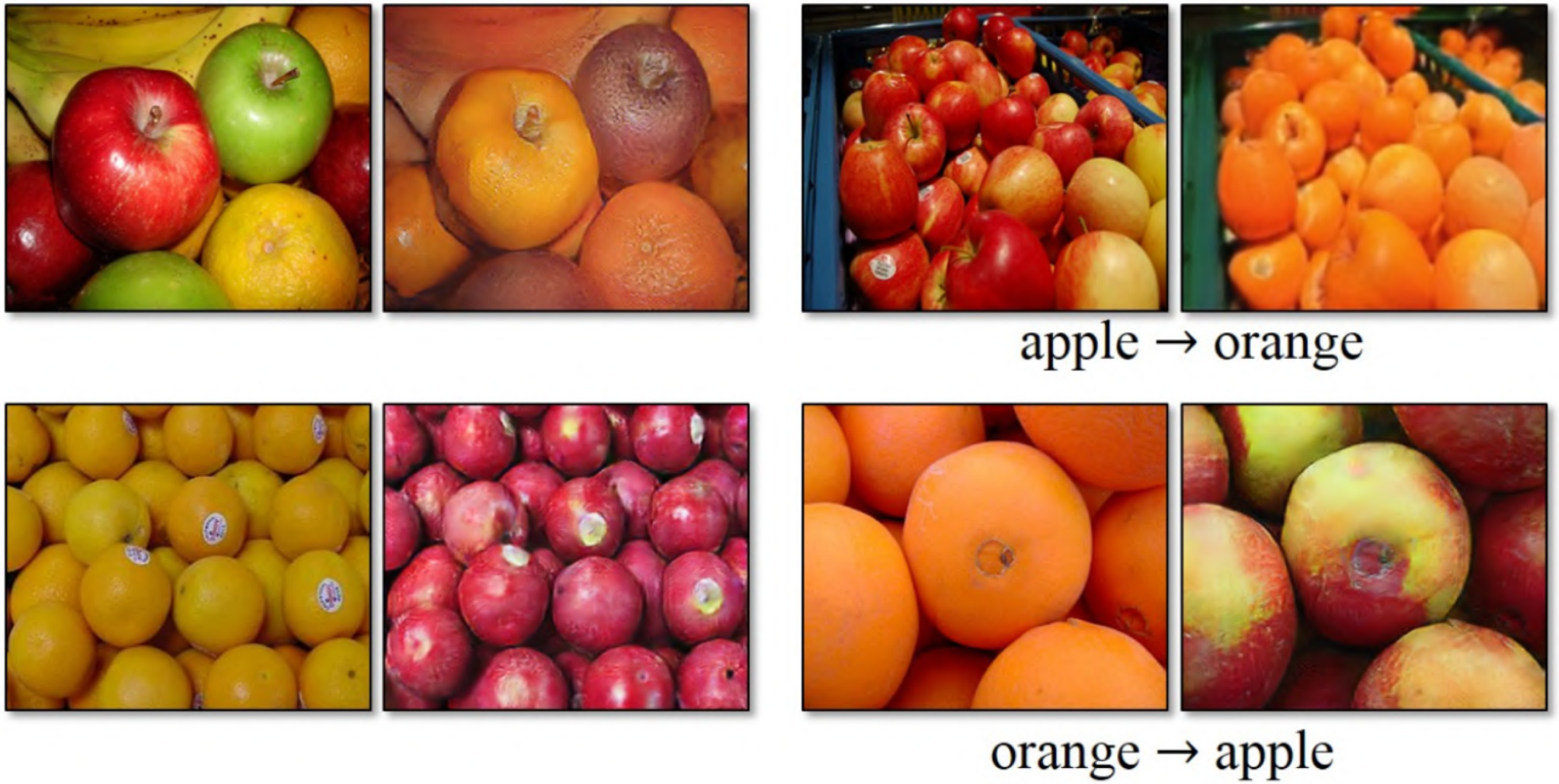
Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." *arXiv preprint arXiv:1609.04802* (2016).

Magic of GANs



Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In *Proceedings of the IEEE international conference on computer vision*, pp. 2223-2232. 2017.

Magic of GANs



Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In *Proceedings of the IEEE international conference on computer vision*, pp. 2223-2232. 2017.

Magic of GANs

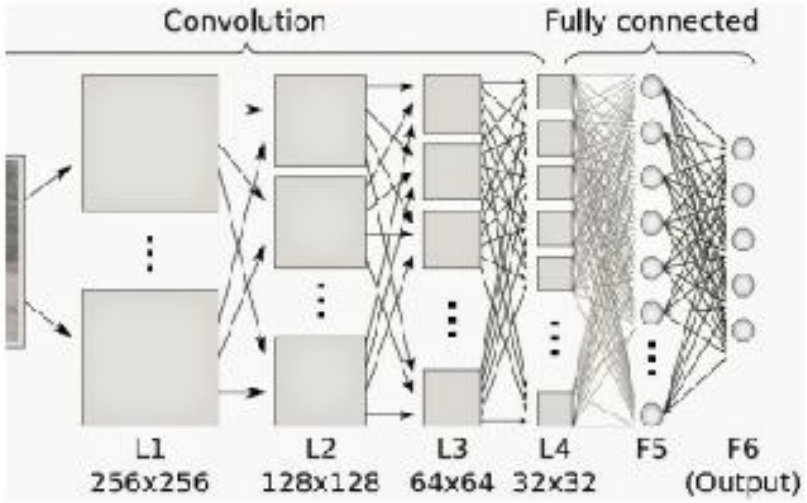
winter Yosemite → summer Yosemite

summer Yosemite → winter Yosemite

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In *Proceedings of the IEEE international conference on computer vision*, pp. 2223-2232. 2017.

what people think he's referring to

what he's actually referring to



GANs by Analogy

Scenario:

Two kids are playing a game where one of them has to fool the other in making counterfeit dollars (where are their parents??)

“Forger”

“Detective”

Slide based on one by Pavlo Lyalyutskyy

GANs by Analogy

- Initially, neither one of them is very good at their job
- The Forger produces horrible doodles on paper
- The Detective just looks for obvious “tells” / mistakes

“Forger”

“Detective”

Slide based on one by Pavlo Lyalyutskyy

GANs by Analogy

- As the Detective spots the Forger's fakes, the Forger has to devise better fakes
- The Detective, in turn, has to get better at spotting the Forger's improved fakes

"Forger"

"Detective"

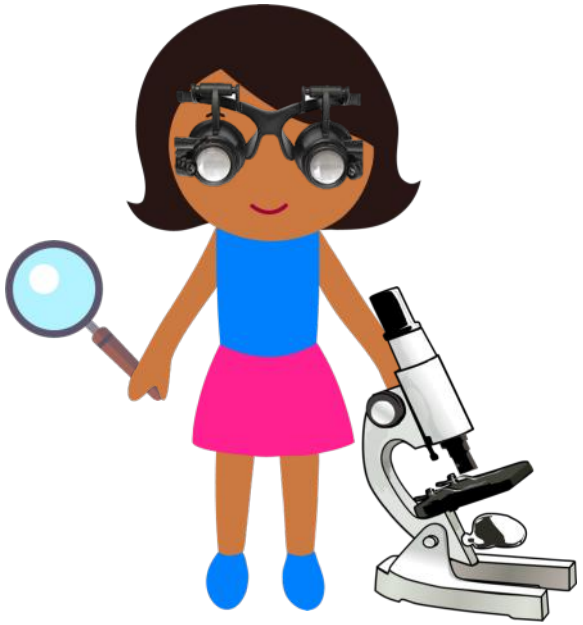
Slide based on one by Pavlo Lyalyutskyy

GANs by Analogy

- If they keep this up long enough, the Forger gets so good that their fakes are virtually indistinguishable from the real thing...
- ...and the Detective has developed ‘superhuman’ abilities to detect them

“Forger”

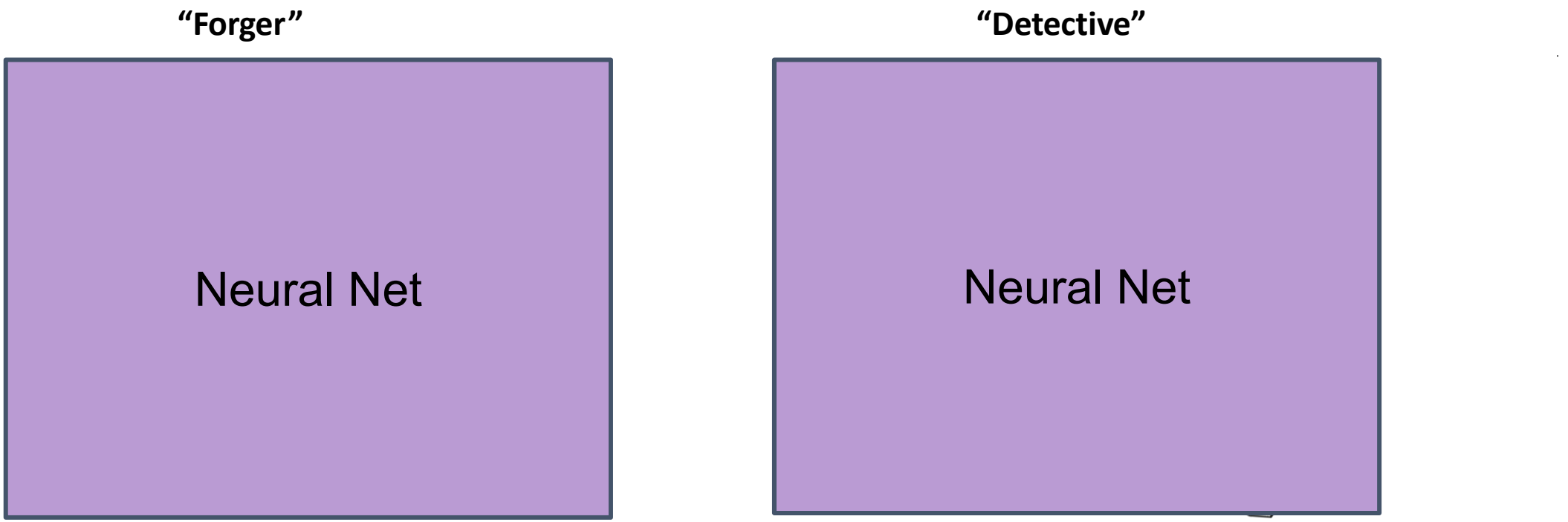
“Detective”



Slide based on one by Pavlo Lyalyutskyy

GANs by Analogy

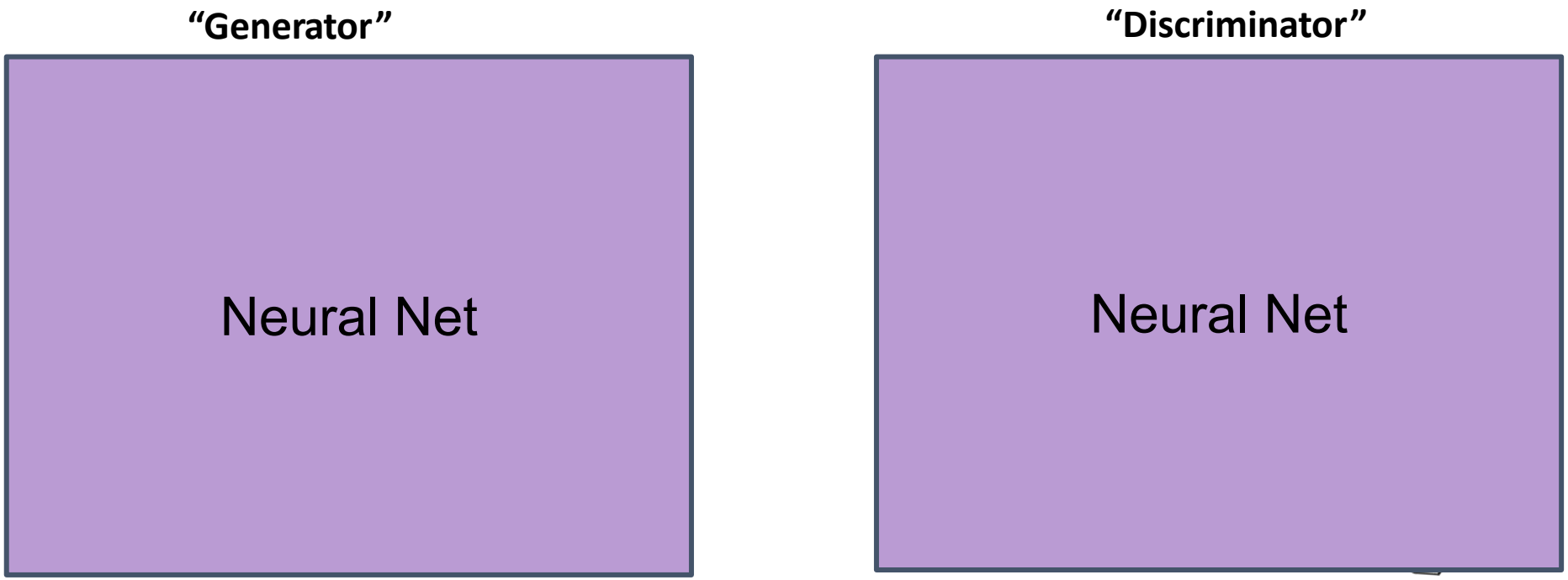
- GANs operationalize this idea by using neural networks to serve both of these roles



Slide based on one by Pavlo Lyalyutskyy

GANs by Analogy

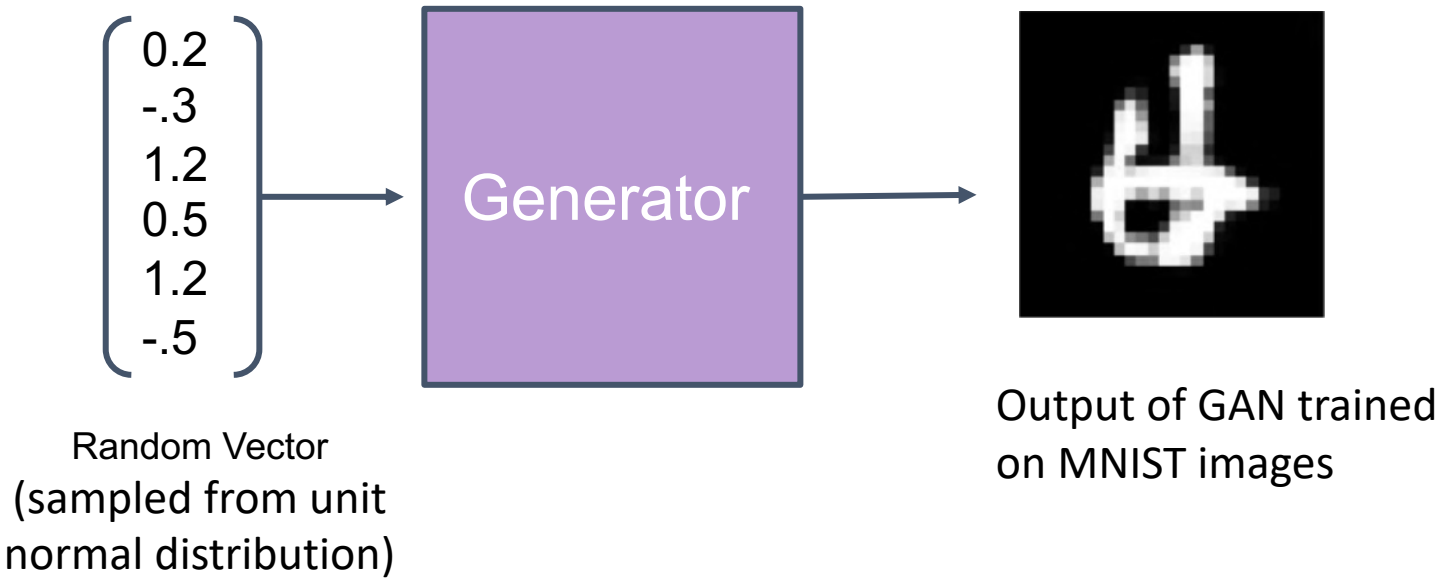
- GANs operationalize this idea by using neural networks to serve both of these roles
- We call these networks the “Generator” and the “Discriminator”



Slide based on one by Pavlo Lyalyutskyy

The Generator

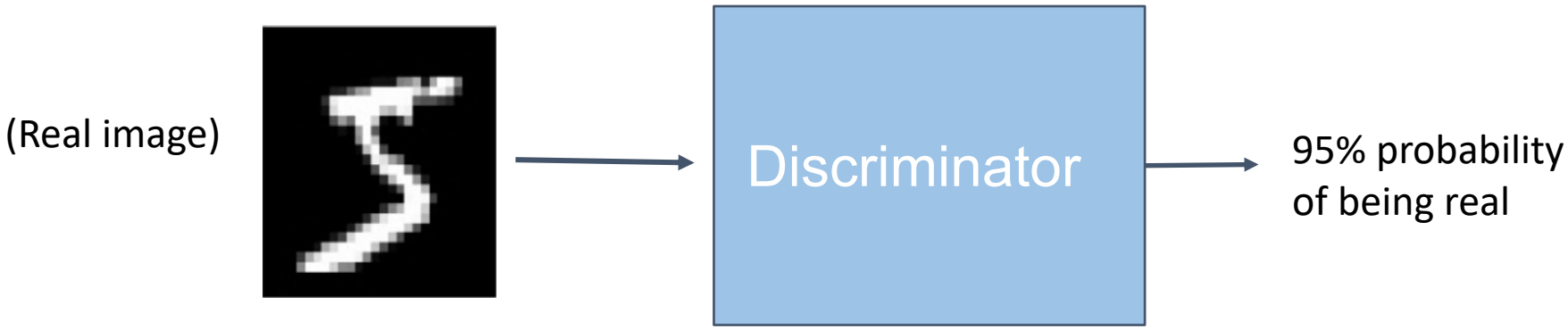
The generator is a neural network that takes in a random vector and produces a "fake" data point



Slide based on one by Pavlo Lyalyutskyy

The Discriminator

The discriminator is a neural network that takes in images and predicts the probability that the image is real:

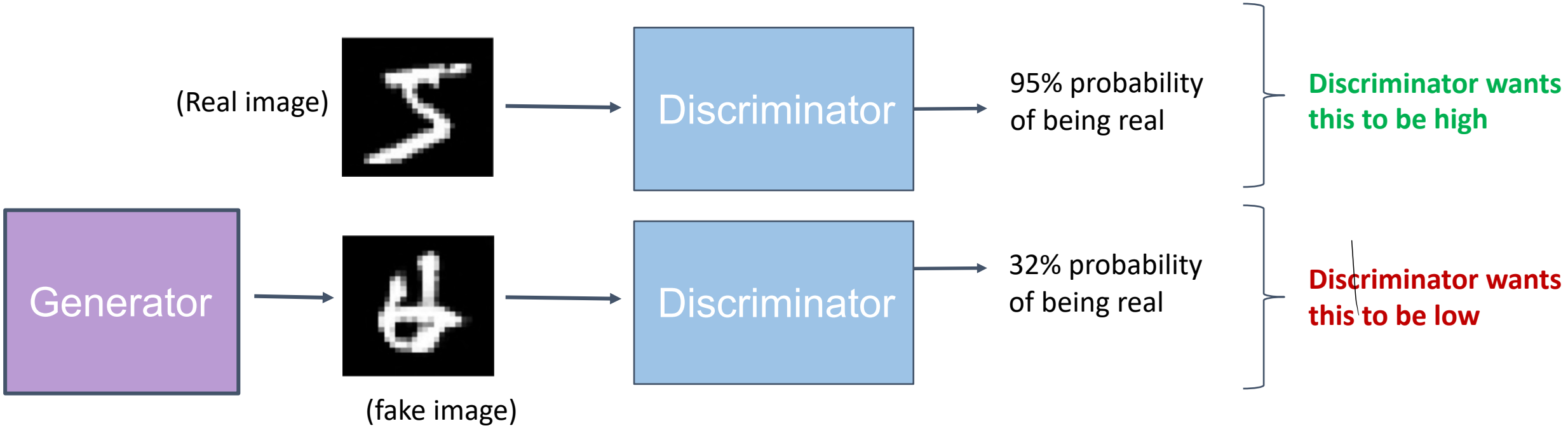


Slide based on one by Pavlo Lyalyutskyy

Training GANs

Discriminator wants:

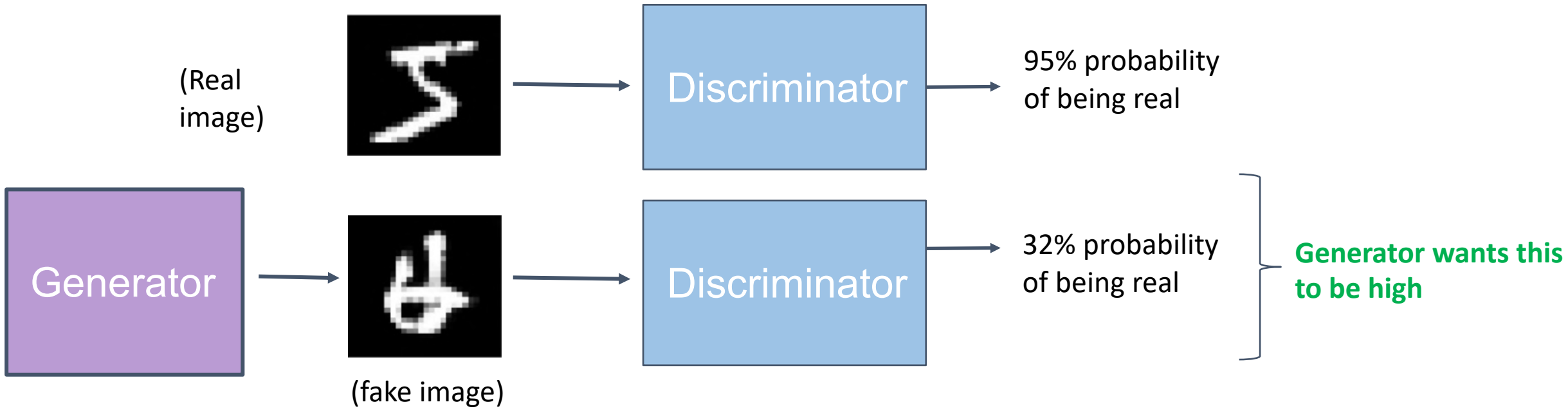
- Real images are real with high probability.
- Fake images are real with low probability.



Slide based on one by Pavlo Lyalyutskyy

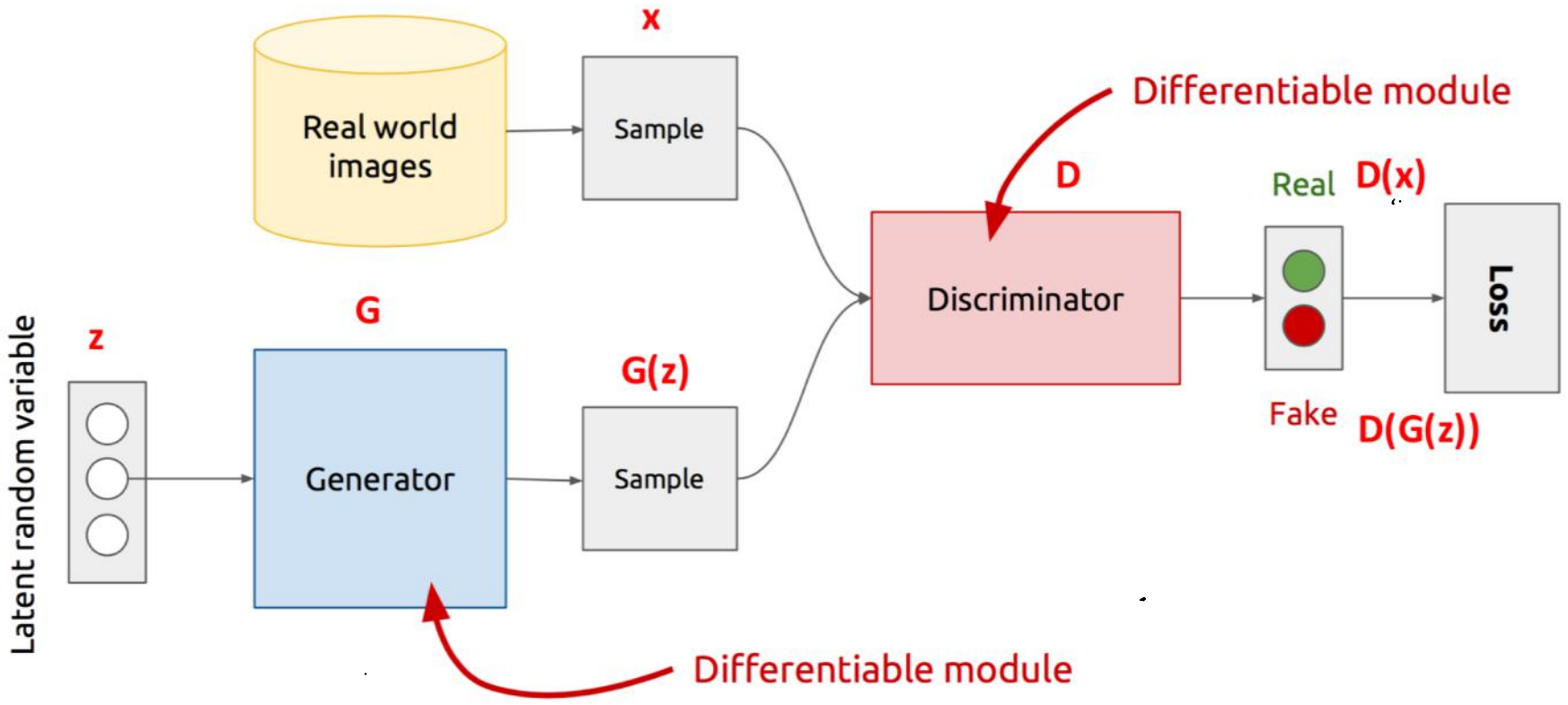
Training GANs

Generator wants to fool the discriminator. It wants the probability of the discriminator saying a fake image is real to be high.



Slide based on one by Pavlo Lyalyutskyy

GANs Architecture



- **Z** is some random noise (Gaussian/Uniform).
- **Z** can be thought as the latent representation of the image.

https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Training the Discriminator

Discriminator wants to **maximize**:

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

Log probability that the real image x is predicted to be **real** by the discriminator.

Log probability that the fake image $G(z)$ is predicted to be **fake** by the discriminator.

Note: Maximizing this quantity is equivalent to minimizing binary cross entropy loss with fake data labelled as 0 and real data labelled as 1.

Slide based on one by Pavlo Lyalyutskyy

Training the Generator

Generator wants to **minimize**:

$$E_z[log(1 - D(G(z)))]$$

Log probability that the fake image $G(z)$ is predicted to be **fake** by the discriminator.

Slide based on one by Pavlo Lyalyutskyy

Training the Generator

Generator wants to **minimize**:

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

This does NOT affect generator

Log probability that the fake image $G(z)$ is predicted to be **true** by the discriminator.

Slide based on one by Pavlo Lyalyutskyy

Training the GAN

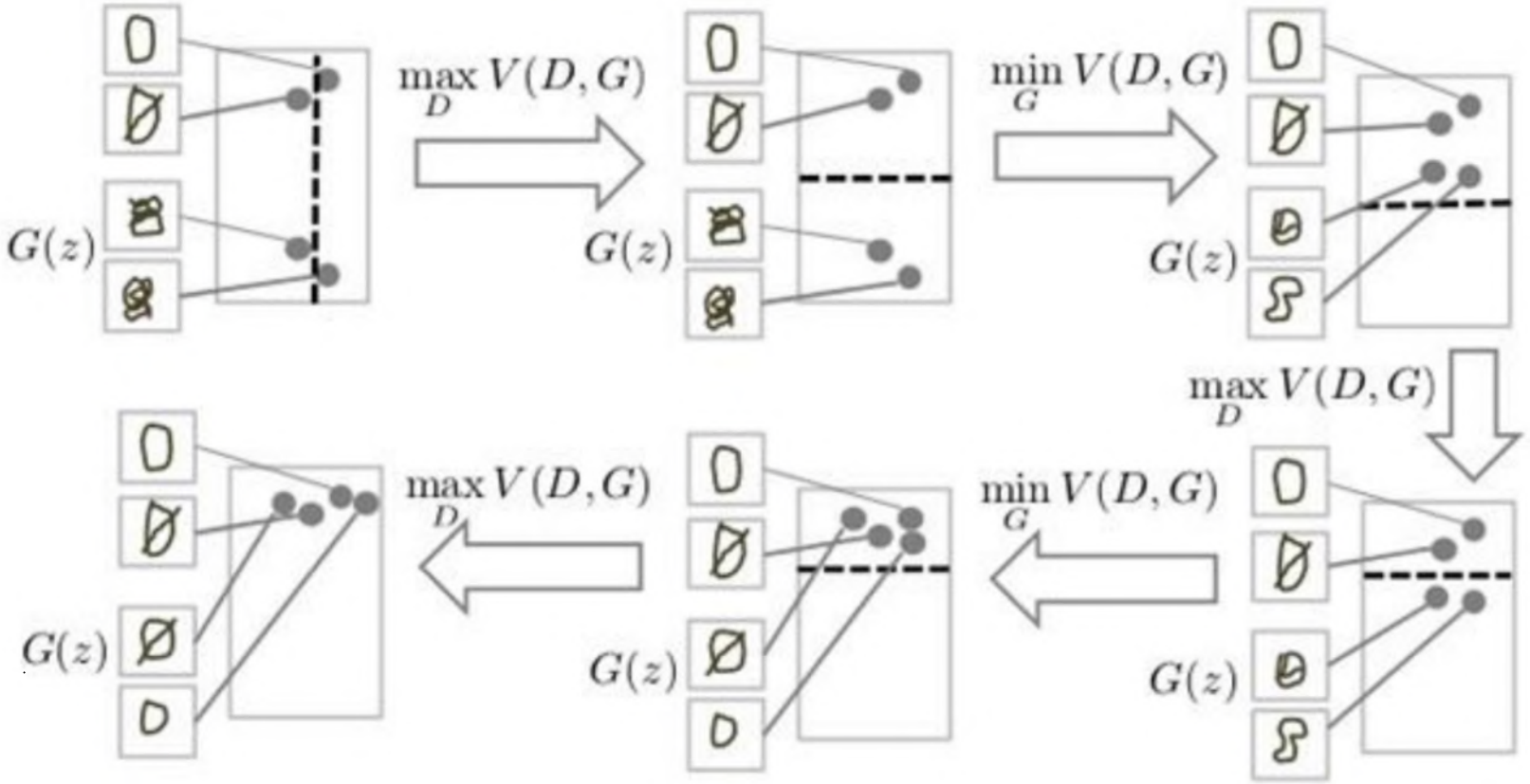
Loss function for the GAN:

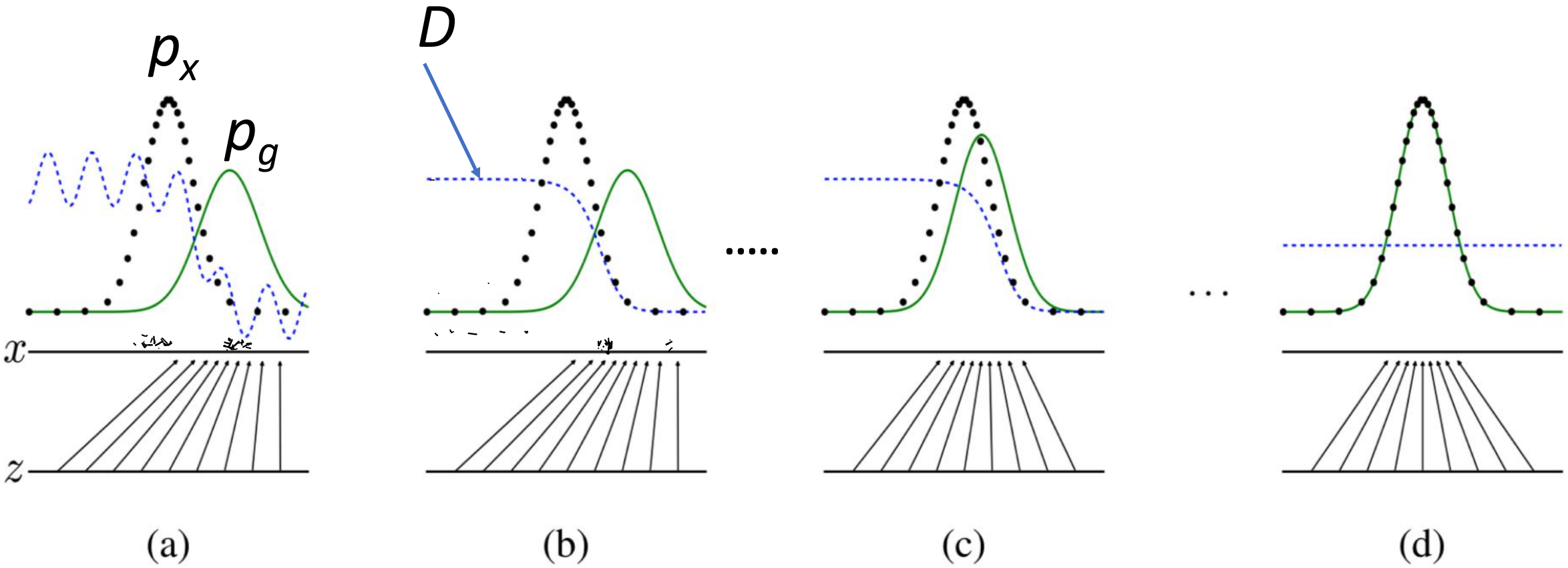
$$\min_{G} \max_{D} \; E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

Slide based on one by Pavlo Lyalyutskyy

Training the GAN

$$\min_G \max_D V(D, G)$$





Black curve: real data distribution

Green curve: generated data distribution

Blue curve: Discriminator's confidence that a sample from the region is real

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Prove Mathematically:

Global Optimality of $p_g = p_{\text{data}}$

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

Proposition 1. *For G fixed, the optimal discriminator D is*

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[1 - logD(G(z))]$$

$$= \int_x p_{data}(x) log(D(x)) dx + \int_z p_z(z) log(1 - D(G(z))) dz$$

$$x = G(z) \Rightarrow z = G^{-1}(x) \Rightarrow dz = (G^{-1})'(x) dx$$

$$\Rightarrow p_g(x) = p_z(G^{-1}(x))(G^{-1})'(x)$$

$$= \int_x p_{data}(x) log(D(x)) dx + \int_x p_z(G^{-1}(x)) log(1 - D(x))(G^{-1})'(x) dx$$

$$= \int_x p_{data}(x) log(D(x)) dx + \int_x p_g(x) log(1 - D(x)) dx$$

$$= \int_x p_{data}(x) log(D(x)) + p_g(x) log(1 - D(x)) dx$$

The whole proof relies on this step, which is not explained in the original paper. This equality comes from the Radon-Nikodym Theorem of measure theory. Colloquially, it's sometimes called the law of the unconscious statistician

$$\max_D V(D,G) = \max_D \int_x p_{data}(x) log(D(x)) + p_g(x) log(1 - D(x)) dx$$

$$\frac{\partial}{\partial D(x)} (p_{data}(x) log(D(x)) + p_g(x) log(1 - D(x))) = 0$$

Note: To find the optimal discriminator, it is desired to find a maximum of the integrand.

$$\Rightarrow \frac{p_{data}(x)}{D(x)} - \frac{p_g(x)}{1 - D(x)} = 0$$

$$\Rightarrow D(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

Cost for the Generator:

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

$$\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} [\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}} [\log(1 - D_G^*(G(\boldsymbol{z})))] \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} [\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g} [\log(1 - D_G^*(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})} \right] + \mathbb{E}_{\boldsymbol{x} \sim p_g} \left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})} \right]
\end{aligned}$$

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

Theorem 1. *The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p_g = p_{data}$. At that point, $C(G)$ achieves the value $-\log 4$.*

One direction is trivial: log(1/2) +log(1/2) = -log4

The other direction: derivation on next page

$$C(G) = -\log(4) + KL\left(p_{\text{data}} \,\middle\|\, \frac{p_{\text{data}} + p_g}{2}\right) + KL\left(p_g \,\middle\|\, \frac{p_{\text{data}} + p_g}{2}\right)$$

$$C(G) = -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \,\|\, p_g\right)$$

Since the Jensen–Shannon divergence between two distributions is always non-negative, and zero iff they are equal, we have shown that $C^* = -\log(4)$ is the global minimum of $C(G)$ and that the only solution is $p_g = p_{\text{data}}$, i.e., the generative model perfectly replicating the data distribution. □

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}\right]+\mathbb{E}_{\boldsymbol{x}\sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}\right]$$

$$= \int_x p_{data}(x) log(\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}) + p_g(x) log(\frac{p_g(x)}{p_{data}(x)+p_g(x)}) dx$$

$$= \int_x p_{data}(x) log(\frac{p_{data}(x)}{\frac{p_{data}(x)+p_g(x)}{2}}) + p_g(x) log(\frac{p_g(x)}{\frac{p_{data}(x)+p_g(x)}{2}}) dx - log(4)$$

$$= KL[p_{data}(x)||\frac{p_{data}(x)+p_g(x)}{2}] + KL[p_g(x)||\frac{p_{data}(x)+p_g(x)}{2}] - log(4)$$

Proposition 2. *If G and D have enough capacity, and at each step of Algorithm 1, the discriminator is allowed to reach its optimum given G, and p_g is updated so as to improve the criterion*

$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))]$$

then p_g converges to p_{data}

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.

Applications

1 — SRGAN
2 — DCGAN
3 — CycleGAN
4 — StyleGAN
5 — Conditional GANs (CGAN)
6 — Progressively Growing GAN (PG GAN)
7 — StackGAN
8 — BigGAN

Applications

SRGAN

Photo-realistic single image super-resolution using a generative adversarial network

C Ledig, L Theis, F Huszár… - Proceedings of the …, 2017 - openaccess.thecvf.com

Despite the breakthroughs in accuracy and speed of single image super-resolution using faster and deeper convolutional neural networks, one central problem remains largely unsolved: how do we recover the finer texture details when we super-resolve at large upscaling factors? The behavior of optimization-based super-resolution methods is principally driven by the choice of the objective function. Recent work has largely focused on minimizing the mean squared reconstruction error. The resulting estimates have high peak …

☆ 99 Cited by 6622 Related articles All 21 versions Import into BibTeX »

Applications

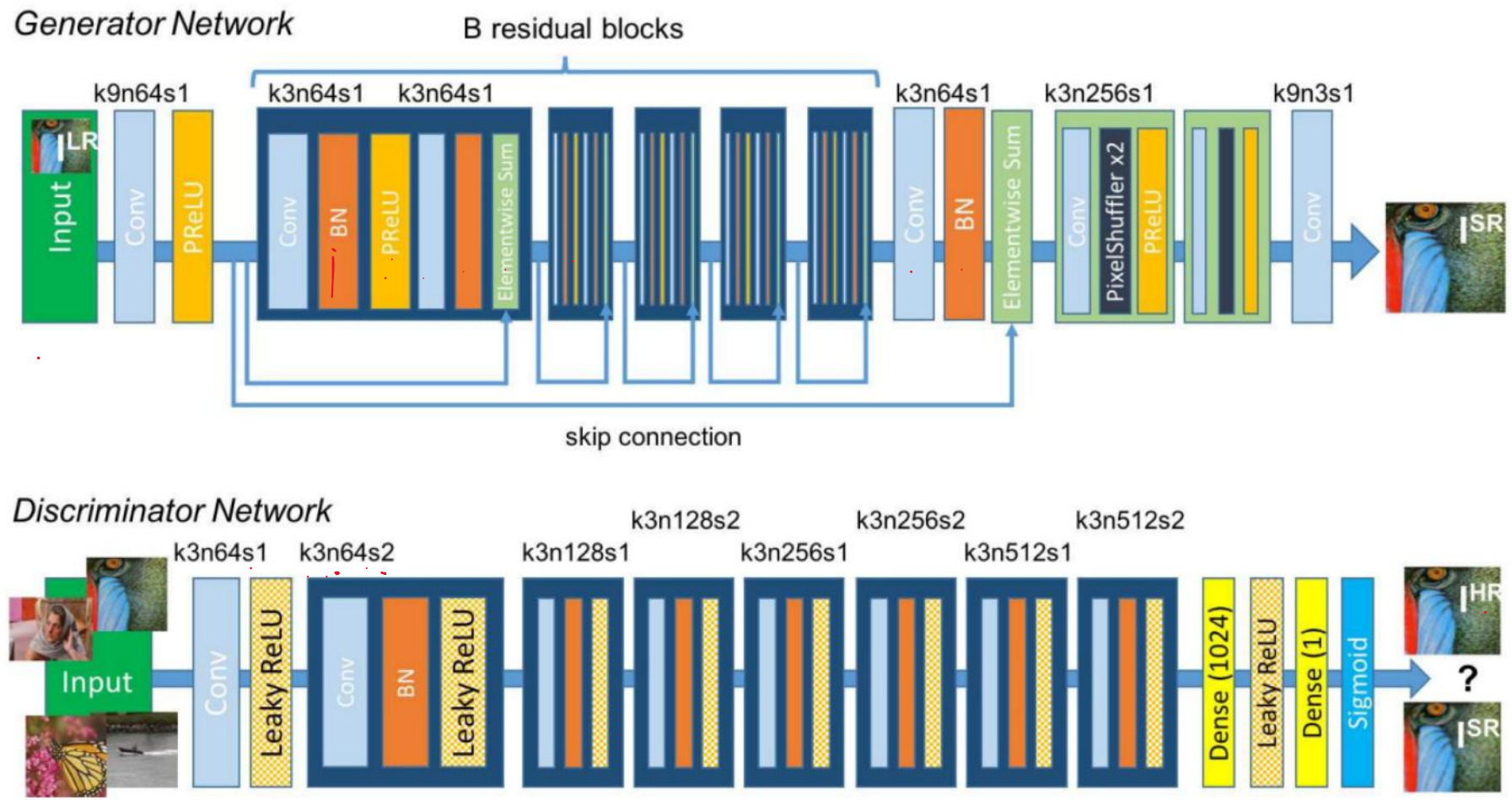


Figure 4: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.

Applications

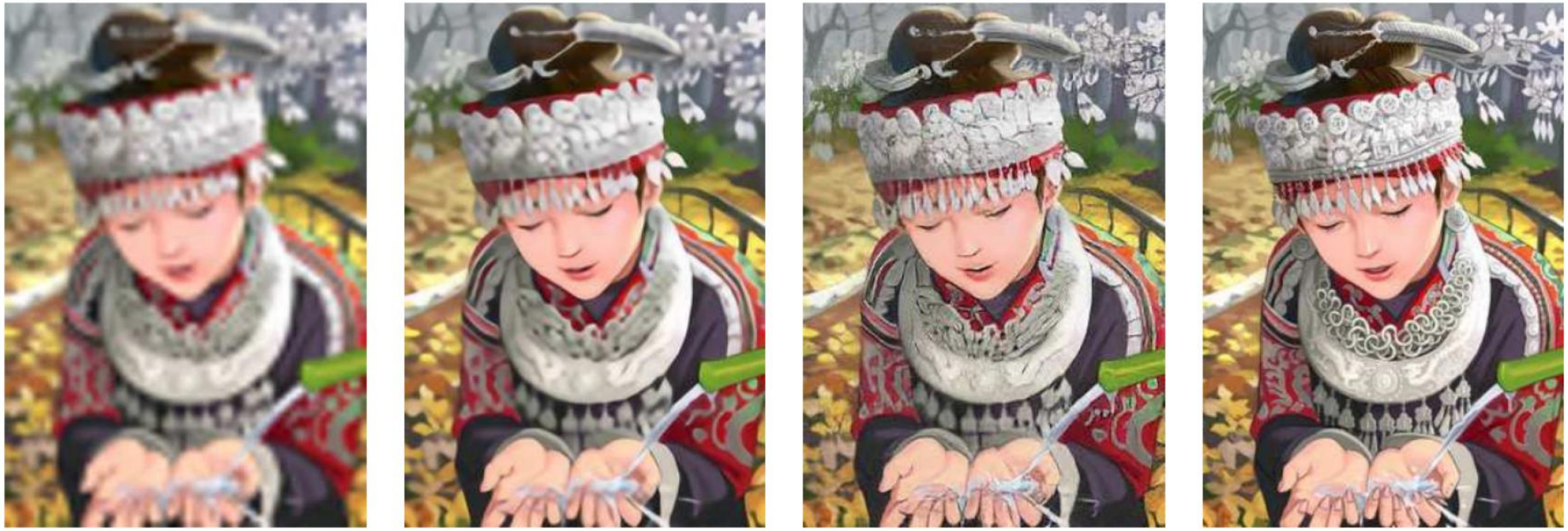


Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

Applications

DCGAN

Unsupervised representation learning with deep convolutional generative adversarial networks

A Radford, L Metz, S Chintala - arXiv preprint arXiv:1511.06434, 2015 - arxiv.org

In recent years, supervised learning with convolutional networks (CNNs) has seen huge adoption in computer vision applications. Comparatively, unsupervised learning with CNNs has received less attention. In this work we hope to help bridge the gap between the success of CNNs for supervised learning and unsupervised learning. We introduce a class of CNNs called deep convolutional generative adversarial networks (DCGANs), that have certain architectural constraints, and demonstrate that they are a strong candidate for …

☆ 99 Cited by 10474 Related articles All 3 versions Import into BibTeX »

Applications

What made this paper special:

- We show that the generators have interesting vector arithmetic properties allowing for easy manipulation of many semantic qualities of generated samples.

Applications

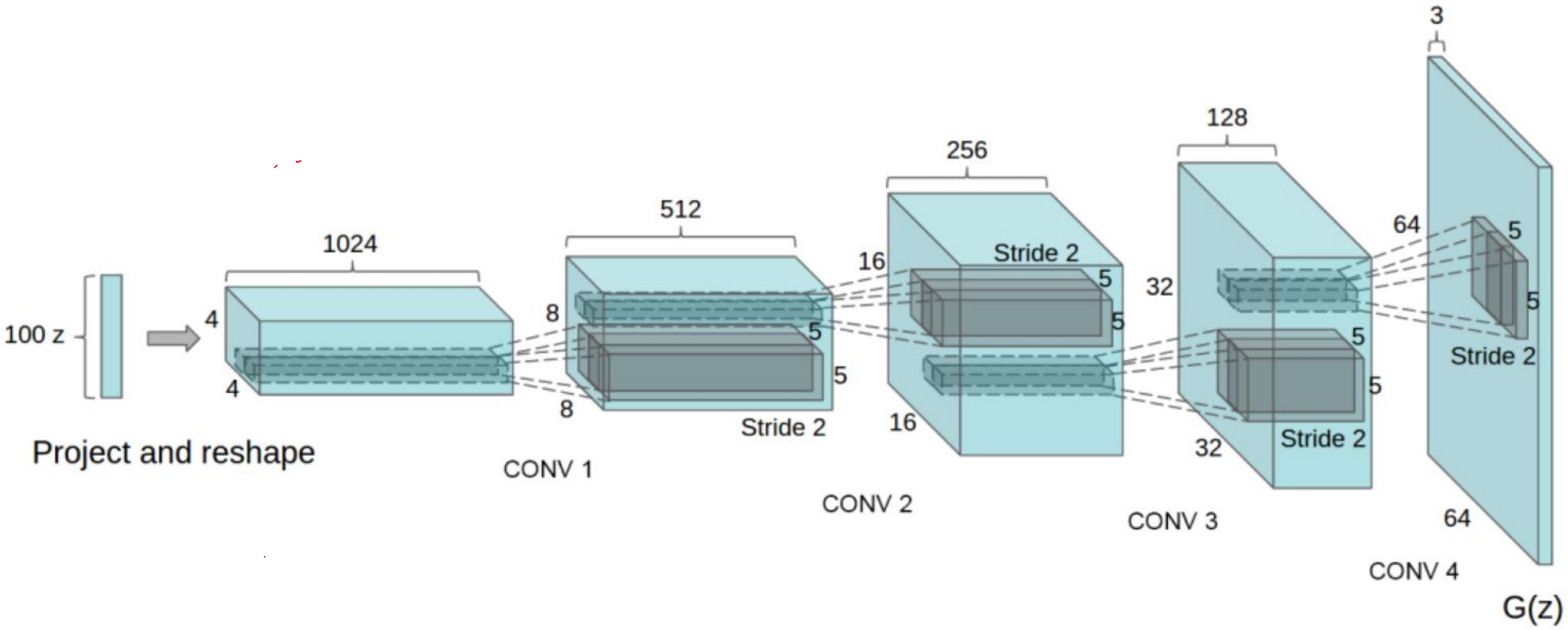


Figure 1: DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64×64 pixel image. Notably, no fully connected or pooling layers are used.

Applications

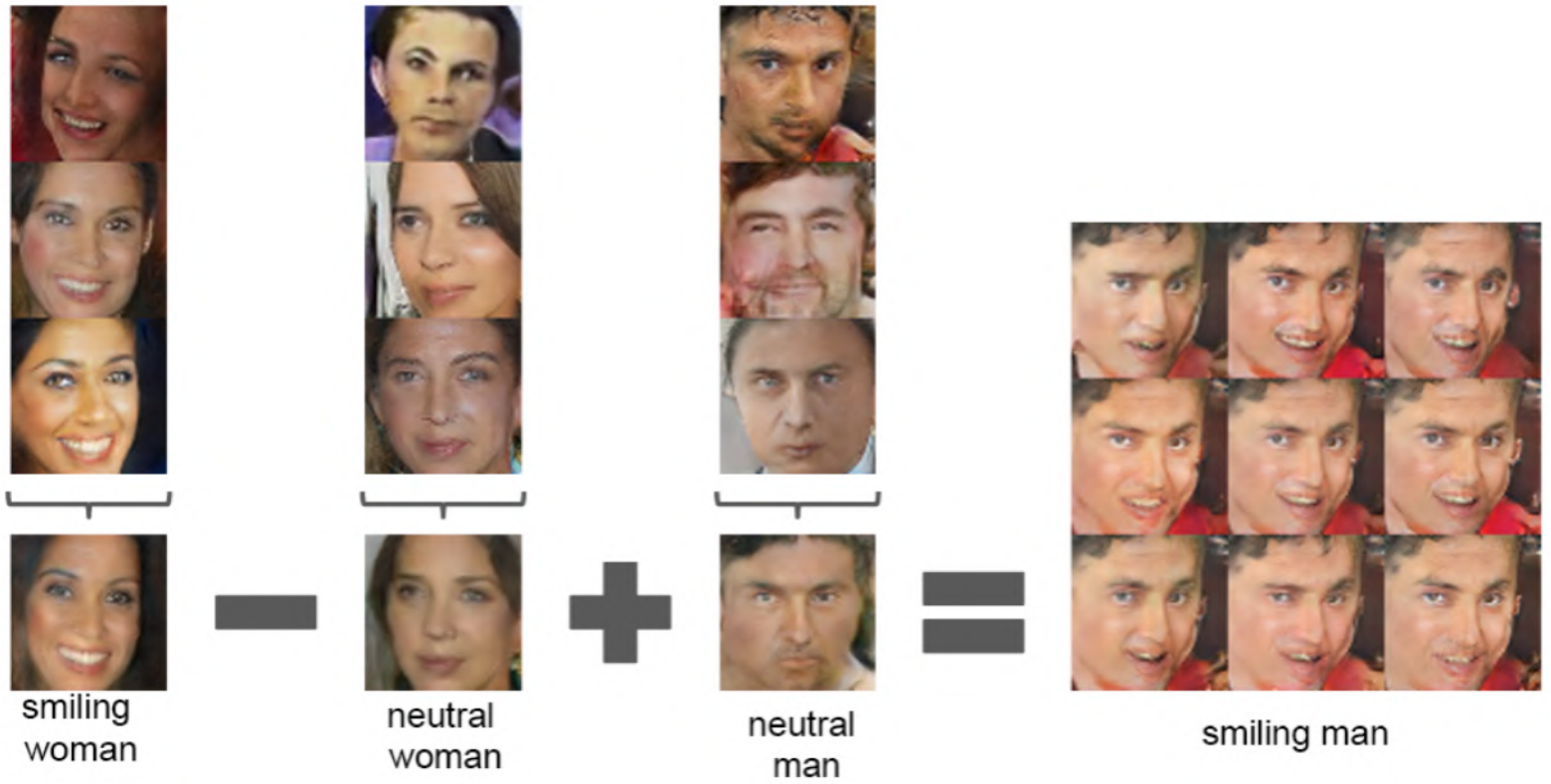


Figure 7: Vector arithmetic for visual concepts. For each column, the Z vectors of samples are averaged. Arithmetic was then performed on the mean vectors creating a new vector Y. The center sample on the right hand side is produce by feeding Y as input to the generator. To demonstrate the interpolation capabilities of the generator, uniform noise sampled with scale +-0.25 was added to Y to produce the 8 other samples. Applying arithmetic in the input space (bottom two examples) results in noisy overlap due to misalignment.

Applications

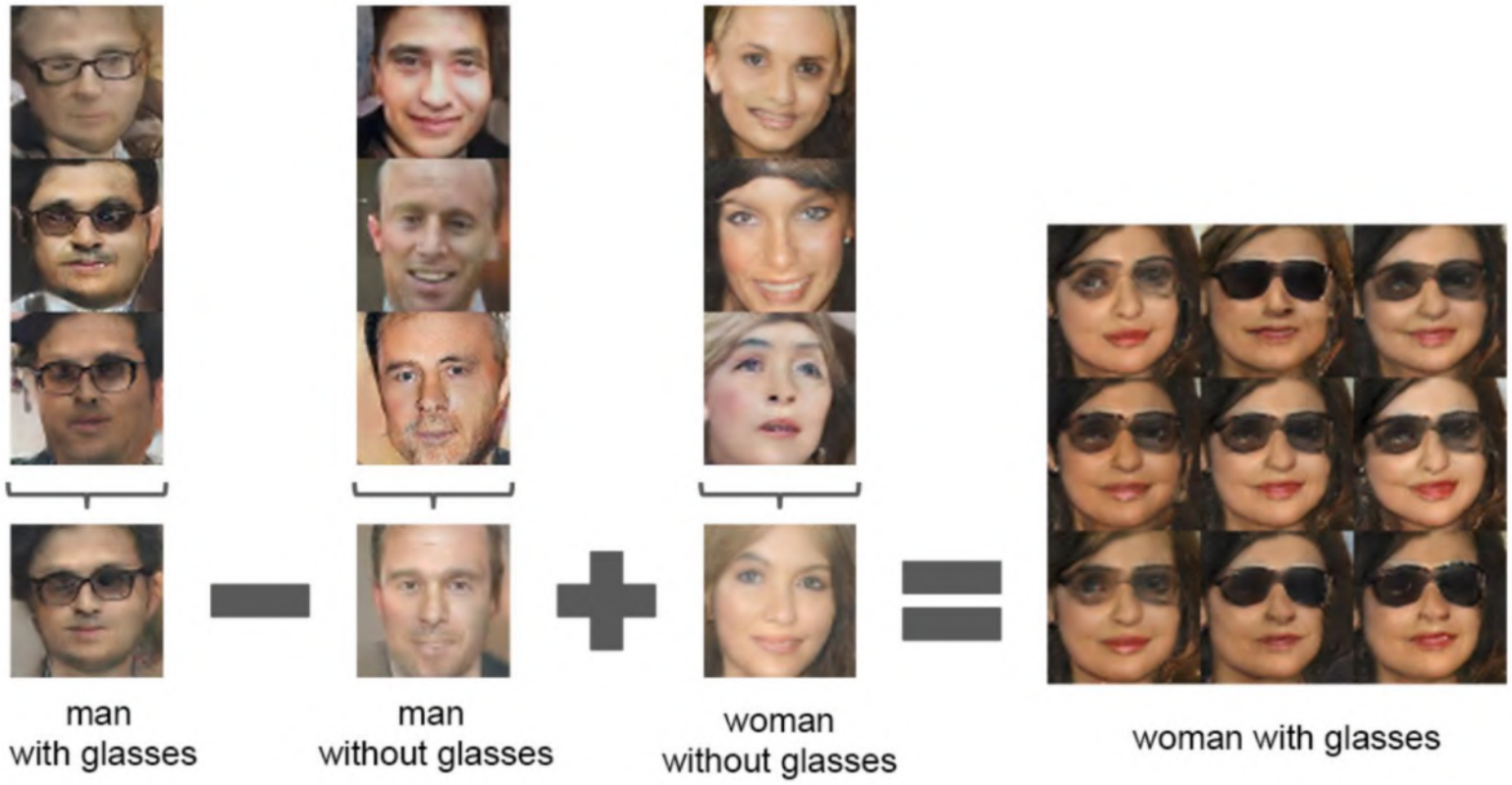


Figure 7: Vector arithmetic for visual concepts. For each column, the Z vectors of samples are averaged. Arithmetic was then performed on the mean vectors creating a new vector Y. The center sample on the right hand side is produce by feeding Y as input to the generator. To demonstrate the interpolation capabilities of the generator, uniform noise sampled with scale +-0.25 was added to Y to produce the 8 other samples. Applying arithmetic in the input space (bottom two examples) results in noisy overlap due to misalignment.

Applications

Figure 8: A "turn" vector was created from four averaged samples of faces looking left vs looking right. By adding interpolations along this axis to random samples we were able to reliably transform their pose.

Applications

CycleGAN

Unpaired image-to-image translation using cycle-consistent adversarial networks

JY Zhu, T Park, P Isola, AA Efros - Proceedings of the IEEE …, 2017 - openaccess.thecvf.com

Image-to-image translation is a class of vision and graphics problems where the goal is to learn the mapping between an input image and an output image using a training set of aligned image pairs. However, for many tasks, paired training data will not be available. We …

☆ 99 Cited by 10193 Related articles All 16 versions Import into BibTeX »

Applications

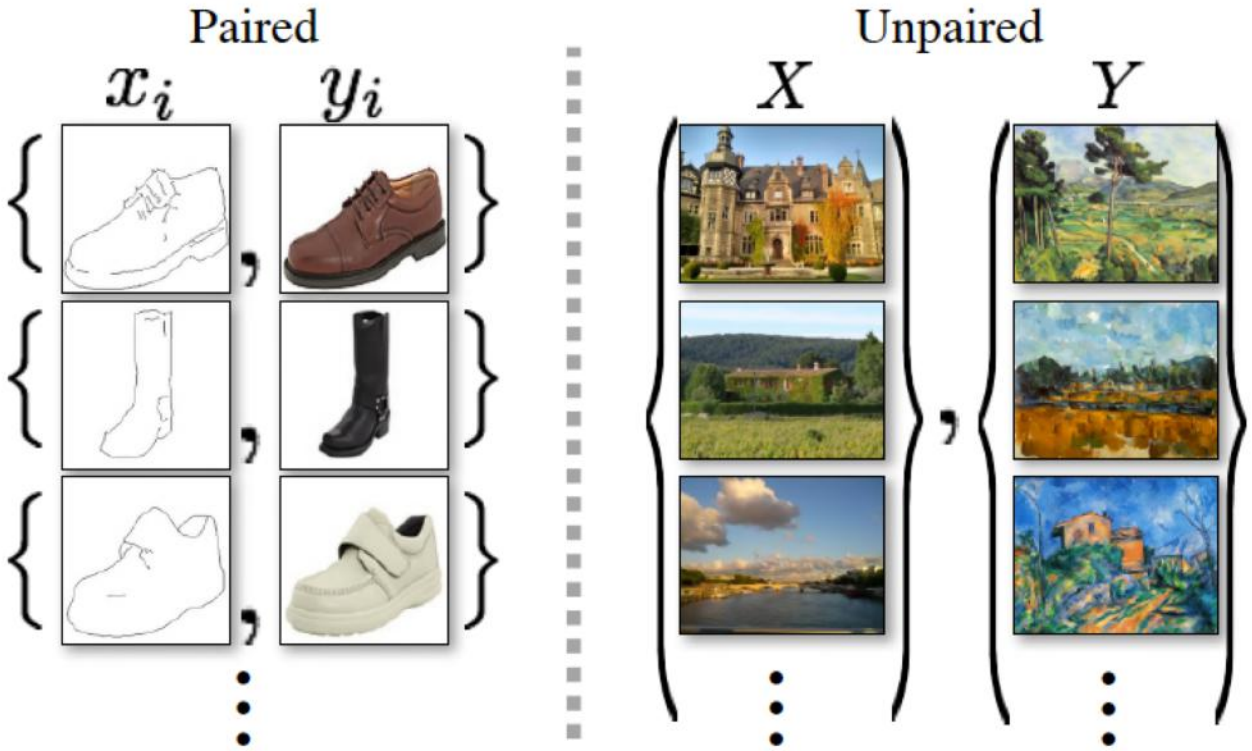


Figure 2: *Paired* training data (left) consists of training examples $\{x_i, y_i\}_{i=1}^{N}$, where the correspondence between x_i and y_i exists [22]. We instead consider *unpaired* training data (right), consisting of a source set $\{x_i\}_{i=1}^{N}$ $(x_i \in X)$ and a target set $\{y_j\}_{j=1}^{M}$ $(y_j \in Y)$, with no information provided as to which x_i matches which y_j.

Applications

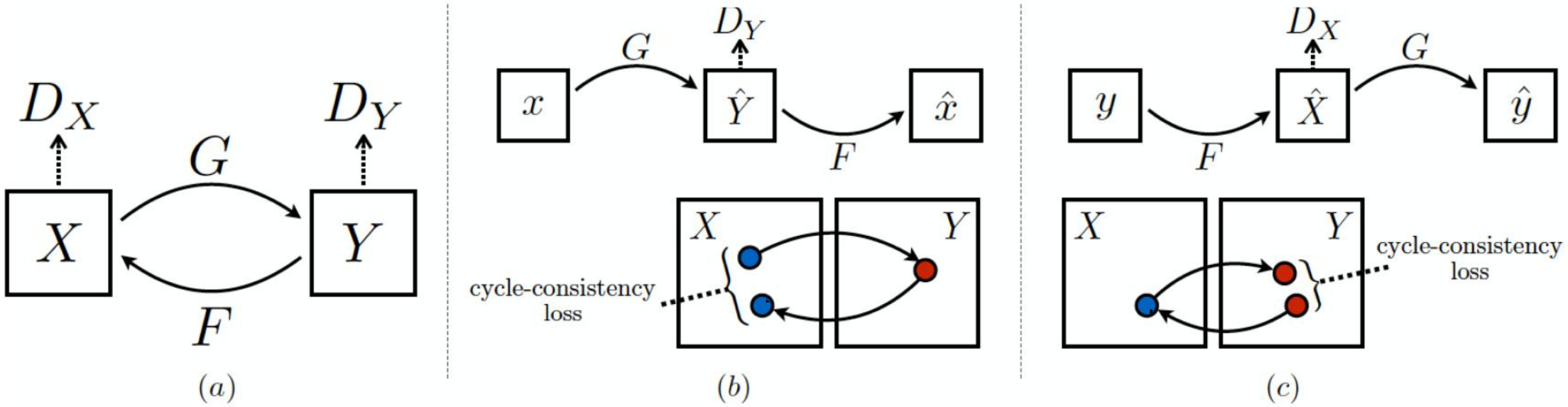


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversarial discriminators D_Y and D_X. D_Y encourages G to translate X into outputs indistinguishable from domain Y, and vice versa for D_X and F. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

Applications

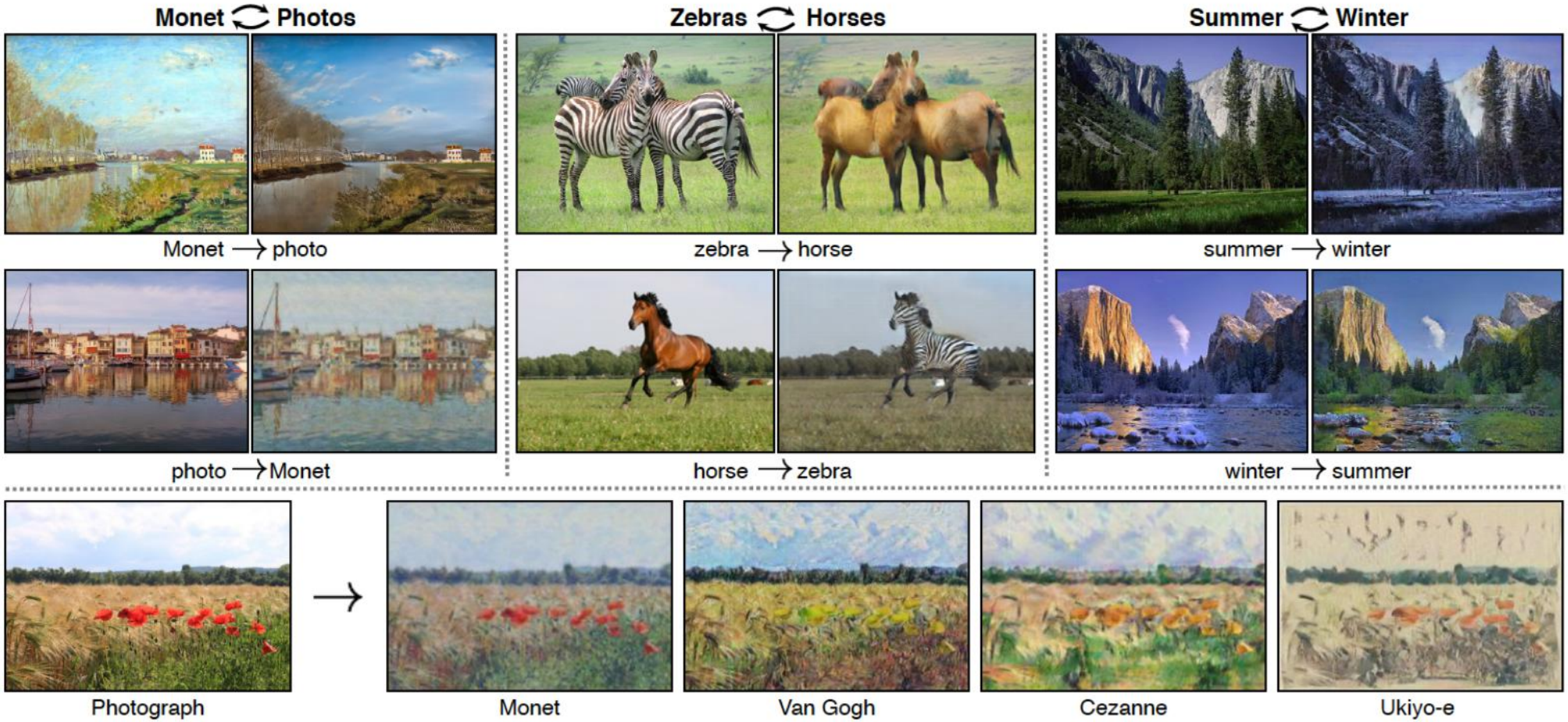


Figure 1: Given any two unordered image collections X and Y, our algorithm learns to automatically "translate" an image from one into the other and vice versa: *(left)* Monet paintings and landscape photos from Flickr; *(center)* zebras and horses from ImageNet; *(right)* summer and winter Yosemite photos from Flickr. Example application *(bottom)*: using a collection of paintings of famous artists, our method learns to render natural photographs into the respective styles.

Applications

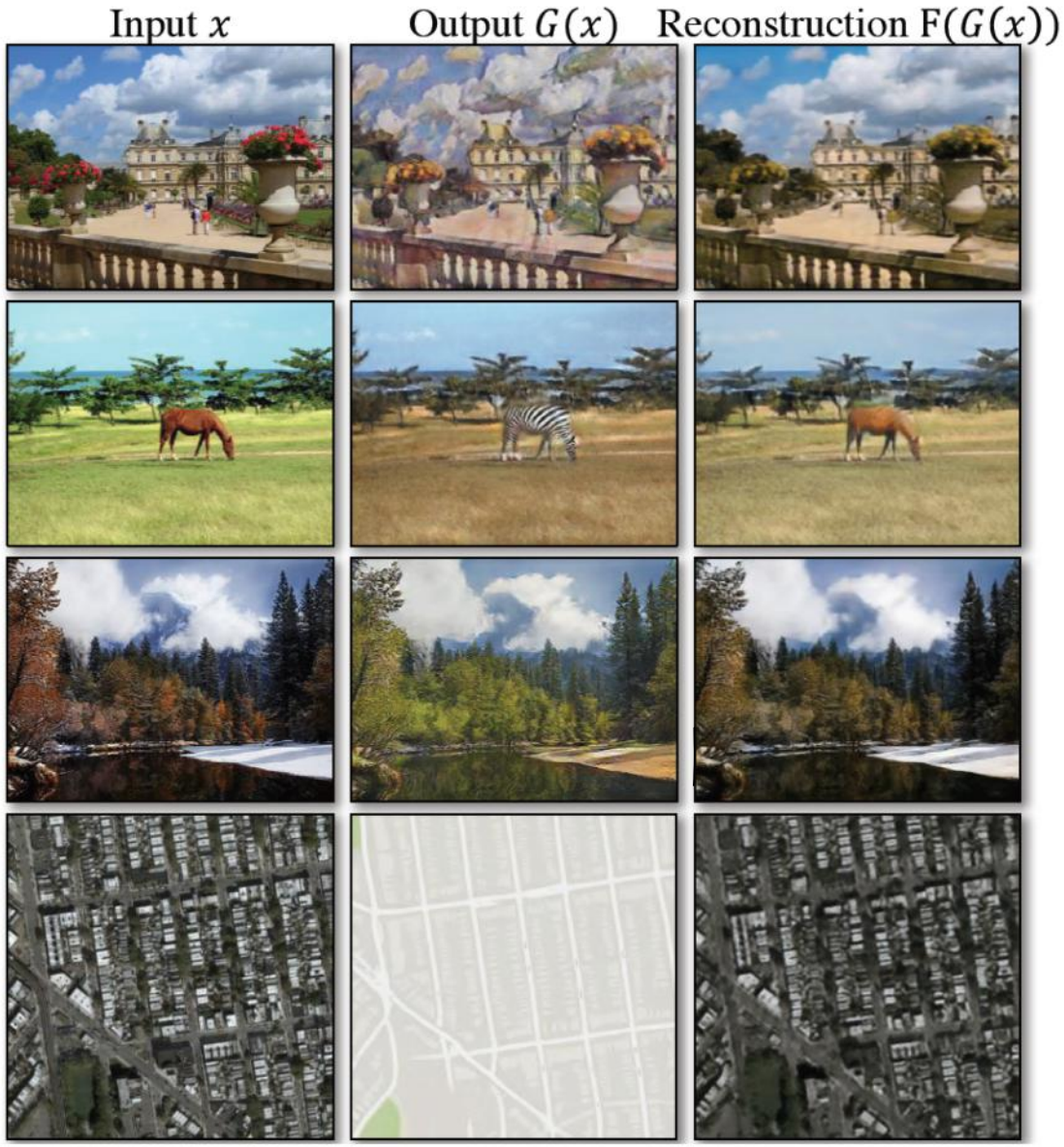
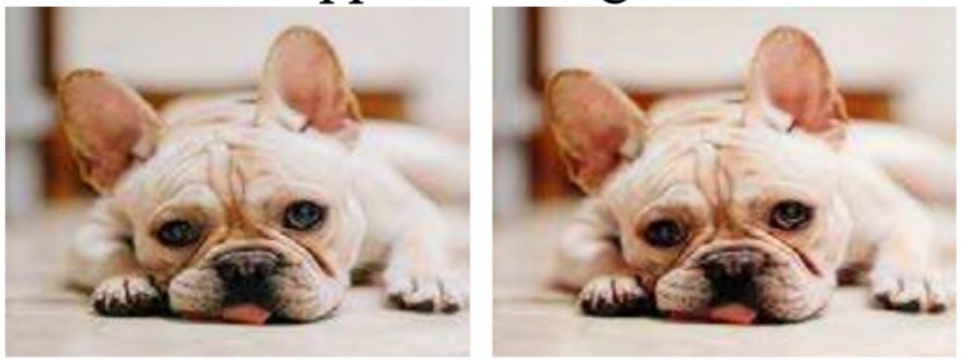


Figure 4: The input images x, output images $G(x)$ and the reconstructed images $F(G(x))$ from various experiments. From top to bottom: photo↔Cezanne, horses↔zebras, winter→summer Yosemite, aerial photos↔Google maps.

Applications

apple → orange



dog → cat

horse → zebra

Some failure cases

Applications

StyleGAN

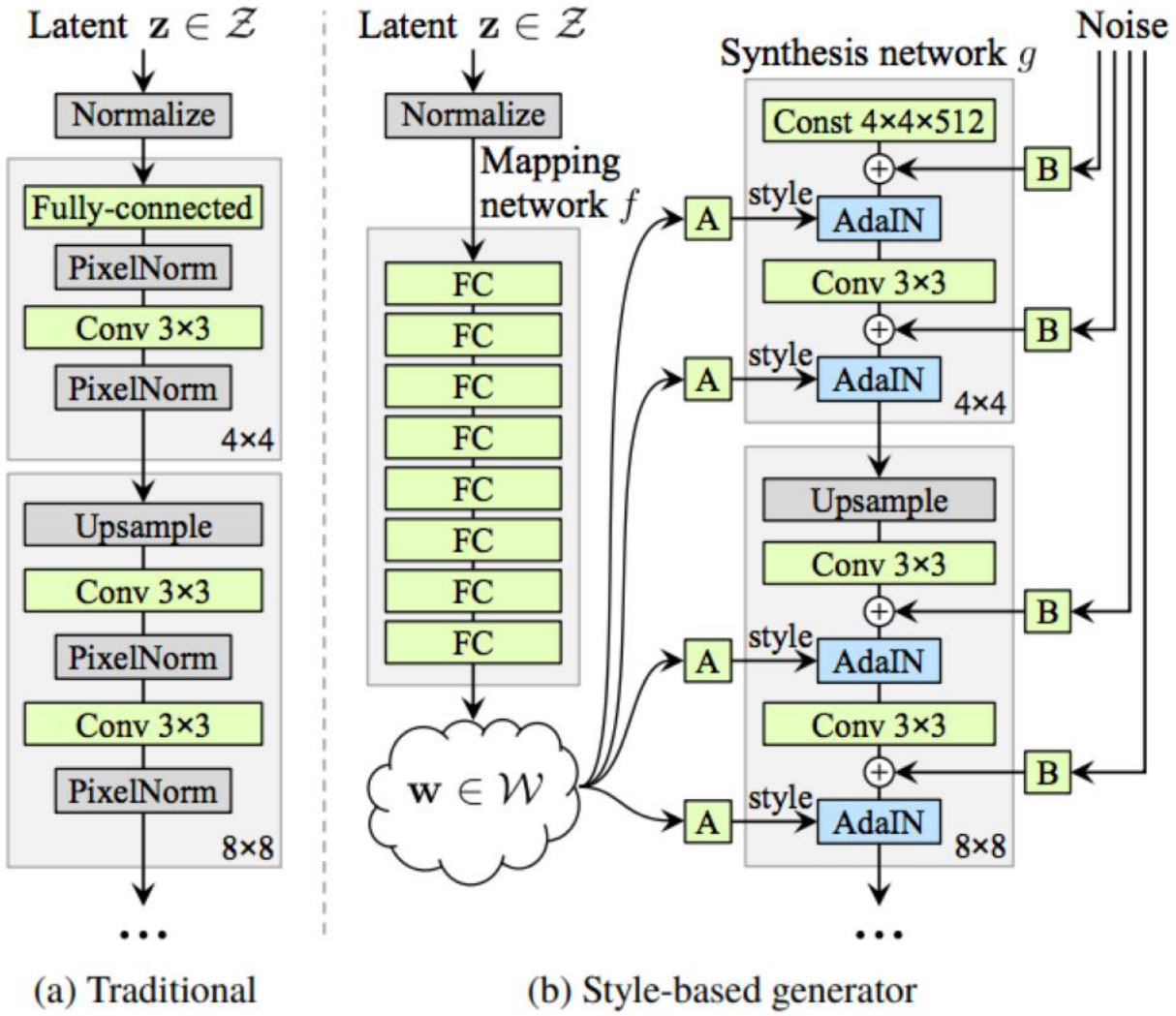
A style-based generator architecture for generative adversarial networks

T Karras, S Laine, T Aila - … of the IEEE/CVF Conference on …, 2019 - openaccess.thecvf.com

We propose an alternative generator architecture for generative adversarial networks, borrowing from style transfer literature. The new architecture leads to an automatically learned, unsupervised separation of high-level attributes (eg, pose and identity when trained on human faces) and stochastic variation in the generated images (eg, freckles, hair), and it enables intuitive, scale-specific control of the synthesis. The new generator improves the state-of-the-art in terms of traditional distribution quality metrics, leads to demonstrably better …

☆ 99 Cited by 2437 Related articles All 9 versions Import into BibTeX »

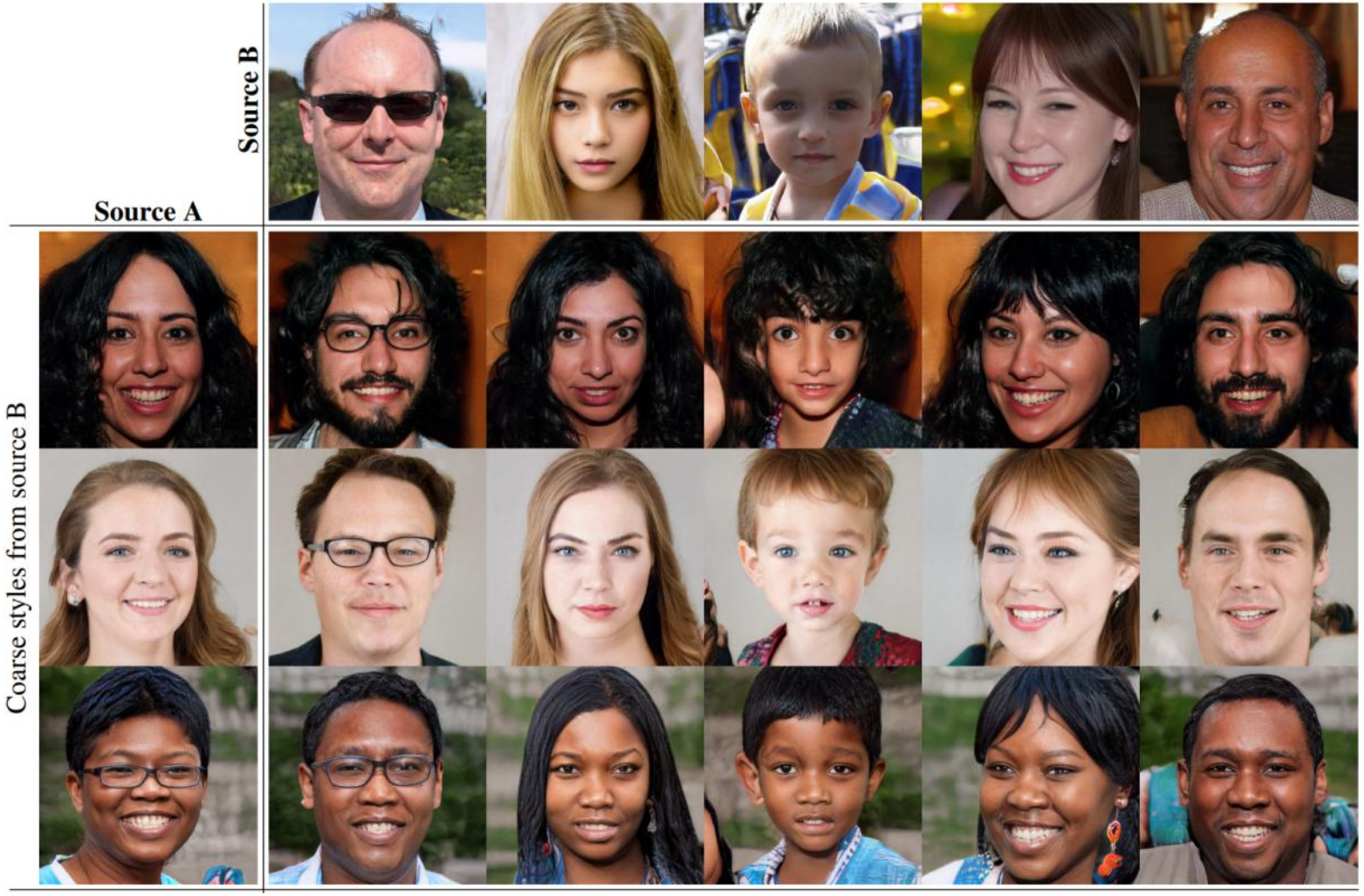
Applications



(a) Traditional (b) Style-based generator

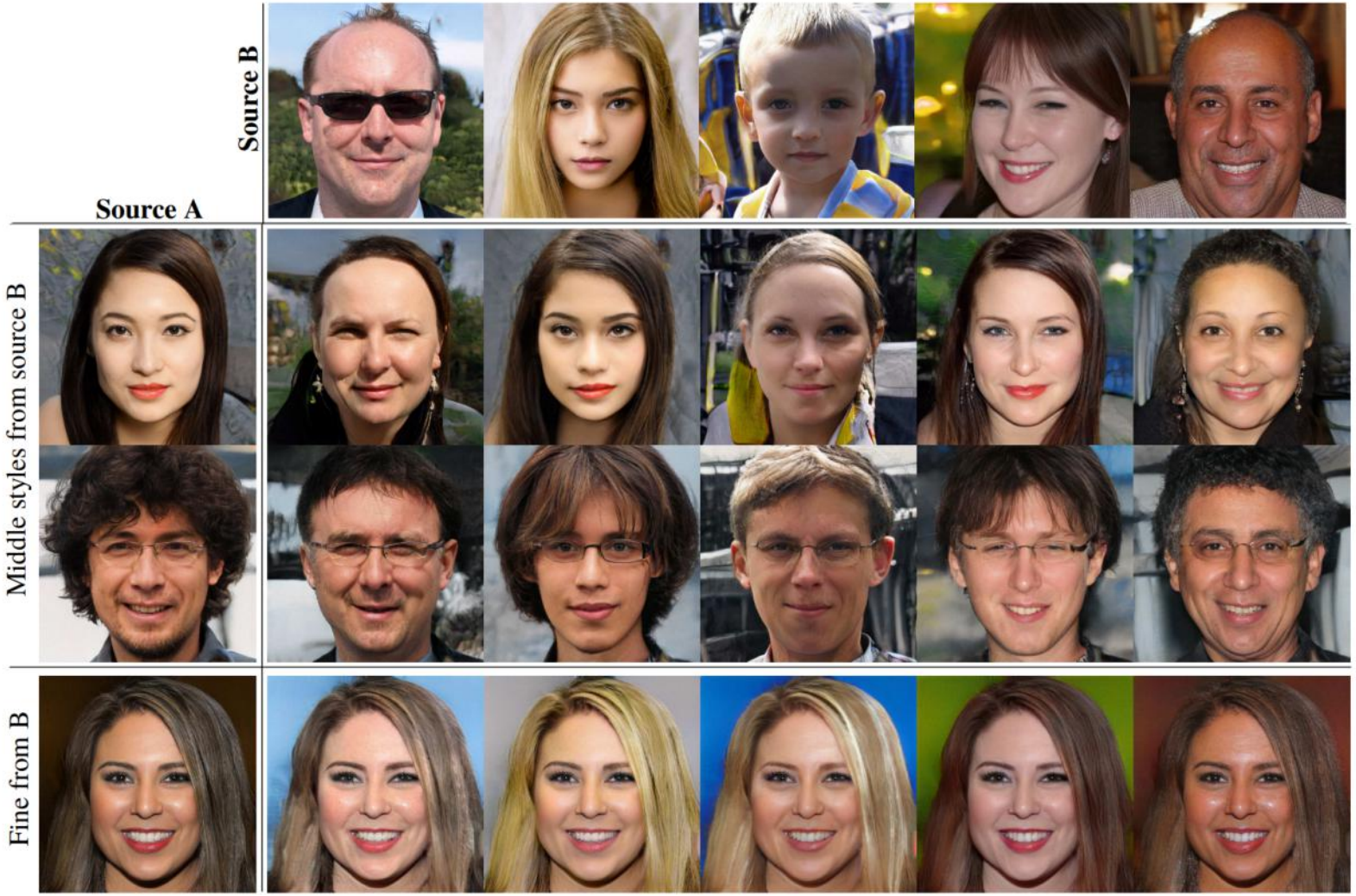
While a traditional generator feeds the latent code though the input layer only, we first map the input to an intermediate latent space W, which then controls the generator through adaptive instance normalization (AdaIN) at each convolution layer. Gaussian noise is added after each convolution, before evaluating the nonlinearity. Here “A” stands for a learned affine transform, and “B” applies learned per-channel scaling factors to the noise input. The mapping network f consists of 8 layers and the synthesis network g consists of 18 layers— two for each resolution (4x4−1024x1024). Our generator has a total of 26.2M trainable parameters, compared to 23.1M in the traditional generator.

Applications



Two sets of images were generated from their respective latent codes (sources A and B); the rest of the images were generated by copying a specified subset of styles from source B and taking the rest from source A. Copying the styles corresponding to coarse spatial resolutions (4 2 – 8 2) brings high-level aspects such as pose, general hair style, face shape, and eyeglasses from source B, while all colors (eyes, hair, lighting) and finer facial features resemble A. If we instead copy the styles of middle resolutions (162 – 322) from B, we inherit smaller scale facial features, hair style, eyes open/closed from B, while the pose, general face shape, and eyeglasses from A are preserved. Finally, copying the fine styles (642 – 10242) from B brings mainly the color scheme and microstructure.

Applications



Two sets of images were generated from their respective latent codes (sources A and B); the rest of the images were generated by copying a specified subset of styles from source B and taking the rest from source A. Copying the styles corresponding to coarse spatial resolutions (4 2 – 8 2) brings high-level aspects such as pose, general hair style, face shape, and eyeglasses from source B, while all colors (eyes, hair, lighting) and finer facial features resemble A. If we instead copy the styles of middle resolutions (162 – 322) from B, we inherit smaller scale facial features, hair style, eyes open/closed from B, while the pose, general face shape, and eyeglasses from A are preserved. Finally, copying the fine styles (642 – 10242) from B brings mainly the color scheme and microstructure.

Applications

CGAN

Conditional generative adversarial nets

M Mirza, S Osindero - arXiv preprint arXiv:1411.1784, 2014 - arxiv.org

Generative Adversarial Nets [8] were recently introduced as a novel way to train generative models. In this work we introduce the conditional version of generative adversarial nets, which can be constructed by simply feeding the data, y, we wish to condition on to both the …

☆ 99 Cited by 6213 Related articles All 4 versions Import into BibTeX »

Applications

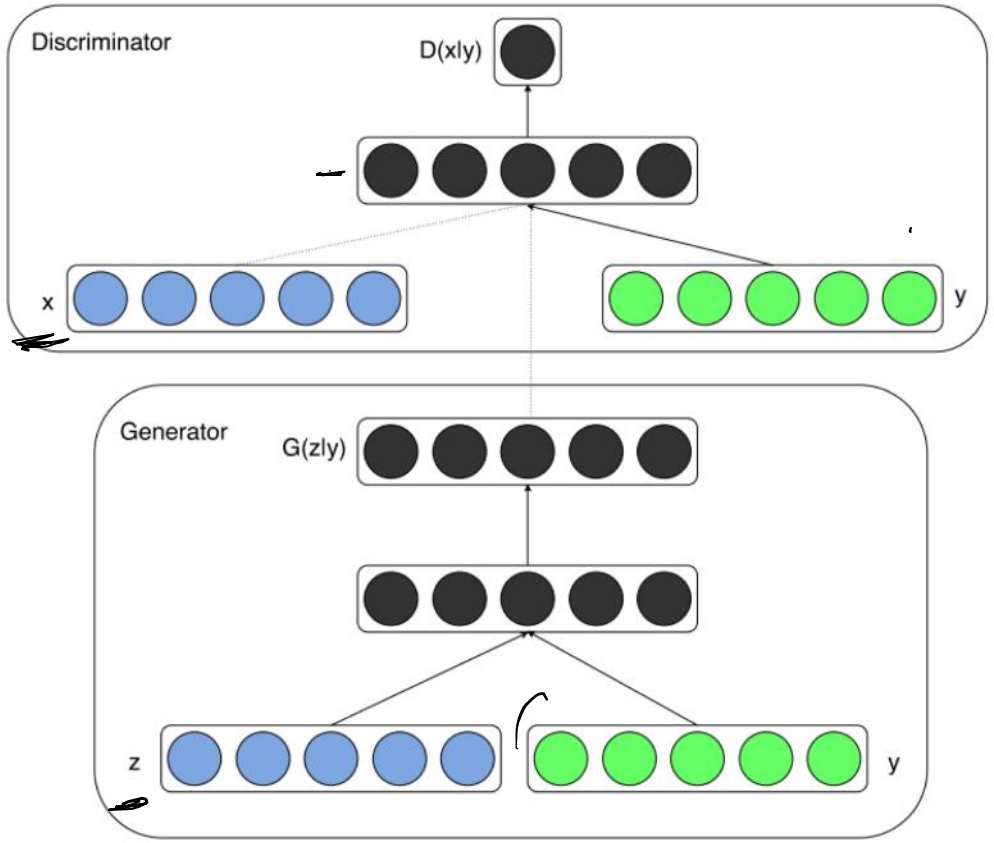


Figure 1: Conditional adversarial net

- Generative adversarial nets can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y.
- y could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding y into the both the discriminator and generator as additional input layer.

Applications

Zhang, Z., Song, Y., & Qi, H. (2017). Age progression/regression by conditional adversarial autoencoder. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5810-5818).

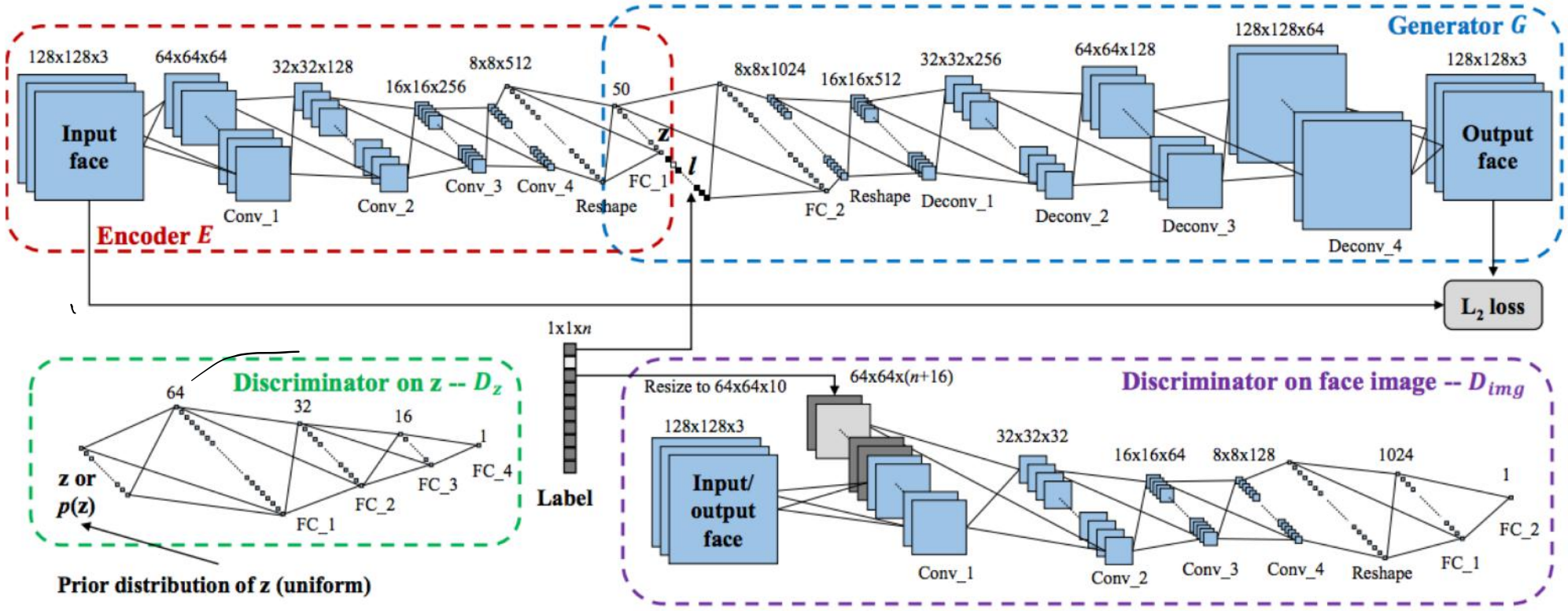


Figure 3. Structure of the proposed CAAE network for age progression/regression. The encoder E maps the input face to a vector z (personality). Concatenating the label l (age) to z, the new latent vector $[z, l]$ is fed to the generator G. Both the encoder and the generator are updated based on the L_2 loss between the input and output faces. The discriminator D_z imposes the uniform distribution on z, and the discriminator D_{img} forces the output face to be photo-realistic and plausible for a given age label.

Applications

Xiao, Y., & Zhao, Y. (2021, August). Preserving Gender and Identity in Face Age Progression of Infants and Toddlers. In *2021 IEEE International Joint Conference on Biometrics (IJCB)* (pp. 1-8).

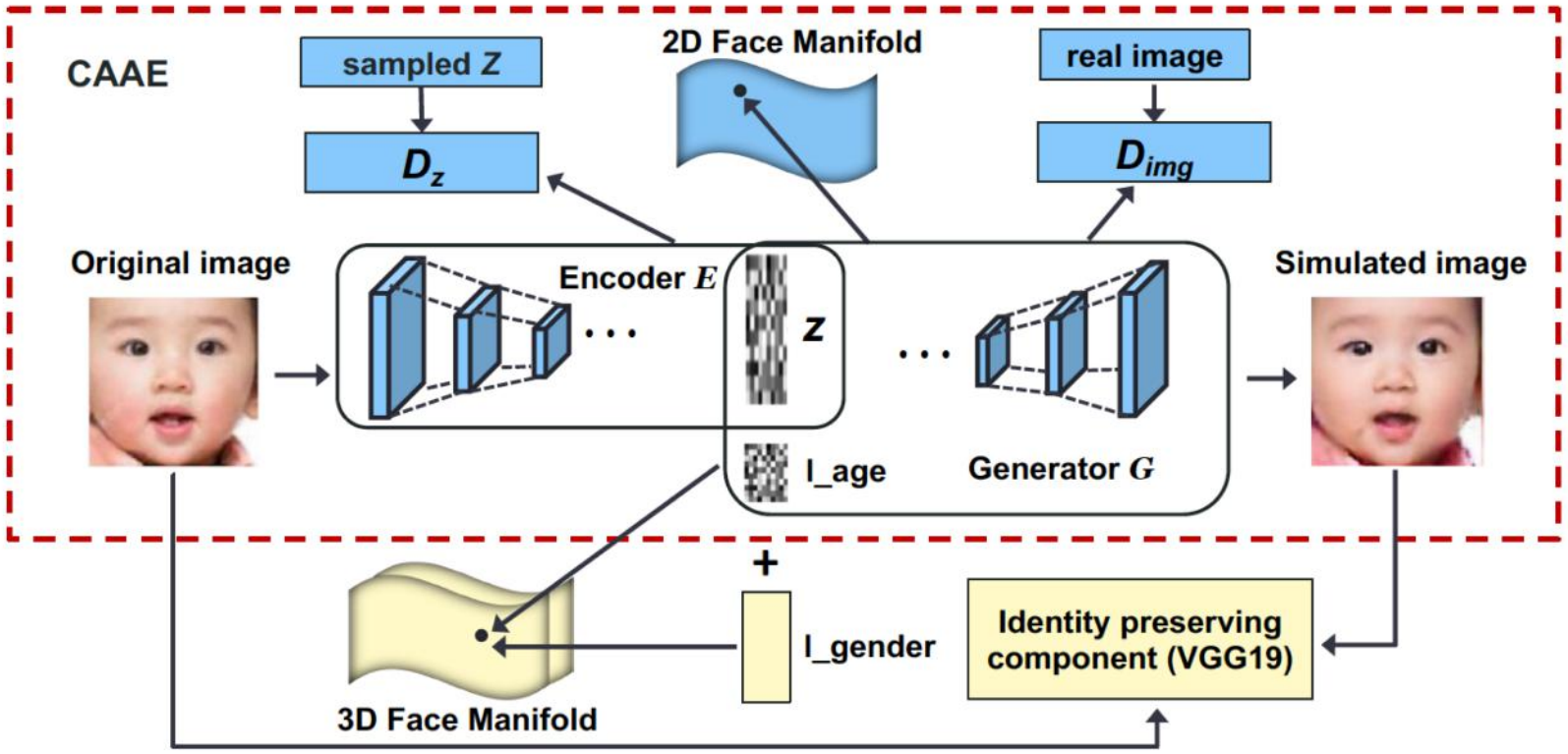
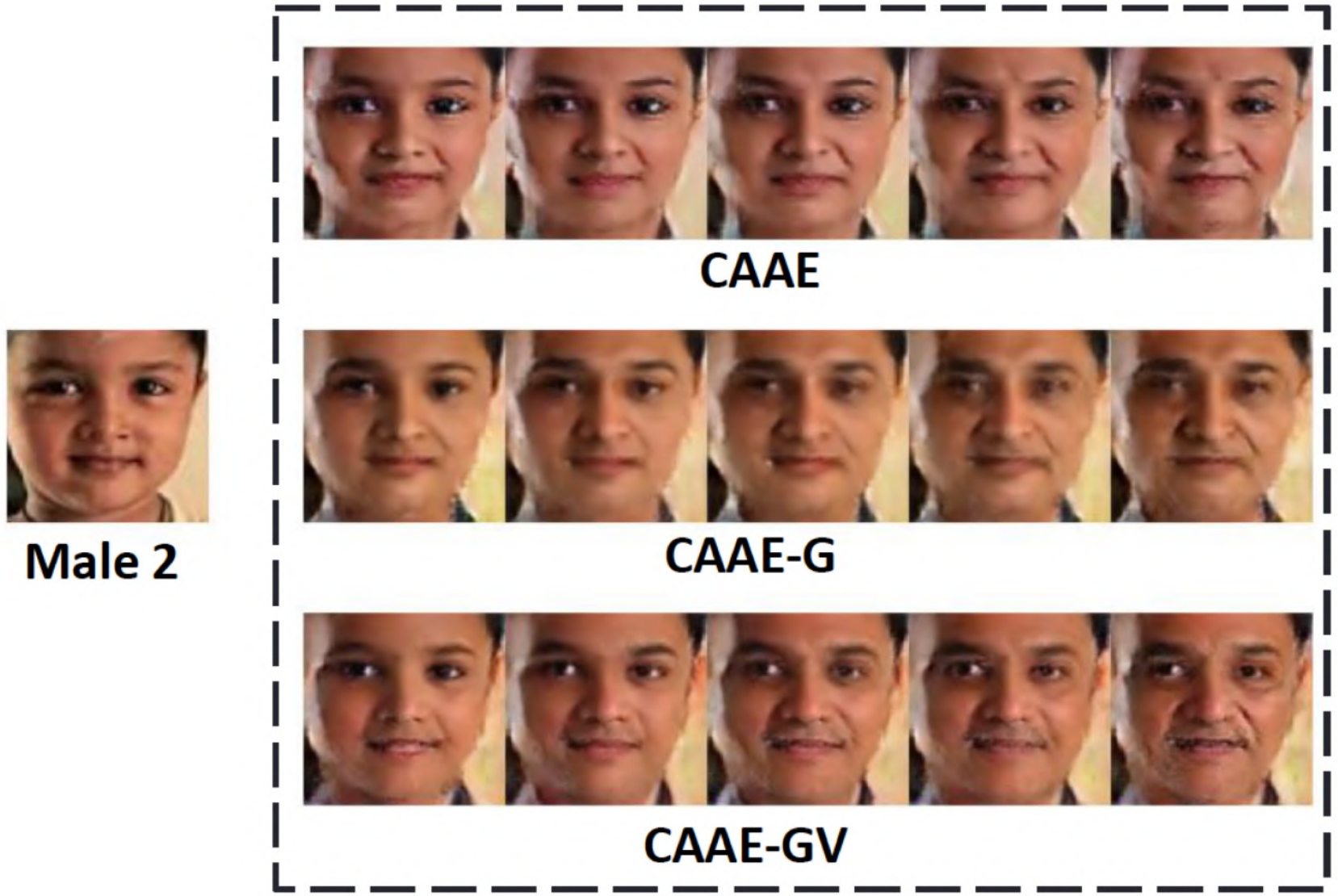


Figure 1: **Architecture of our Proposed Model.**

Applications



Applications

PG GAN

Progressive growing of gans for improved quality, stability, and variation

T Karras, T Aila, S Laine, J Lehtinen - arXiv preprint arXiv:1710.10196, 2017 - arxiv.org

We describe a new training methodology for generative adversarial networks. The key idea is to grow both the generator and discriminator progressively: starting from a low resolution, we add new layers that model increasingly fine details as training progresses. This both speeds the training up and greatly stabilizes it, allowing us to produce images of unprecedented quality, eg, CelebA images at 1024^ 2. We also propose a simple way to increase the variation in generated images, and achieve a record inception score of 8.80 in …

☆ 99 Cited by 3585 Related articles All 7 versions Import into BibTeX »

Applications

PG GAN

- Our primary contribution is a training methodology for GANs where we start with low-resolution images, and then progressively increase the resolution by adding layers to the networks as visualized in Figure 1. This incremental nature allows the training to first discover large-scale structure of the image distribution and then shift attention to increasingly finer scale detail, instead of having to learn all scales simultaneously.

- We use generator and discriminator networks that are mirror images of each other and always grow in synchrony. All existing layers in both networks remain trainable throughout the training process.

Applications

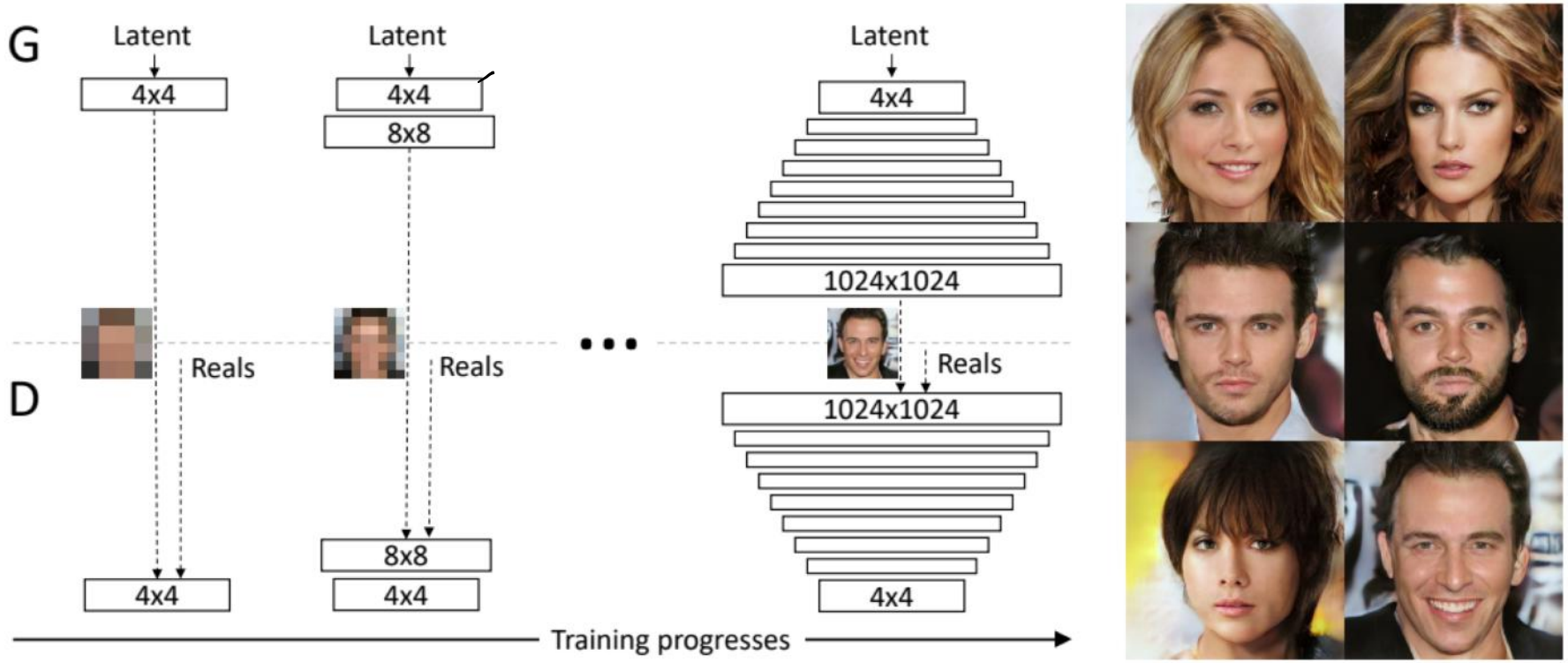


Figure 1: Our training starts with both the generator (G) and discriminator (D) having a low spatial resolution of 4×4 pixels. As the training advances, we incrementally add layers to G and D, thus increasing the spatial resolution of the generated images. All existing layers remain trainable throughout the process. Here $N \times N$ refers to convolutional layers operating on $N \times N$ spatial resolution. This allows stable synthesis in high resolutions and also speeds up training considerably. One the right we show six example images generated using progressive growing at 1024×1024.

Applications

Mao et al. (2016b) (128 × 128) Gulrajani et al. (2017) (128 × 128) Our (256 × 256)

Figure 6: Visual quality comparison in LSUN BEDROOM; pictures copied from the cited articles.

Applications

StackGAN

Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks

H Zhang, T Xu, H Li, S Zhang, X Wang… - Proceedings of the …, 2017 - openaccess.thecvf.com

Synthesizing high-quality images from text descriptions is a challenging problem in computer vision and has many practical applications. Samples generated by existing text-to-image approaches can roughly reflect the meaning of the given descriptions, but they fail to contain necessary details and vivid object parts. In this paper, we propose Stacked Generative Adversarial Networks (StackGAN) to generate 256x256 photo-realistic images conditioned on text descriptions. We decompose the hard problem into more manageable …

☆ 99 Cited by 1878 Related articles All 29 versions Import into BibTeX »

Applications

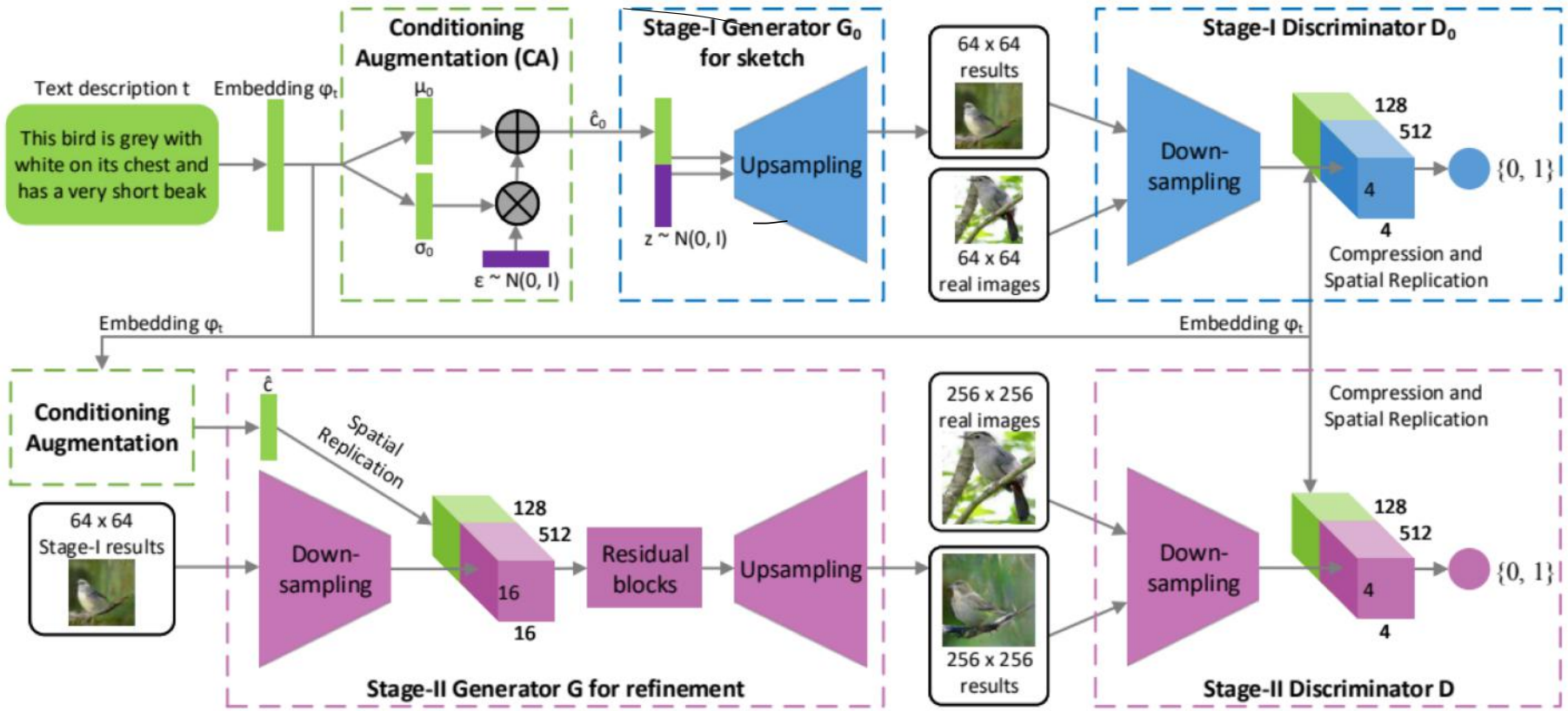


Figure 2. The architecture of the proposed StackGAN. The Stage-I generator draws a low-resolution image by sketching rough shape and basic colors of the object from the given text and painting the background from a random noise vector. Conditioned on Stage-I results, the Stage-II generator corrects defects and adds compelling details into Stage-I results, yielding a more realistic high-resolution image.

Applications

A unique combination of Conditional GANs and Progressively Growing GANs. More results in the paper.

Figure 3. Example results by our StackGAN, GAWWN [24], and GAN-INT-CLS [26] conditioned on text descriptions from CUB test set.

Applications

A unique combination of Conditional GANs and Progressively Growing GANs. More results in the paper.

Figure 5. Samples generated by our StackGAN from unseen texts in CUB test set. Each column lists the text description, images generated from the text by Stage-I and Stage-II of StackGAN.

Applications

BigGAN

Large scale GAN training for high fidelity natural image synthesis

A Brock, J Donahue, K Simonyan - arXiv preprint arXiv:1809.11096, 2018 - arxiv.org

Despite recent progress in generative image modeling, successfully generating high-resolution, diverse samples from complex datasets such as ImageNet remains an elusive goal. To this end, we train Generative Adversarial Networks at the largest scale yet attempted, and study the instabilities specific to such scale. We find that applying orthogonal regularization to the generator renders it amenable to a simple" truncation trick," allowing fine control over the trade-off between sample fidelity and variety by reducing the variance of …

☆ 99 Cited by 2188 Related articles All 4 versions Import into BibTeX »

Applications

Capitalizes on many previous findings

3 SCALING UP GANS

In this section, we explore methods for scaling up GAN training to reap the performance benefits of larger models and larger batches. As a baseline, we employ the SA-GAN architecture of Zhang et al. (2018), which uses the hinge loss (Lim & Ye, 2017; Tran et al., 2017) GAN objective. We provide class information to **G** with class-conditional BatchNorm (Dumoulin et al., 2017; de Vries et al., 2017) and to **D** with projection (Miyato & Koyama, 2018). The optimization settings follow Zhang et al. (2018) (notably employing Spectral Norm in **G**) with the modification that we halve the learning rates and take two **D** steps per **G** step. For evaluation, we employ moving averages of **G**'s weights following Karras et al. (2018); Mescheder et al. (2018); Yazc et al. (2018), with a decay of 0.9999. We use Orthogonal Initialization (Saxe et al., 2014), whereas previous works used $\mathcal{N}(0, 0.02I)$ (Radford et al., 2016) or Xavier initialization (Glorot & Bengio, 2010). Each model is trained on 128 to 512 cores of a Google TPUv3 Pod (Google, 2018), and computes BatchNorm statistics in **G** across all devices, rather than per-device as is typical. We find progressive growing (Karras et al., 2018) unnecessary even for our 512×512 models. Additional details are in Appendix C.

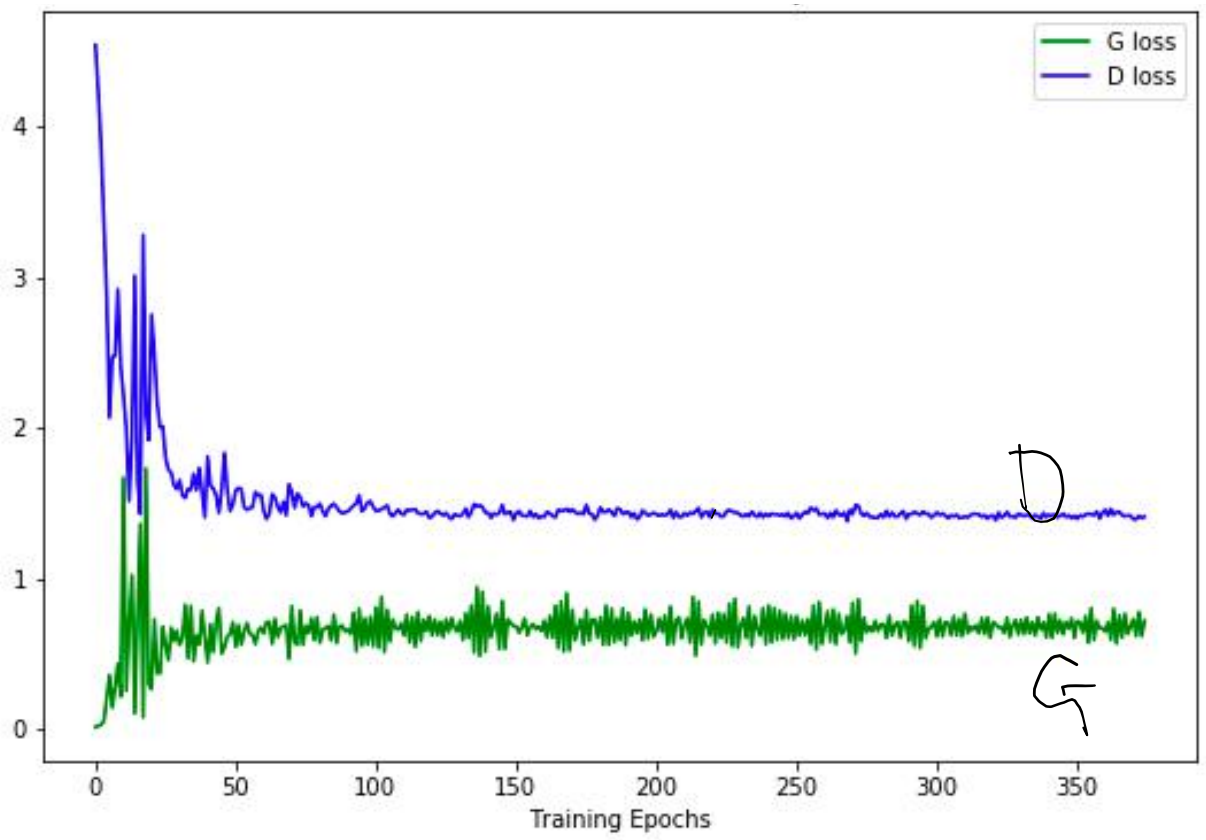
Applications

State-of-the-art on ImageNet generation. Difficult to implement on a local machine.

Figure 1: Class-conditional samples generated by our model.

“ When trained on ImageNet at 128×128 resolution, our models (BigGANs) achieve an Inception Score (IS) of 166.5 and Frechet Inception Distance (FID) of 7.4, improving over the previous best IS of 52.52 and FID of 18.65. “

GAN training can be ***very*** unstable



- This picture? You get this if everything is working well
- Turns out, equilibria are hard to find
  - With every other net we've trained, the loss function is with respect to a fixed target value we're trying to hit
  - Here, we have a "moving target" (G's target is fool D, D's target is detect G)
- These curves can oscillate a lot

- What happens if the discriminator ever becomes perfect at detecting G's fakes?
  - The discriminator always returns probability zero
  - The gradient through D is zero
  - The generator stops training

Mode Collapse

- Generator loss says: “generate an output that looks real”
- It does not say: “generate ***every*** output that looks real”
- The generator can “cheat” by finding one output / a few outputs that reliably fool the discriminator (the specific one(s) it finds can shift over training)

[PDF] **Improved techniques** for **training gans**

T Salimans, I Goodfellow, W Zaremba… - Advances in neural …, 2016 - papers.nips.cc

We present a variety of new architectural features and training procedures that we apply to the generative adversarial networks (GANs) framework. Using our new techniques, we achieve state-of-the-art results in semi-supervised classification on MNIST, CIFAR-10 and …

☆ 99 Cited by 5739 Related articles All 12 versions Import into BibTeX »