10707: Deep Learning

Russ Salakhutdinov

Machine Learning Department
rsalakhu@cs.cmu.edu
http://www.cs.cmu.edu/~rsalakhu/10707/

# Generative Adversarial Networks

Statistical Generative Models

+
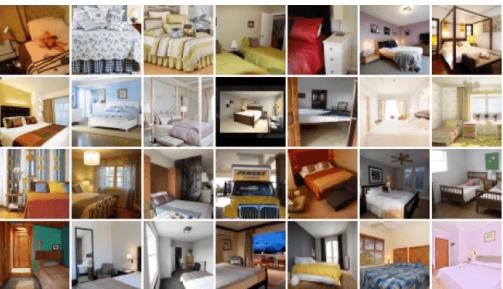
Model family, loss function, optimization algorithm, etc.

Data
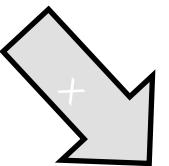
Learning
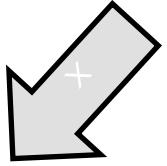
Prior Knowledge

Image x

A probability distribution p(x)

probability p(x)

Sampling from p(x) **generates** new images:

Grover and Ermon, DGM Tutorial

Conditional Generation

▶ Conditional generative model P(zebra images| horse images)

▶ Style Transfer

| Input Image | Monet | Van Gogh |

Zhou el al., Cycle GAN 2017

Fully Observed Models

▸ Density Estimation by Autoregression

$$p(x_1, \ldots, x_d) = \prod_{i=1}^{d} p(x_i | x_{i-1}, \ldots, x_1) \approx \prod_{i=1}^{d} p(x_i | g(x_{i-1}, \ldots, x_1))$$

Each conditional can be a deep neural network

$\emptyset \rightarrow \boxed{h_1 = g(\emptyset, h_0)} \rightarrow \boxed{\sim} \quad p(x_1)$

$x_1 \rightarrow \boxed{h_2 = g(x_1, h_1)} \rightarrow \boxed{\sim} \quad p(x_2|x_1)$

$x_2 \rightarrow \boxed{h_3 = g(x_3, h_2)} \rightarrow \boxed{\sim} \quad p(x_3|x_2, x_1)$

$x_{d-1} \rightarrow \boxed{h_d = g(x_{d-1}, h_{d-1})} \rightarrow \boxed{\sim} \quad p(x_d|x_d, \ldots, x_1)$
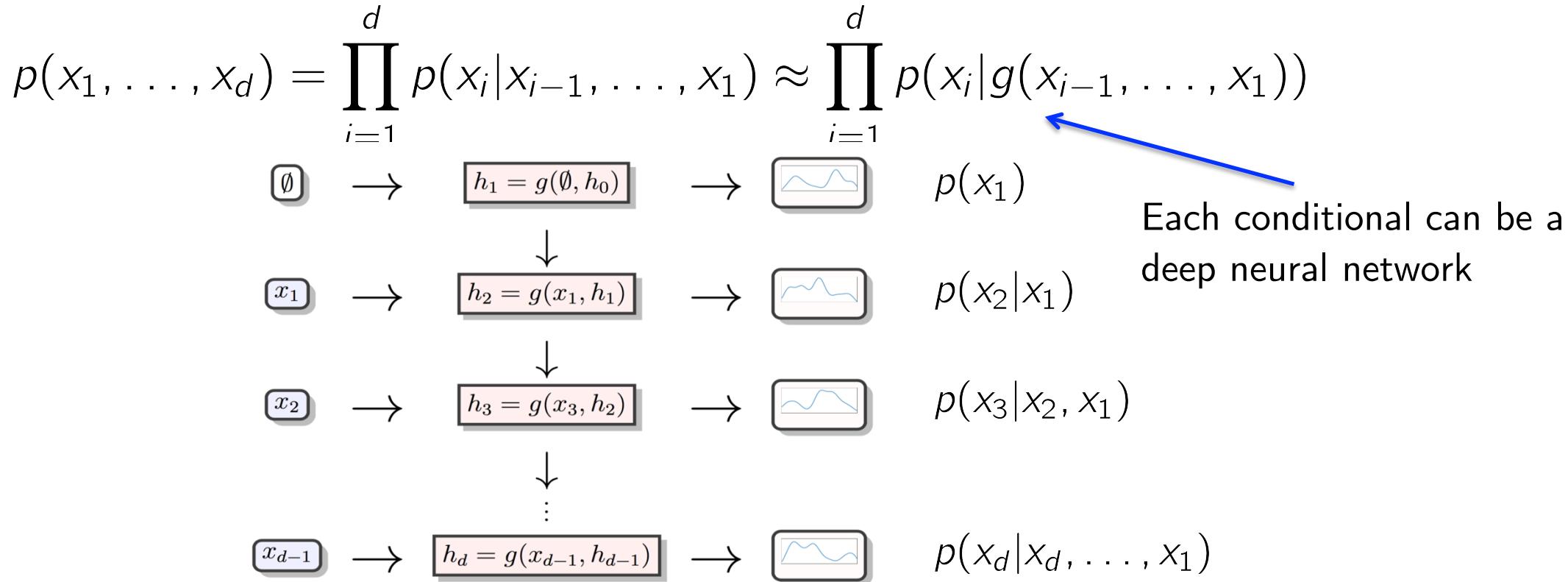
▸ Ordering of variables is crucial

NADE (Uria 2013), MADE (Germain 2017), MAF (Papamakarios 2017), PixelCNN (van den Oord, et al, 2016)

# Fully Observed Models

▶ Density Estimation by Autoregression
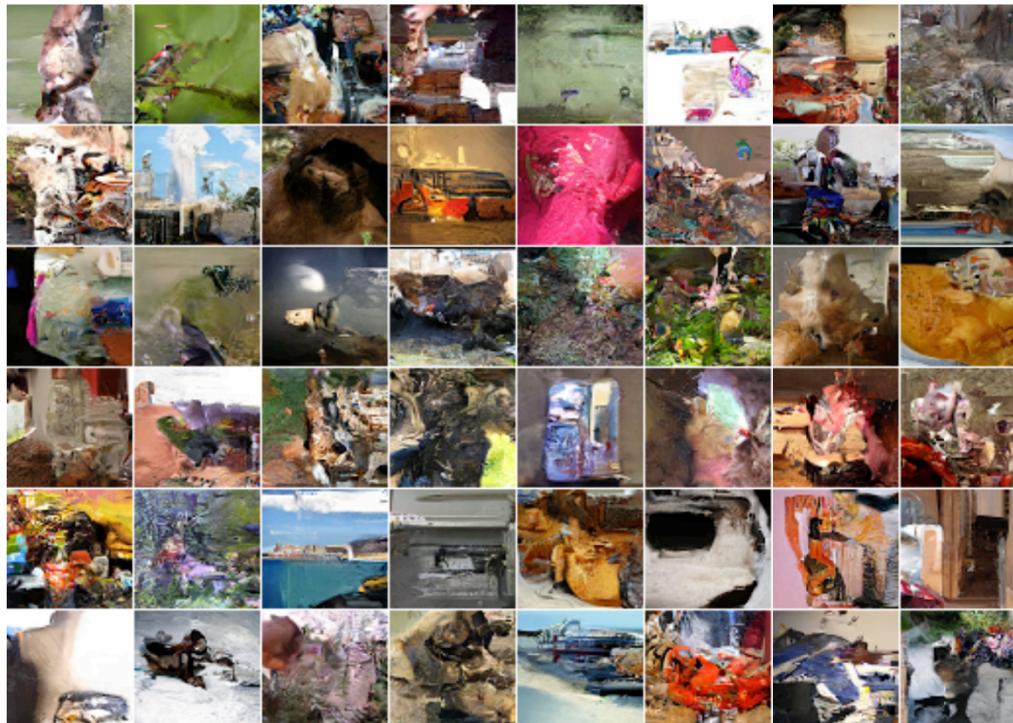


PixelCNN (van den Oord, et al, 2016)

NADE (Uria 2013), MADE (Germain 2017), MAF
(Papamakarios 2017), PixelCNN (van den Oord, et al, 2016)

WaveNet

▸ Generative Model of Speech Signals
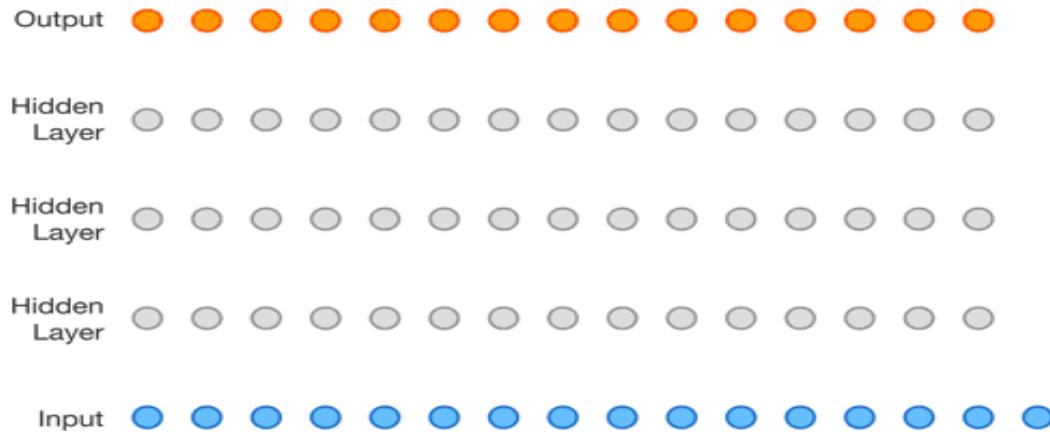
Quality: Mean Opinion Scores
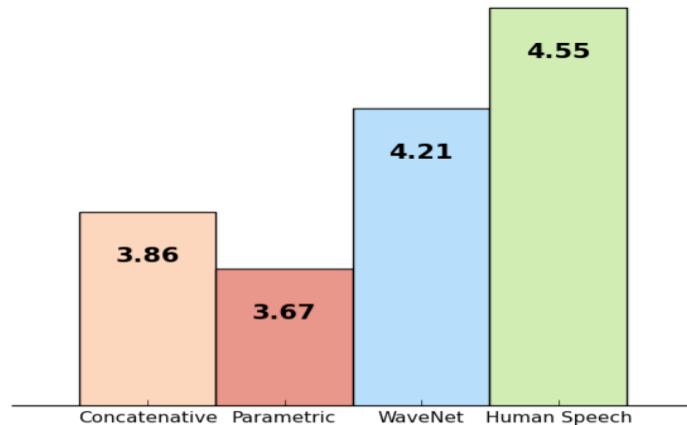
van den Oord et al, 2016

Deep Directed Generative Models

Code Z

▶ Latent Variable Models

$p_\theta(\mathbf{x}|\mathbf{z})$ $q_\phi(\mathbf{z}|\mathbf{x})$
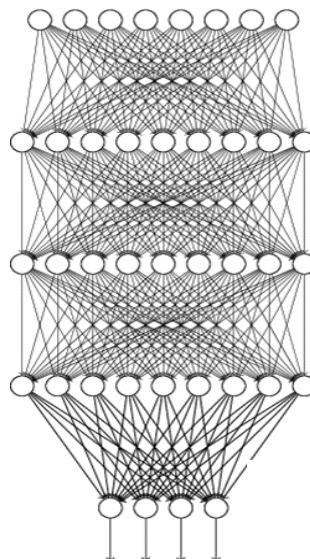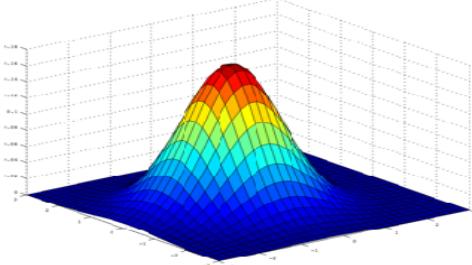
▶ Recognition

▶ Bottom-up

▶ Q(z|x)
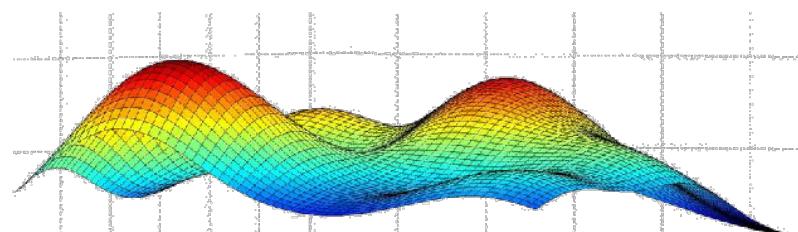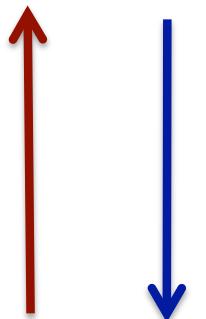
▶ Generative

▶ Top-Down

▶ P(x|z)
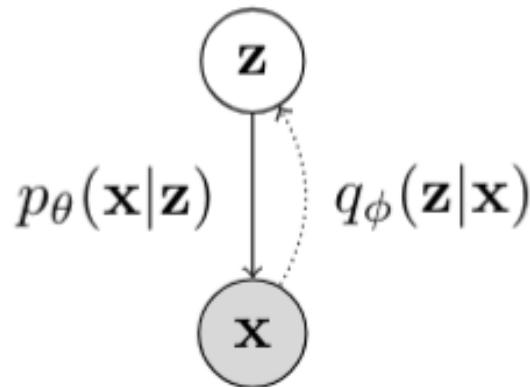
$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z}) \mathrm{d}\mathbf{z}$$

▶ Conditional distributions are parameterized by deep neural networks

D_{real}

Directed Deep Generative Models

▸ Directed Latent Variable Models with Inference Network

$p_\theta(\mathbf{x}|\mathbf{z})$ $q_\phi(\mathbf{z}|\mathbf{x})$
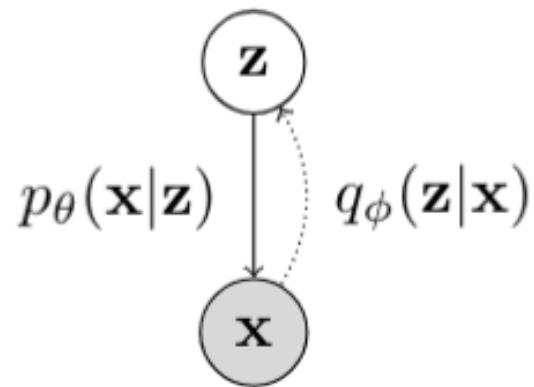
▸ Maximum log-likelihood objective

$$\max_\theta \sum_{\mathbf{x} \in \mathcal{D}} \log p_\theta(\mathbf{x})$$

▸ Marginal log-likelihood is intractable:

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z}) \mathrm{d}\mathbf{z}$$

▸ Key idea: Approximate true posterior p(z|x) with a simple, tractable distribution q(z|x) (inference/recognition network).

Grover and Ermon, DGM Tutorial

Variational Autoencoders (VAEs)

▶ Single stochastic (Gaussian) layer, followed by many deterministic layers

z
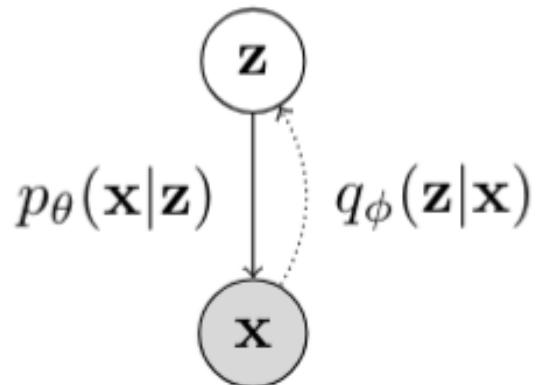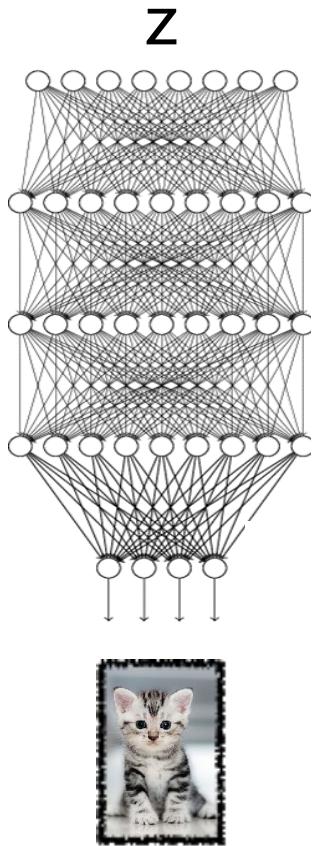
$$p(\mathbf{z}) = \mathcal{N}(0, I)$$

$$p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}\big(\mu(\mathbf{z}, \theta), \Sigma(\mathbf{z}, \theta)\big)$$

Deep neural network parameterized by θ.
(Can use different noise models)

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}\big(\mu(\mathbf{x}, \phi), \Sigma(\mathbf{x}, \phi)\big)$$

Deep neural network parameterized by φ.

Generative Adversarial Networks (GAN)

▶ Implicit generative model for an unknown target density $p(x)$

▶ Converts sample from a known noise density $p_Z(z)$ to the target $p(x)$

Unknown target density $p(x)$ of data over domain $\mathcal{X}$, e.g. $\mathbb{R}^{32 \times 32}$
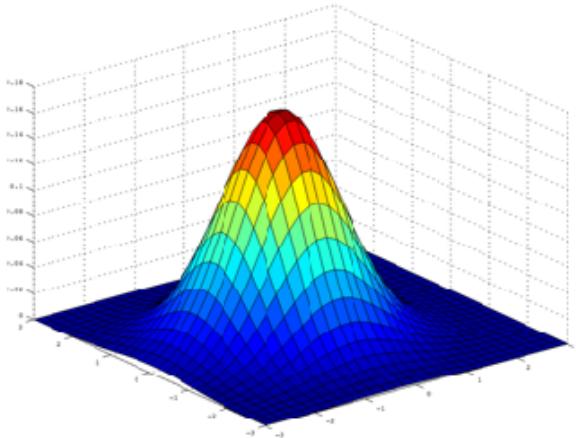
Noise density $p_Z(z)$ over space $\mathcal{Z}$
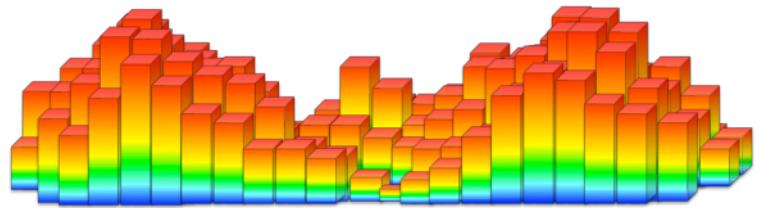
Distribution of generated samples should follow target density $p(x)$

[Slide Credit: Manzil Zaheer]

Goodfellow et al, 2014

# GAN Formulation

▶ GAN consists of two components

Generator

$$G : \mathcal{Z} \to \mathcal{X}$$

Random input
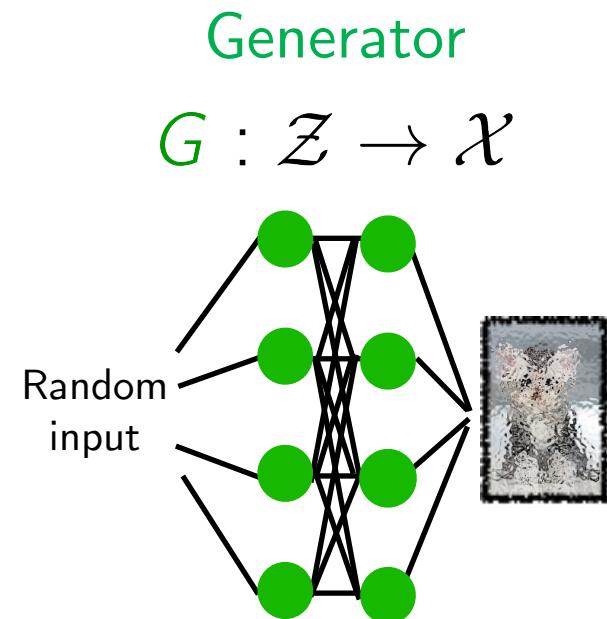
Goal: Produce samples indistinguishable from true data

Discriminator

$$D : \mathcal{X} \to \mathbb{R}$$
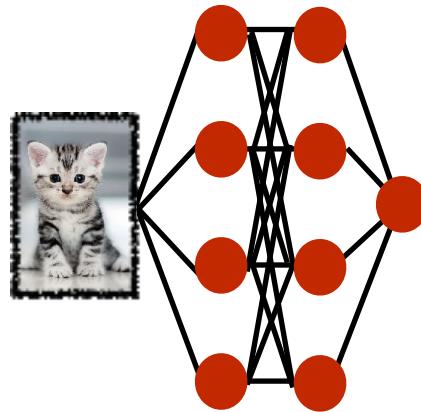
Goal: Distinguish true and generated data apart

[Slide Credit: Manzil Zaheer]

Goodfellow et al, 2014

GAN Formulation: Discriminator

▸ Discriminator's objective: Tell real and generated data apart like a classifier

$$\max_{D} \mathbb{E}_{x \sim p} \left[\log D(x) \right] + \mathbb{E}_{z \sim p_Z} \left[\log \left(1 - D(G(z)) \right) \right]$$

Real Data $p(x)$

Discriminator

D outputs:

$D(x) = 1$ real

$D(x) = 0$ generated

Generator

$p_Z(z)$ Random input

[Slide Credit: Manzil Zaheer]

Goodfellow et al, 2014
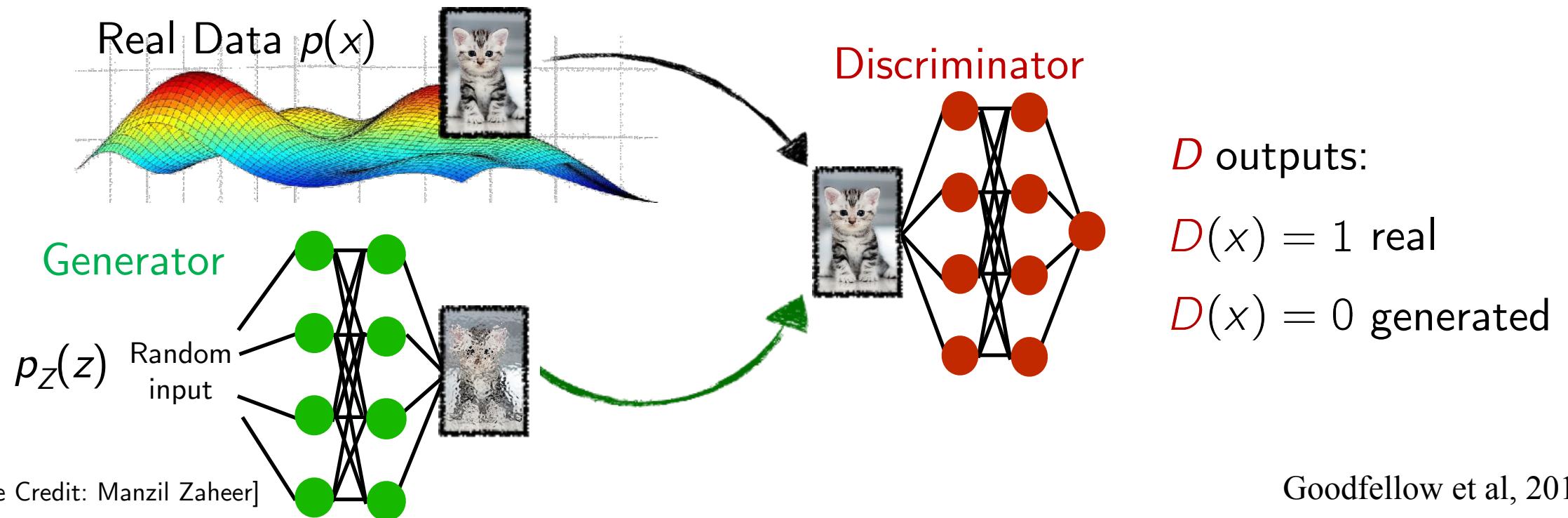
GAN Formulation: Generator

▸ Generator's objective: Fool the best discriminator

$$\min_{G} \max_{D} \mathbb{E}_{x \sim p}\big[\log D(x)\big] + \mathbb{E}_{z \sim p_Z}\big[\log\big(1 - D(G(z))\big)\big]$$

Real Data $p(x)$

Discriminator

D outputs:

$D(x) = 1$ real

$D(x) = 0$ generated

Generator

$p_Z(z)$ Random input

[Slide Credit: Manzil Zaheer]

Goodfellow et al, 2014
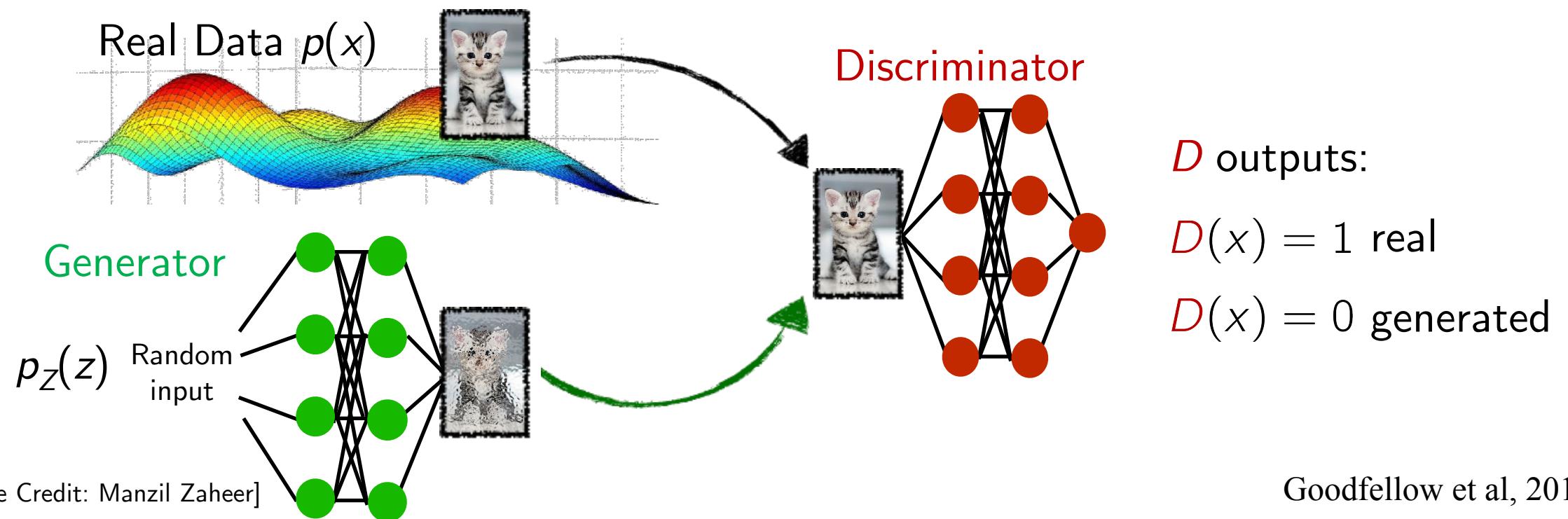
GAN Formulation: Optimization

▸ Overall GAN optimization

$$\min_{G} \max_{D} V(G, D) = \mathbb{E}_{x \sim p}\big[\log D(x)\big] + \mathbb{E}_{z \sim p_Z}\big[\log\big(1 - D(G(z))\big)\big]$$

▸ The generator-discriminator are iteratively updated using SGD to find "equilibrium" of a "min-max objective" like a game

$$G \leftarrow G - \eta_G \nabla_G V(G, D)$$

$$D \leftarrow D - \eta_D \nabla_D V(G, D)$$

[Slide Credit: Manzil Zaheer]
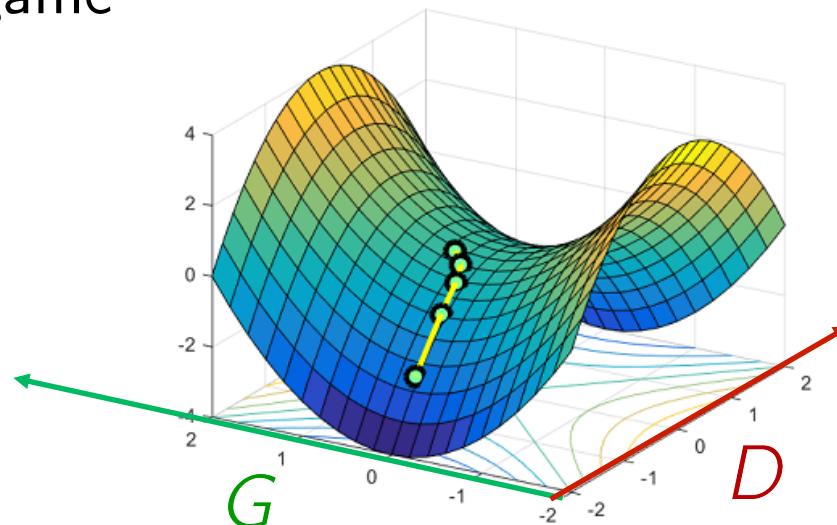
Distributional perspective - Discriminator

$$\min_{G} \max_{D} V(G, D) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]$$

▸ For a fixed generator, discriminator is maximizing negative cross entropy

▸ Optimal discriminator is given by:

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

Goodfellow et al, 2014

A minimax learning objective

▸ During learning, generator and discriminator are updated alternatively

$$\min_\theta \max_\phi V(G_\theta, D_\phi) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}}[\log D_\phi(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]$$


Data distribution .... Discriminator ———— Generator

(a)	(b)	(c)	(d)
Current state	Update discriminator	Update generator	Convergence

Goodfellow et al, 2014
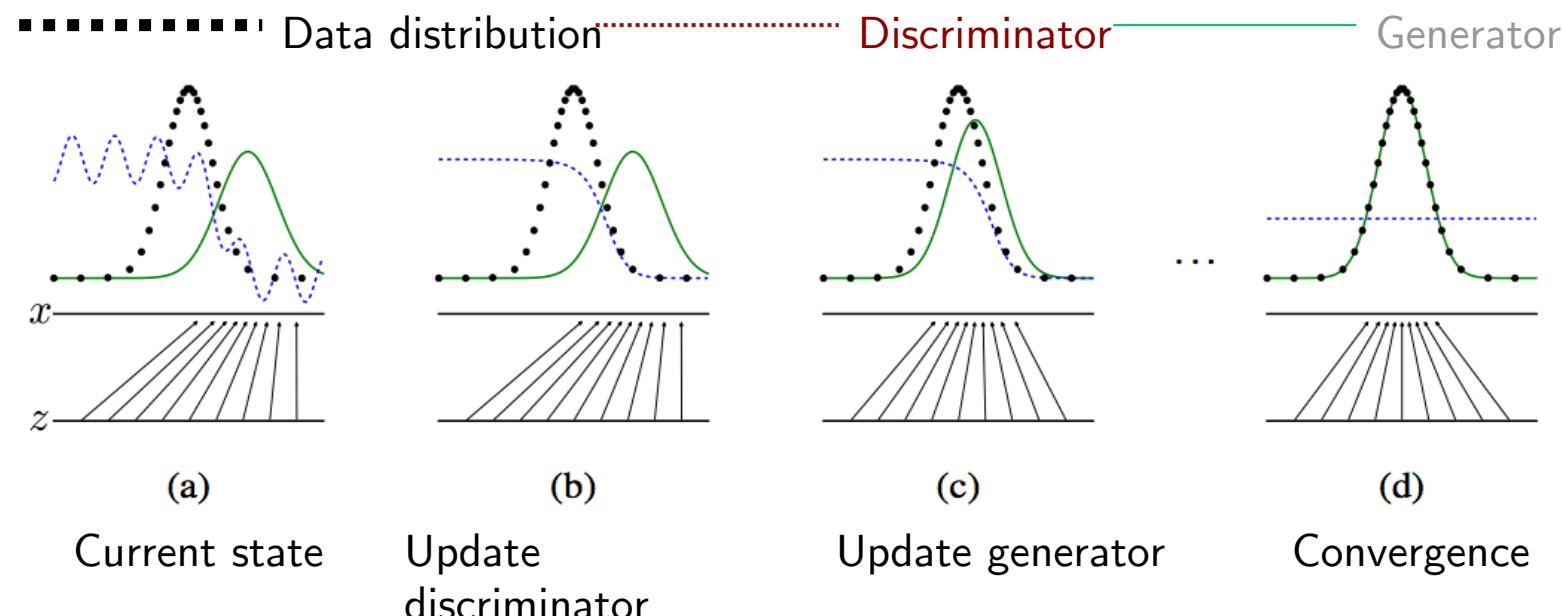
Evaluation

▸ Likelihoods may not be defined or tractable

▸ Directed model permits ancestral sampling
  - ▸ For labelled datasets, metrics such as inception scores quantify sample diversity and quality using pretrained classifiers

Wu et al., 2017, Grover et al., 2018, Salimans et al., 2016, Heusel et al., 2018

Mode Collapse

► In practice, GANs suffer from mode collapse

Arjovsky et al., 2017

Wasserstein GAN

▸ WGAN optimization

$$\min_{G} \max_{D} W(G, D) = \mathbb{E}_{x \sim p}\big[D(x)\big] - \mathbb{E}_{z \sim p_Z}\big[D(G(z))\big]$$

▸ Difference in expected output on real vs. generated images

    ▸ Generator attempts to drive objective ≈ 0

▸ More stable optimization

Compare to training DBMs

$$\frac{\partial \log P_\theta(\mathbf{v})}{\partial W^1} = \mathbb{E}_{P_{data}}[\mathbf{v}\mathbf{h^1}^\top] - \mathbb{E}_{P_\theta}[\mathbf{v}\mathbf{h^1}^\top]$$

D outputs:

$D(x) = 1$ real

$D(x) = 0$ generated

Arjovsky et al., 2017

LSUN Bedroom: Samples

Radford et al., 2015
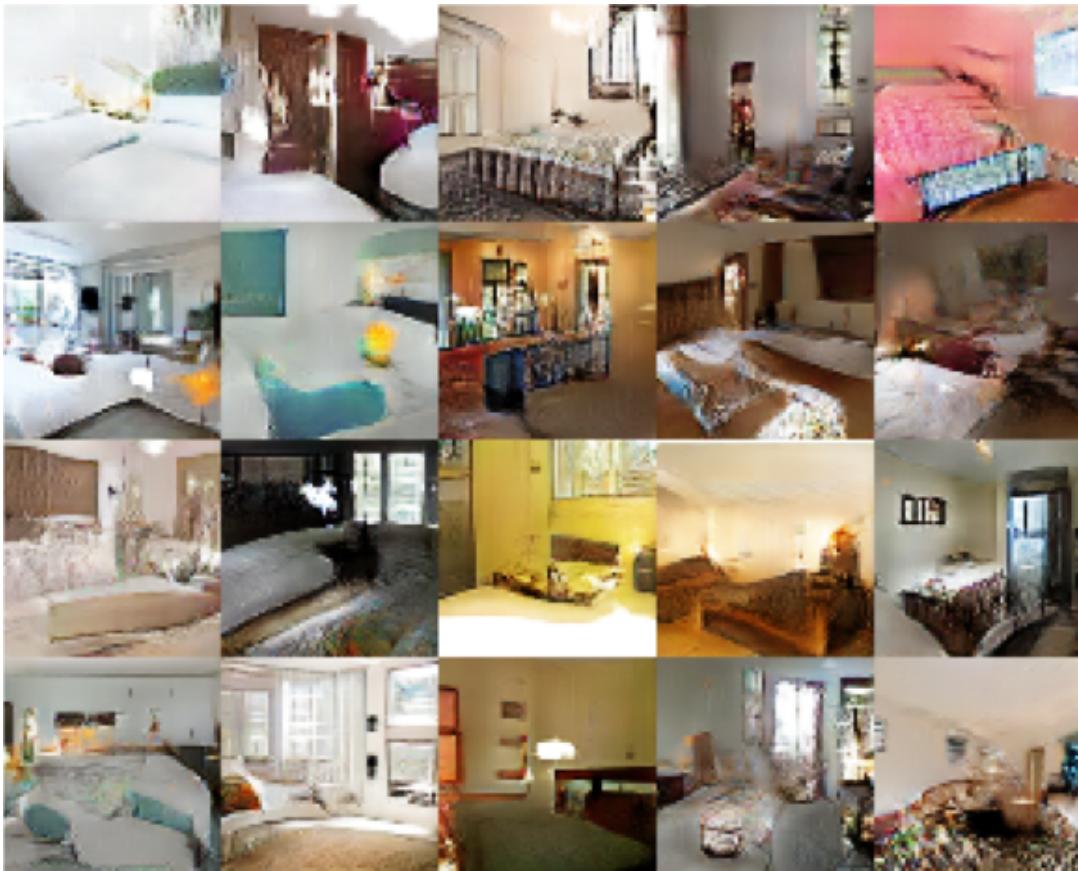
CIFAR Dataset

Training

Samples

Salimans et. al., 2016
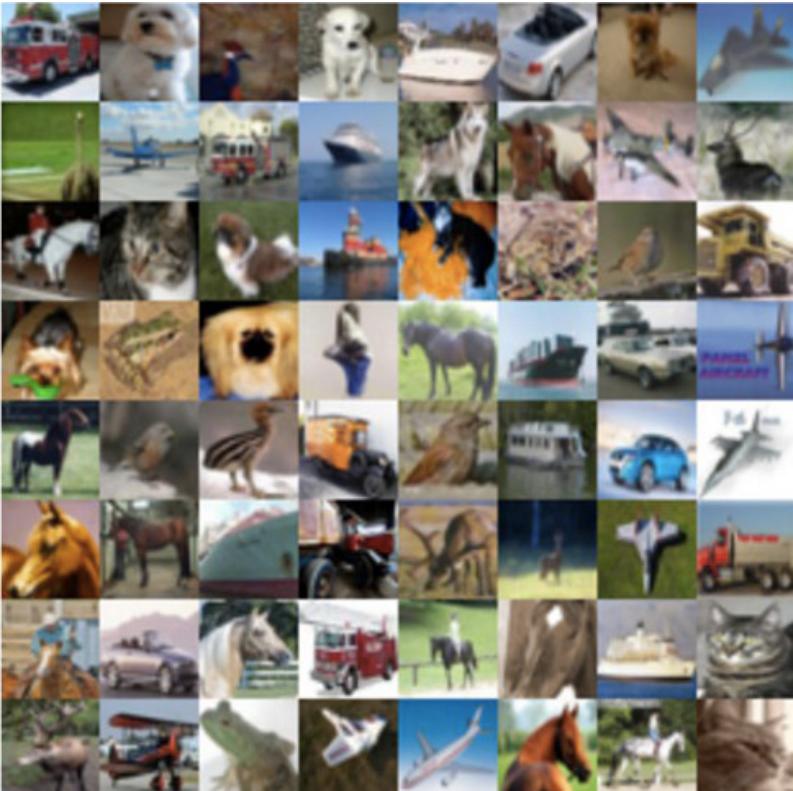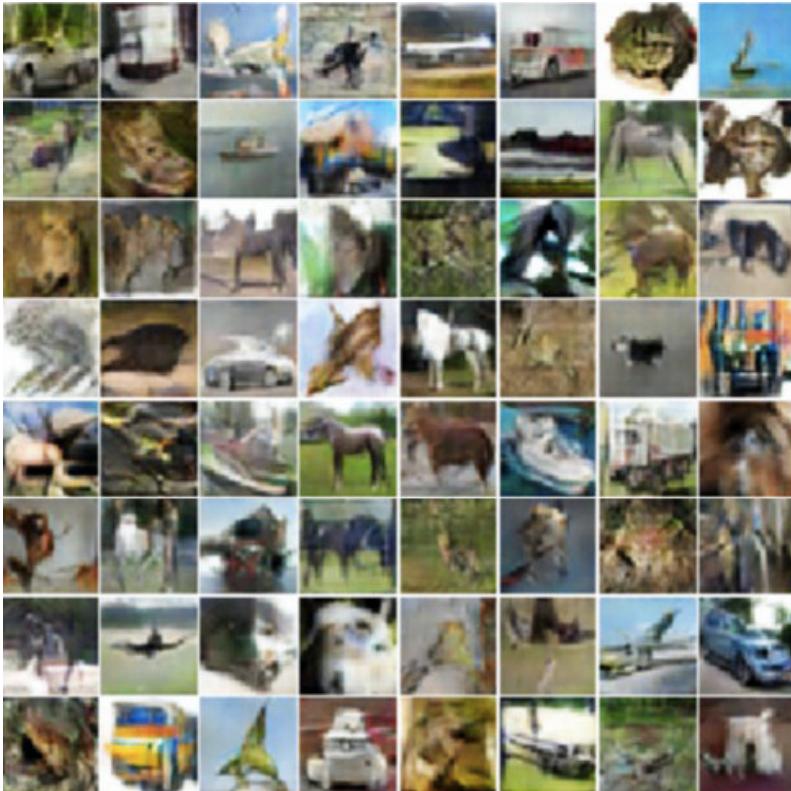
ImageNet: Cherry-Picked Samples
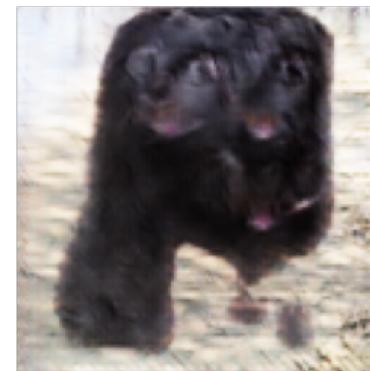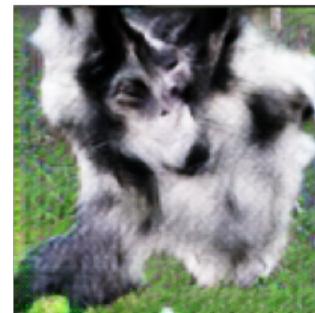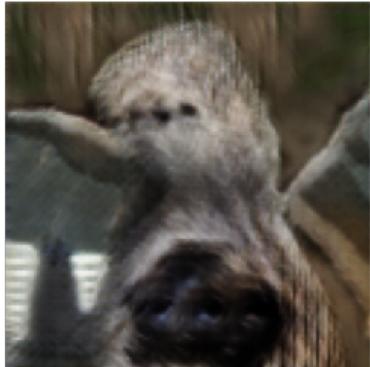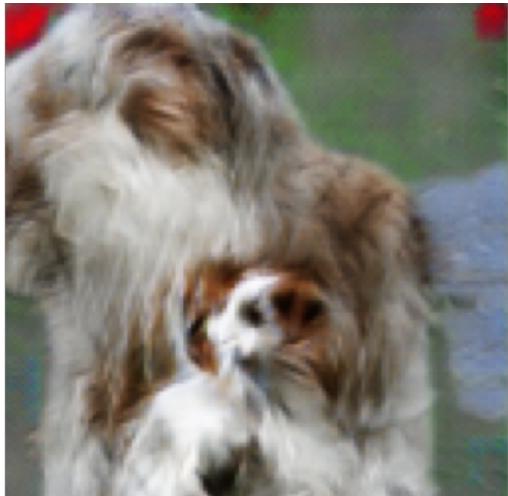
▶ Open Question: How can we quantitatively evaluate these models!

Slide Credit: Ian Goodfellow

Modelling Point Cloud Data

| Data | AAE | PC-GAN | Data | AAE | PC-GAN |

(a) Lamp (b) Chair

(c) Plane (d) Guitar

Zaheer et al. Point Cloud GAN 2018
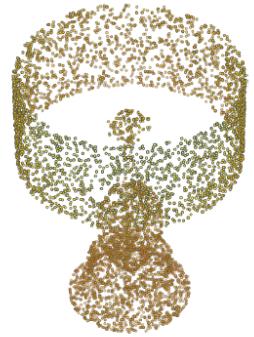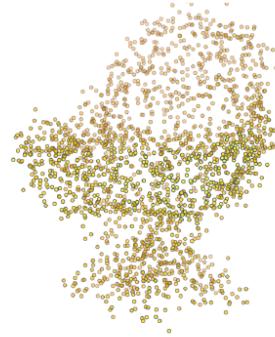
Interpolation in Latent Space

Interpolate

z ⇢ z

x   x

Chair   Table

Cycle GAN

Paired

x_i　　y_i
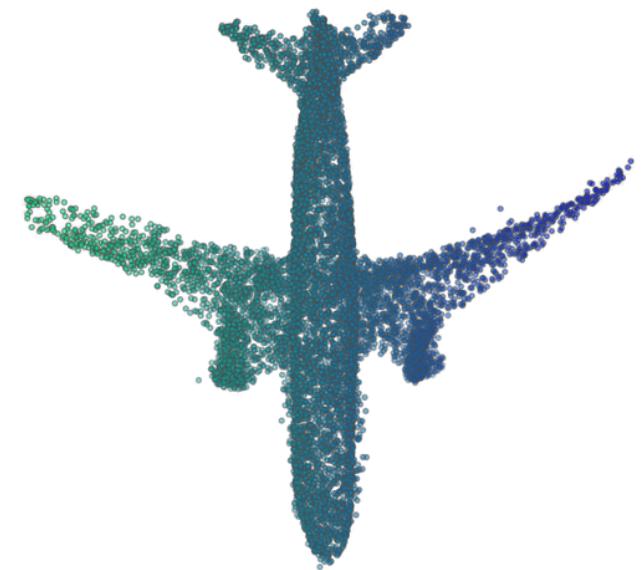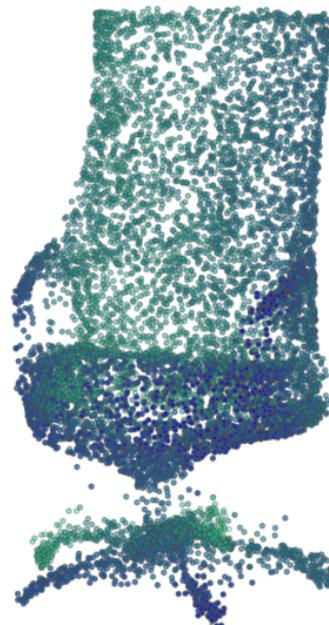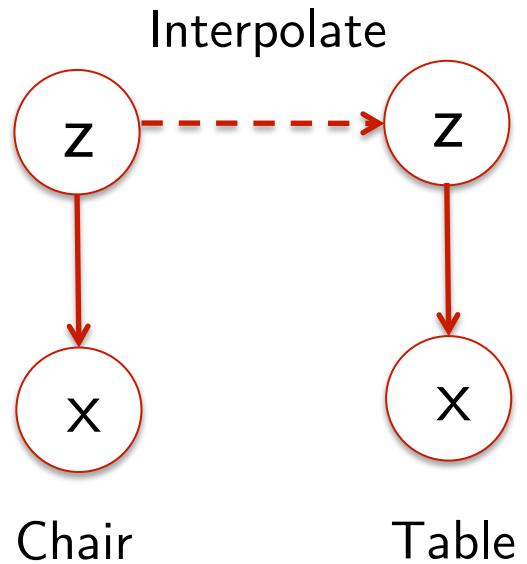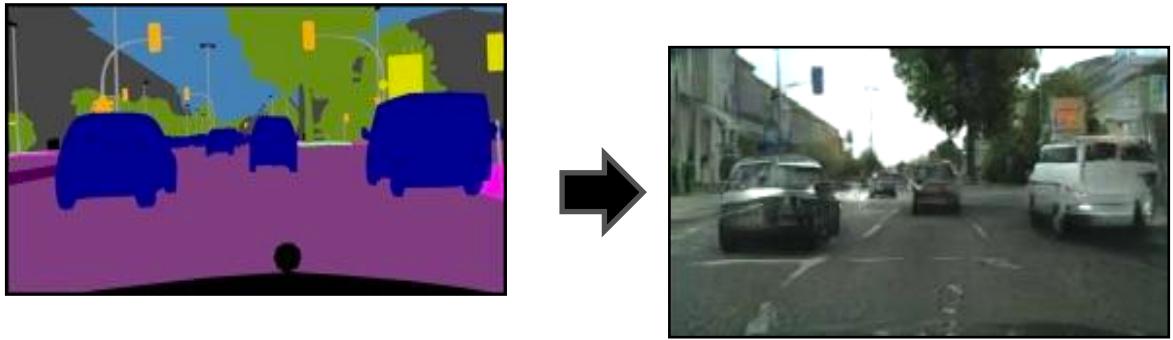

Label ↔ photo: per-pixel labeling
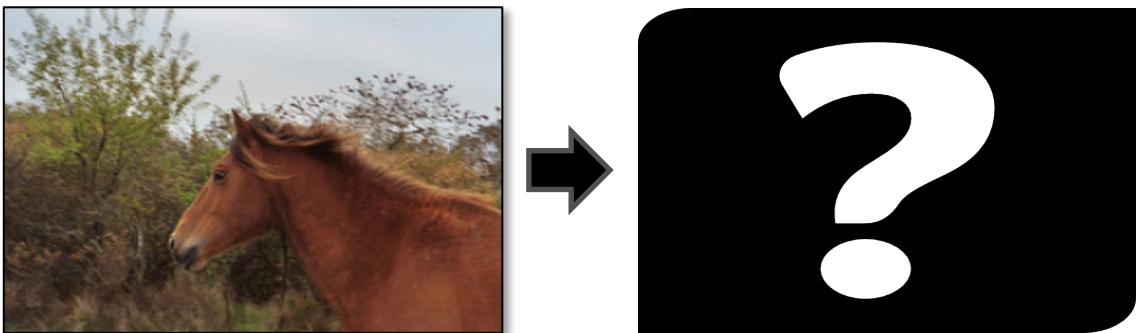
Horse ↔ zebra: how to get zebras?

- Expensive to collect pairs.
- Impossible in many scenarios.

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

Slide credit: Jun-Yan Zhu

Cycle GAN

Paired

x_i y_i

Unpaired

X Y
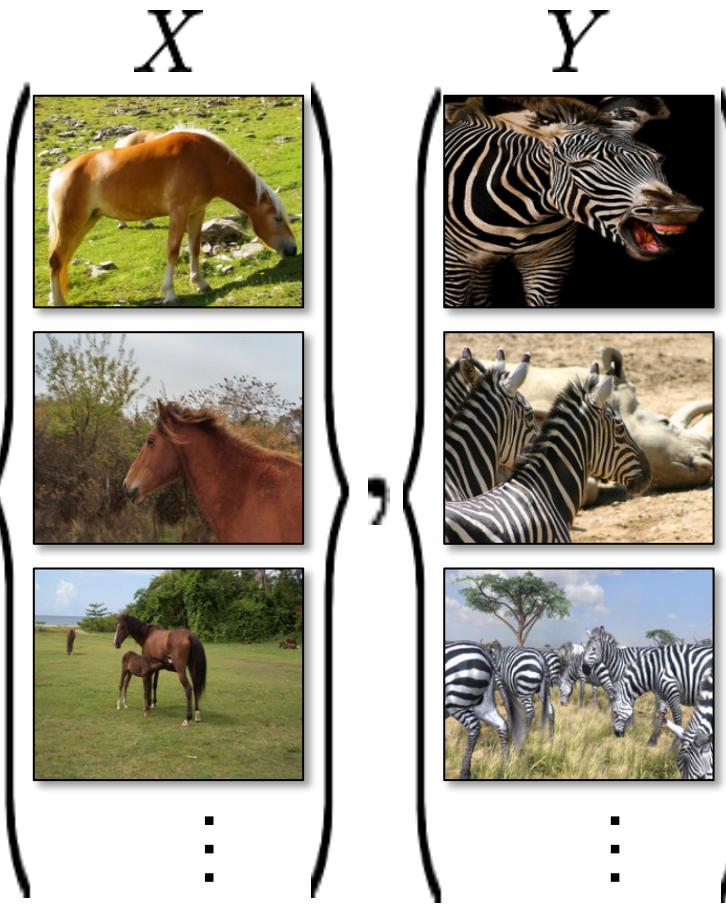
[Zhu*, Park*, Isola, and Efros, ICCV 2017]

No input-output pairs!

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

Generator → Discriminator → **Real**

[Zhu*, Park*, Isola, and Efros, ICCV 2017]
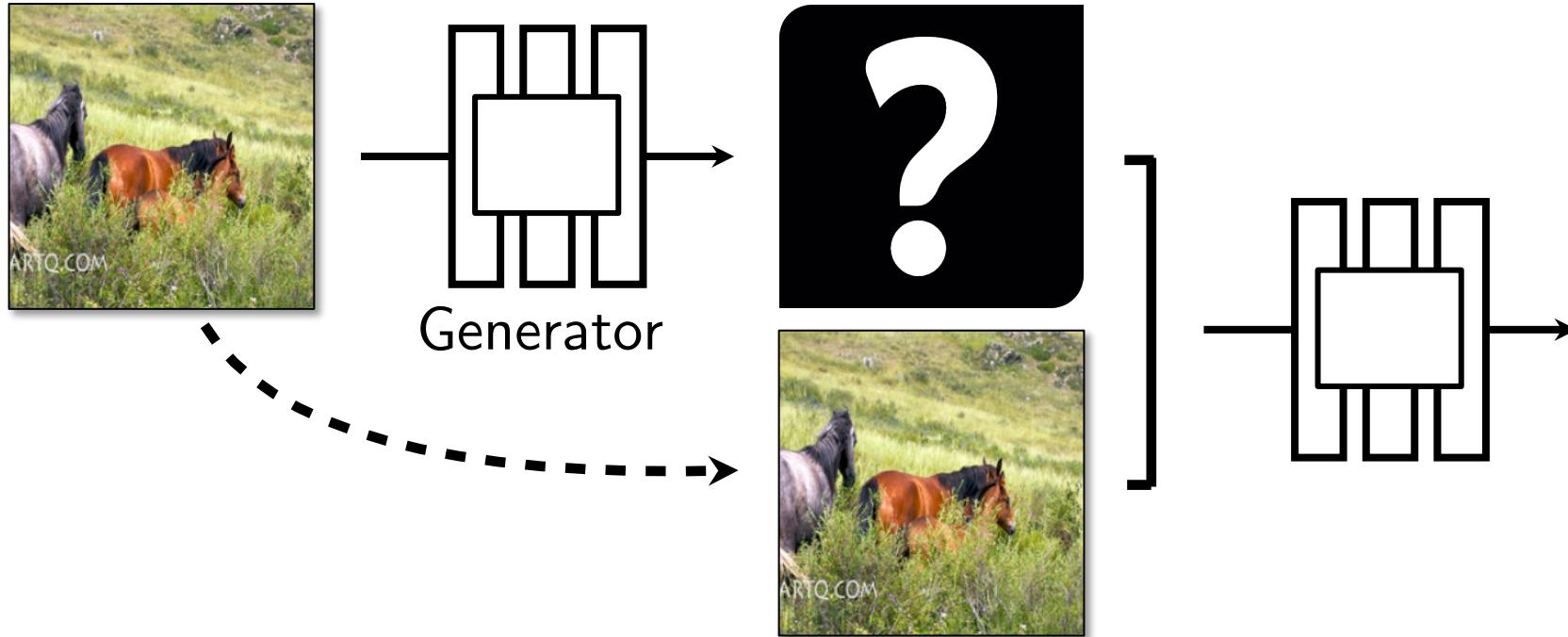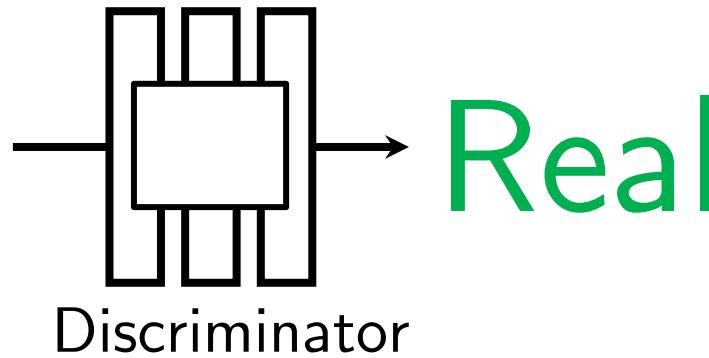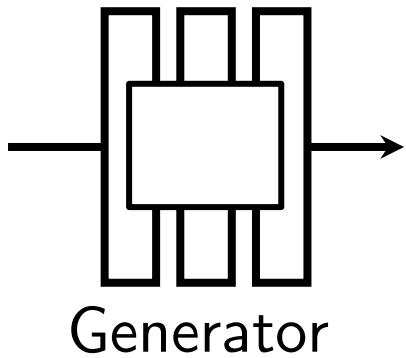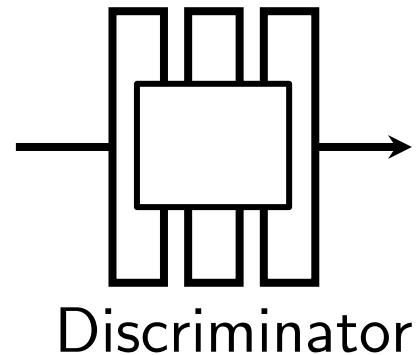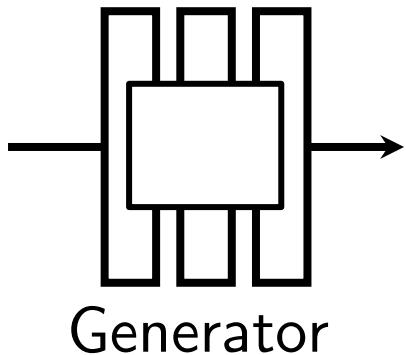
Generator · Discriminator · Real too

GANs doesn't force output to correspond to input

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

mode collapse

Slide credit: Jun-Yan Zhu

Cycle Consistent Adversarial Networks

Slide credit: Jun-Yan Zhu

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

Cycle Consistent Adversarial Networks

X

Y

$X \rightleftarrows Y$

D_Y

[Mark Twain, 1903]

Slide credit: Jun-Yan Zhu

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

Cycle Consistency Loss

$$D_Y(G(x))$$

Reconstruction error

$$\|F(G(x)) - x\|_1$$

[Zhu*, Park*, Isola, and Efros, ICCV 2017]
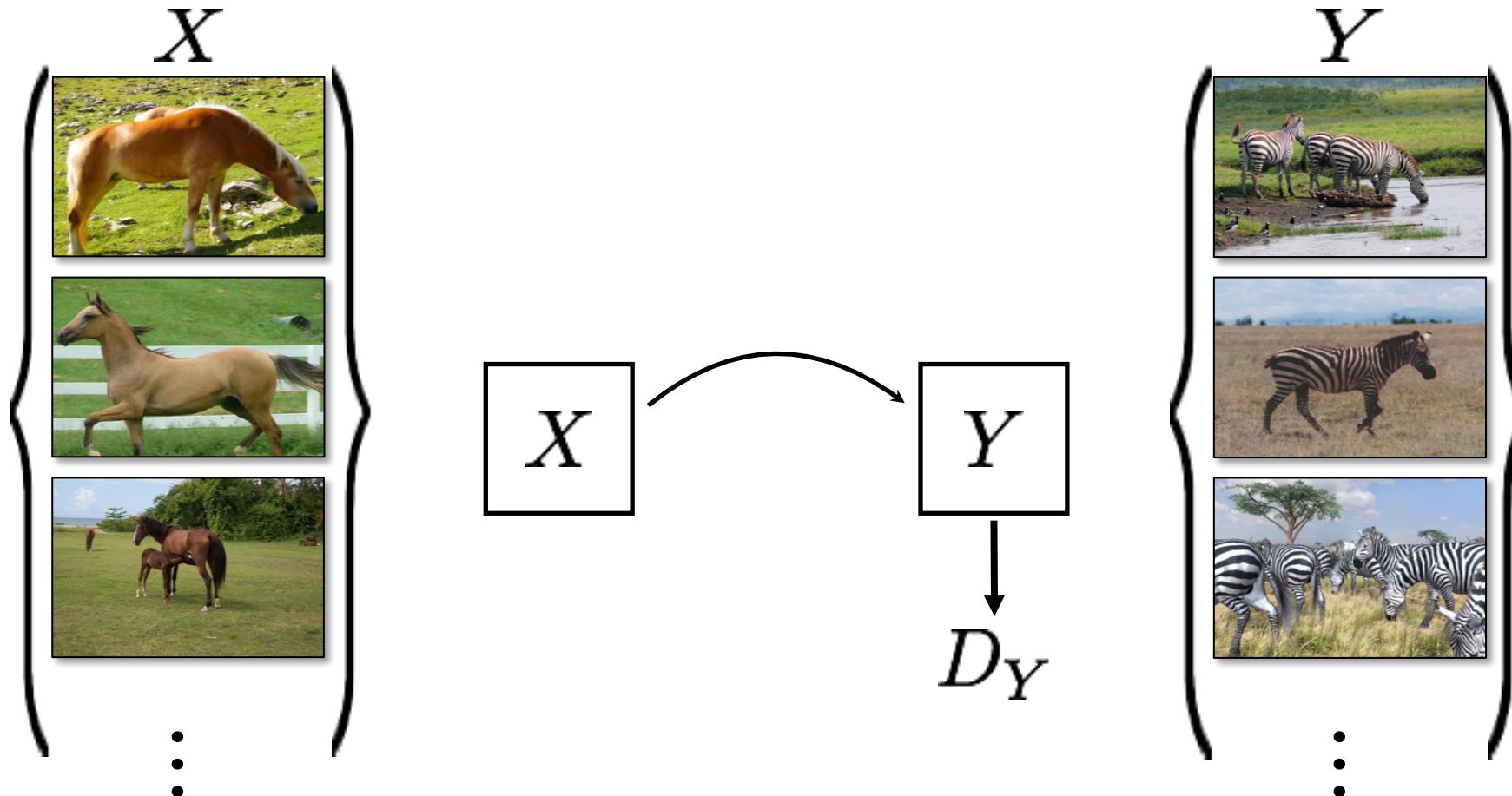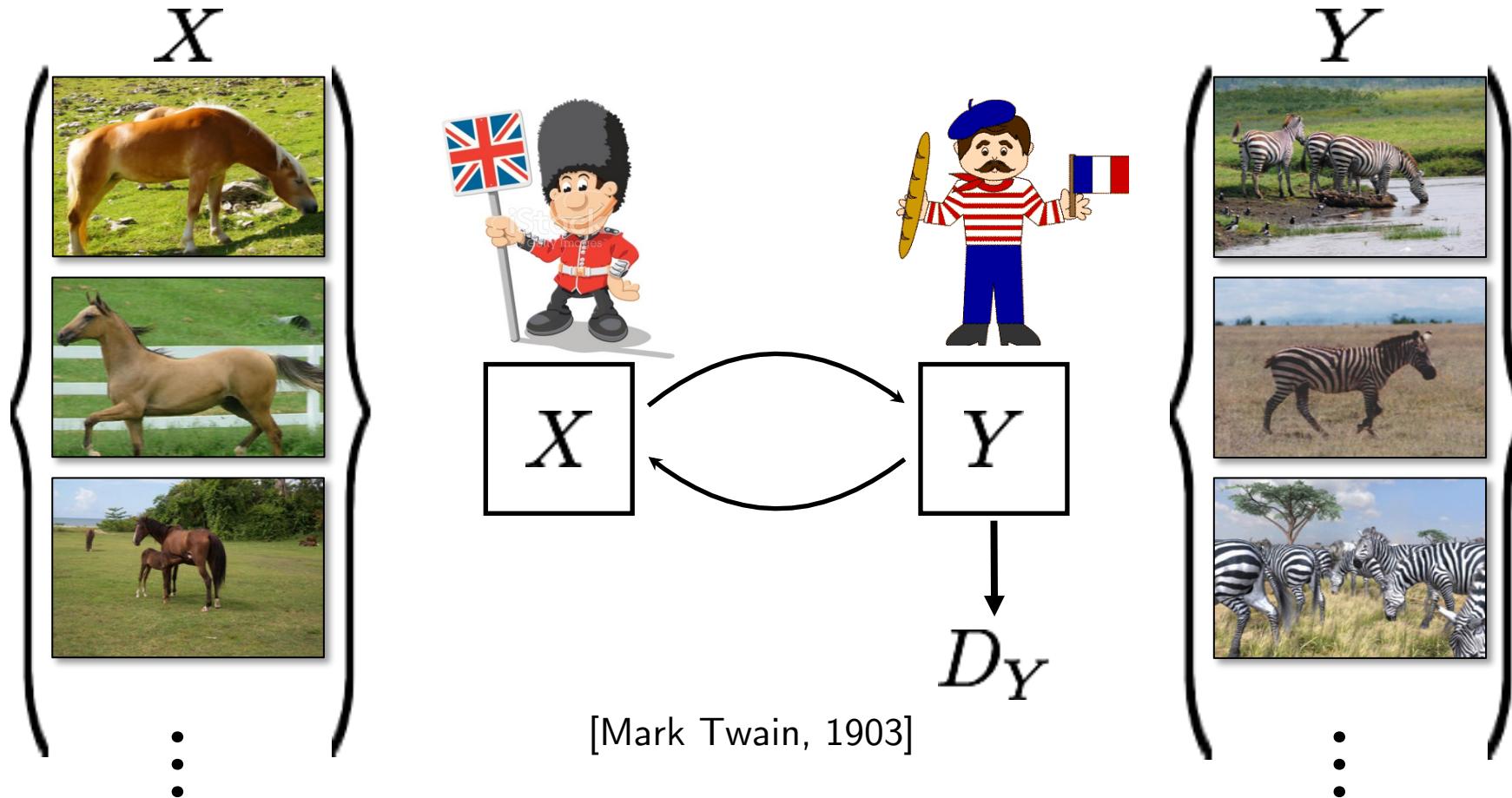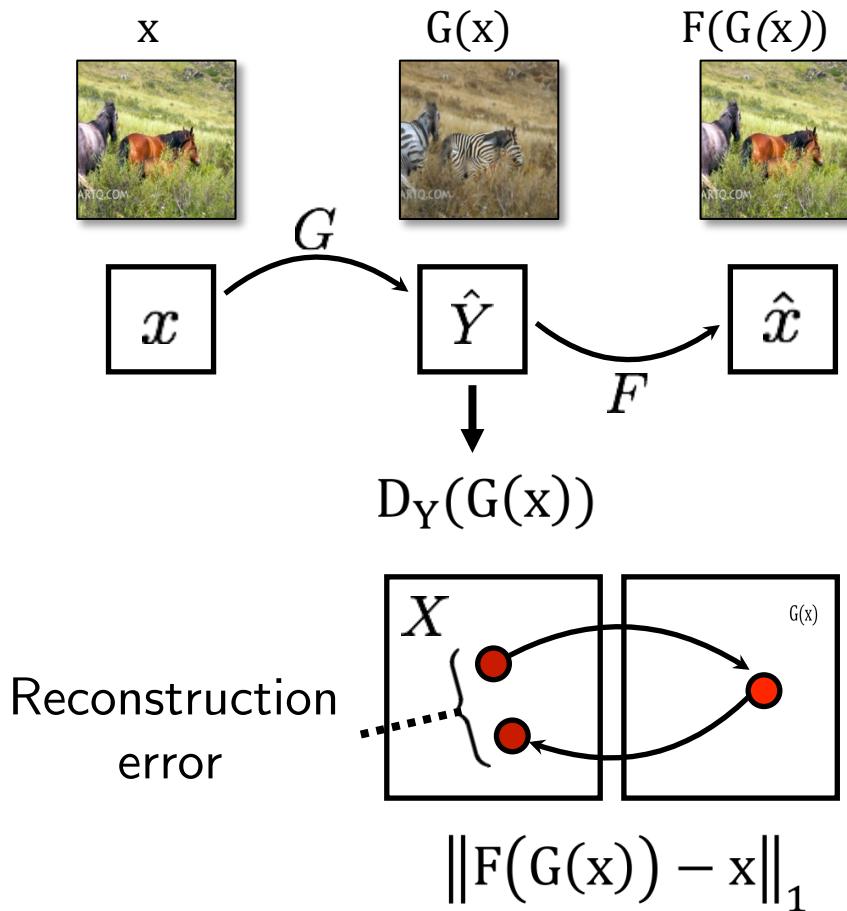
Cycle Consistency Loss
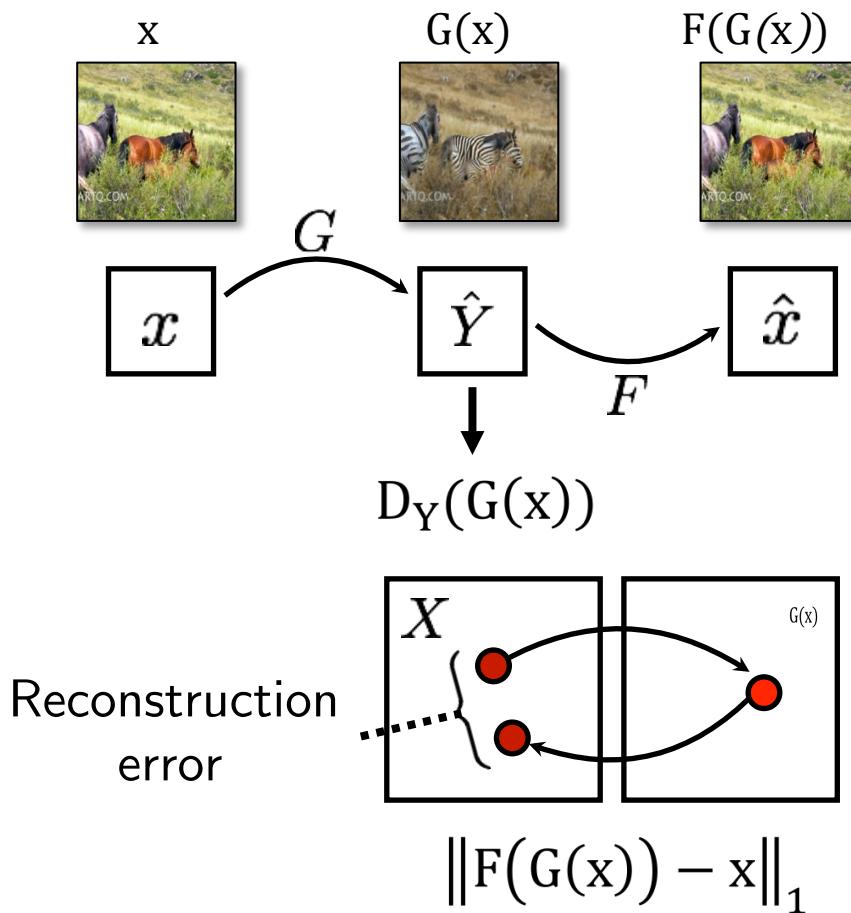
Small cycle loss
Large cycle loss

Reconstruction error $\|F(G(x)) - x\|_1$

[Zhu*, Park*, Isola, and Efros, ICCV 2017]

Slide credit: Jun-Yan Zhu

Cycle Consistency Loss

Reconstruction error

Reconstruction error

[Zhu*, Park*, Isola, and Efros, ICCV 2017]
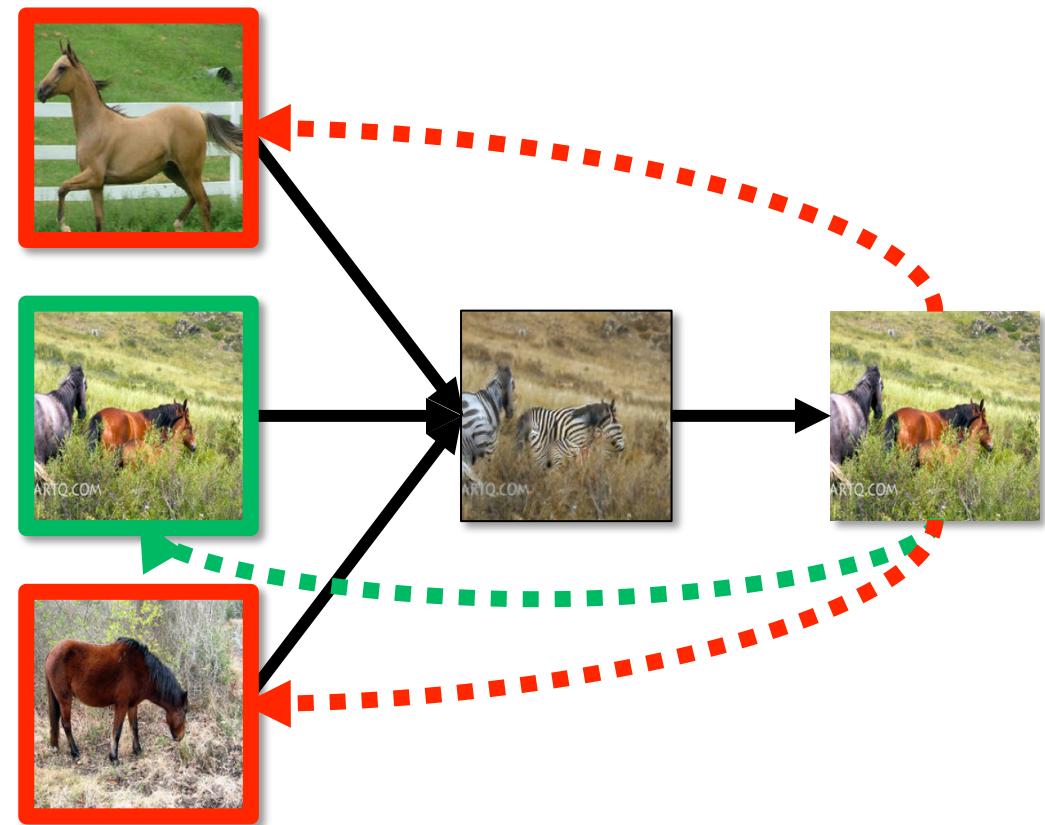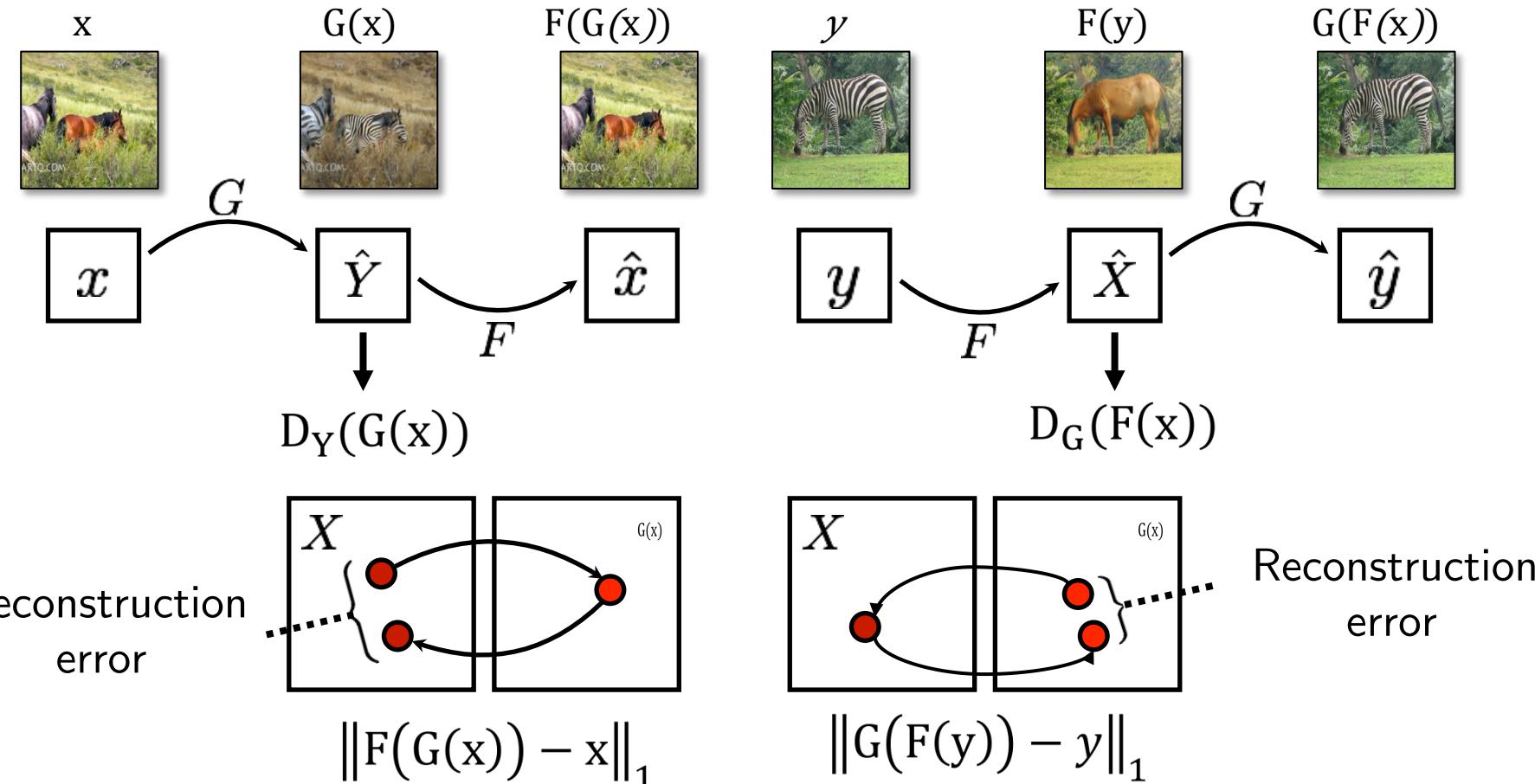
Collection Style Transfer

Ukiyo-e   Cezanne

Van Gogh   Monet

| Input | Monet | Van Gogh | Cezanne | Ukiyo-e |

Conditional Generation

▶ Conditional generative model P(zebra images| horse images)

▶ Style Transfer

Input Image   Monet   Van Gogh

Zhou el al., Cycle GAN 2017

Normalizing Flows

▸ Directed Latent Variable Invertible models
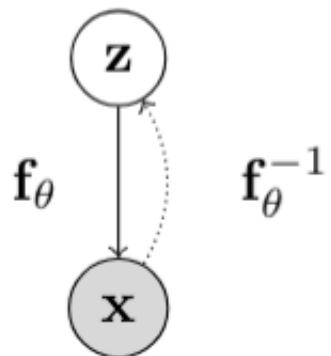
> ▸ The mapping between x and z is deterministic and invertible:

$$\mathbf{x} = \mathbf{f}_\theta(\mathbf{z})$$
$$\mathbf{z} = \mathbf{f}_\theta^{-1}(\mathbf{x})$$

▸ Use change-of-variables to relate densities between z and x

$$p_X(\mathbf{x}; \theta) = p_Z(\mathbf{z}) \left| \det \frac{\partial \mathbf{f}_\theta^{-1}(\mathbf{x})}{\partial X} \right|_{X=\mathbf{x}}$$

Grover and Ermon DGM Tutorial, NICE (Dinh et al. 2014),
Real NVP (Dinh et al. 2016)

Normalizing Flows

▶ Invertible transformations can be composed:

$$\mathbf{z}^M \triangleq \mathbf{f}_\theta^M = \mathbf{f}_\theta^M \circ \cdots \circ \mathbf{f}_\theta^1(\mathbf{z}^0), \quad p_X(\mathbf{x}; \theta) = p_Z(\mathbf{z}^0) \prod_{m=1}^{M} \left| \det \frac{\partial(\mathbf{f}_\theta^m)^{-1}}{\partial Z^m} \right|_{Z^m = \mathbf{z}^m}$$

▶ Planar Flows

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}g(\mathbf{w}^\top \mathbf{z} + b)$$

Rezendre and Mohamed, 2016

Rezendre and Mohamed, 2016, Grover and Ermon DGM Tutorial
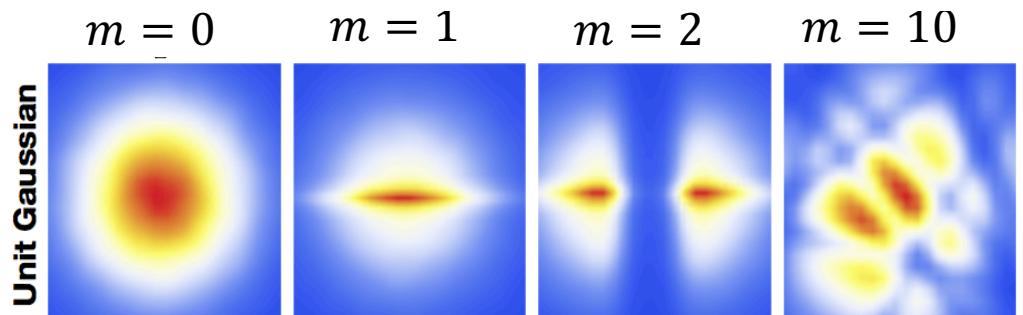
Normalizing Flows

▸ Maximum log-likelihood objective

$$\max_\theta \log p_X(\mathcal{D}; \theta) = \sum_{\mathbf{x} \in \mathcal{D}} \left(\log p_Z(\mathbf{z}) + \log \left| \det \frac{\partial (\mathbf{f}_\theta)^{-1}}{\partial X} \right|_{X=\mathbf{x}} \right)$$

▸ Exact log-likelihood evaluation via inverse transformations

▸ Sampling from the model

$$\mathbf{z} \sim p_Z(\mathbf{z}), \quad \mathbf{x} = \mathbf{f}_\theta(\mathbf{z})$$

▸ Inference over the latent representations:

$$\mathbf{z} = \mathbf{f}_\theta^{-1}(\mathbf{x})$$

Rezendre and Mohamed, 2016, Grover and Ermon DGM Tutorial

Example: GLOW

► Generative Flow with Invertible 1x1 Convolutions

  https://blog.openai.com/glow/

Latent factors of variation

Kingma, Dhariwal, 2018
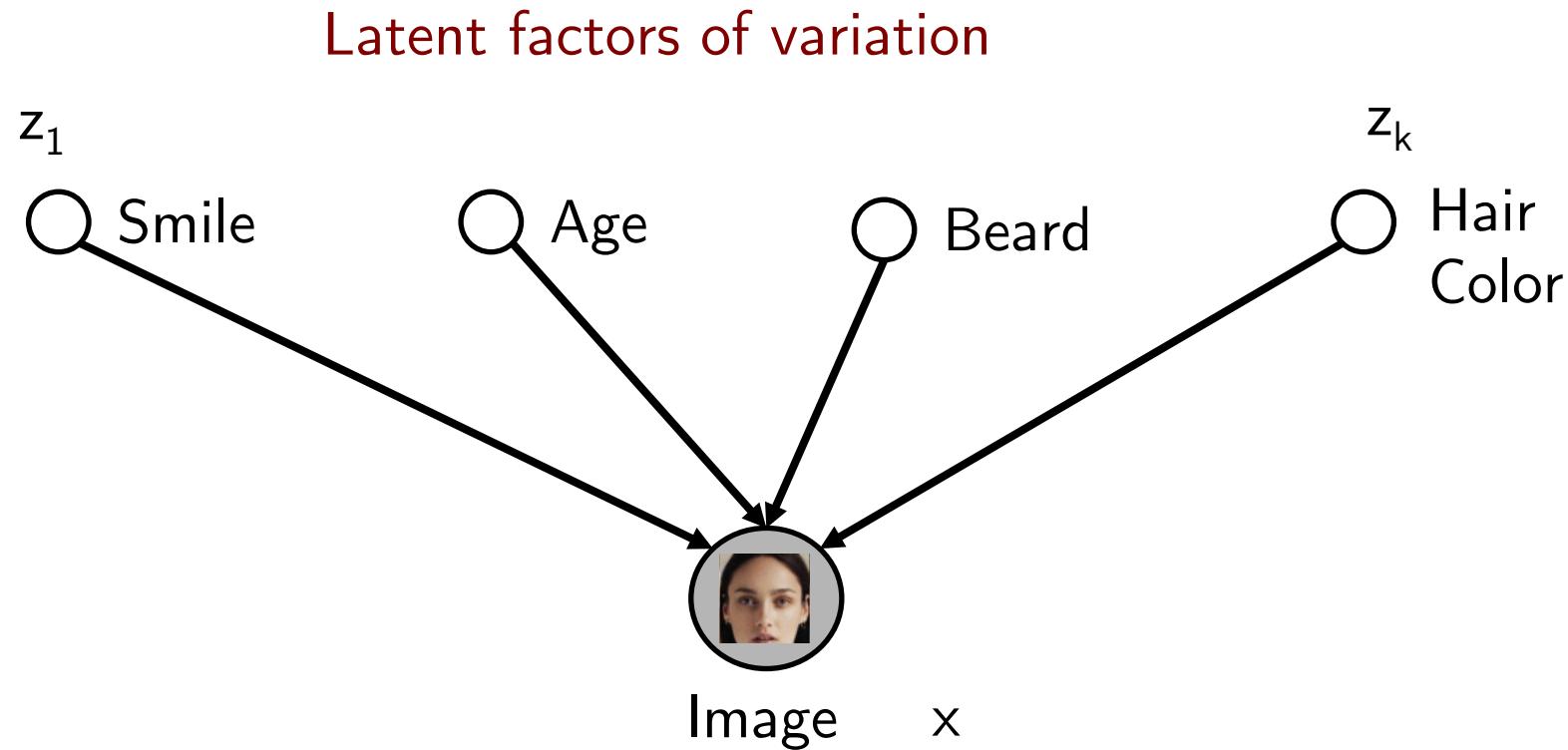
Example: GLOW

Input   Smile   Add Beard   Increase Age

https://blog.openai.com/glow/

Flow Models

▶ Simple prior that allows for sampling and tractable likelihood evaluation e.g., isotropic Gaussian

▶ Invertible transformations with tractable evaluation:

   ▶ Likelihood evaluation requires efficient evaluation of inverse

   ▶ Sampling requires efficient evaluation of inverse

▶ Tractable evaluation of determinants of Jacobian for large

   ▶ Computing determinants for a large matrix is prohibitive

   ▶ Key idea: Determinant of triangular matrices is the product of the diagonal entries, i.e., an operation

Grover and Ermon, DGM Tutorial

Thank you