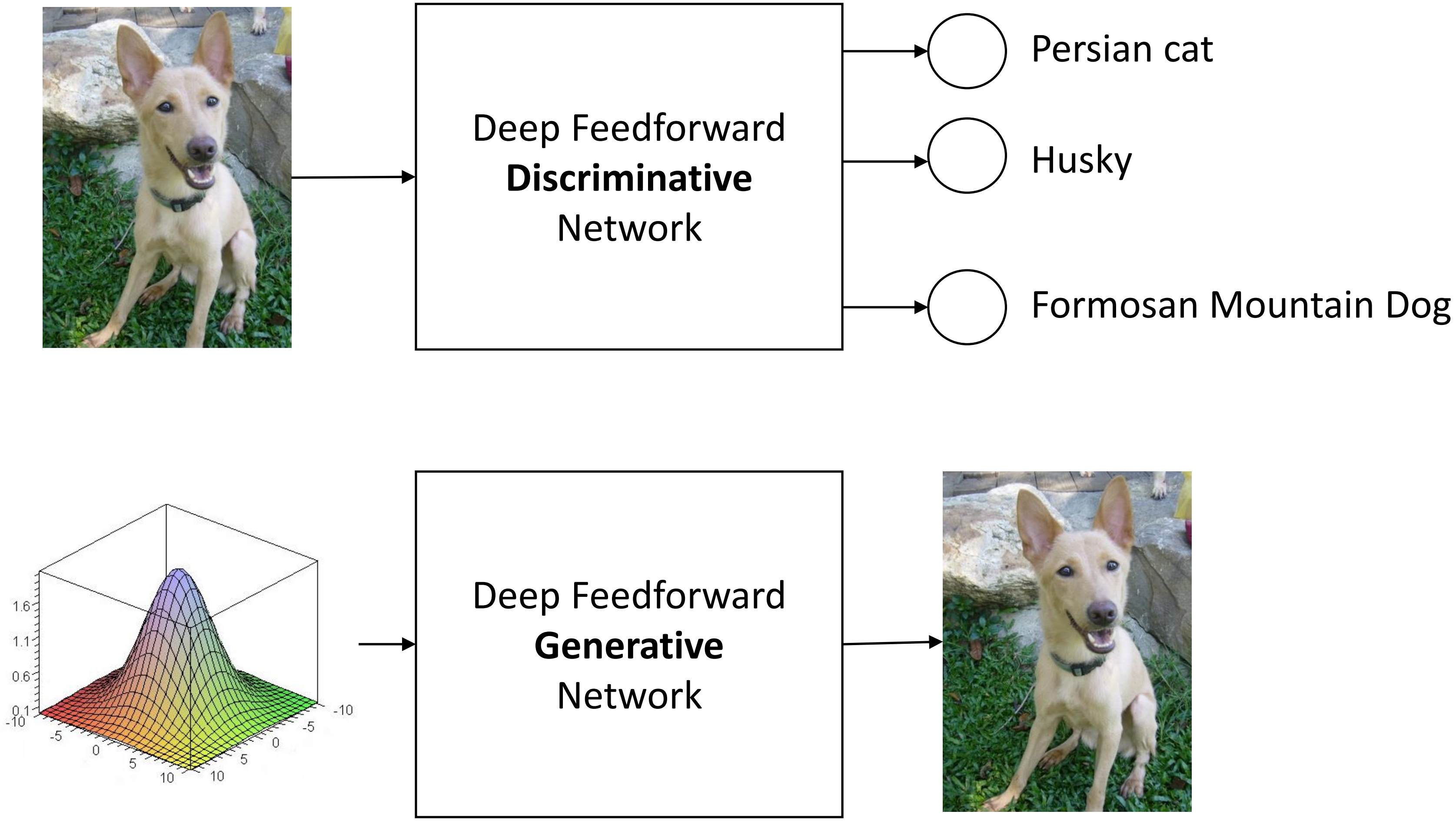
Deep Feedforward Generative Models

Ming-Yu Liu, NVIDIA

Deep Feedforward Generative Models

- A generative model is a model for randomly generating data.
- Many deep learning-based generative models exist including Restrictive Boltzmann Machine (RBM), Deep Boltzmann Machines (DBM), Deep Belief Networks (DBN) ….
- We will focus on deep feedforward generative models.
- We will focus on models that maps a random sample z from a parametric probability distribution to an image x.
  - Variational Autoencoders (Kingma and Welling 2014)
  - Generative Adversarial Networks (Goodfellow et al 2014)

Comparison

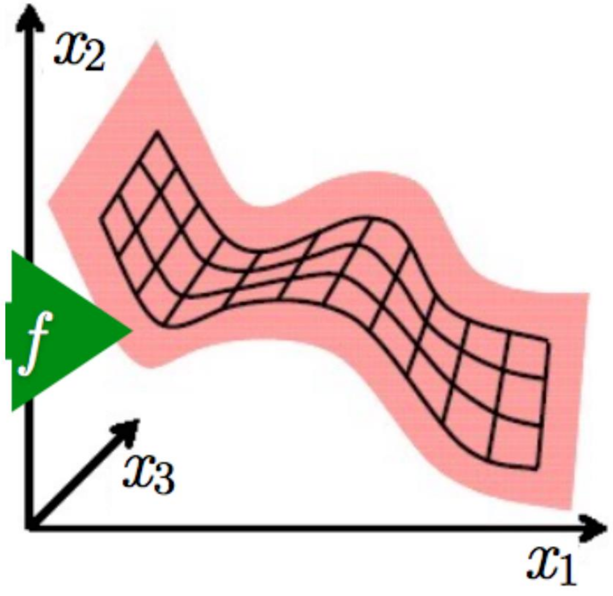

Deep Feedforward
Discriminative
Network
Persian cat
Husky
Formosan Mountain Dog
Deep Feedforward
Generative
Network

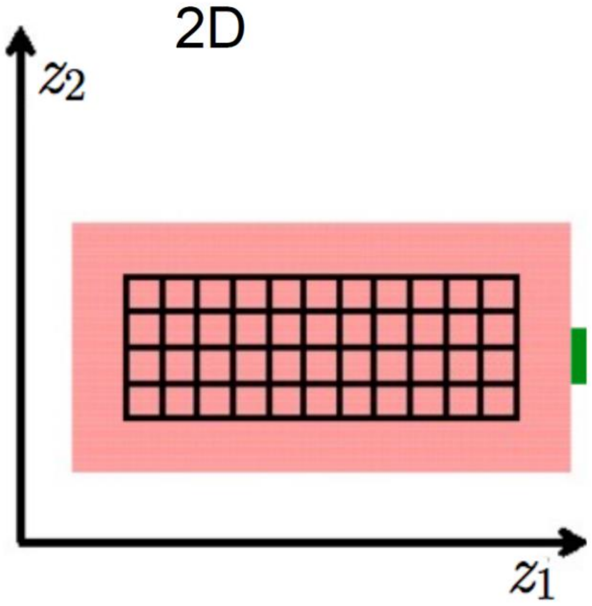
Comparison

Model	Input	Output	Operation
Deep Feedforward Discriminative Networks	• Image • High-dimensional	• A probability distribution of class labels • Low-dimensional	• “Compression” • Many down-sampling operations
Deep Feedforward Generative Networks	• Random sample from a parametric probabilistic distribution • Low-dimensional	• A probability distribution of images • High-dimensional	• “Decompression” • Many up-sampling operations

Manifold Hypothesis

- Structured high-dimensional data (images) live in a low-dimensional manifold.





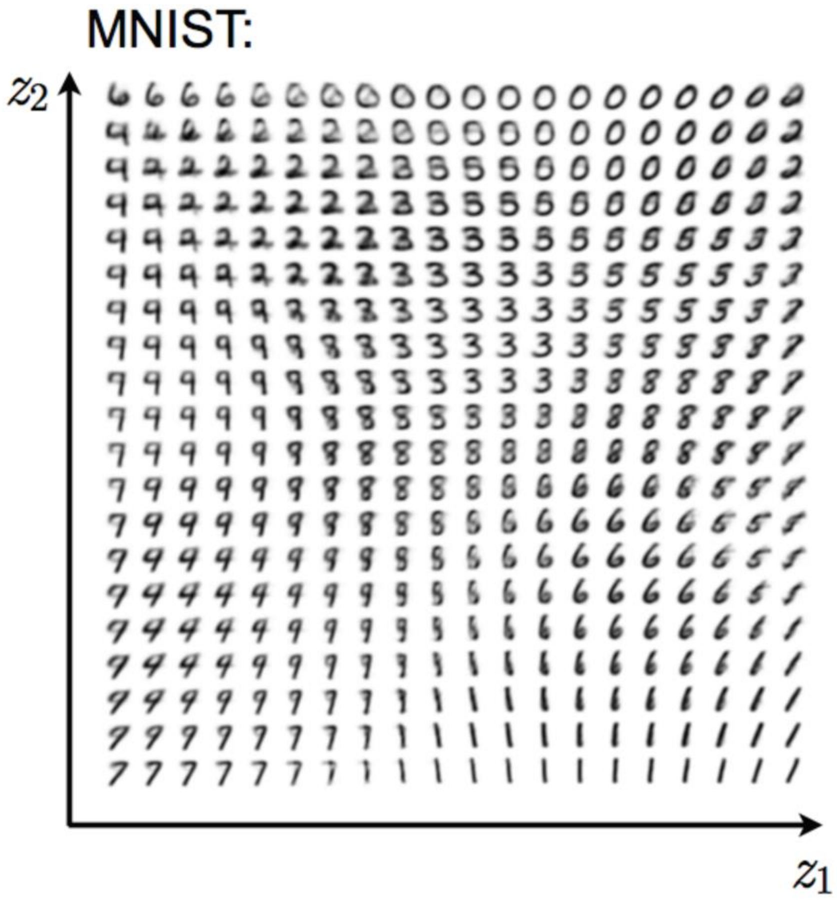
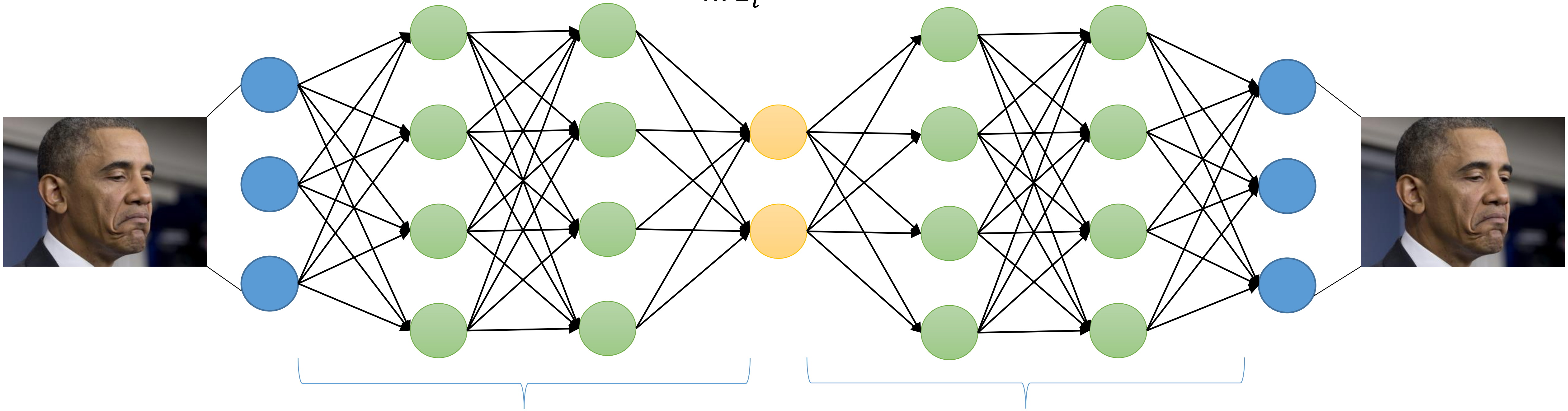


Image credit Goodfellow et al 2016

Autoencoder

Latent representation: z_i



Encoder Network: $f_\theta^{enc}(x_i)$

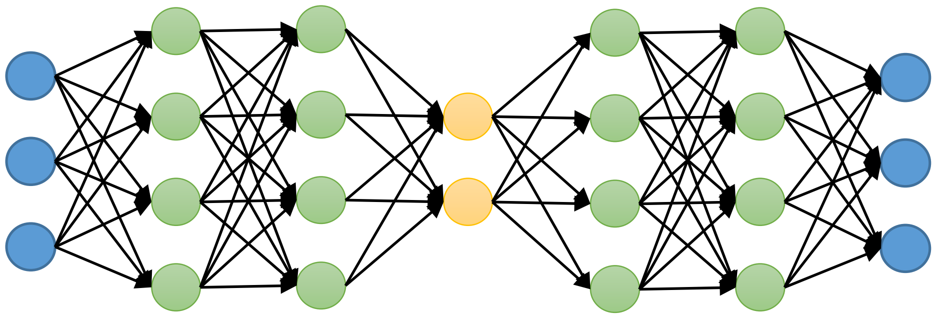
Decoder Network: $f_\theta^{dec}(z_i)$

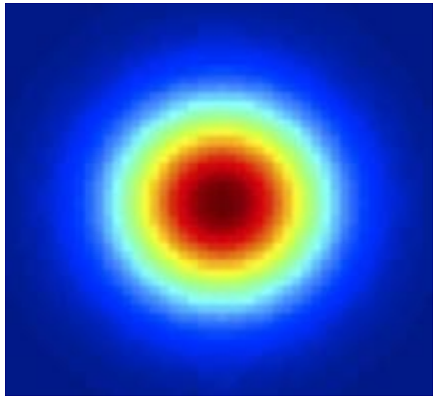
Learning is usually done by solving $\min_\theta \sum_{x_i \in D} ||f_\theta(x_i) - x_i||_2^2$

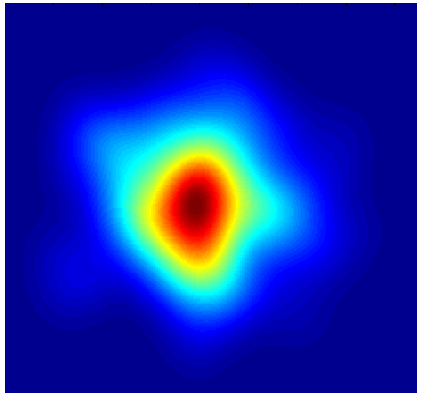
where $f_\theta(x_i) = f_\theta^{dec}(f_\theta^{enc}(x_i))$

Remarks on Autoencoder

- One of the architectures that led to the renaissances of neural networks in 2007.
- In order to avoid learning a trivial identify function, the input sample is noise corrupted. Denoising Autoencoder (Vincent et al 2010)
- The hierarchical representation learned in the encoder can be used as a feature extractor for a supervised learning task.
- However, difficult to sample from the latent space.
- Poor generalization: the decoder often just remember the input samples.







Variational Autoencoder

- Put a constraint on the latent space to make sampling easier.
- Constraint the encoder to output a conditional Gaussian distribution for an input sample x_i

$$f_\theta^{enc}(x_i) = q_\theta(z|x_i) = \mathcal{N}(z|\mu_\theta(x_i), I)$$

- The decoder reconstructs the input from a random sample from the conditional distribution $z_i \sim q_\theta(z|x_i)$

$$f_\theta^{dec}(z_i) = p_\theta(x_i|z_i \sim q_\theta(z|x_i))$$

- Train the encoder to output a zero mean Gaussian distribution and the decoder to reconstruct the input.

$$\min_\theta \sum_{x_i \in D} D_{KL}(\mathcal{N}(z|\mu_\theta(x_i), I)\ ||\mathcal{N}(z|0, I)\ ) + E_{z_i \sim q_\theta(z|x_i)}\left[\frac{1}{2}\left\|f_\theta^{dec}(z_i) - x_i\right\|_2^2\right]$$

Variational Lower Bound

$$L(\theta|D) = \sum_{x_i \in D} \log(p_\theta(x))$$
$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log(p_\theta(x))$$
$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log(\frac{p_\theta(z,x)}{p_\theta(z|x)})$$
$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log(\frac{p_\theta(z,x)}{q_\theta(z|x)} \frac{q_\theta(z|x)}{p_\theta(z|x)})$$
$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log\left(\frac{p_\theta(z,x)}{q_\theta(z|x)}\right) + \sum_z q_\theta(z|x) \log\left(q_\theta(z|x) \log\left(\frac{q_\theta(z|x)}{p_\theta(z|x)}\right)\right)$$
$$= L_V(\theta|D) + \sum_{x_i \in D} D_{KL}(q_\theta(z|x)||p_\theta(z|x))$$
$$\geq L_V(\theta|D)$$

$L_V(\theta|D)$ is the variational lower bound of the log-likelihood function $L(\theta|D)$.

Maximize the Variational Lower Bound

$$L_V(\theta|D) = \sum_{x_i \in D} \sum_z q_\theta(z|x) \log\left(\frac{p_\theta(z,x)}{q_\theta(z|x)}\right)$$

$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log\left(\frac{p_\theta(x|z)p(z)}{q_\theta(z|x)}\right)$$

$$= \sum_{x_i \in D} \sum_z q_\theta(z|x) \log\left(\frac{p(z)}{q_\theta(z|x)}\right) + \sum_z q_\theta(z|x) \log\big(p_\theta(x|z)\big)$$

$$= \sum_{x_i \in D} \underbrace{-D_{KL}(q_\theta(z|x)||p(z))}_{\text{Regularization}} + \underbrace{E_{z_i \sim q_\theta(z|x_i)}[\log\big(p_\theta(x|z_i)\big)]}_{\text{Reconstruction}}$$

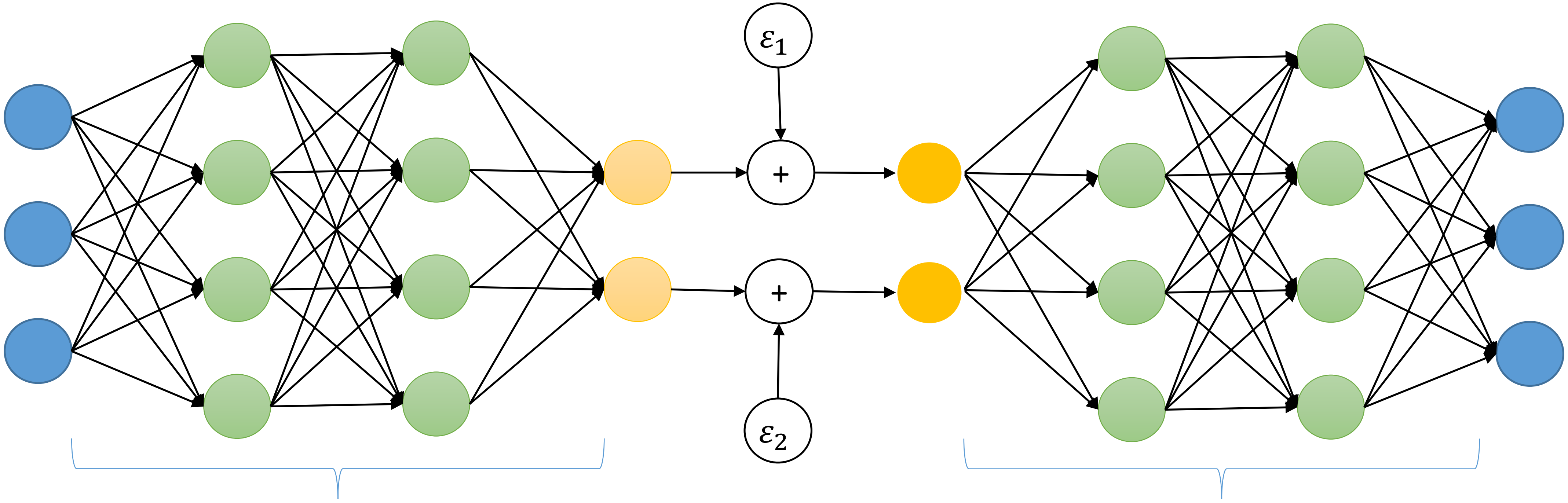
Regularization Reconstruction

$$\max_\theta L_V(\theta|D) \Leftrightarrow \min_\theta \sum_{x_i \in D} D_{KL}(\mathcal{N}(z|\mu_\theta(x_i), I)\ ||\mathcal{N}(z|0, I)\ ) + E_{z_i \sim q_\theta(z|x_i)}\left[\frac{1}{2}\left\|f_\theta^{dec}(z_i) - x_i\right\|_2^2\right]$$

Implementation of VAE

- $D_{KL}(\mathcal{N}(z|\mu_\theta(x_i), I)\ ||\mathcal{N}(z|0, I)\ ) = \frac{1}{2}\sum_d (\mu_{d,\theta}(x_i))^2$
- Sample approximation $E_{z_i \sim q_\theta(z|x_i)}\left[\log\left(p_\theta(x|z_i)\right)\right] \approx \frac{1}{L}\sum_{l=1}^{L}\frac{1}{2}\left\|f_\theta^{dec}\left(z_i^{(l)}\right) - x_i\right\|_2^2$

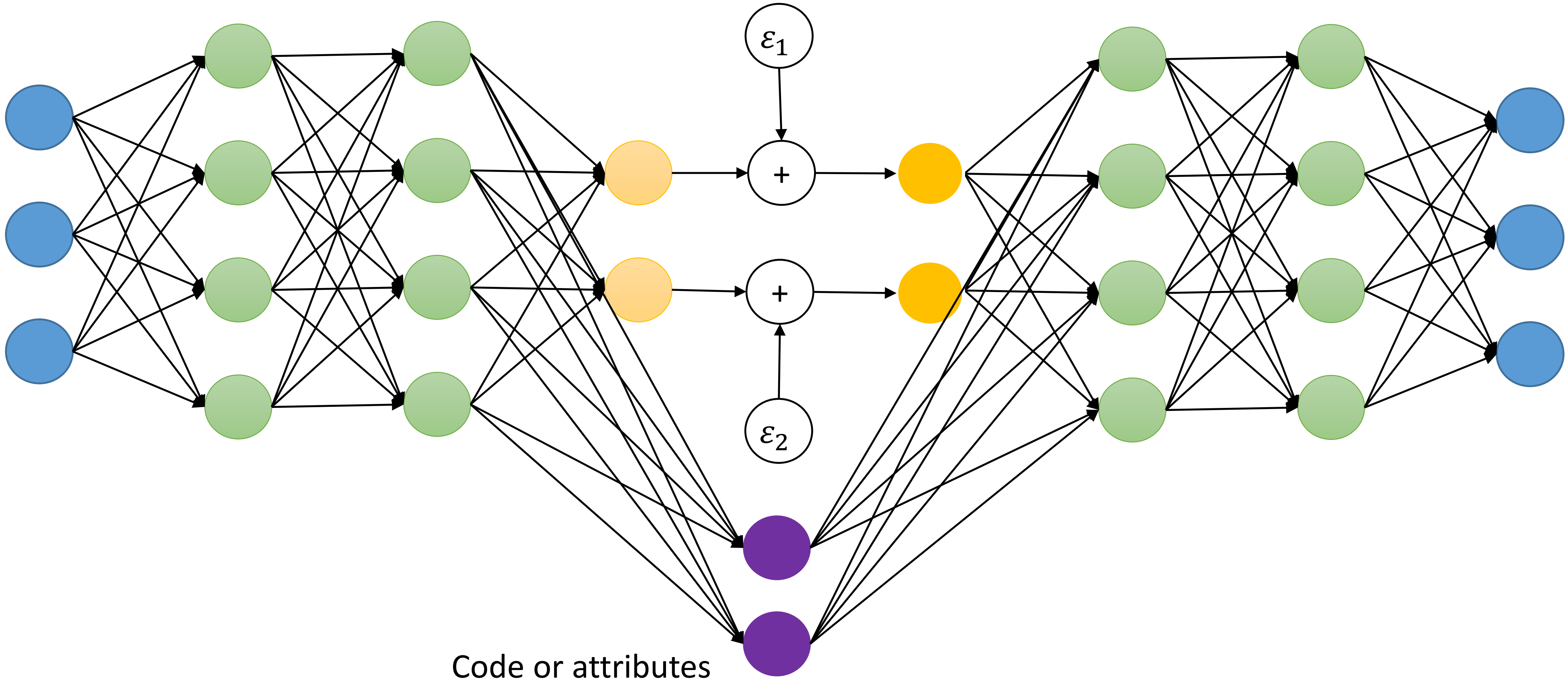
  where $z_i^{(l)} = q_\theta(z|x_i) + \varepsilon^{(l)}$ and $\varepsilon^{(l)} \sim \mathcal{N}(0, I)$



Encoder Network: $f_\theta^{enc}(x_i)$

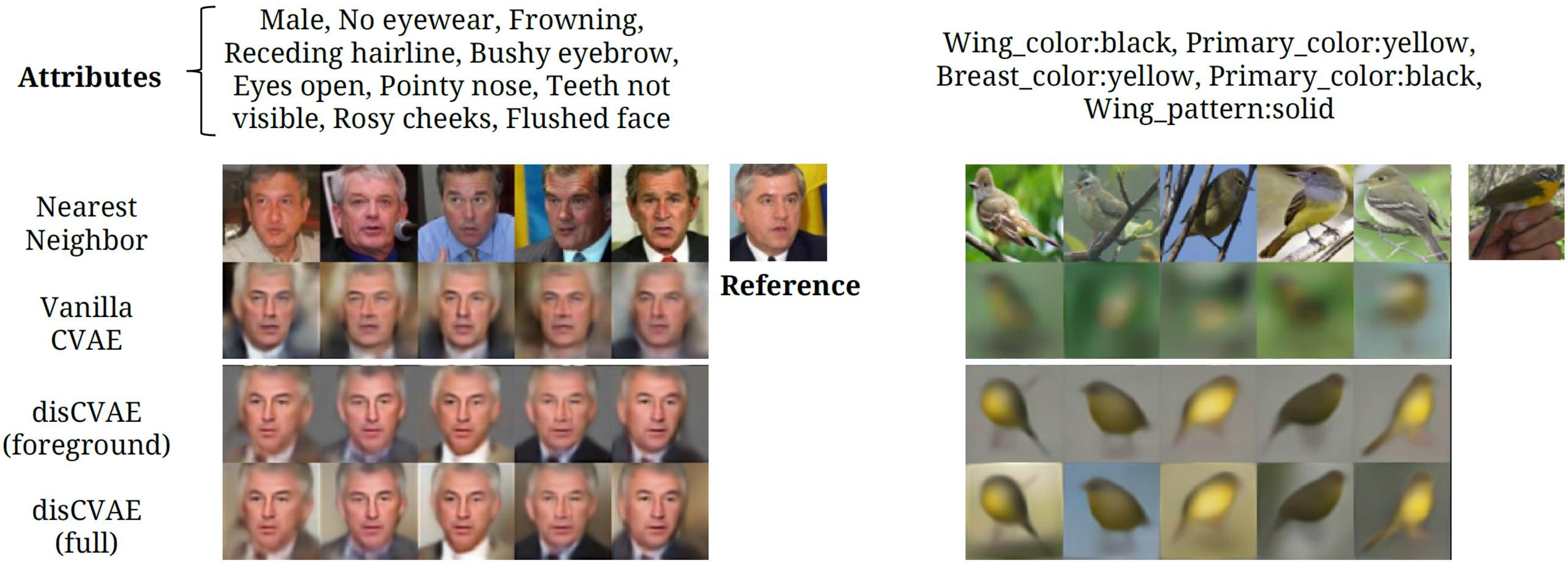
Decoder Network: $f_\theta^{dec}(z_i)$

Conditional Variational Autoencoder



Kingma et al. “Semi-supervised learning with deep generative models.” NIPS 2014

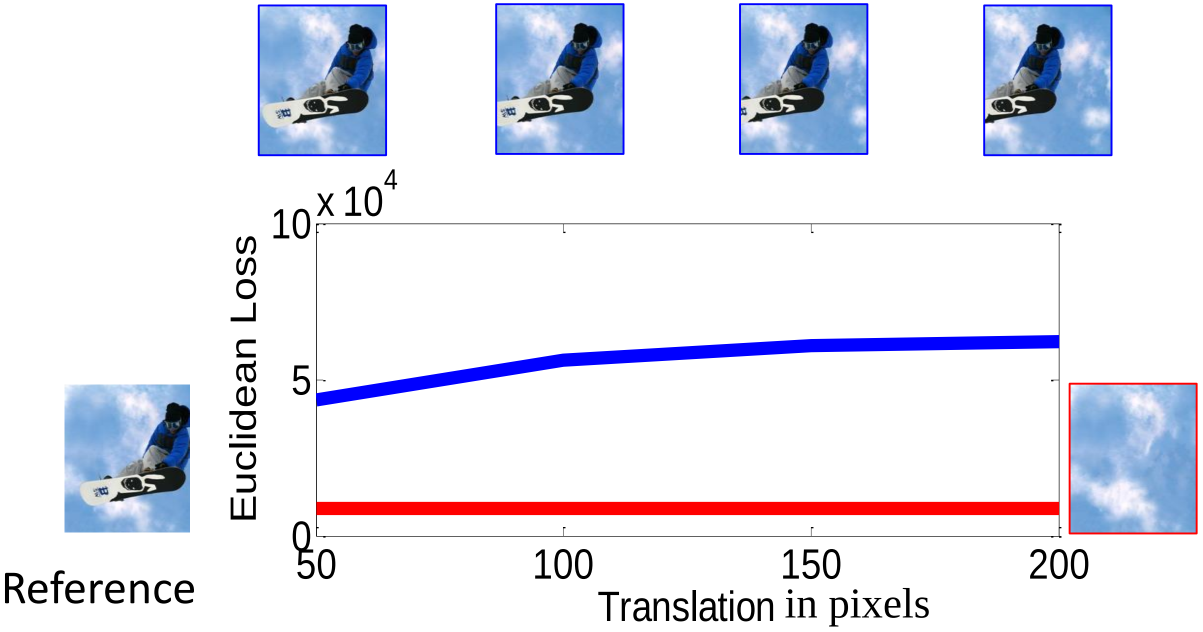
Attribute2Image



Yan et al "Attribute2Image: Conditional Image Generation from Visual Attribute" ECCV 2016

Drawback of VAE

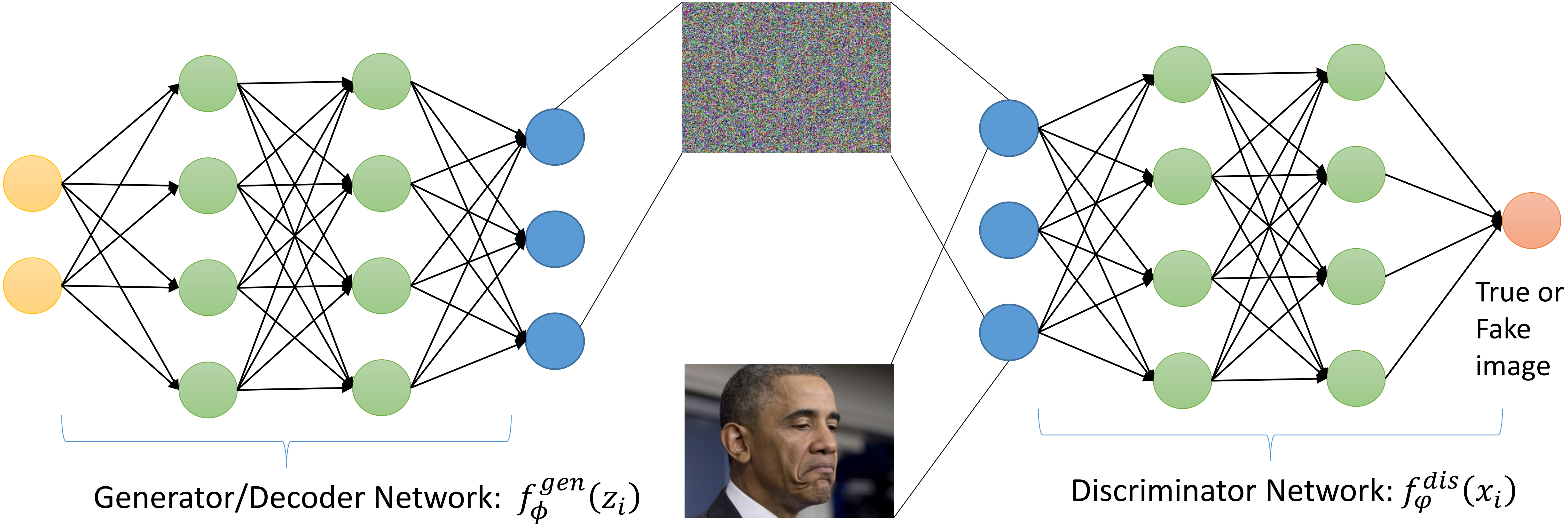
- Euclidean loss is not a good perceptual loss.
- Regress to the mean and render blurry images
- Difficult to hand-craft a good perceptual loss function.
- Why not learn one?



The blue curve plots the Euclidean loss between a reference image and its translations. The red bar is the Euclidean loss between the reference image and a background image. The Euclidean loss suggests that the background image is more similar to the reference image.

Generative Adversarial Networks

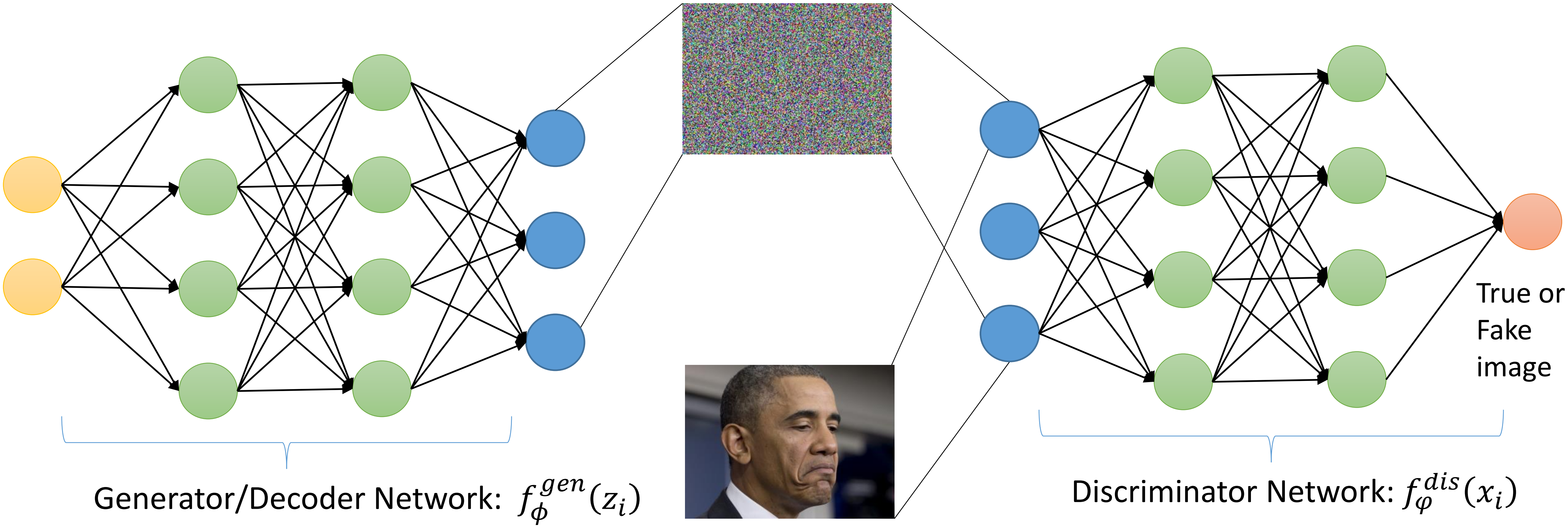
- Forget about how to design an image similarity loss. Let use a deep feedforward discriminative network to verify if a generated image is similar to a real image. (Goodfellow et al 2014)



Goodfellow et al "Generative Adversarial Networks" NIPS 2014

Generative Adversarial Networks

- Generator: map a random sample from a Gaussian distribution to an image.
- Discriminator: Differentiate a generated image from a real image.



Generative Adversarial Networks

- The generator and the discriminator is playing a zero-sum game.
- $\min_{\phi} \max_{\varphi} E_{x \sim p_{data}(x)}\left[\log f_{\varphi}^{dis}(x)\right] + E_{z \sim p_Z(z)}\left[\log(1 - f_{\varphi}^{dis}\left(f_{\phi}^{gen}(z)\right))\right]$

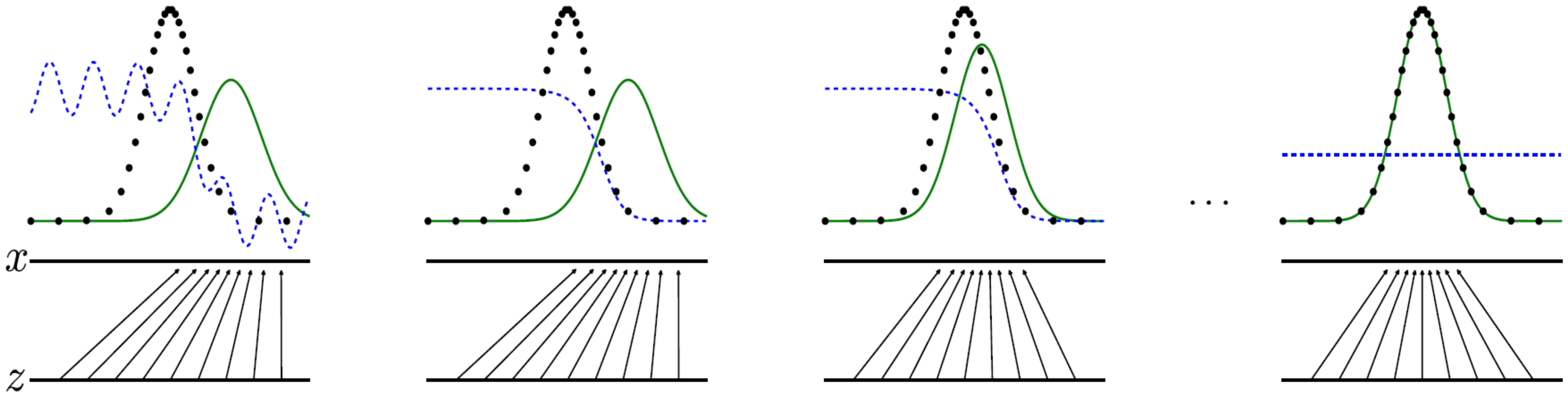


Image credit Goodfellow et al 2014

Generative Adversarial Networks

- What does this optimization do?
- For a fixed generator $f_\phi^{gen}(z)$, the optimal discriminator is $f_\varphi^{dis}(x) = \frac{p_{data}(x)}{p_{data}(x)+f_\phi^{gen}(z)}$.

$$\min_\phi \max_\varphi E_{x\sim p_{data}(x)}[\log f_\varphi^{dis}(x)] + E_{z\sim p_Z(z)}\left[\log(1 - f_\varphi^{dis}\left(f_\phi^{gen}(z)\right))\right]$$

$$= \min_\phi E_{x\sim p_{data}(x)}\left[\log \frac{p_{data}(x)}{p_{data}(x)+f_\phi^{gen}(z)}\right] + E_{z\sim p_Z(z)}\left[\log \frac{f_\phi^{gen}(z)}{p_{data}(x)+f_\phi^{gen}(z)}\right]$$

$$= \min_\phi D_{KL}(p_{data}(x)|| \frac{p_{data}(x)+f_\phi^{gen}(z)}{2}) + D_{KL}(f_\phi^{gen}(z)|| \frac{p_{data}(x)+f_\phi^{gen}(z)}{2}) - \log(4)$$

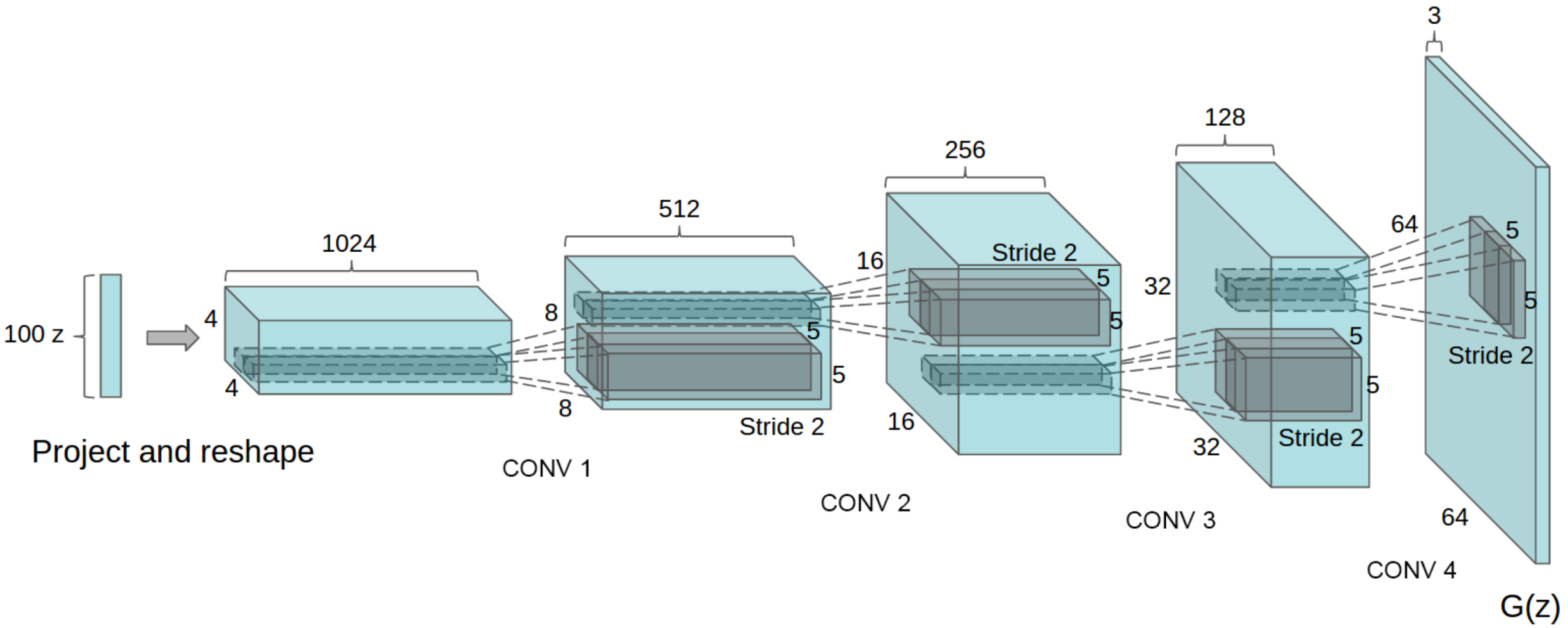
$$= \min_\phi D_{JS}(p_{data}(x)||f_\phi^{gen}(z)) - \log(4)$$

Jensen-Shannon Divergence

Generative Adversarial Network Training

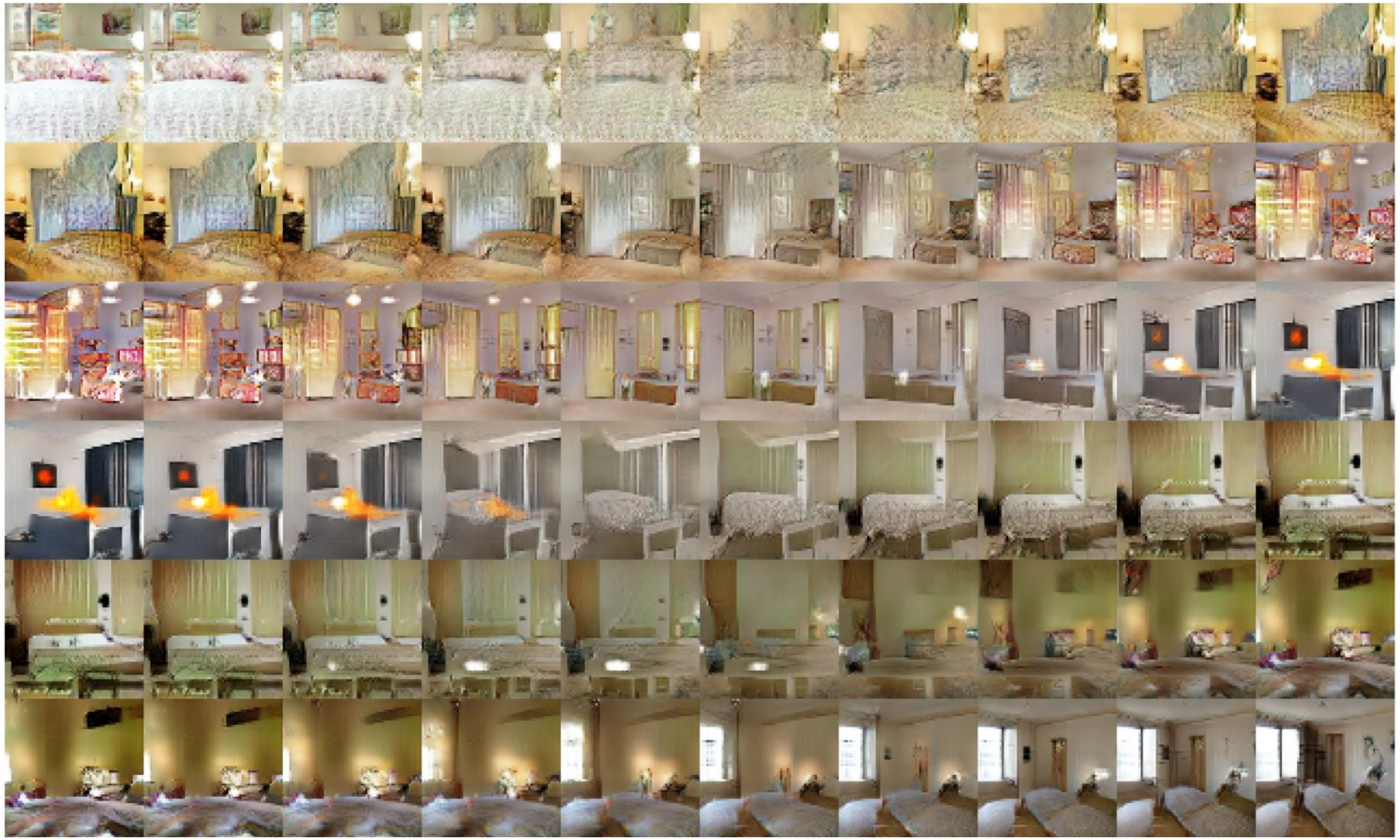
- $V(\varphi, \phi) = E_{x \sim p_{data}(x)}[\log f_{\varphi}^{dis}(x)] + E_{z \sim p_Z(z)}\left[\log(1 - f_{\varphi}^{dis}\left(f_{\phi}^{gen}(z)\right))\right]$
- $\min_{\phi} \max_{\varphi} V(\varphi, \phi)$
- Alternating gradient descent
- Fix ϕ (generator), apply a stochastic gradient ascent step on $V(\varphi, \phi)$.
- Fix φ (discriminator), apply a stochastic gradient descent step on $V(\varphi, \phi)$.

Deep Convolutional Generative Adversarial Networks



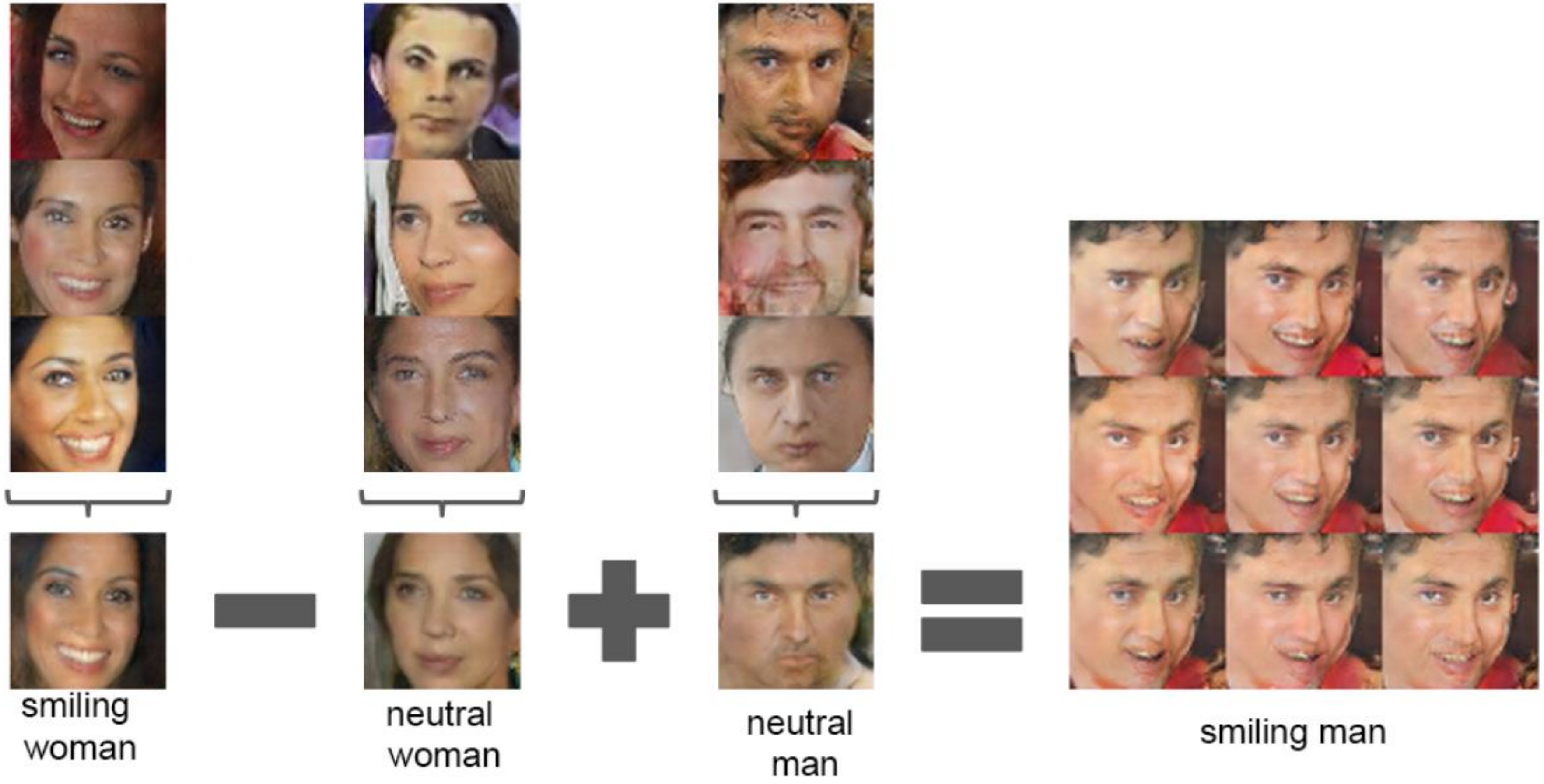
Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Deep Convolutional Generative Adversarial Networks



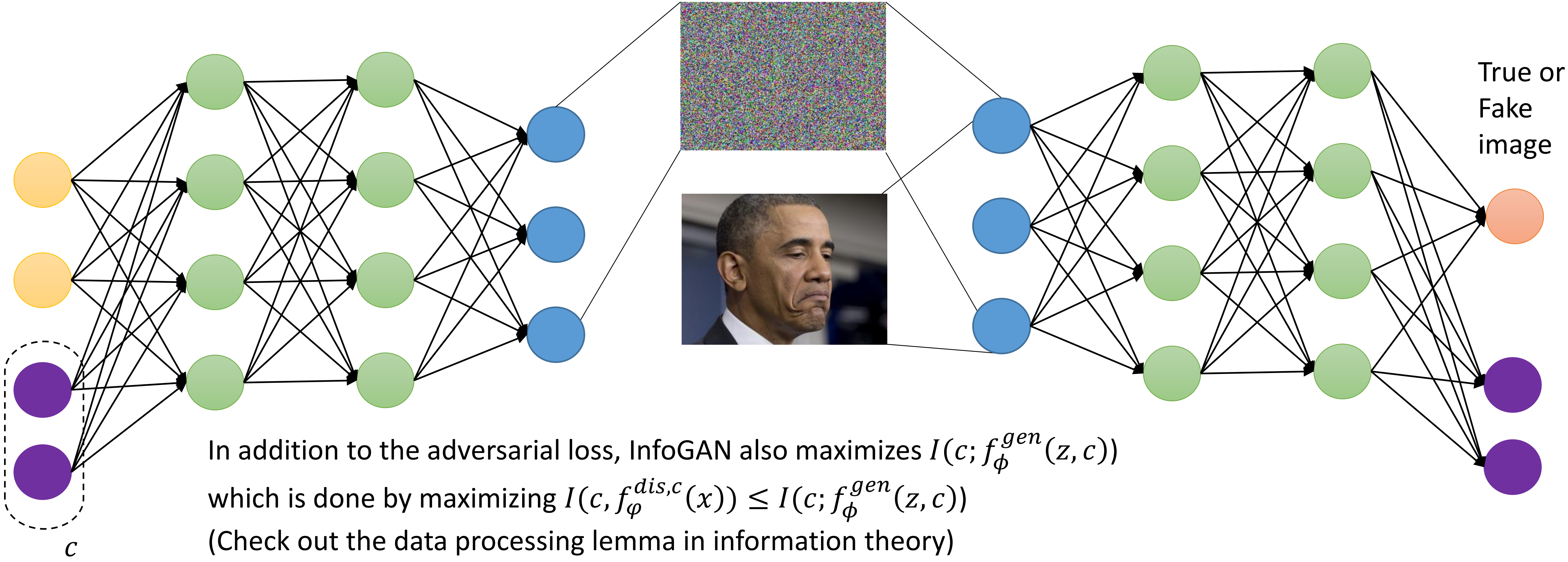
Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Deep Convolutional Generative Adversarial Networks



Radford et al. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR 2016

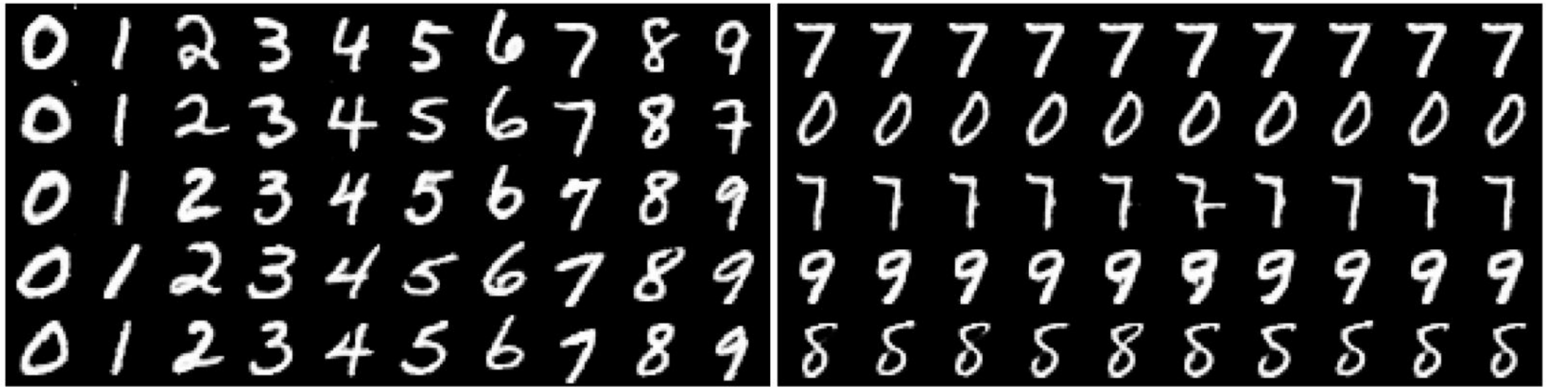
InfoGAN: Interpretable Representation Learning by Information Maximizing GAN



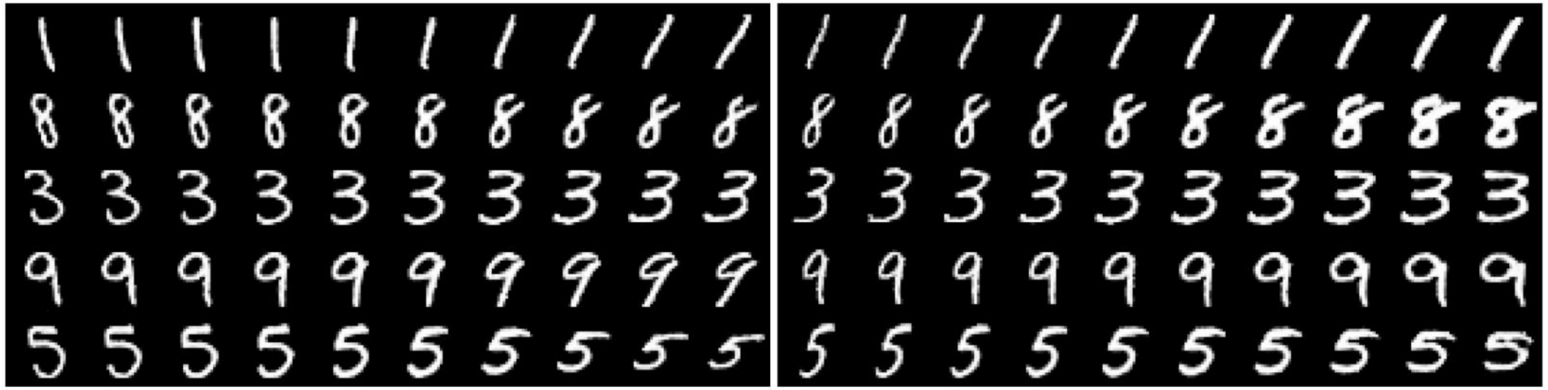
In addition to the adversarial loss, InfoGAN also maximizes $I(c; f_{\phi}^{gen}(z,c))$
which is done by maximizing $I(c, f_{\varphi}^{dis,c}(x)) \leq I(c; f_{\phi}^{gen}(z,c))$
(Check out the data processing lemma in information theory)

Chen et al. "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets", NIPS 2016

InfoGAN: Interpretable Representation Learning by Information Maximizing GAN



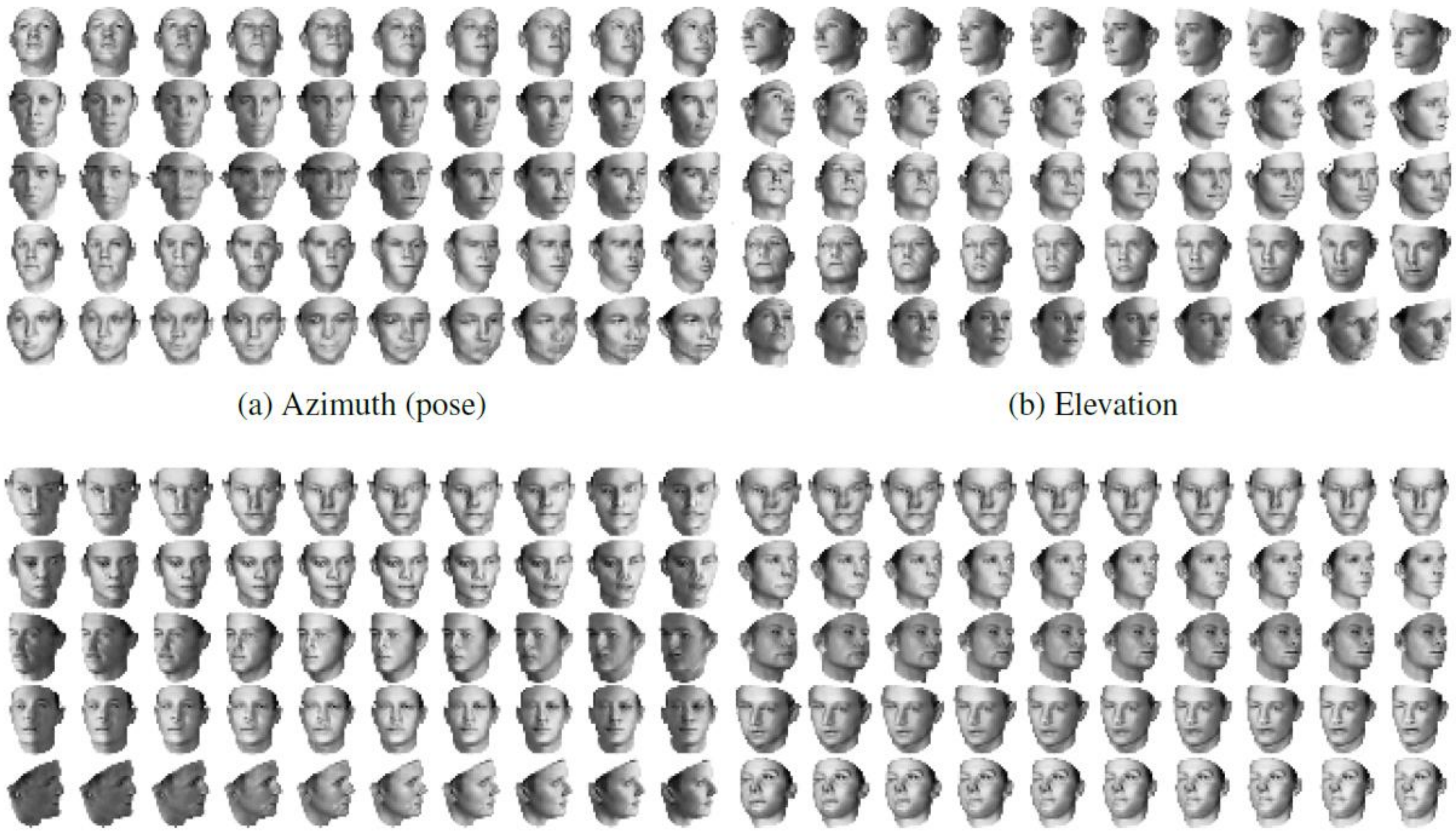
(a) Varying c_1 on InfoGAN (Digit type) (b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation) (d) Varying c_3 from -2 to 2 on InfoGAN (Width)

Chen et al. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”, NIPS 2016

InfoGAN: Interpretable Representation Learning by Information Maximizing GAN



(a) Azimuth (pose) (b) Elevation

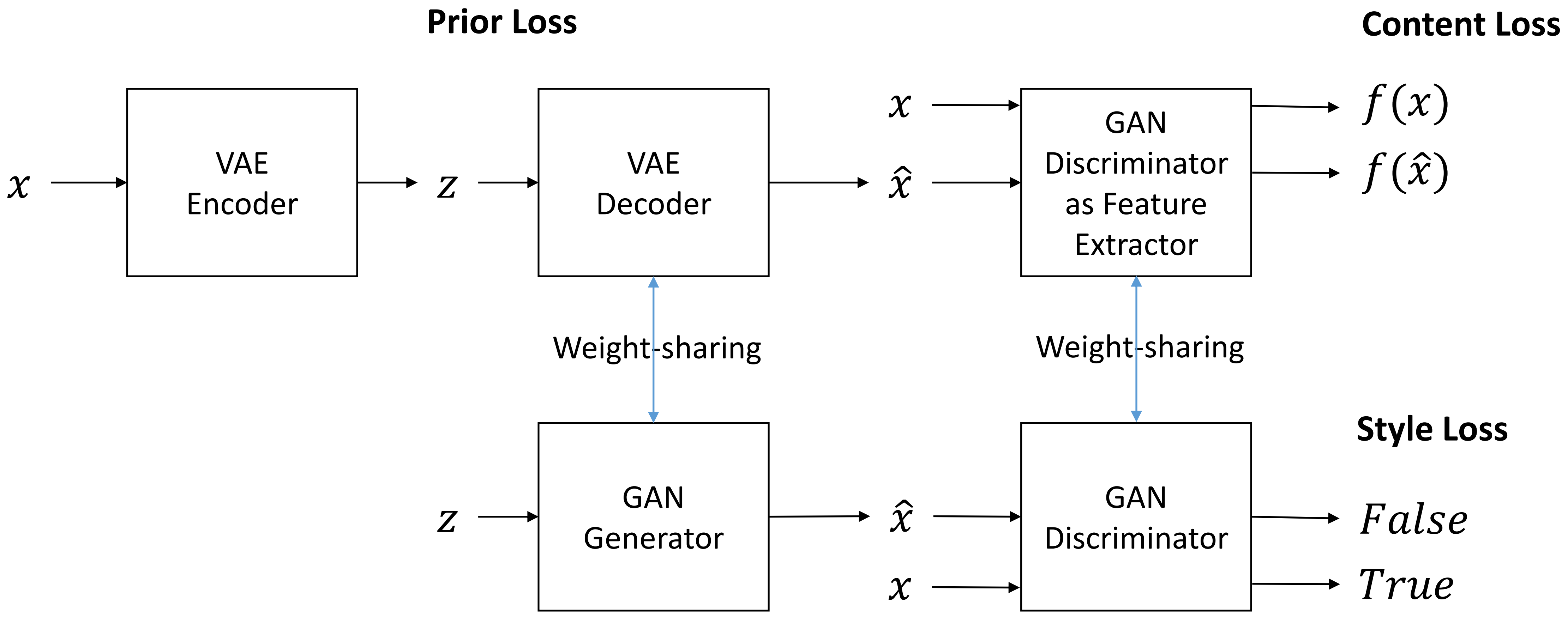
(c) Lighting (d) Wide or Narrow

Chen et al. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”, NIPS 2016

VAE and GAN Comparison

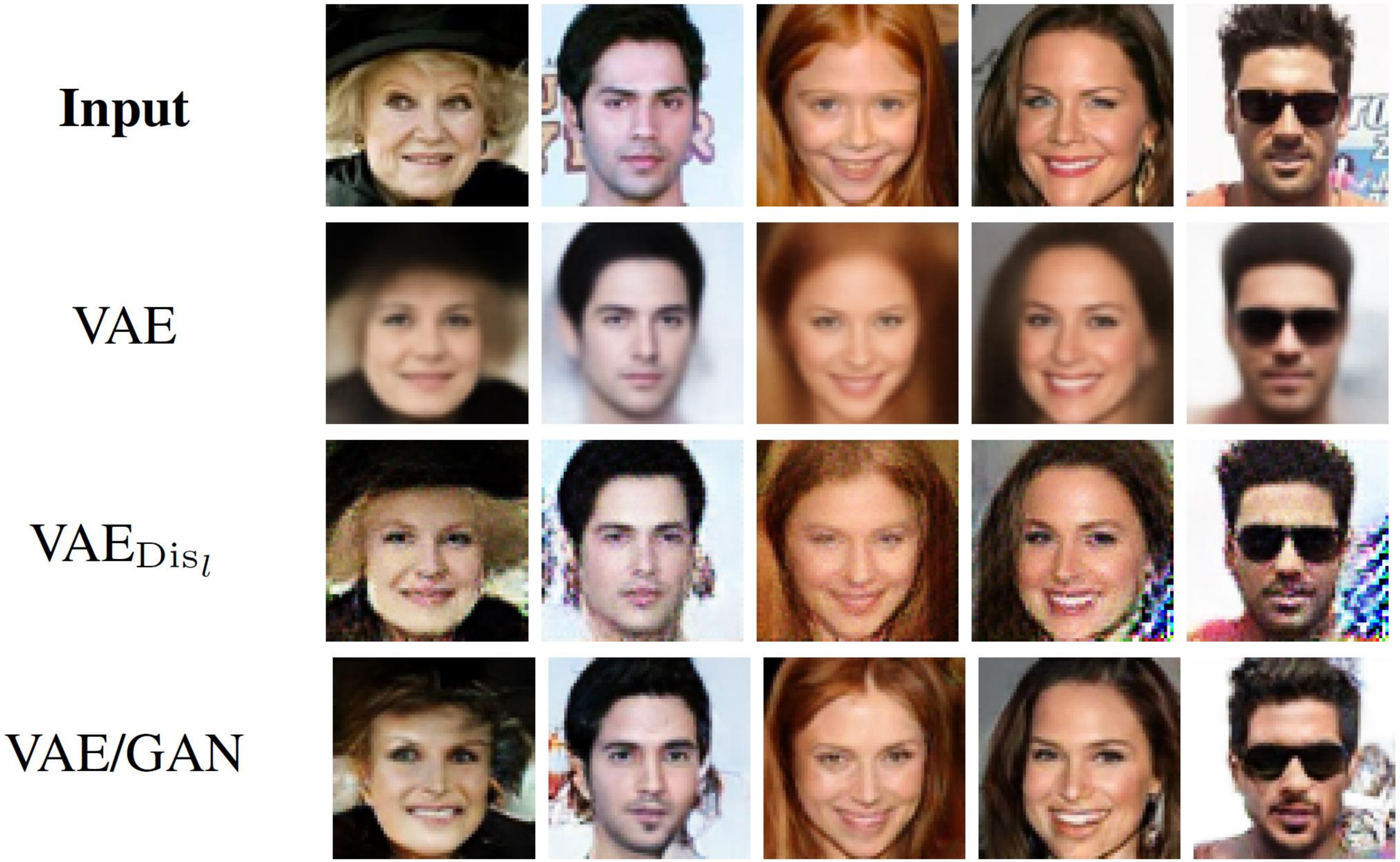
Model	Optimization	Image Quality	Generalization
Variational Autoencoders (VAE)	• Stochastic gradient descent • Converge to local minimum • Easier	• Smooth • Blurry	• Tend to remember input images
Generative Adversarial Networks (GAN)	• Alternating stochastic gradient descent • Converge to saddle points • Harder	• Sharp • Artifact	• Generate new unseen images

VAE/GAN Model



Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”, ICML 2016

VAE/GAN Model



Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”, ICML 2016

VAE/GAN Model

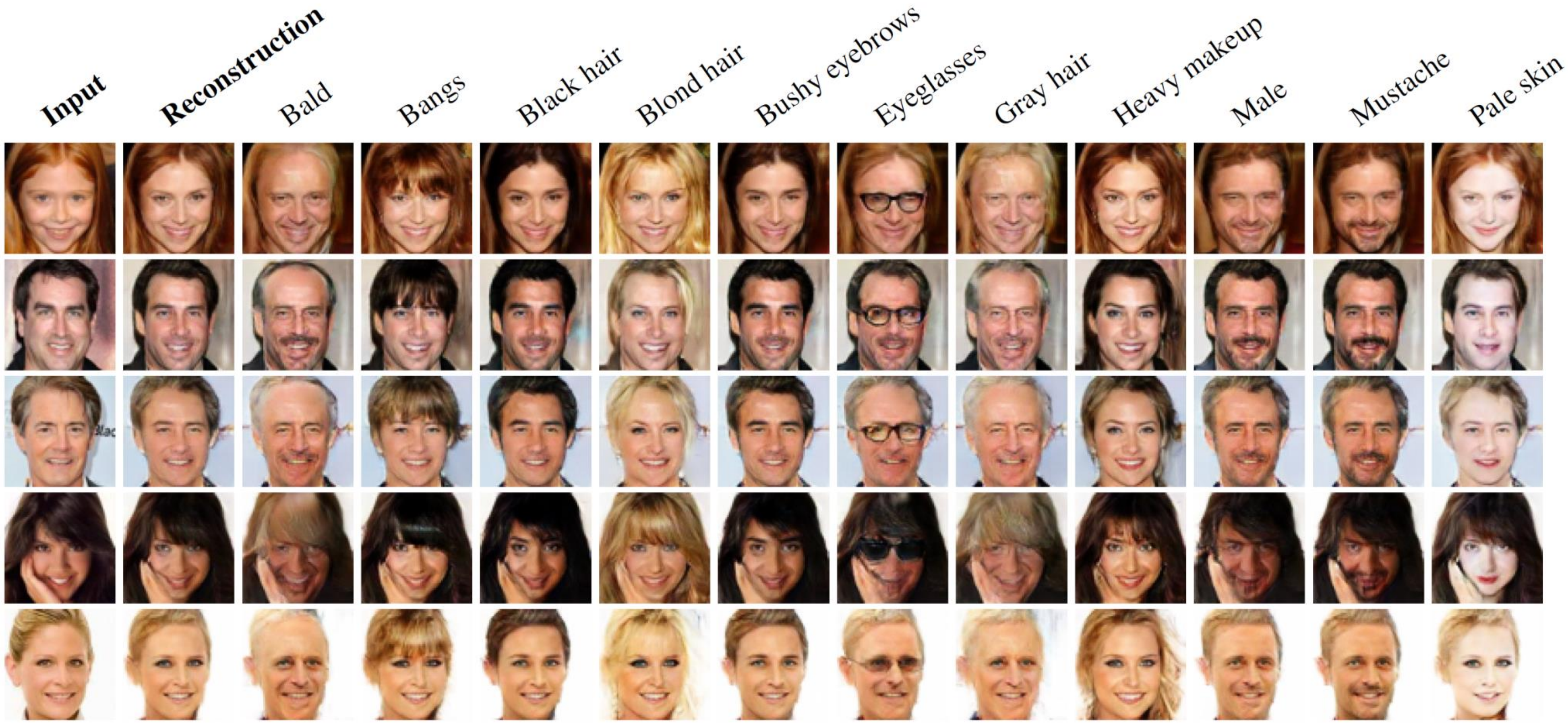


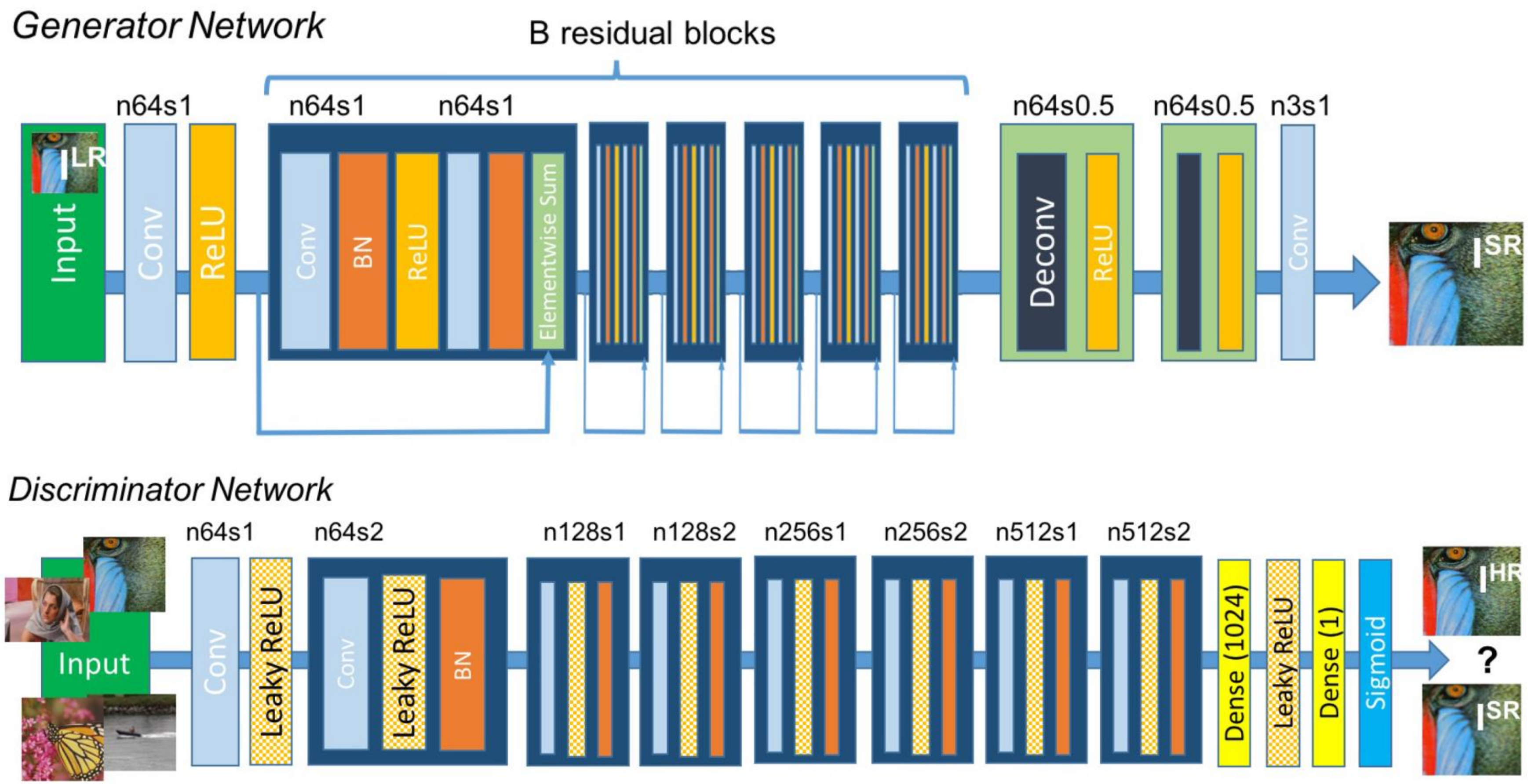
Figure 5. Using the VAE/GAN model to reconstruct dataset samples with visual attribute vectors added to their latent representations.

Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”, ICML 2016

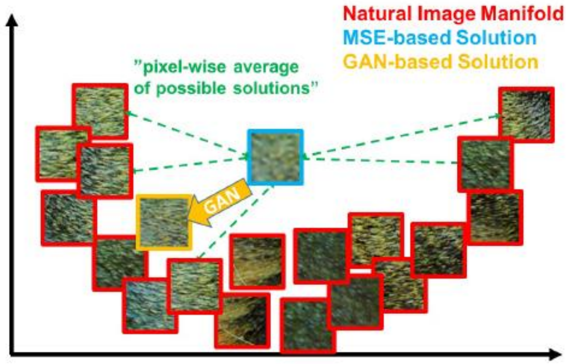
Applications

- Image Superresolution
- Inpainting
- Image Editing
- Domain Adaptation

Application: Image Super-resolution



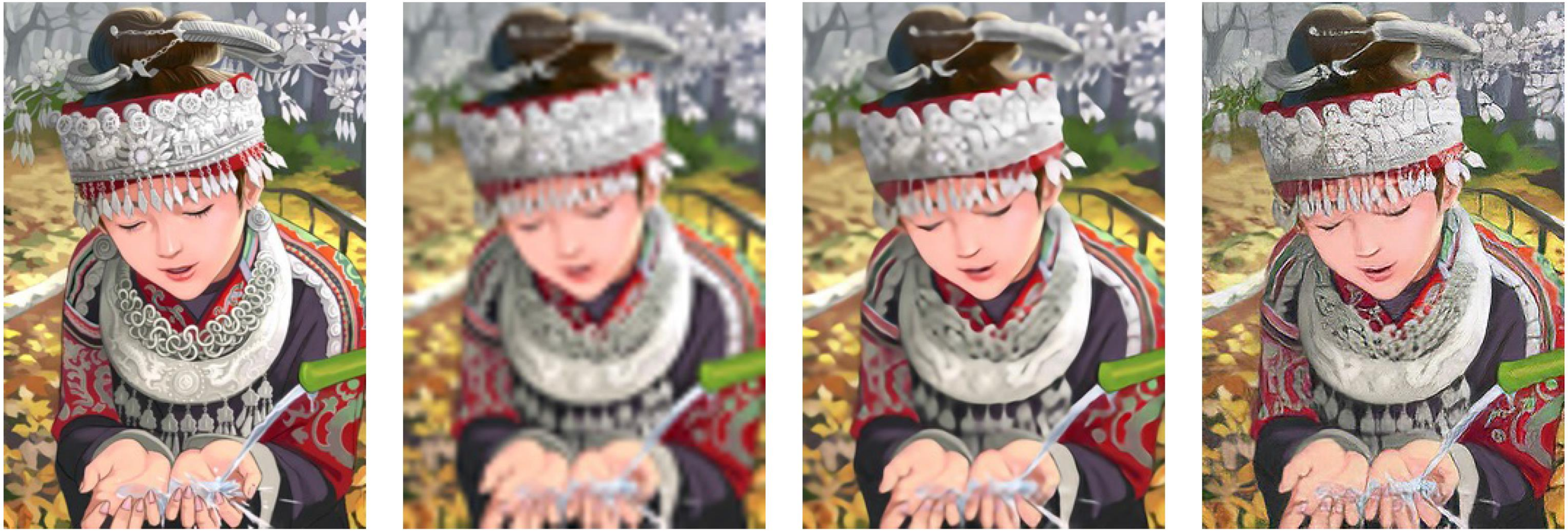
Minimize
- Adversarial Loss
- Content Loss
- TV-norm



Ledig et al, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network" arXiv 1609.0480

Application: Image Super-resolution

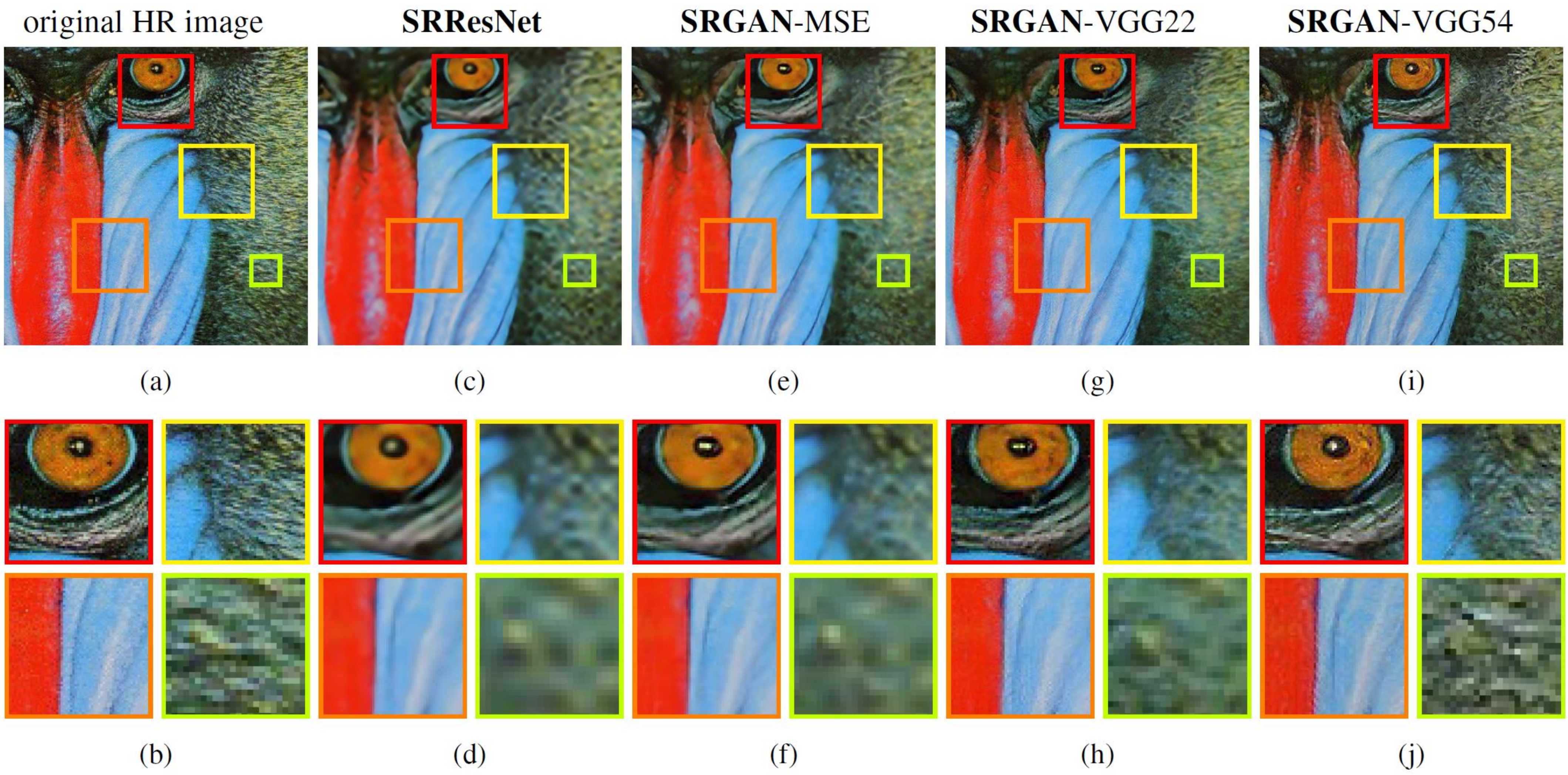
original | bicubic (21.59dB/0.6423) | SRResNet (23.44dB/0.7777) | SRGAN (20.34dB/0.6562)



PSNR/SSIM

Ledig et al, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network” arXiv 1609.0480

Application: Image Super-resolution



Ledig et al, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network” arXiv 1609.0480

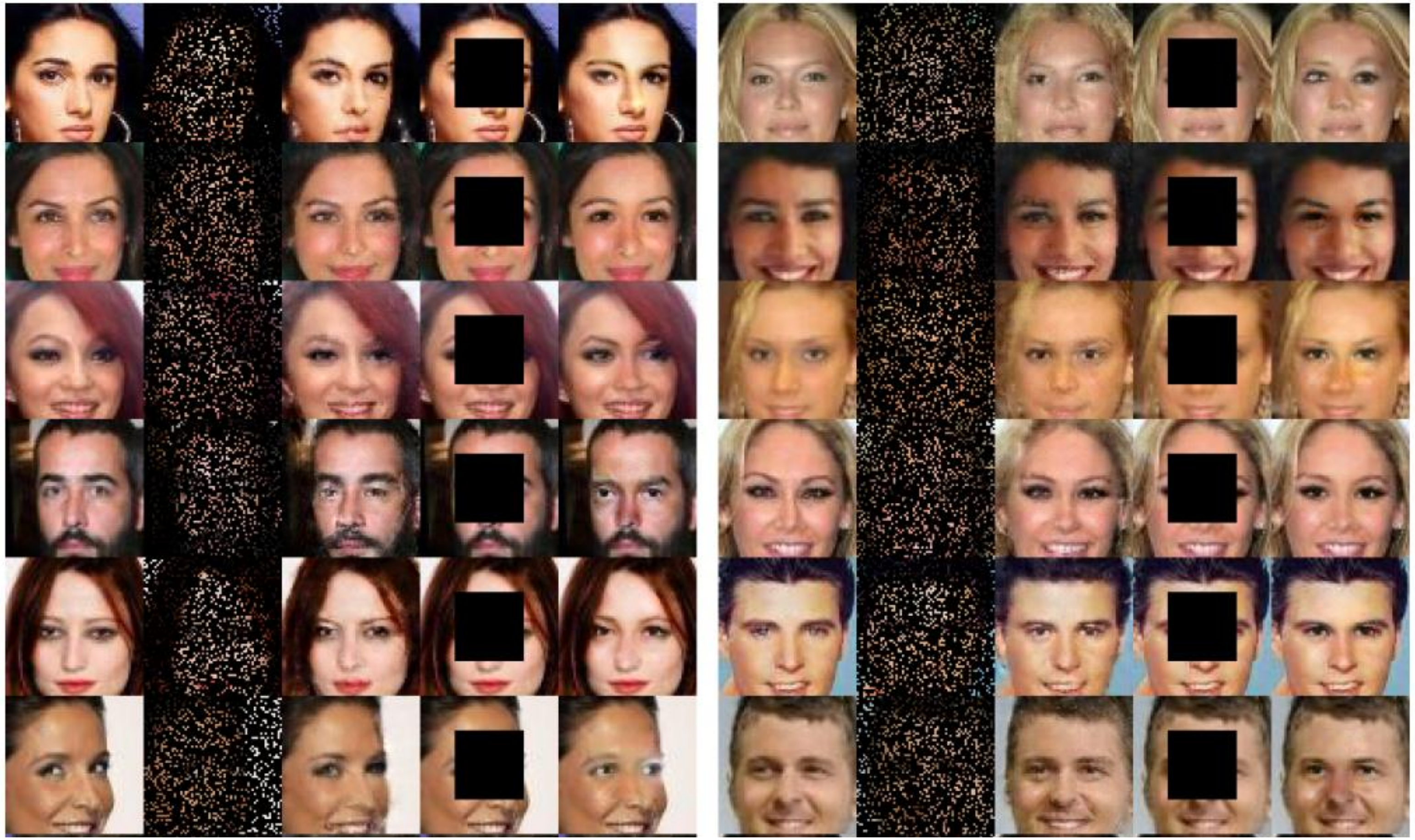
Image Inpaiting

Let $\bar{x}$ be a corrupted images. By solving

$$z^* = \underset{z}{\operatorname{argmin}} \log(1 - f_\varphi^{dis}(f_\phi^{gen}(z)) + \left\| M \odot f_\phi^{gen}(z) - M \odot \bar{x} \right\|_2^2$$

We can get the inpainted image by

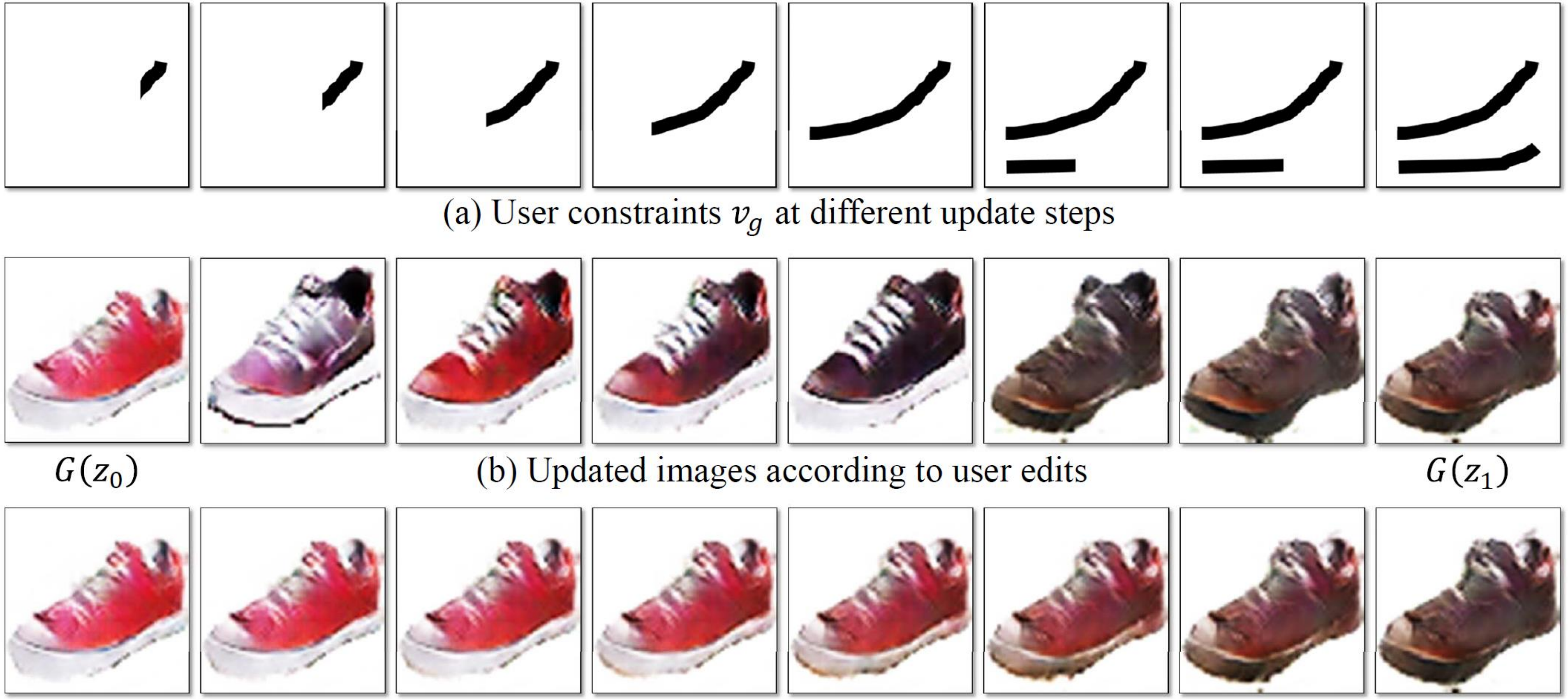
$$x = f_\phi^{gen}(z^*)$$



Inpainted images w/wo perceptual loss

Yeh et al, "Semantic Image Inpainting with Perceptual and Contextual Losses" arXiv 1607.07539

Generative Visual Manipulation on the Natural Image Manifold



(a) User constraints v_g at different update steps

(b) Updated images according to user edits

(c) Linear interpolation between $G(z_0)$ and $G(z_1)$

Zhu et al, "Generative Visual Manipulation on the Natural Image Manifold" ECCV 2016

Generative Visual Manipulation on the Natural Image Manifold

Let x_0 be an input image. Find the hidden code that the generator would use

$$z_0 = \underset{z}{\operatorname{argmin}}\, L(x_0, G(z))$$

The user then made some edits. The edits are given as constraints. We then solve the optimization problem for find a new hidden code that resembles the original image while satisfying the constraints by solving

$$z^* = \underset{z \in \mathbb{Z}}{\arg\min} \underbrace{\sum_g \|f_g(G(z)) - v_g\|^2}_{\text{data term}} + \underbrace{\lambda_s \cdot \|z - z_0\|^2}_{\substack{\text{manifold} \\ \text{smoothness}}} + \lambda_D \cdot \log(1 - D(G(z)))$$

Perceptual loss

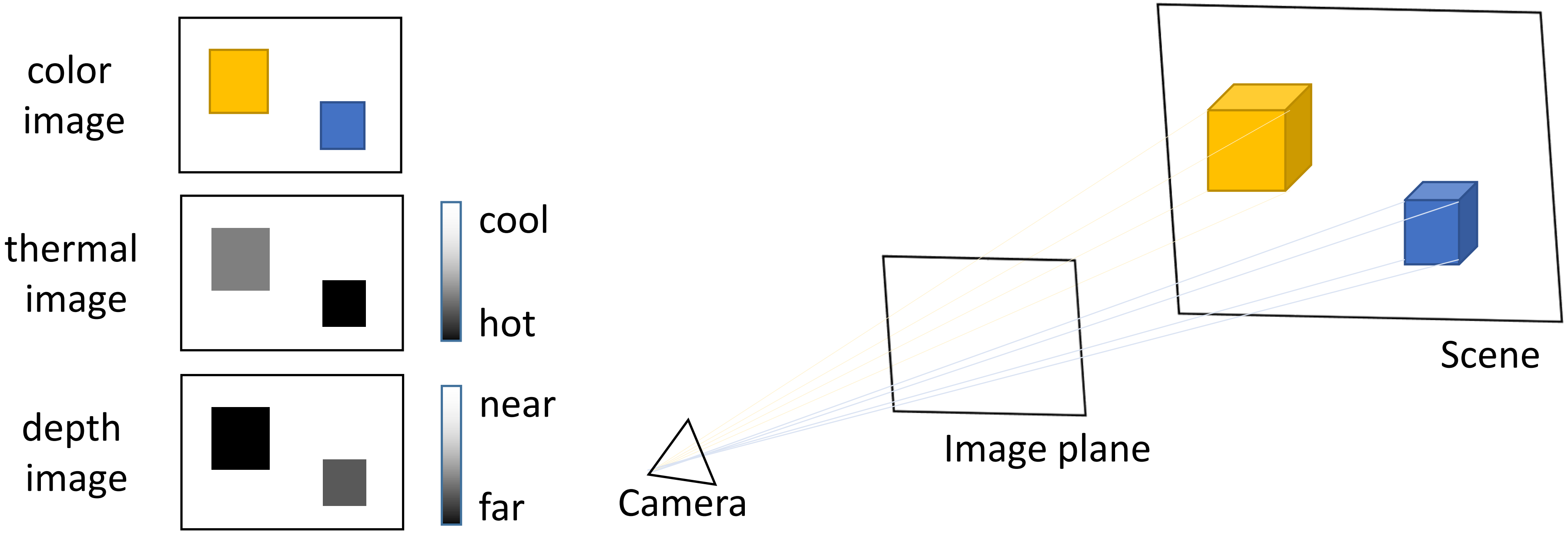
Zhu et al, “Generative Visual Manipulation on the Natural Image Manifold” ECCV 2016

Generative Visual Manipulation on the Natural Image Manifold

Zhu et al, “Generative Visual Manipulation on the Natural Image Manifold” ECCV 2016

Coupled Generative Adversarial Networks

Learn joint distribution of multi-domain images without any corresponding images in the different domains.



- $p(X_1, X_2, \dots, X_N)$: where X_i are images of the scene in different modalities.
- Ex. $p(X_{color}, X_{thermal}, X_{depth})$:

Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

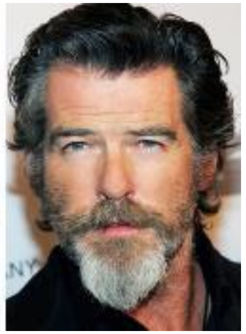
Coupled Generative Adversarial Networks

- Define domain by attribute.
- Multi-domain images are views of an object with different attributes.

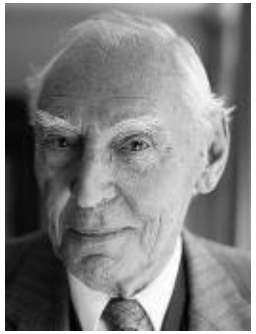

Non-smiling


Smiling


Non-beard


Beard


Young


Senior

Font#1 Font#2


images

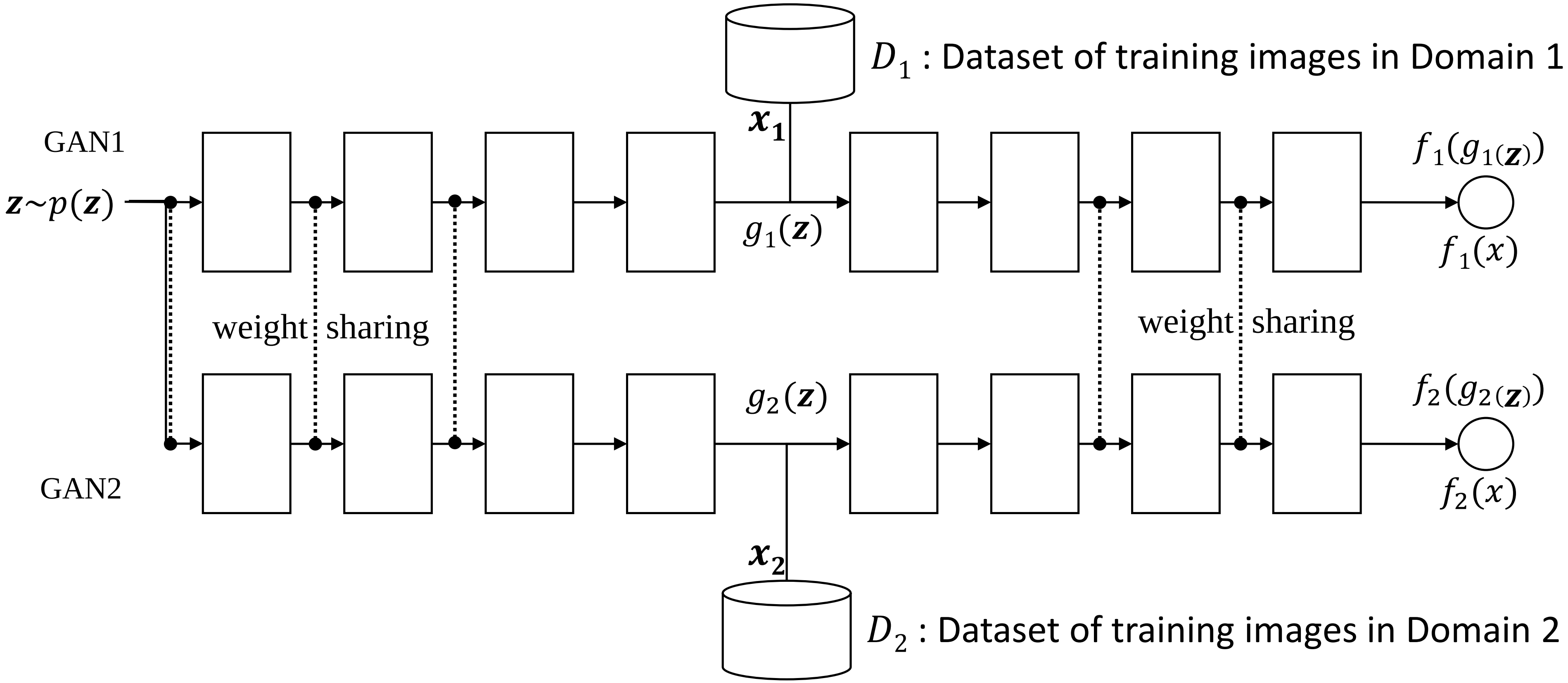

Hand-drawings


summer


winter

Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

Coupled Generative Adversarial Networks

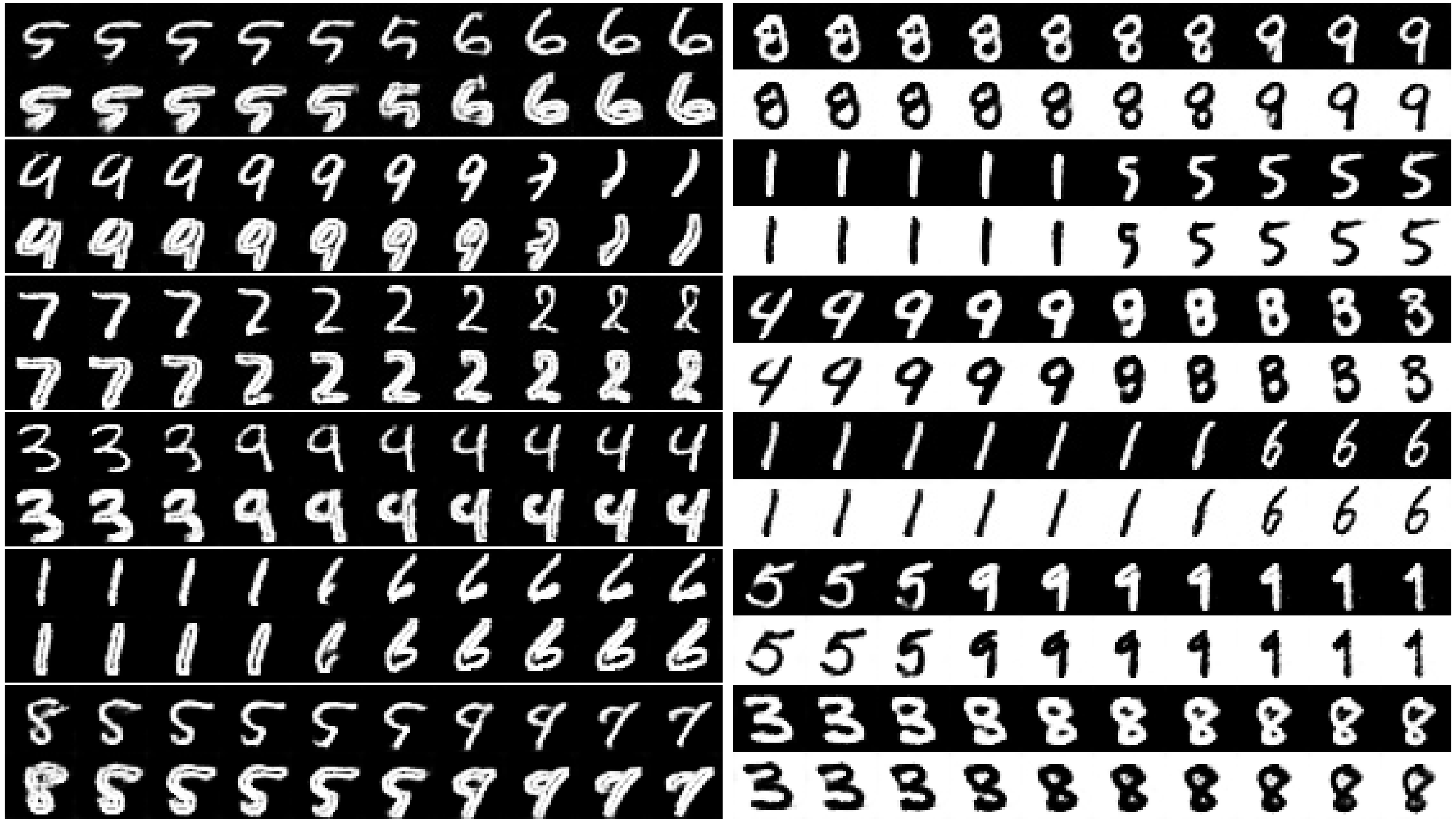


Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

NO CORRESPONDING IMAGES

Table 1: Numbers of training images in Domain 1 and Domain 2 in the MNIST experiments.

	Task $\mathbb{A}$ Pair generation of digits and corresponding edge images	Task $\mathbb{B}$ Pair generation of digits and corresponding negative images
# of images in Domain 1	30,000	30,000
# of images in Domain 2	30,000	30,000



Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

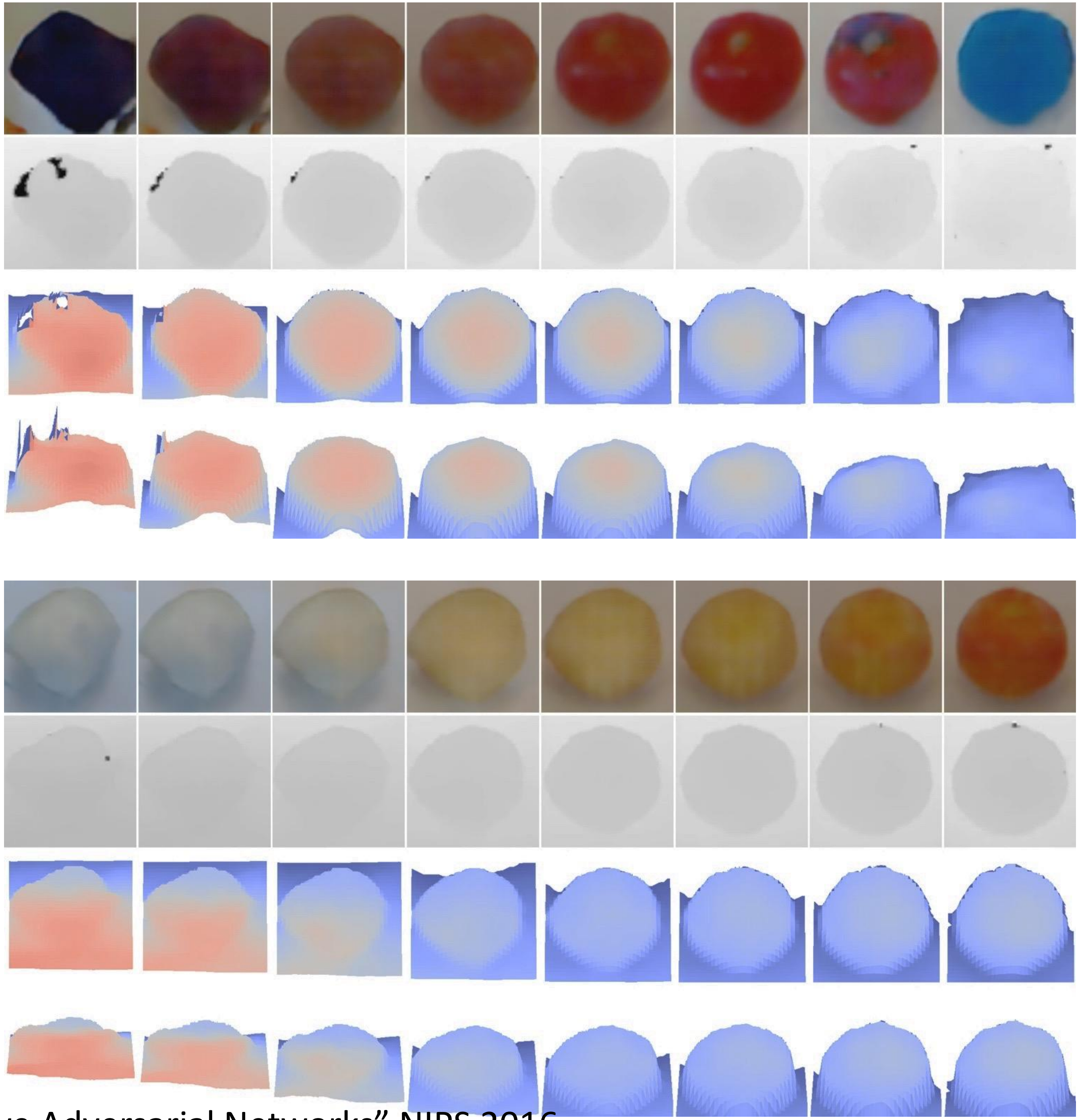
NO CORRESPONDING IMAGES

Figure 3: Training images from the RGBD dataset [3].

Table 3: Numbers of RGB and depth training images in the RGBD experiments.

# of RGB images	125,000
# of depth images	125,000

Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016



Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

NO CORRESPONDING IMAGES

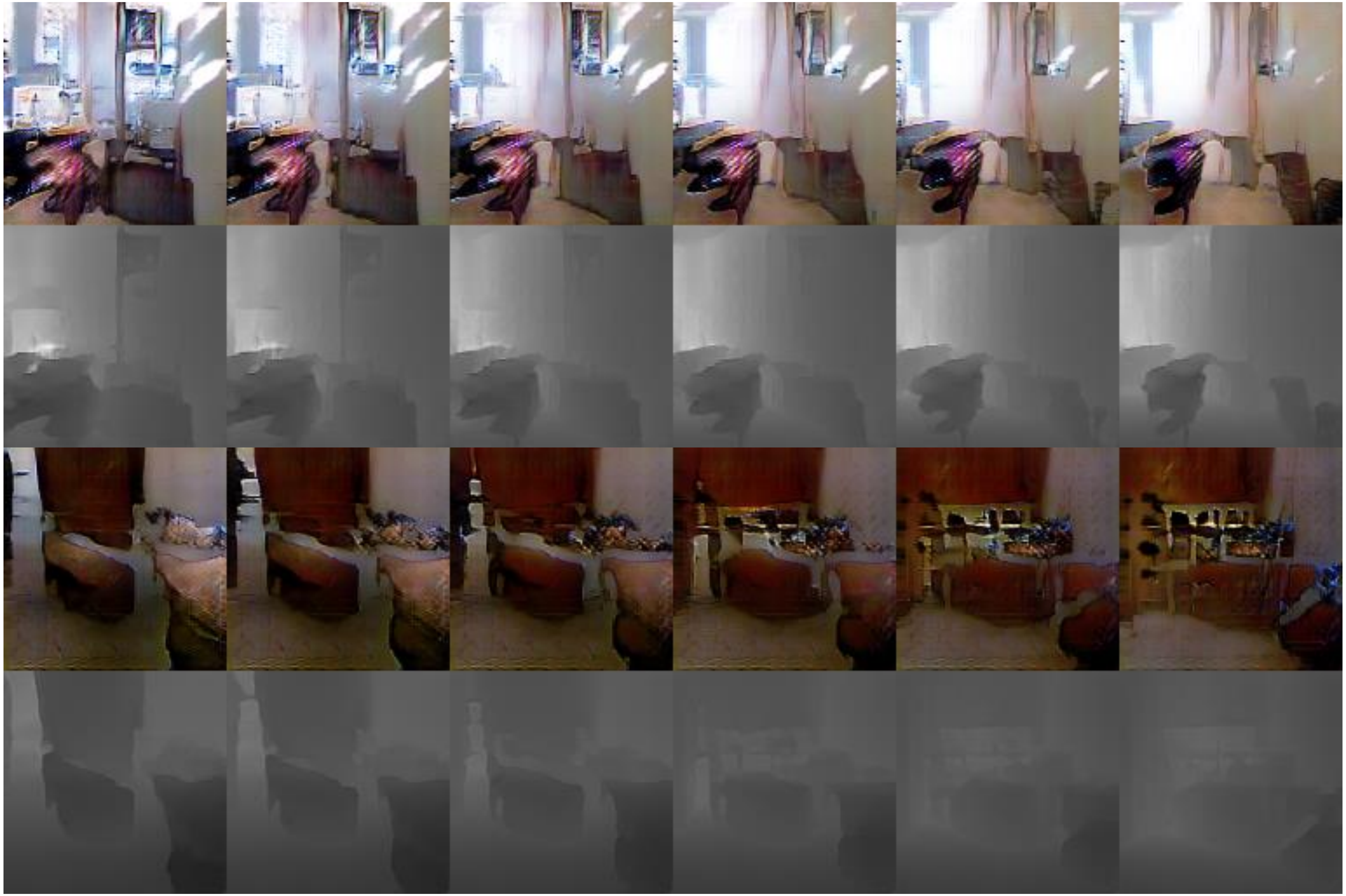
Figure 4: Training images from the NYU dataset [4].

Table 4: Numbers of RGB and depth training images in the NYU experiments.

# of RGB images	514,192
# of depth images	1,449

Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016



Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

NO CORRESPONDING IMAGES

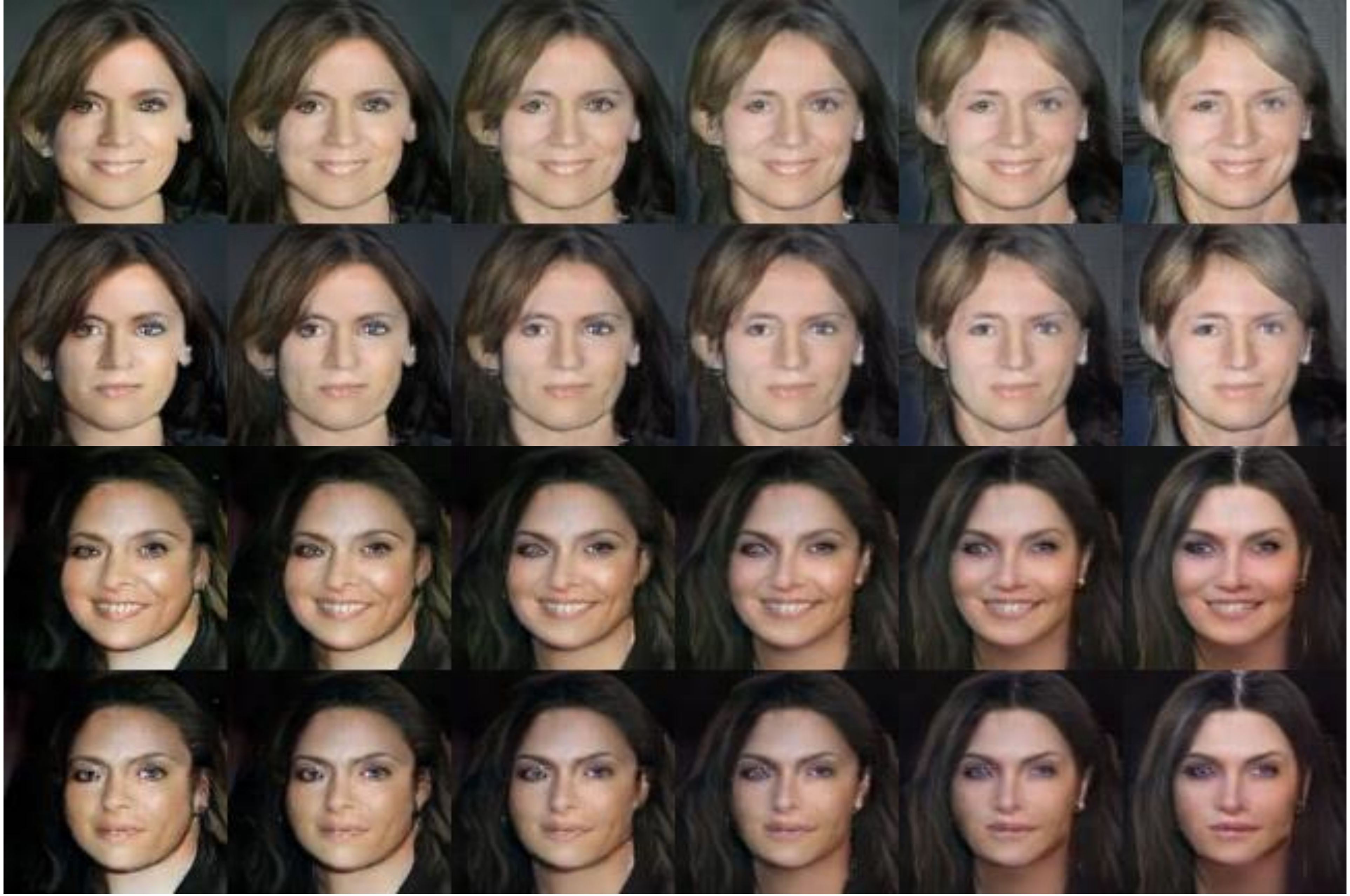
Figure 2: Training images from the Celeba dataset [2].

Table 2: Numbers of training images of different attributes in the pair face generation experiments.

Attribute	Smiling	Blond hair	Glasses
# of images with the attribute	97,669	29,983	13,193
# of images without the attribute	104,930	172,616	189,406

Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

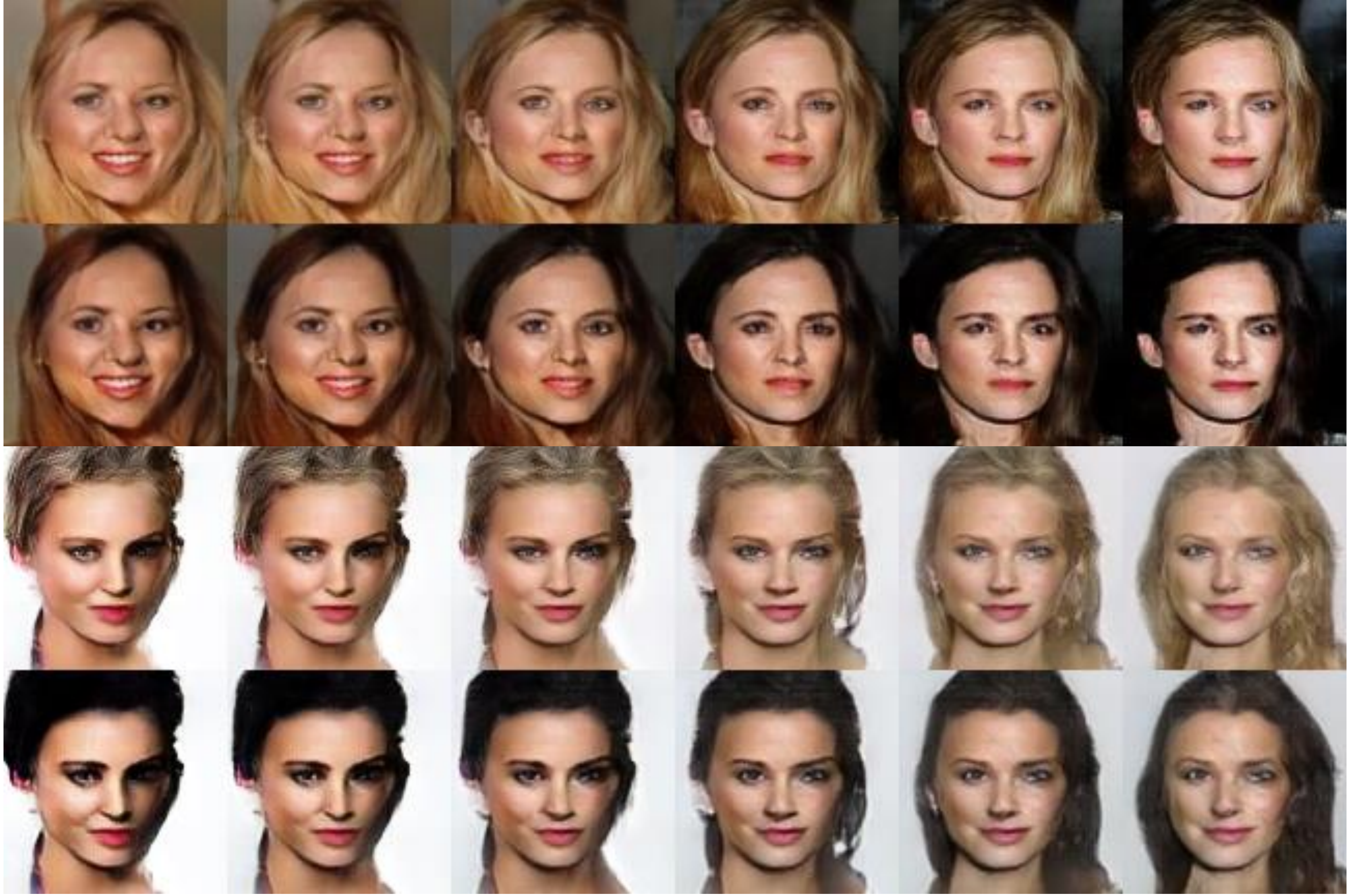
Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016



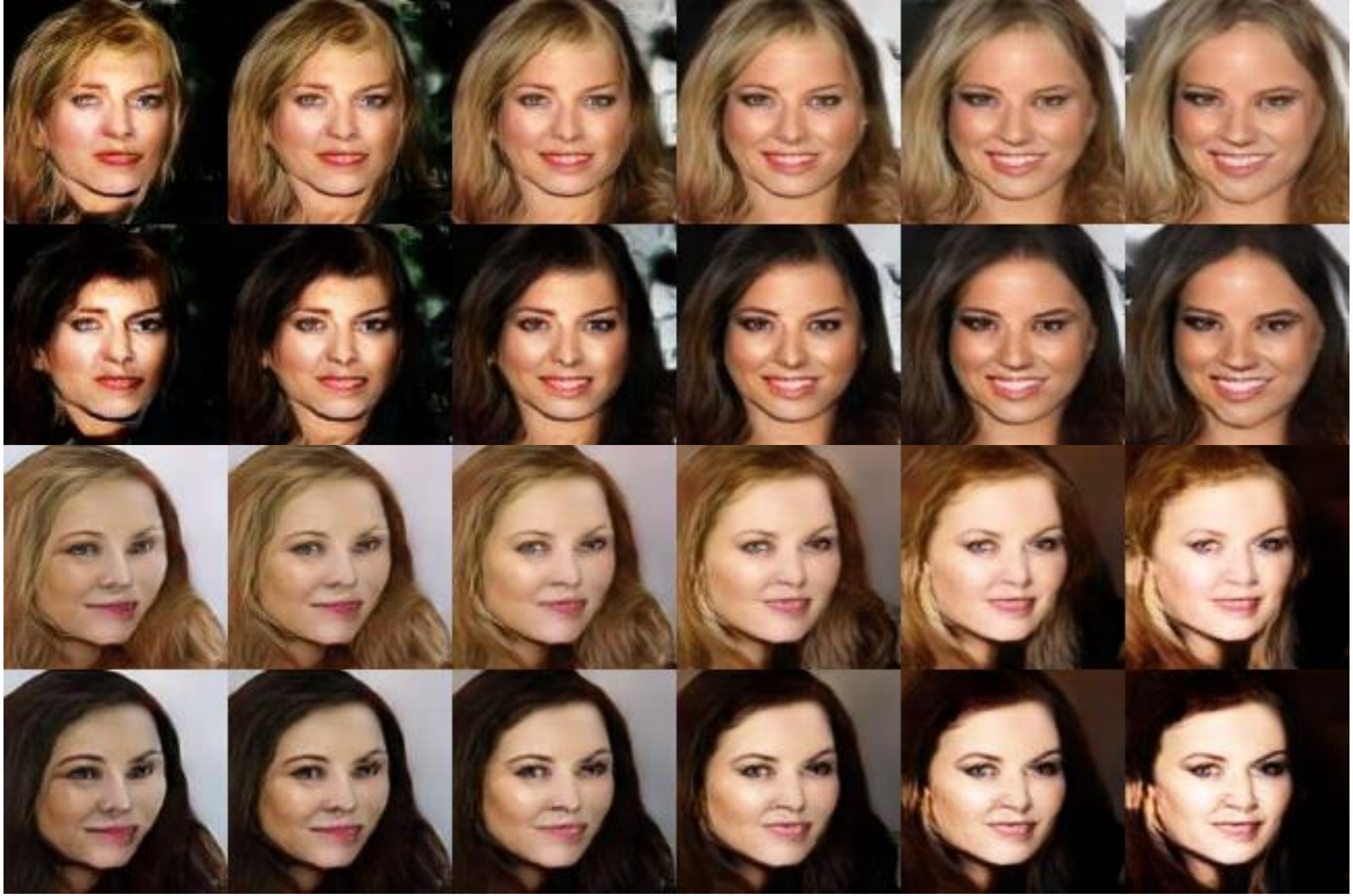
Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

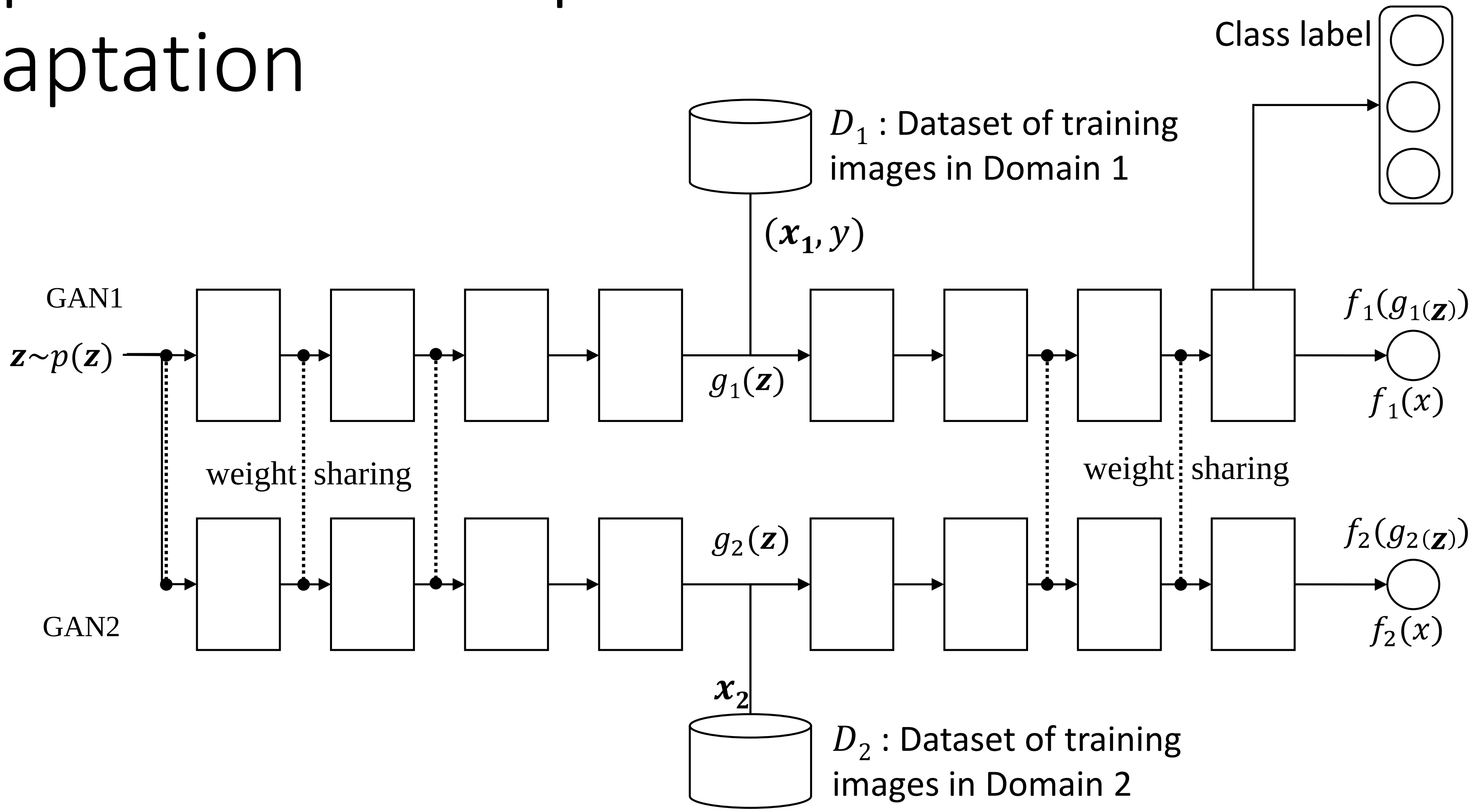


Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

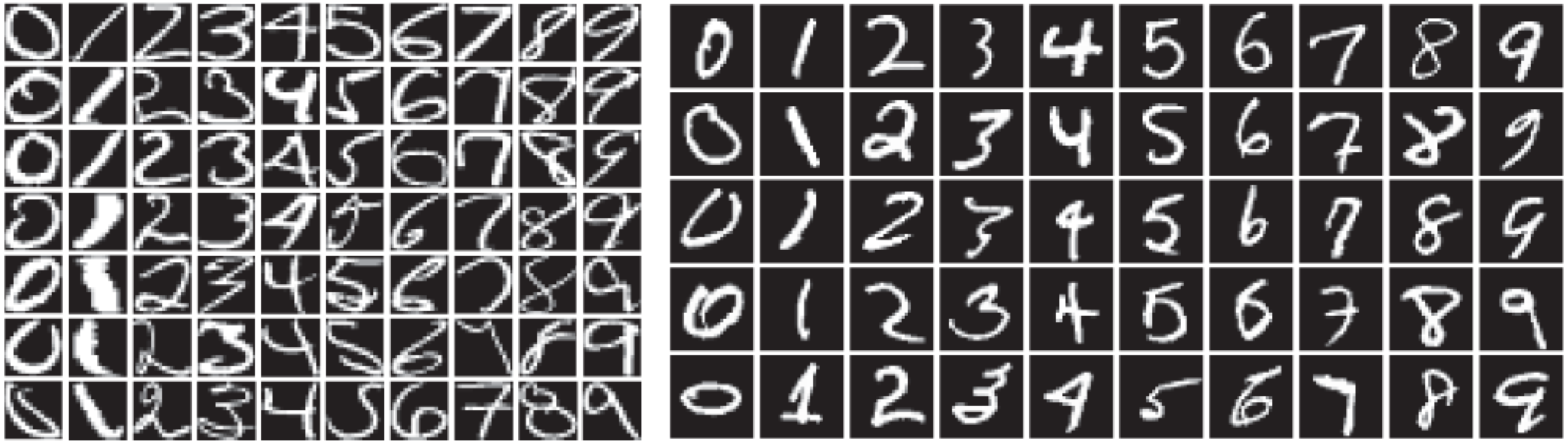


Liu et al, “Coupled Generative Adversarial Networks” NIPS 2016

Application: Unsupervised Domain Adaptation



Unsupervised Domain Adaptation



(a) USPS (b) MNIST

Task \ Method	[18]	[19]	[20]	[21]	CoGAN
MNIST→USPS	0.408	0.467	0.478	0.607	**0.912** ±0.008
USPS→MNIST	0.274	0.355	0.631	0.673	**0.891** ±0.008
Average	0.341	0.411	0.554	0.640	**0.902**

Liu et al, "Coupled Generative Adversarial Networks" NIPS 2016

Conclusions

- We discussed two popular deep generative models
  - Variational Autoencoders
  - Generative Adversarial Networks
- We discussed their pros and cons and how to take the best from both.
- We discussed several computer vision applications of these models.
- Many other applications and interesting properties of these deep generative models are waiting for your exploration.