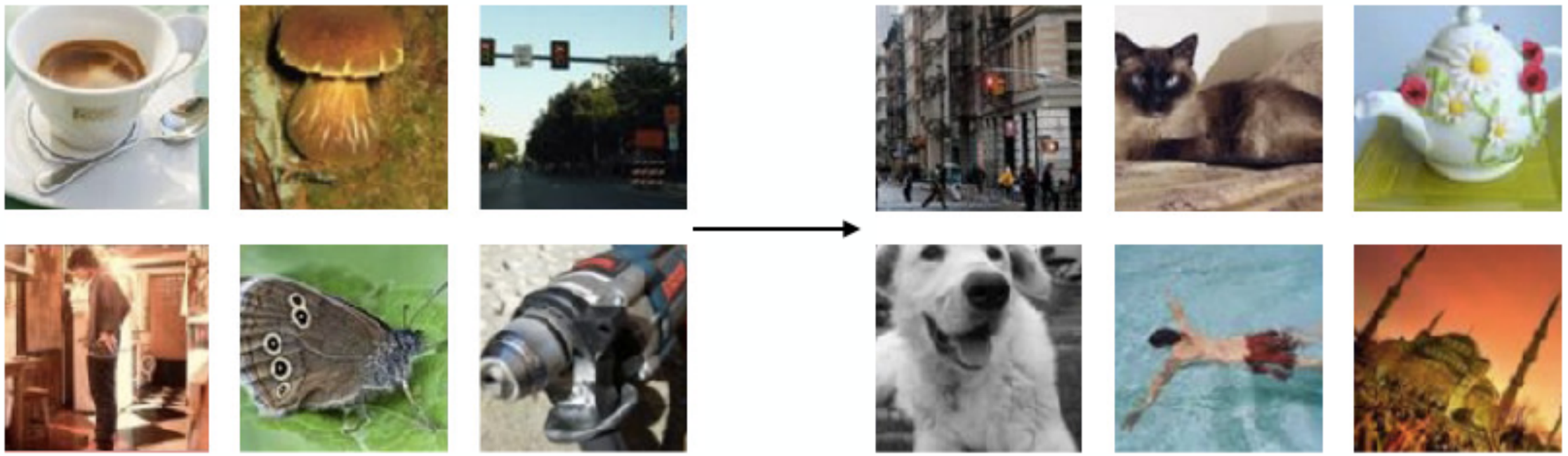


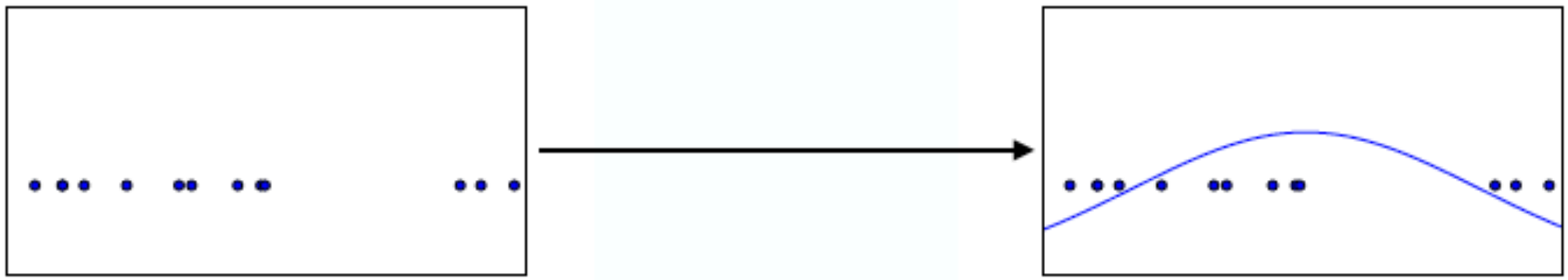
Generative Modeling

Zhanxing Zhu
Peking University

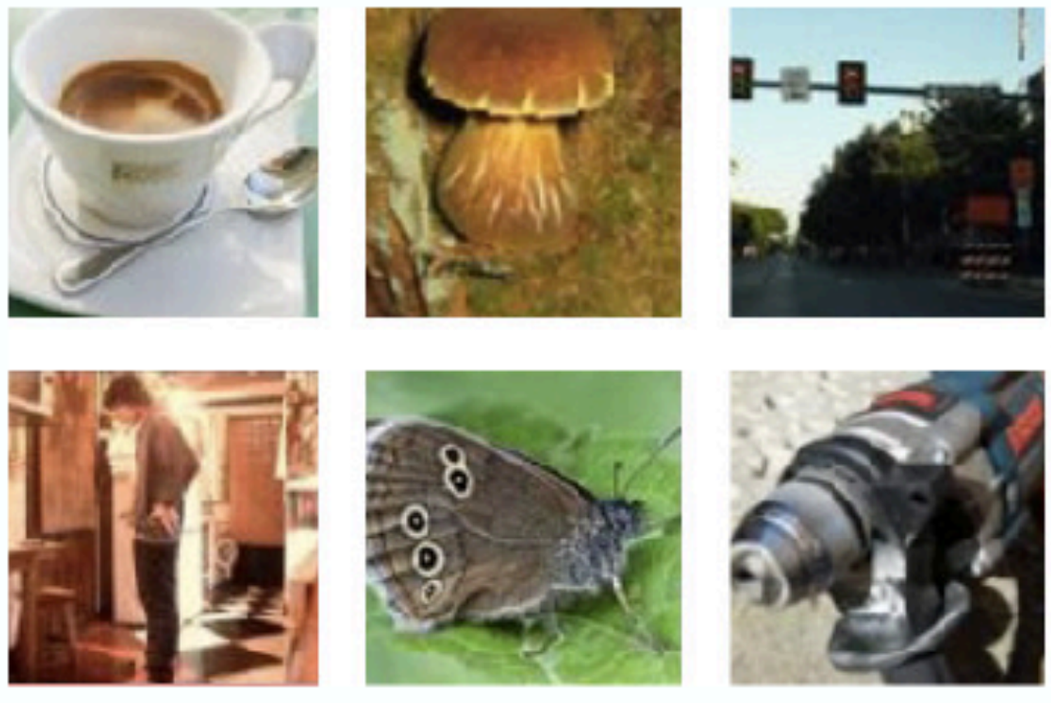
Some of contents are based on: https://arxiv.org/abs/1701.00160

Generative Modeling

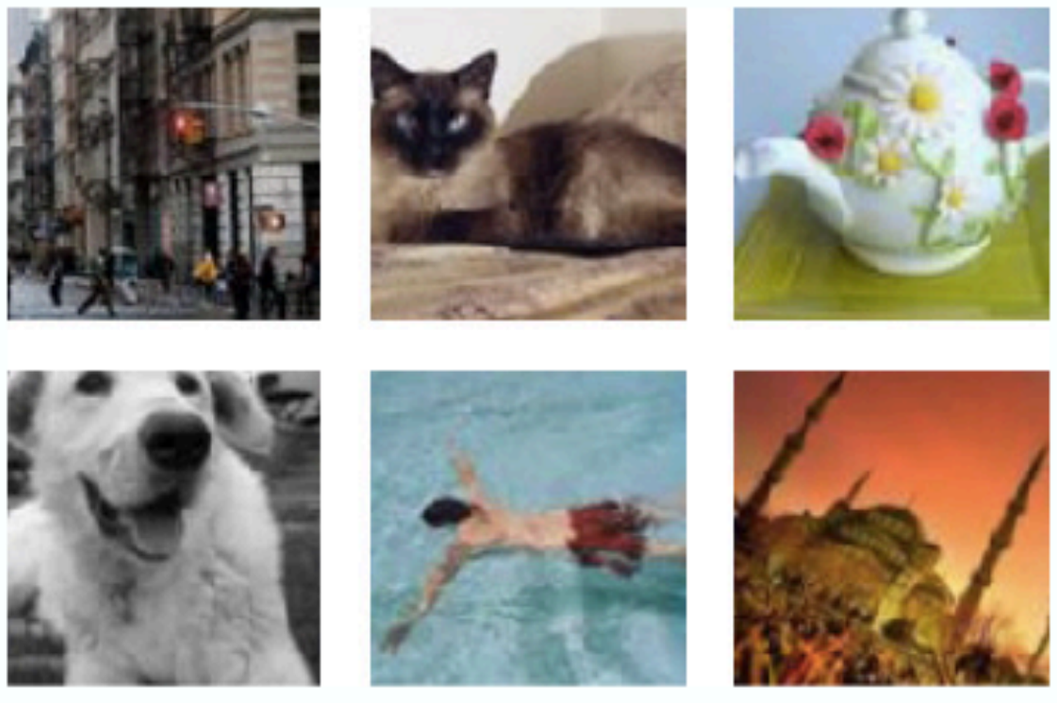
- Density estimation



- Sample generation



Training examples



Model samples

Generative Models v.s. Discriminative Models

Mixture of Gaussians
Hidden Markov Models
Topics Models
Markov Random Fields
(Restricted Boltzmann Machine)
NADE
VAE
GAN
et.al.

SVM
AdaBoost
Random Forest
Conditional Random Fields
MLP
CNN
et.al.

Contents

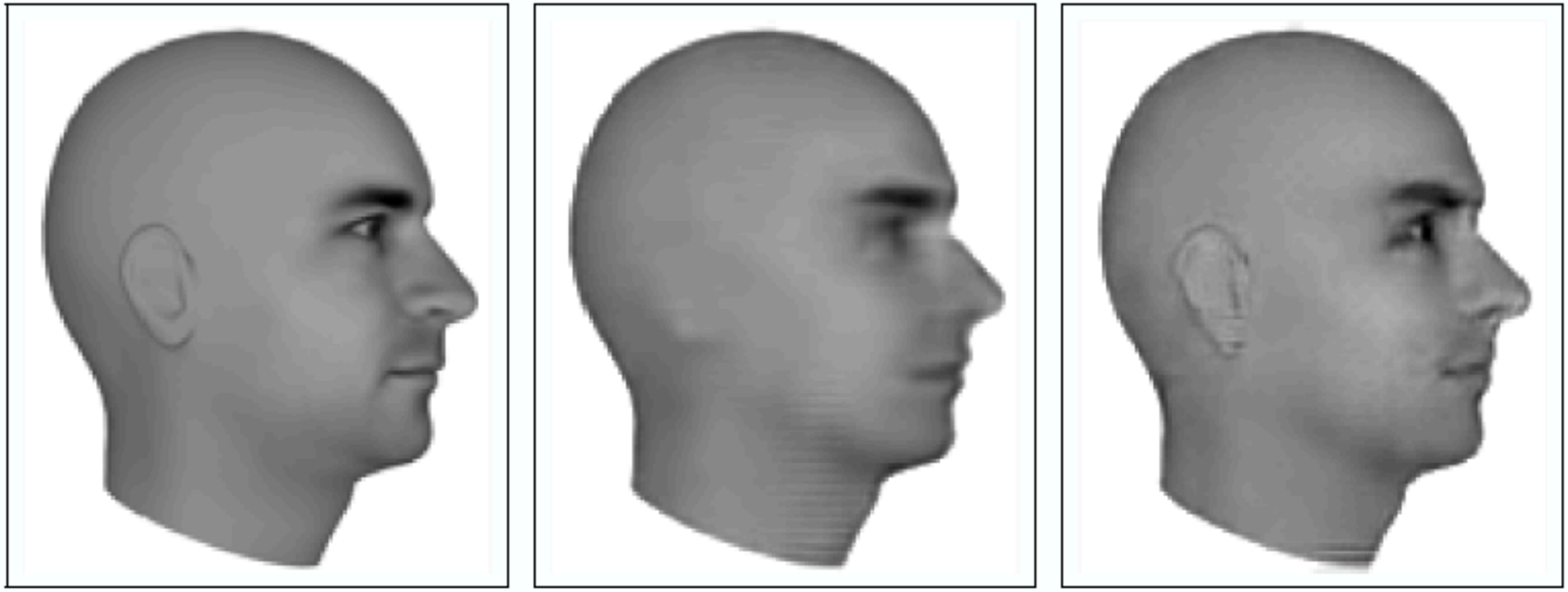
- Why do we study generative models?
- Deep Generative Models
- Explicit and Implicit Models
- Some popular generative models
- Two important types
  - VAE, GAN
- Research Frontiers

Motivation

- Excellent test of our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

Next Video Frame Prediction

Ground Truth | MSE | Adversarial



(Lotter et al 2016)

Single Image Super-Resolution

original

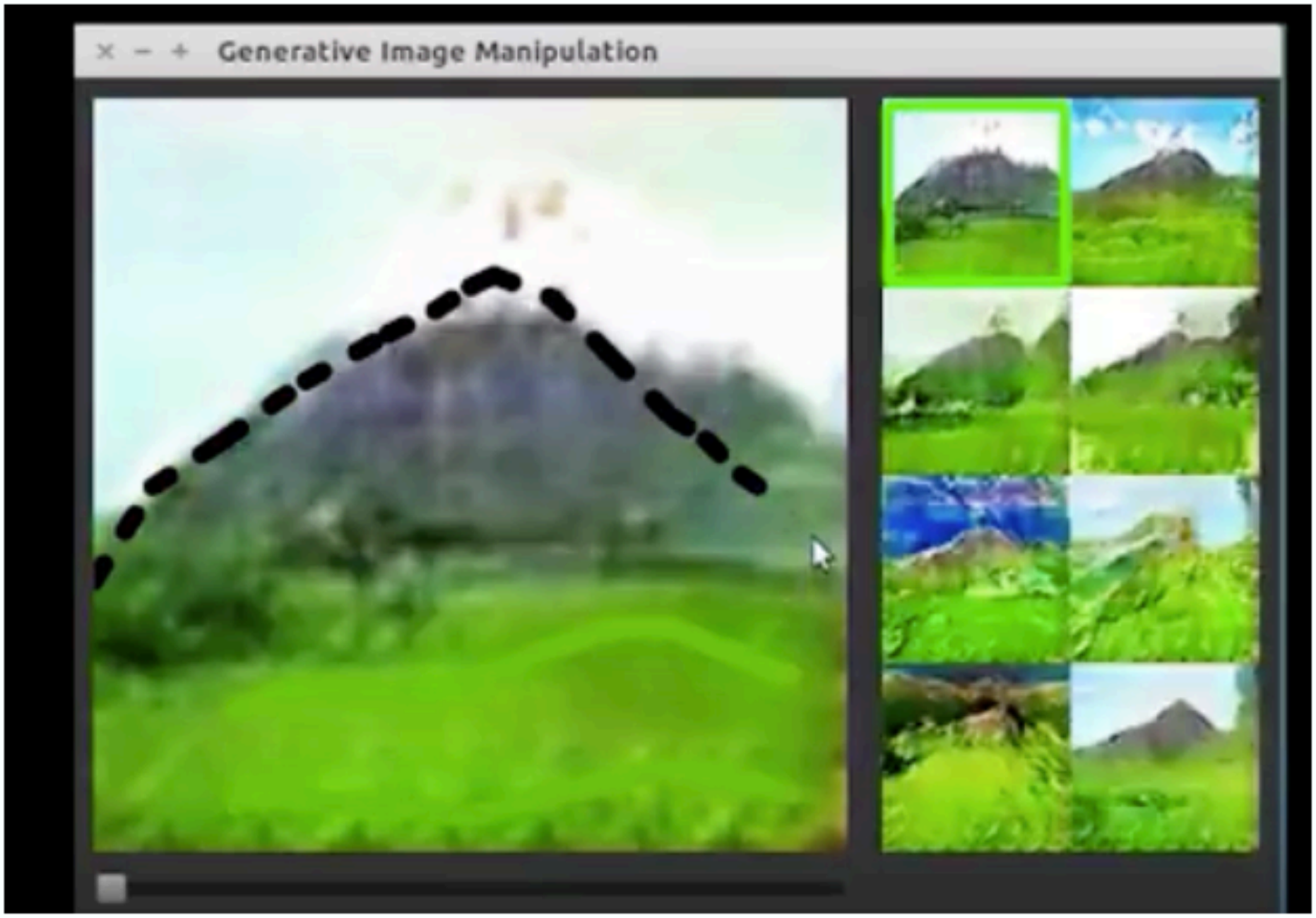
bicubic
(21.59dB/0.6423)

SRResNet
(23.44dB/0.7777)

SRGAN
(20.34dB/0.6562)

(Ledig et al 2016)

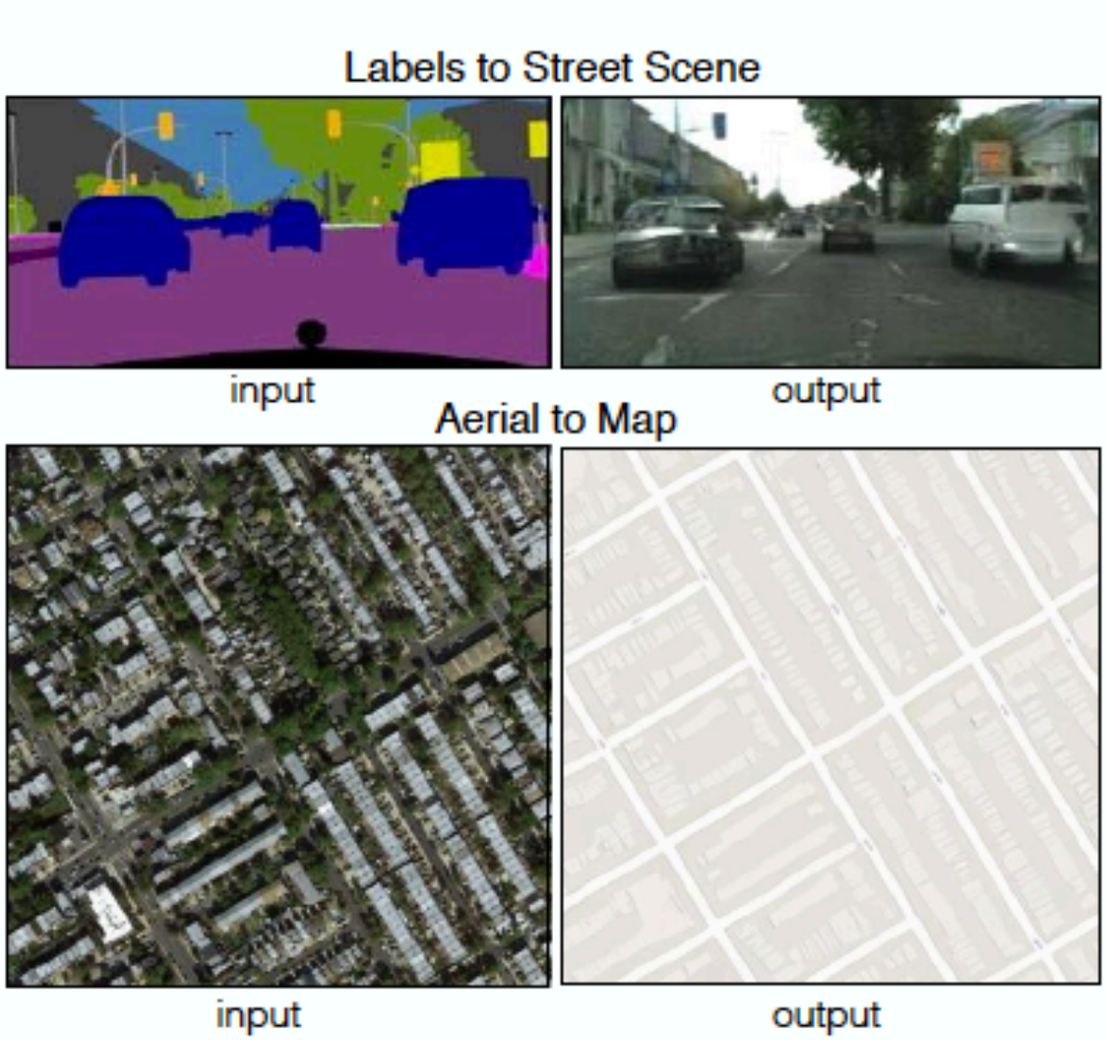
iGAN



youtube

(Zhu et al 2016)

Image to Image Translation



(Isola et al 2016)

Taxonomy of (deep) generative models

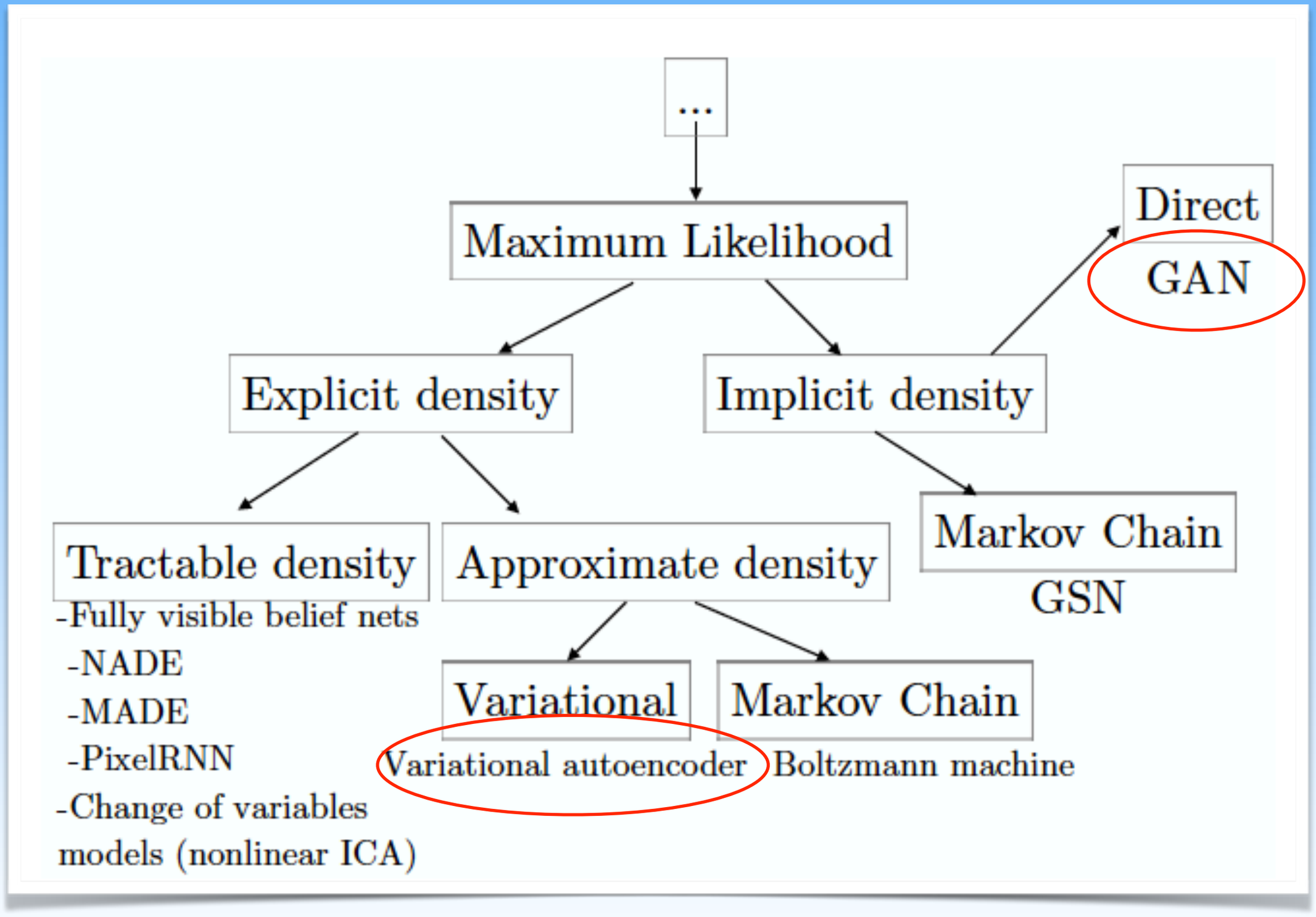


Figure from Goodfellow's NIPS 2016 Tutorial

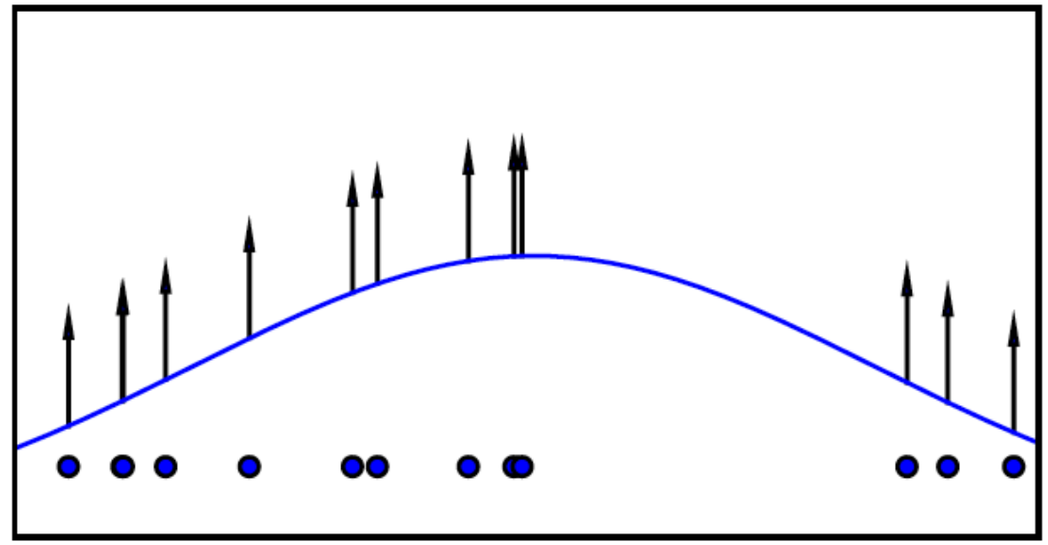
Maximum Likelihood Estimation

- Data distribution & model distribution

$$p_{\text{data}}(x) \quad p_{\text{model}}(x;\theta)$$

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{m} p_{\text{model}}\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right) \\ &= \arg\max_{\boldsymbol{\theta}} \log \prod_{i=1}^{m} p_{\text{model}}\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right) \\ &= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log p_{\text{model}}\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)\end{aligned}$$

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} D_{\text{KL}}\left(p_{\text{data}}(\boldsymbol{x}) \| p_{\text{model}}(\boldsymbol{x};\boldsymbol{\theta})\right)$$



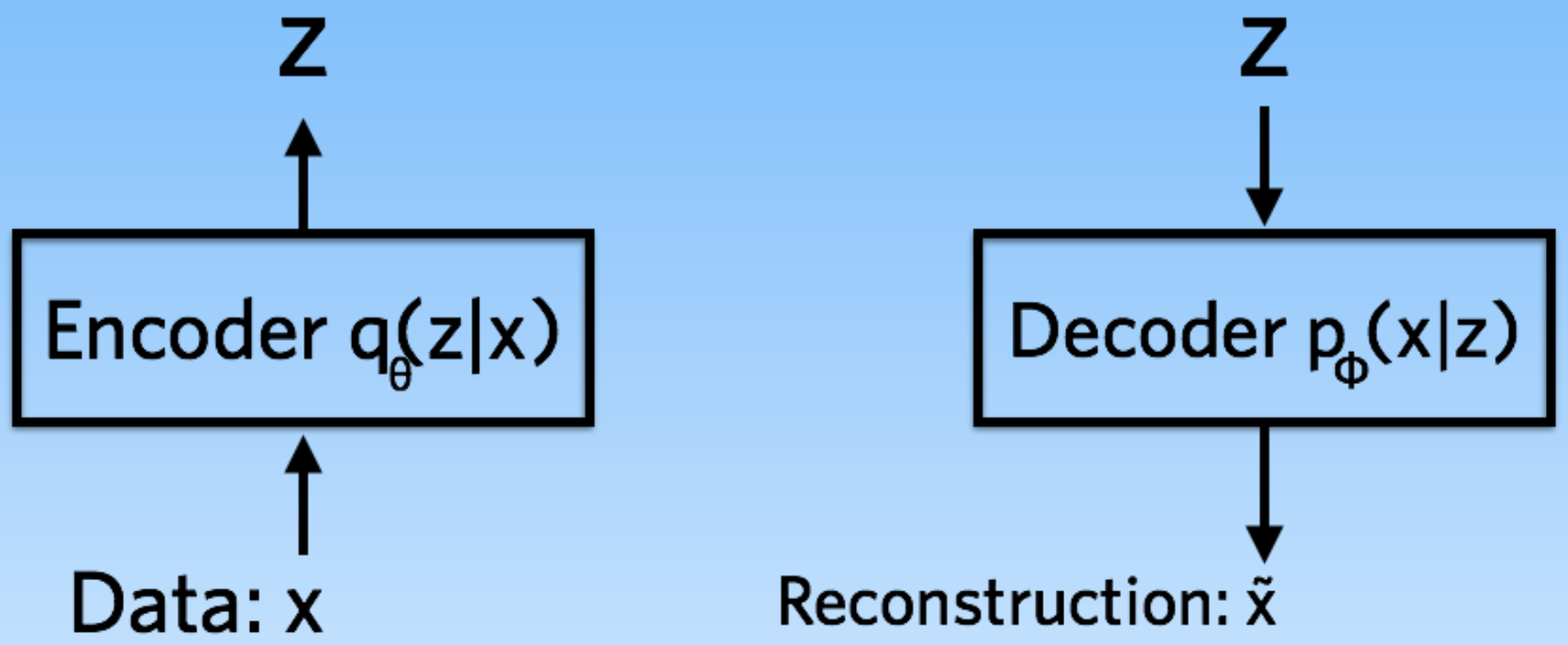
$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{x\sim p_{\text{data}}} \log p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{\theta})$$

Explicitly or implicitly

Variational AutoEncoder

(Kingma & Welling 2014)

- Neural network perspective

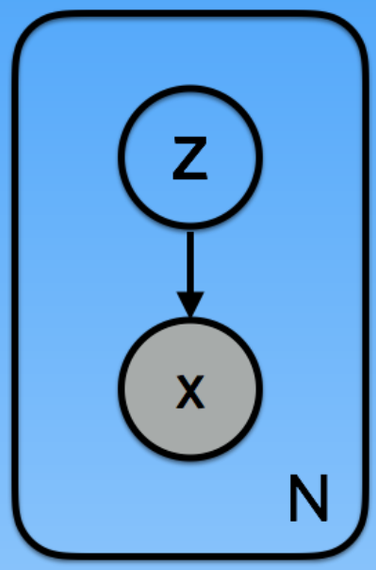


$$l_i(\theta, \phi) = -E_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i|z)] + KL(q_\theta(z|x_i)||p(z))$$

reconstruction error

regularizer

- Graphical model perspective



- Draw latent variables $z_i \sim p(z)$
- Draw datapoint $x_i \sim p(x|z)$

evidence lower bound

$$\log p(\boldsymbol{x}_i) \geq ELBO_i(\lambda) = E_{q_\lambda(z|x_i)}[\log p(x_i|z)] - KL(q_\lambda(z|x_i)||p(z))$$

parameterizing by NN

$$ELBO_i(\theta, \phi) = E_{q_\theta(z|x_i)}[\log p_\phi(x_i|z)] - KL(q_\theta(z|x_i)||p(z))$$

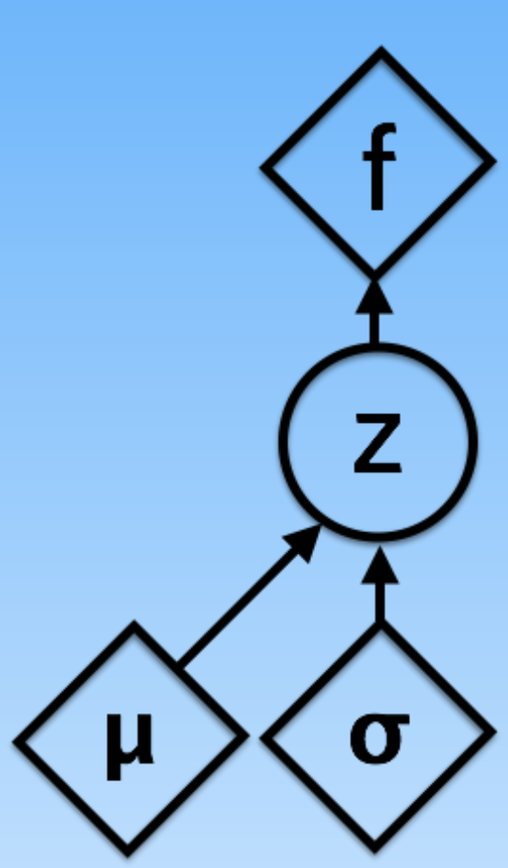
$$\log q_\theta(\boldsymbol{z}|\boldsymbol{x}_i) = \log N(\boldsymbol{z}; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2 \boldsymbol{I})$$

Output of NN

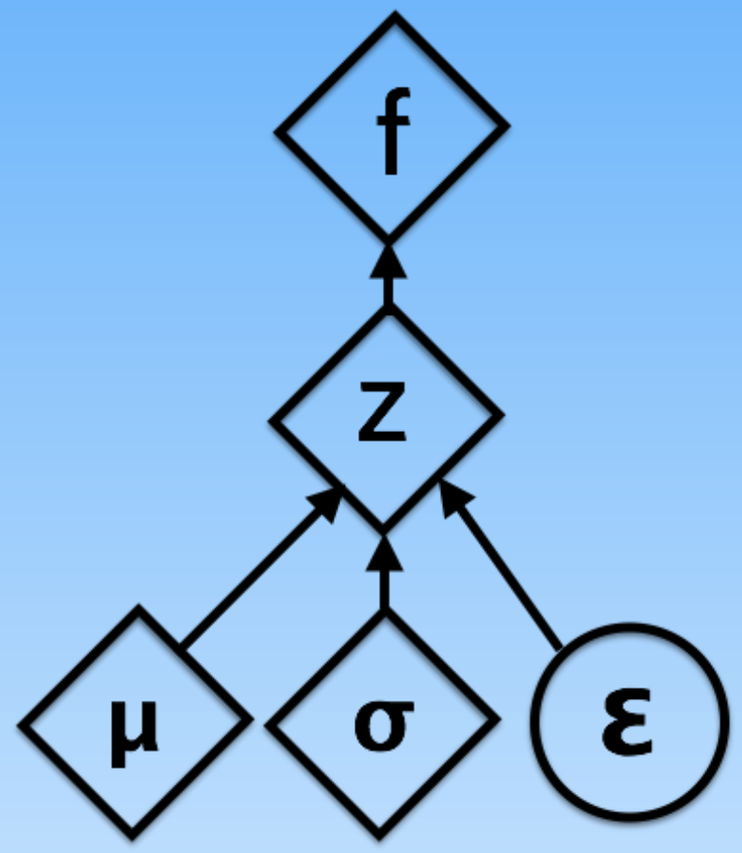
$$\mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)} \qquad \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$$

reparameterization to be differentiable

Reparameterization



Original

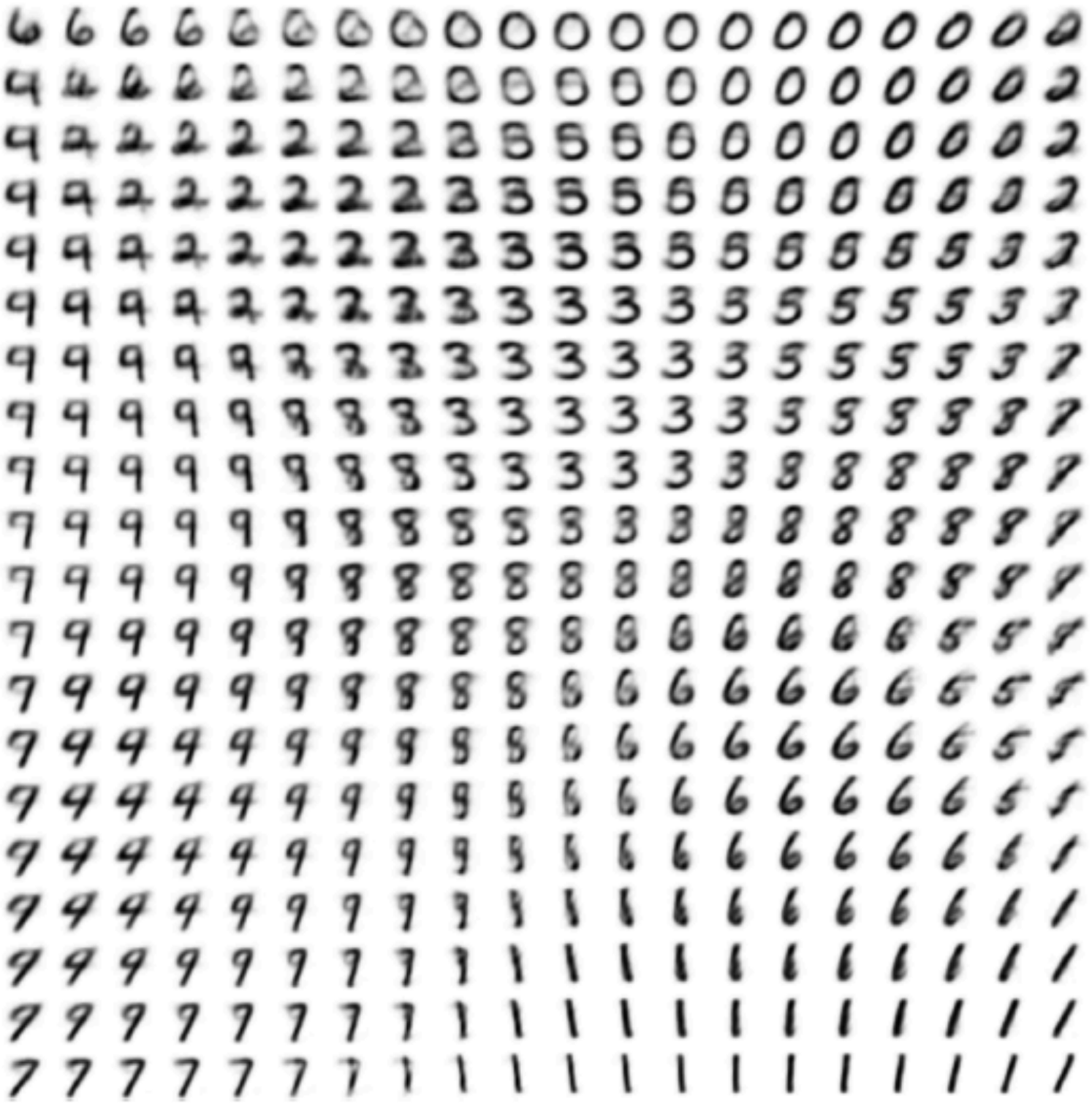


Reparametrized

$$\mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)}$$

$$\boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$$

(a) Learned Frey Face manifold



(b) Learned MNIST manifold

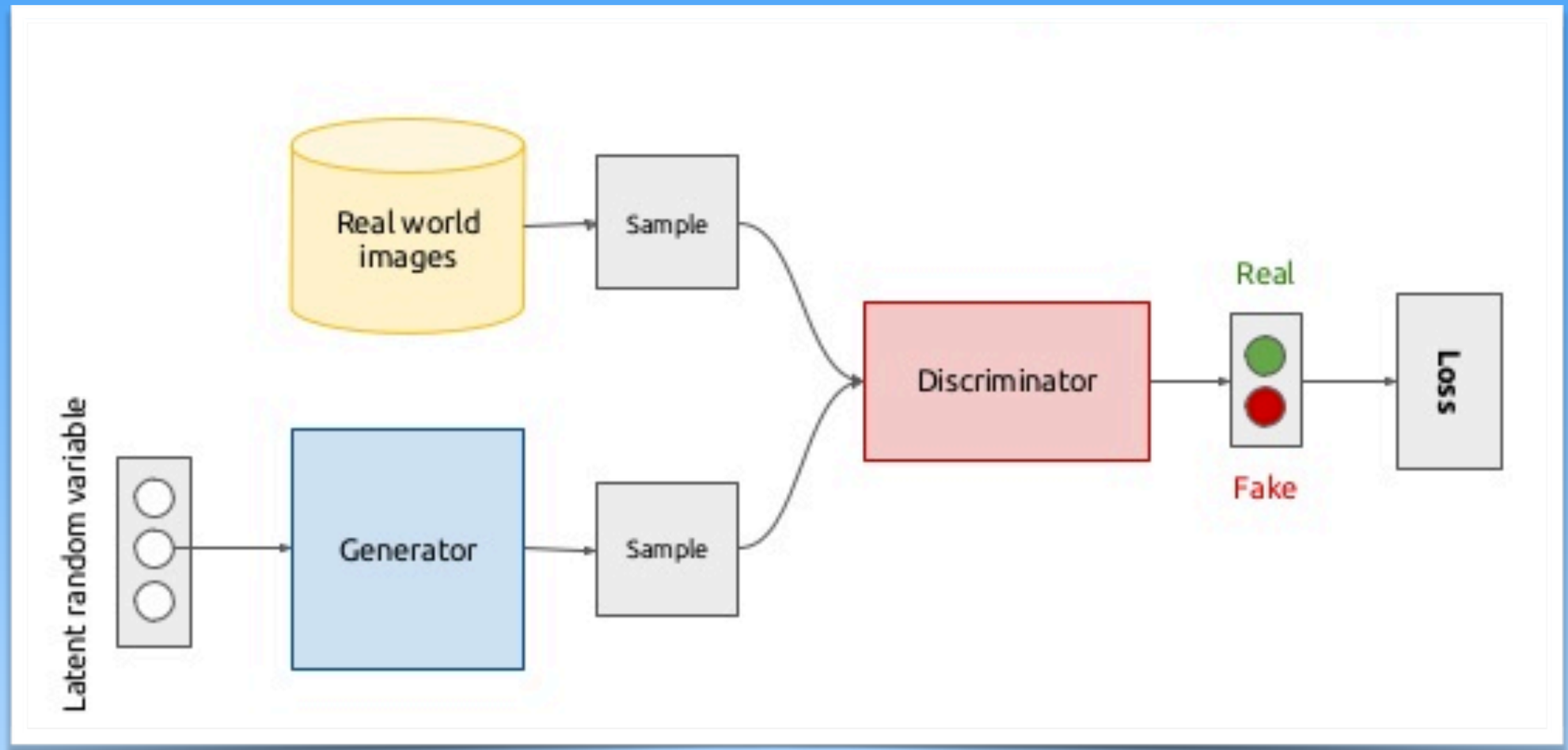
Generative Adversarial Nets

(Goodfellow et.al. 2014)

- Advantages compared to other generative models
  - Generate samples in parallel v.s. NADE, WaveNet et.al
  - Few restrictions on designing the generating function
    - v.s. RBM, has to be easy for MCMC
  - No variational bound needed, v.s. VAE
  - Produce better samples than other methods
- Disadvantages….

GAN

- A game between two players
  - Generator net
    - try to generate samples similar with real data
  - Discriminator net, i.e. a classifier
    - try to distinguish whether the generated samples are real or fake
  - Min-max game: Nash equilibrium



Min-max optimization

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Optimal solutions for D $D_G^*(\boldsymbol{x}) = \dfrac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$

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:

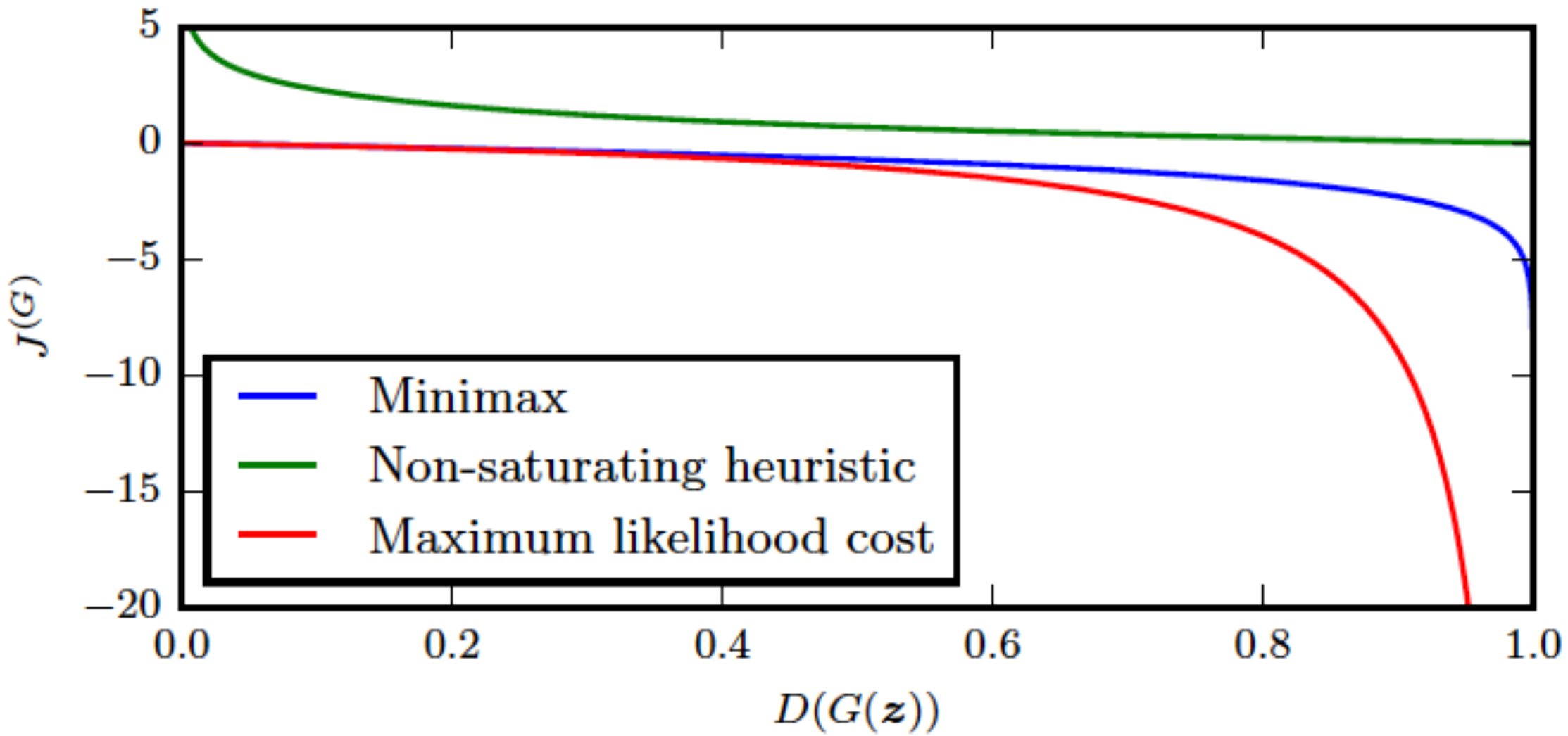
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

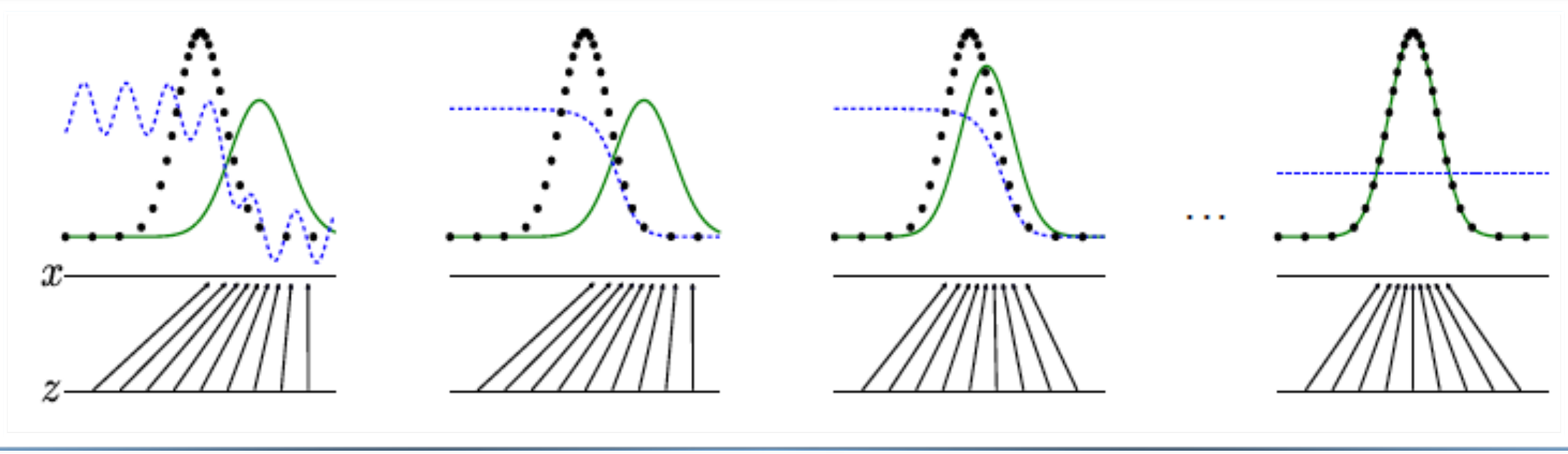
end for

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

From Goodfellow et.al. 2014

Saturating issue and instability: in early stage, G is weak.
log D(G(z)) trick!


5
0
−5
−10
−15
−20
$J(G)$
Minimax
Non-saturating heuristic
Maximum likelihood cost
0.0
0.2
0.4
0.6
0.8
1.0
$D(G(\boldsymbol{z}))$



$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

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

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

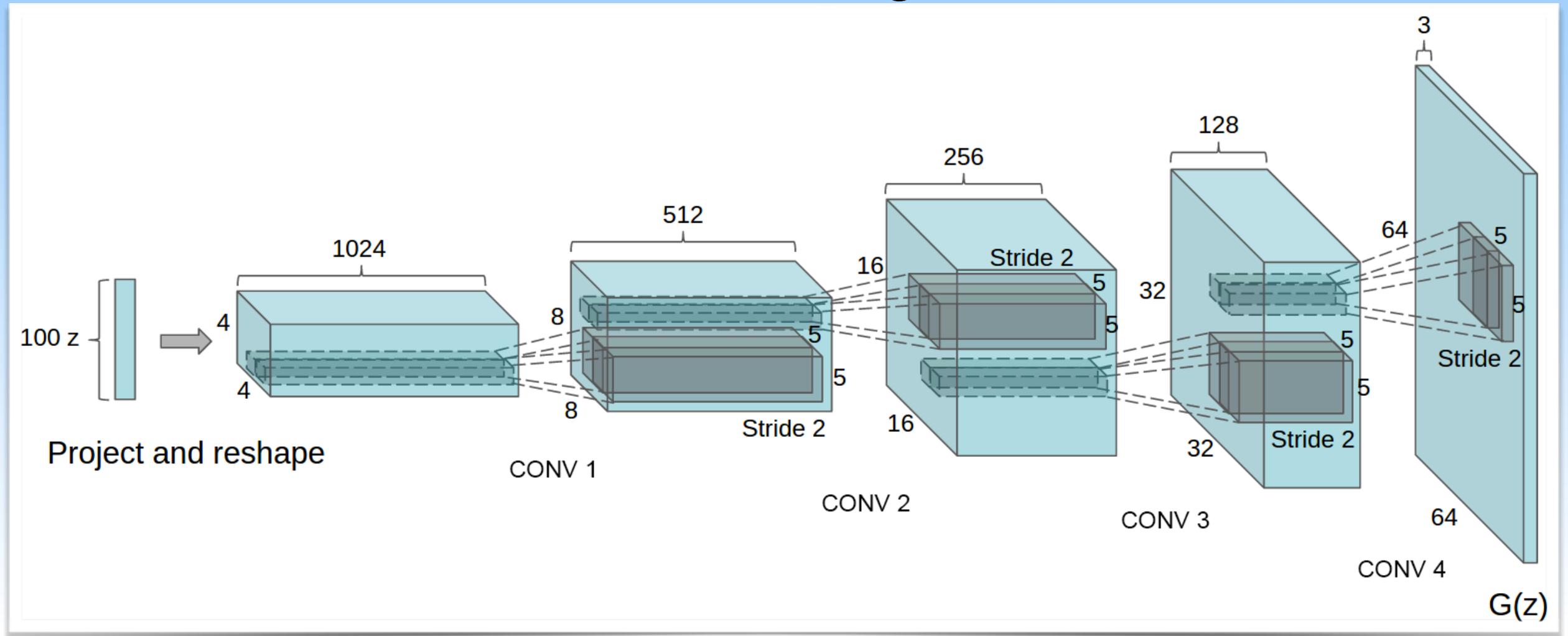
Jensen-Shannon divergence

Some issues of GAN

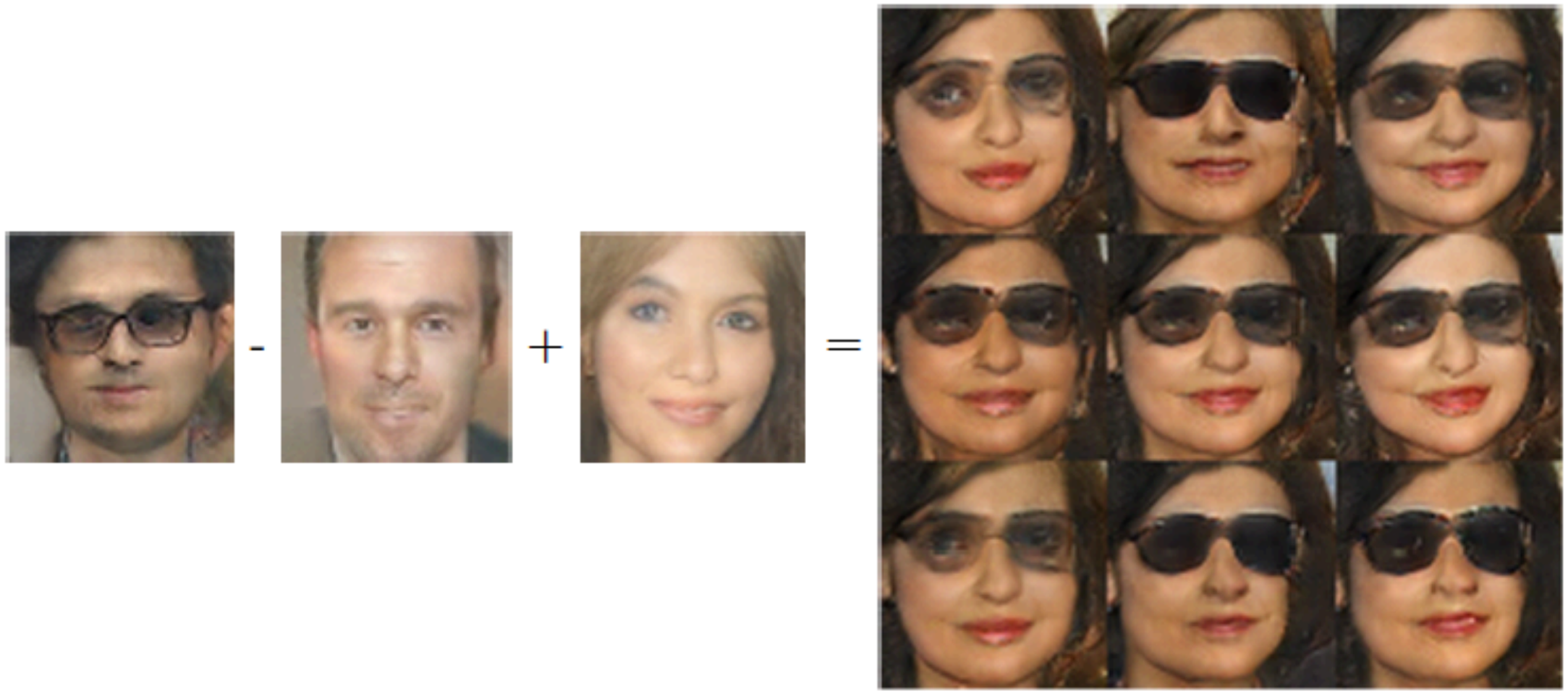
- Training instability (need balance G and D)
- Mode collapse
  - only generate a small number of modes
- Why does GAN can generate sharp samples, compared with other methods? Not clear.

DCGAN (Radford et.al. 2015)

- Most GANs loosely rely on the architecture of DCGAN.
  - Batch Normalization used
  - All-convolution, increase spatial size by "deconvolution" with stride larger than 1



DCGAN on LSUN dataset



Arithmetic operation in latent space

Wasserstein GAN

(Arjovsky et.al. 2017)

- A remarkable milestone for GAN
- Analyze different distance measures over distributions
  - largely solve the instability issue
  - more meaningful loss metric
- Wasserstein distance / Earth-mover distance
  - Optimal transport

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\, \|x - y\| \,\right]$$

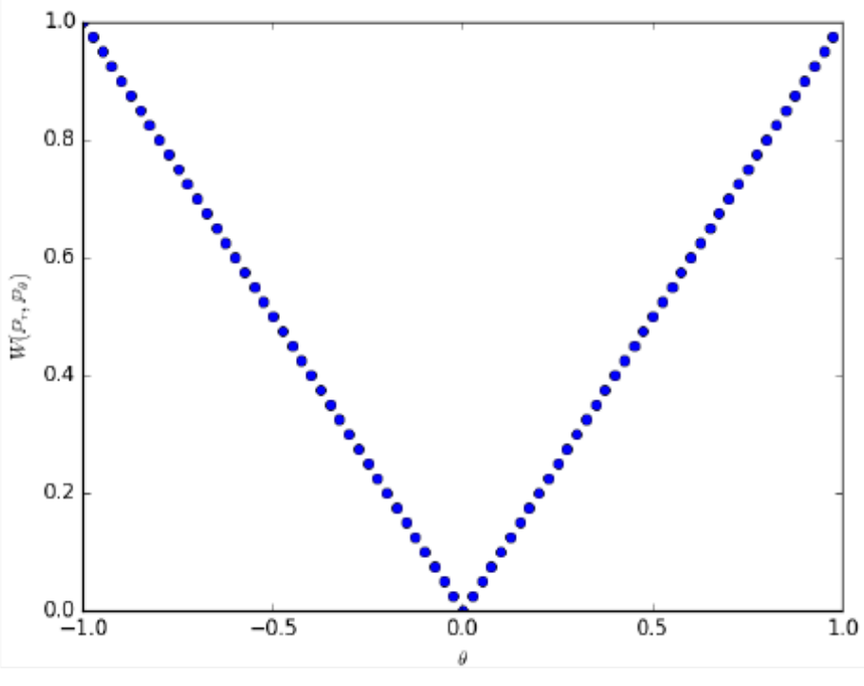
joint distribution with marginal distribution preserved

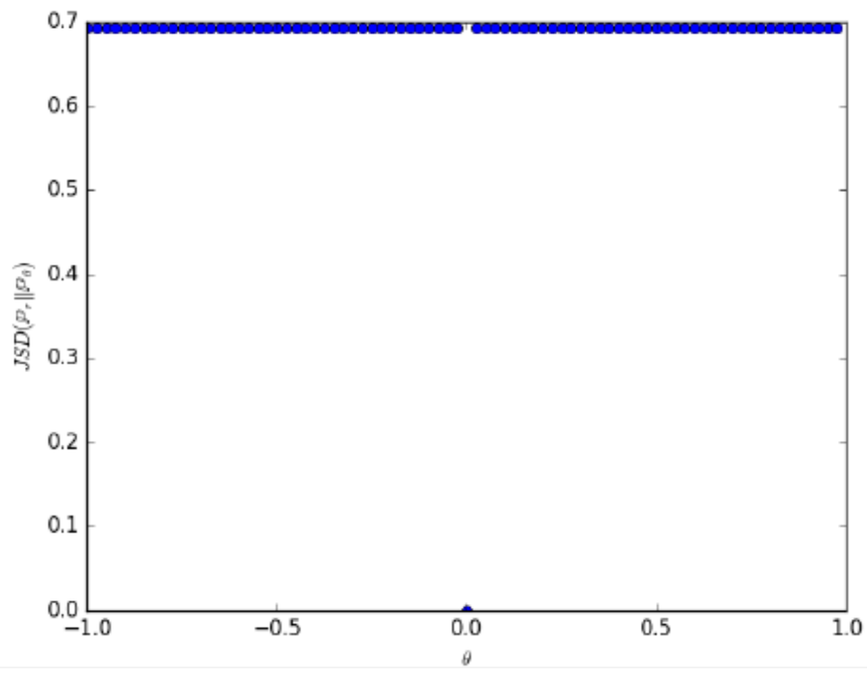
Learning parallel line

P0: uniform over (0,Z)

Ptheta: g(z) = (theta, z)

- $W(\mathbb{P}_0, \mathbb{P}_\theta) = |\theta|,$
- $JS(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} \log 2 & \text{if } \theta \neq 0\ , \\ 0 & \text{if } \theta = 0\ , \end{cases}$
- $KL(\mathbb{P}_\theta \| \mathbb{P}_0) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta) = \begin{cases} +\infty & \text{if } \theta \neq 0\ , \\ 0 & \text{if } \theta = 0\ , \end{cases}$





- Minimizing (after Kantorovich-Rubinstein dual transform)

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)]$$

Lipschitz constant

$$\max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim p(z)}[f_w(g_\theta(z)]$$

f, also called test function in functional analysis

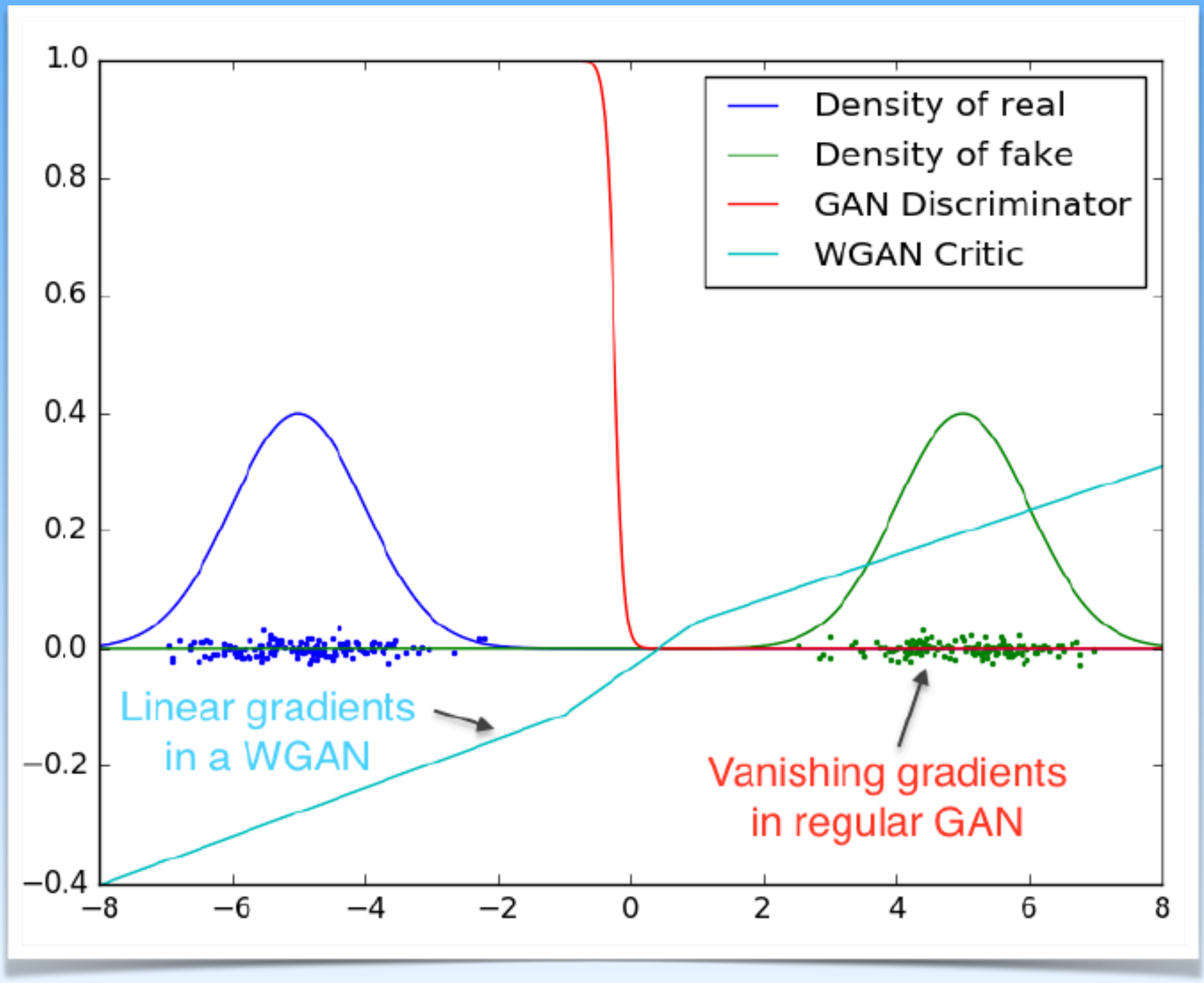
- Control Lipschitz constant by clipping weights
- RMSProp optimization
- f, “critic”
  - output arbitrary real-value, not a probability (GAN)

Require: : α, the learning rate. c, the clipping parameter. m, the batch size.
n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

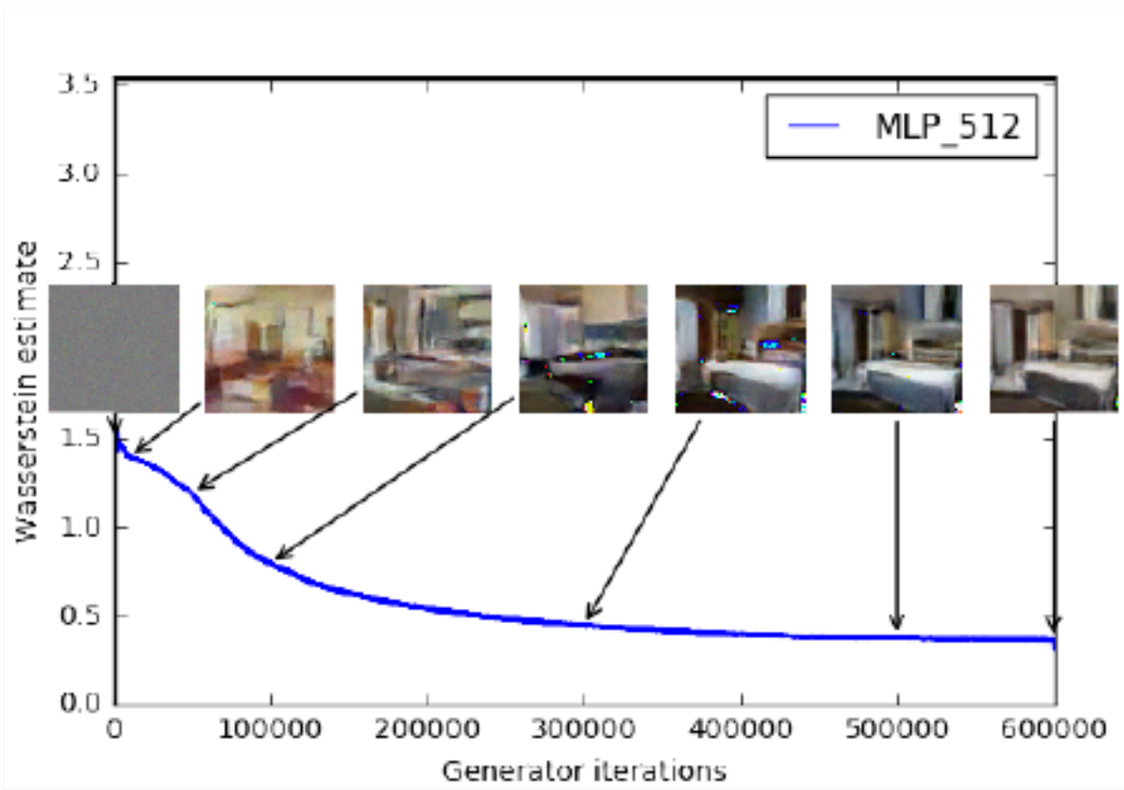
From Arjovsky et.al. 2017

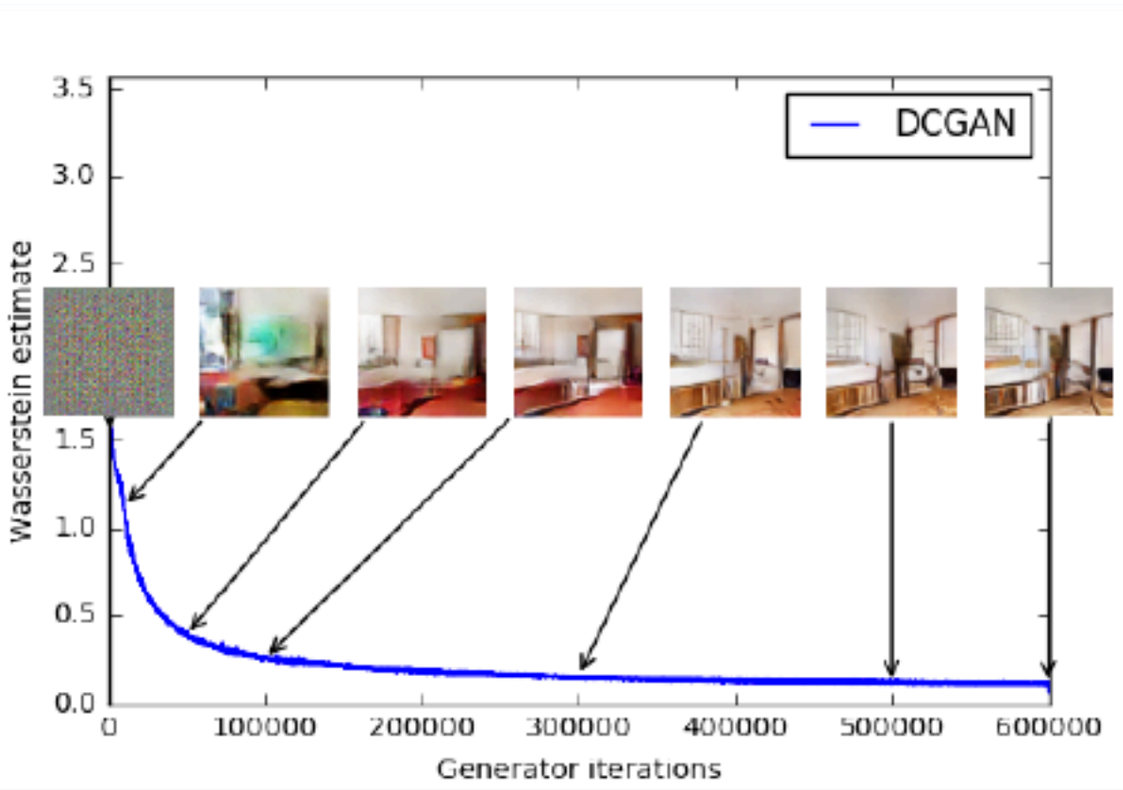
- Avoid “gradient vanishing” issue in GAN

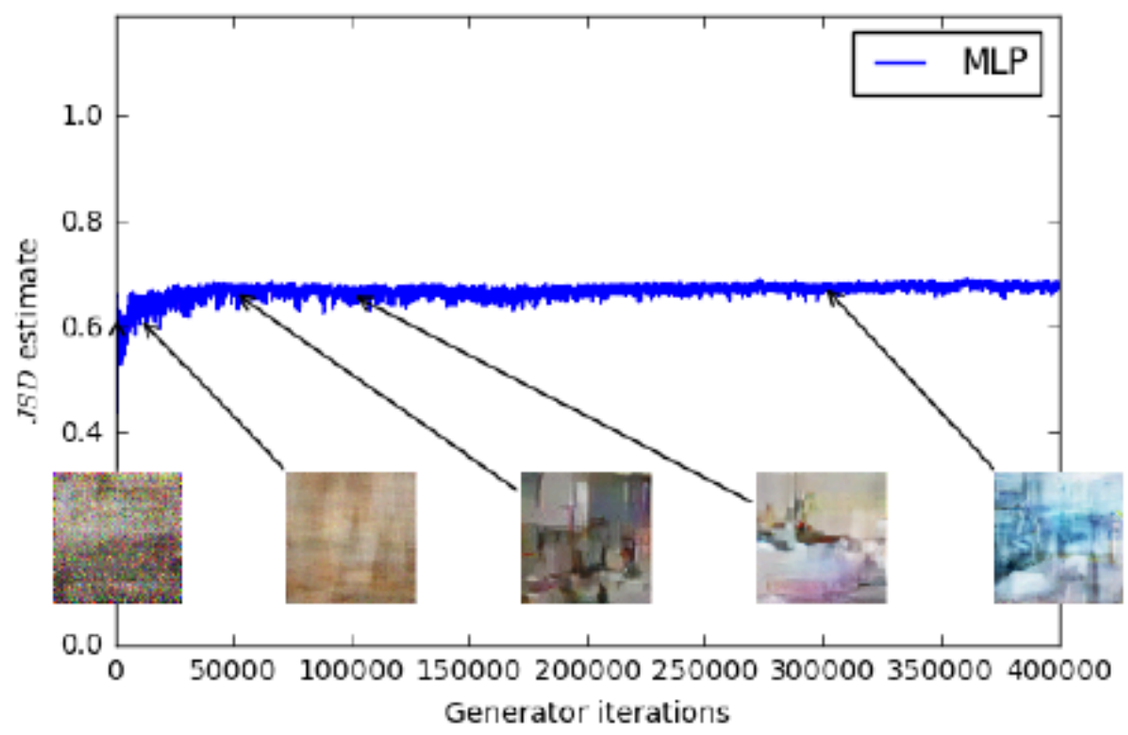


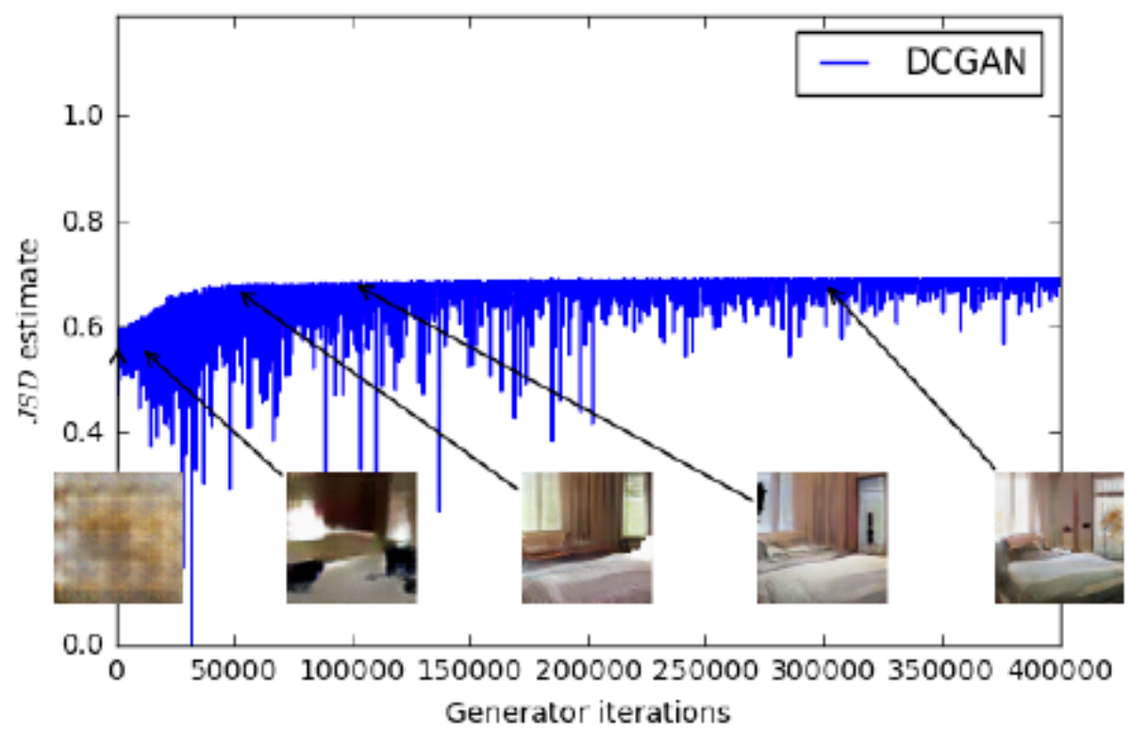
From Arjovsky et.al. 2017

- Meaningful loss metric









From Arjovsky et.al. 2017

CycleGAN (Zhu et.al 2017)

- Unpaired image-to-image translation

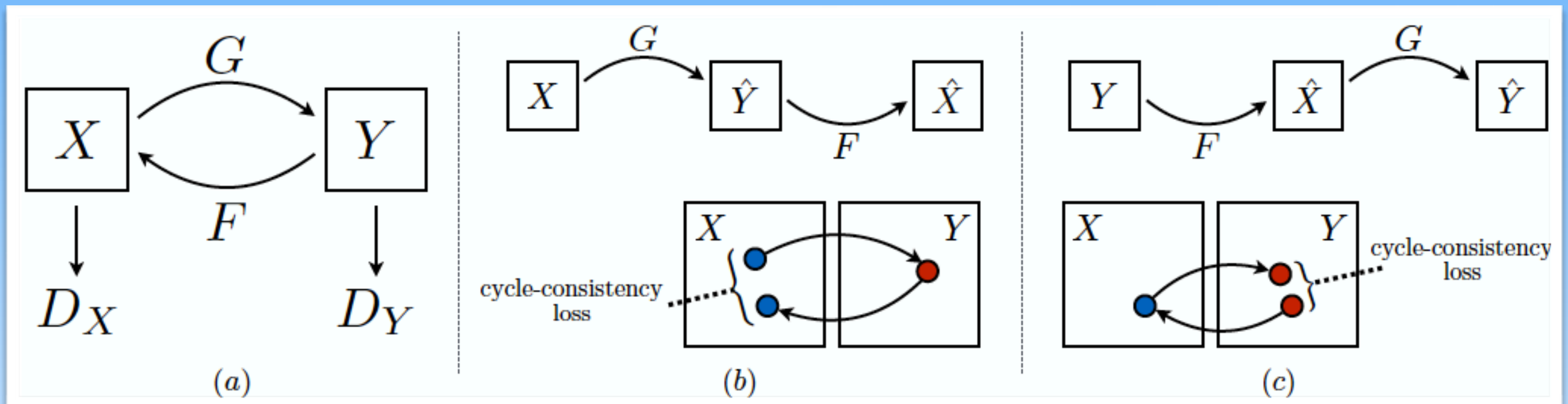


Figure from Zhu et.al (2017)

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F),$$

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))]$$

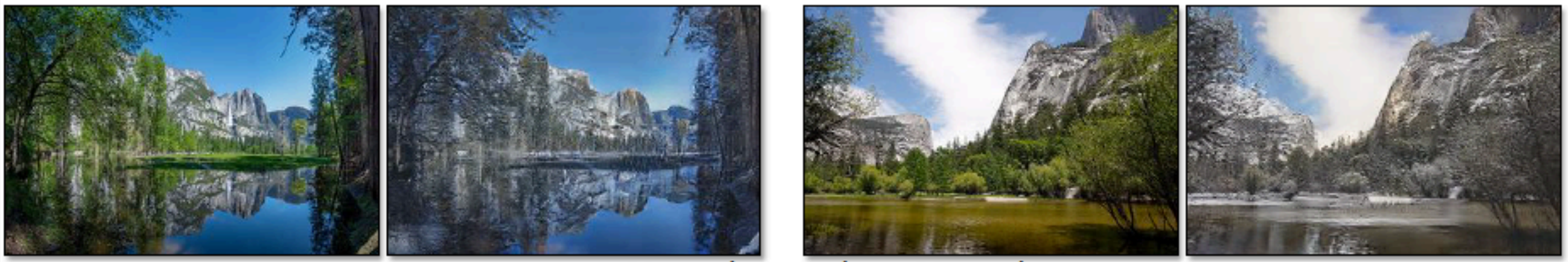
$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]$$

Input Output Input Output Input Output

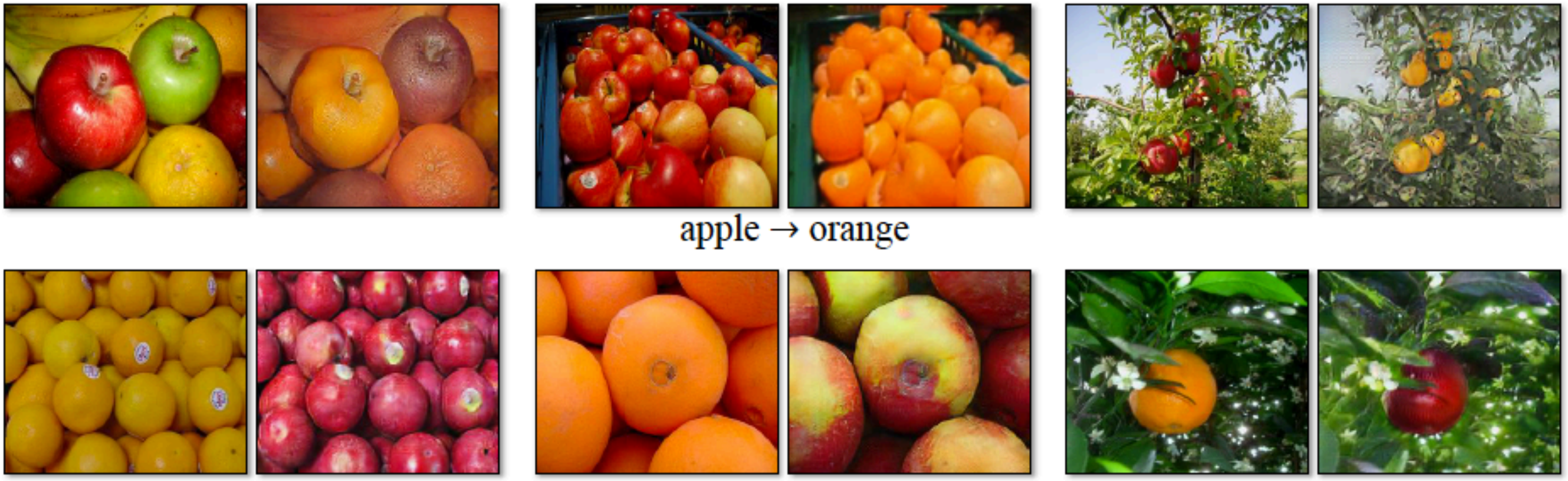
horse → zebra

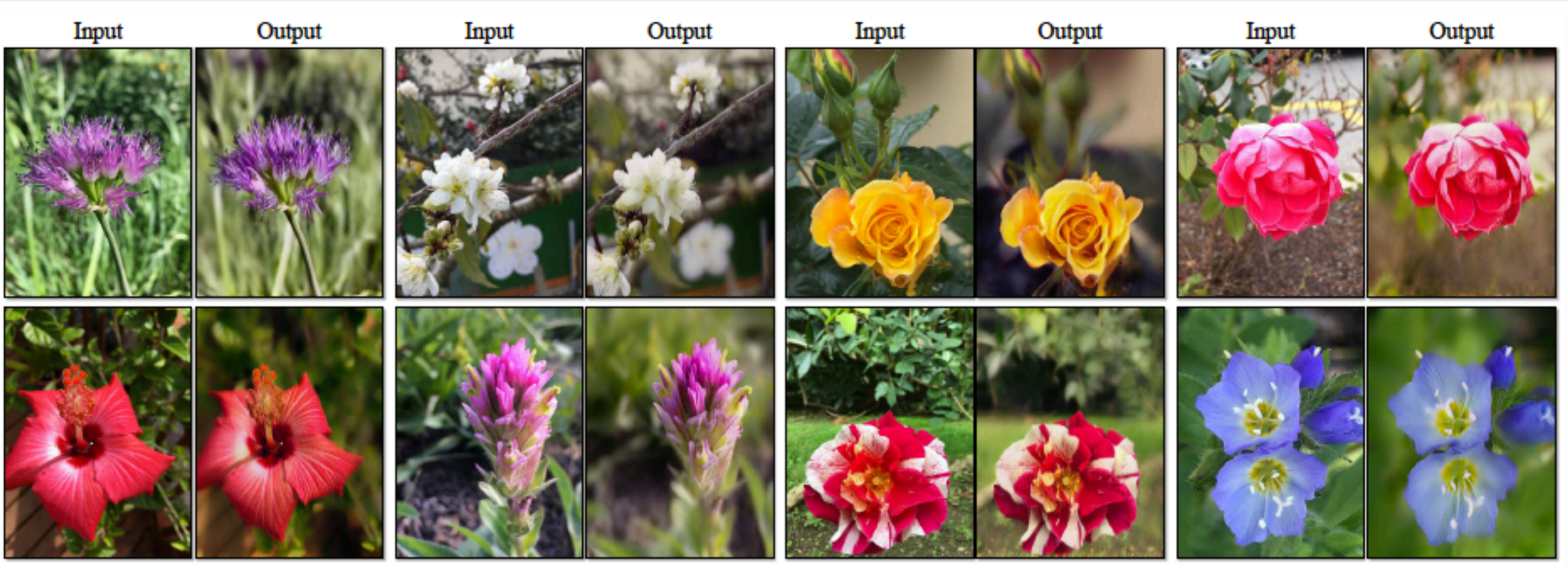
zebra → horse

winter Yosemite → summer Yosemite



summer Yosemite → winter Yosemite


apple → orange
orange → apple


Input
Output
Input
Output
Input
Output
Input
Output