Introduction to Machine Learning CS182

Lu Sun

School of Information Science and Technology

ShanghaiTech University

December 28, 2023

Today:

- Deep Generative Networks (DGN)

Readings:

- Deep Learning (DL), Chapters 14&20

Today's Agenda

深度生成网络：无监督

Deep Generative Models (DGM)

- Overview
- Representation Learning with Autoencoder 自编码器
- Generative Adversarial Network (GAN) 生成对抗网络
- Applications of GANs

Acknowledgement: Hugo Larochelle's, MehryarMohri@NYU's, YingyuLiang@Princeton's, BhikshaRaj@CMU's&FeifeiLi@Stanford'scourse notes

Deep Generative Networks-DGM

- **Overview**
- Representation Learning with Autoencoder
- Generative Adversarial Network (GAN)
- Applications of GANs

Unsupervised Learning

- Task formulation

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Unsupervised Learning

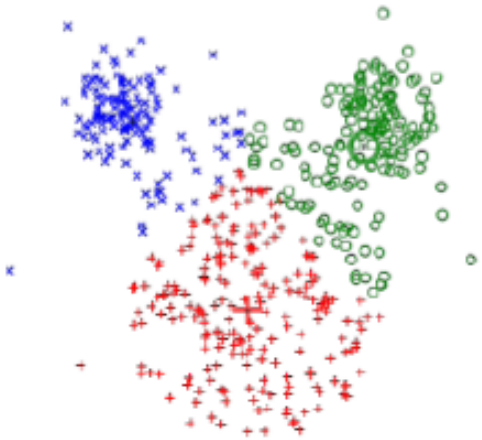
- Task formulation

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.



K-means clustering

Unsupervised Learning

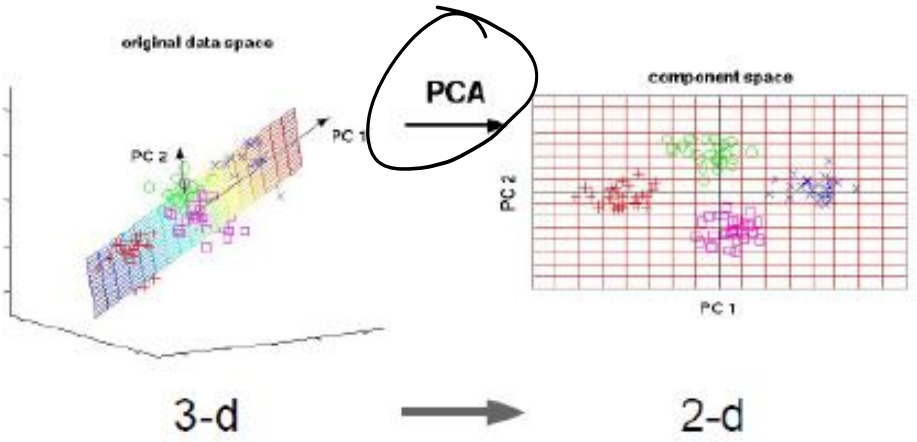
- Task formulation

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.



Principal Component Analysis
(Dimensionality reduction)

Unsupervised Learning

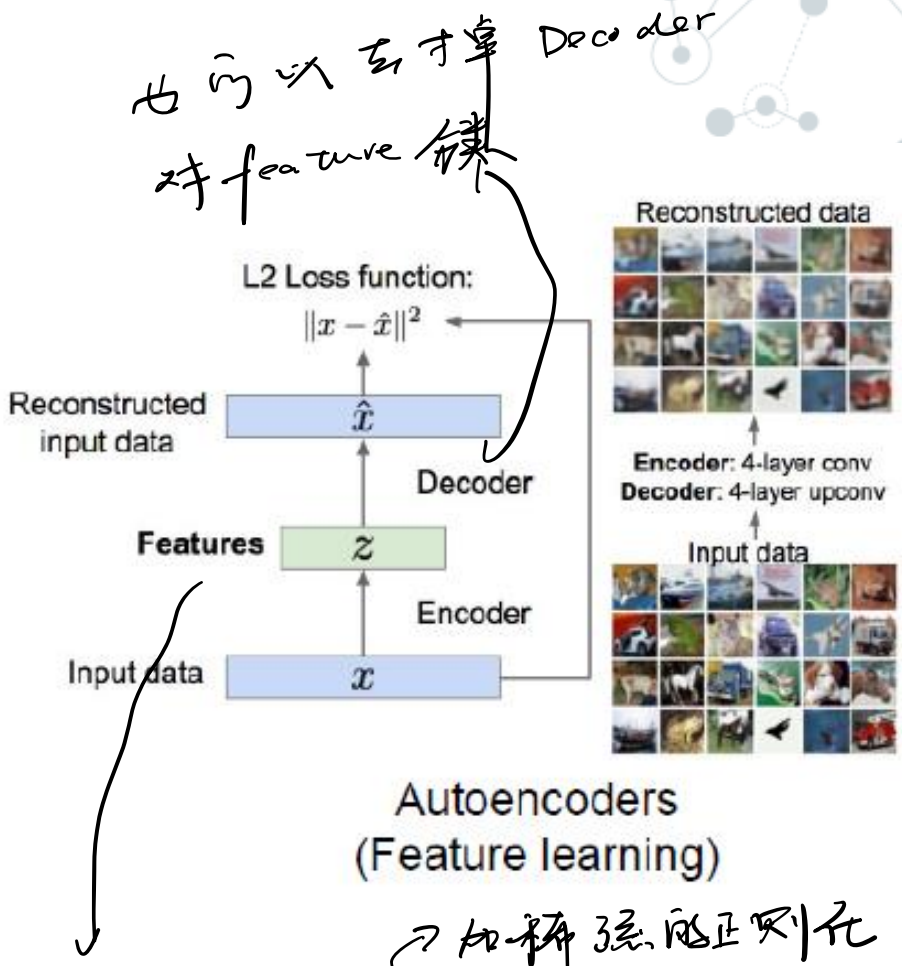
- Task formulation

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.



Unsupervised Learning

- Task formulation

Unsupervised Learning

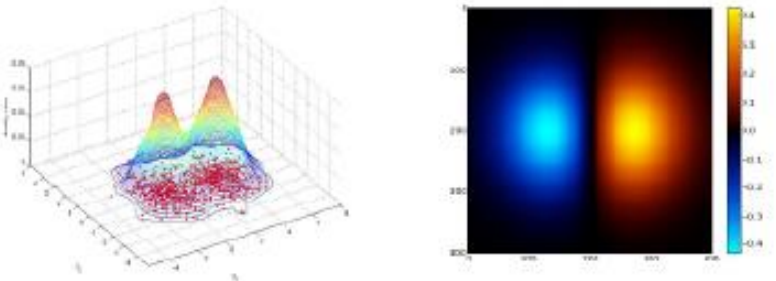
Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation



2-d density estimation

Deep Generative Networks-DGM

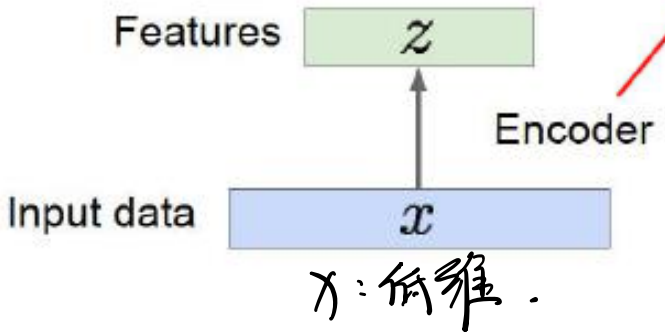
- Overview
- **Representation Learning with Autoencoder**
- Generative Adversarial Network (GAN)
- Applications of GANs

Autoencoder

- Feature representation learning

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

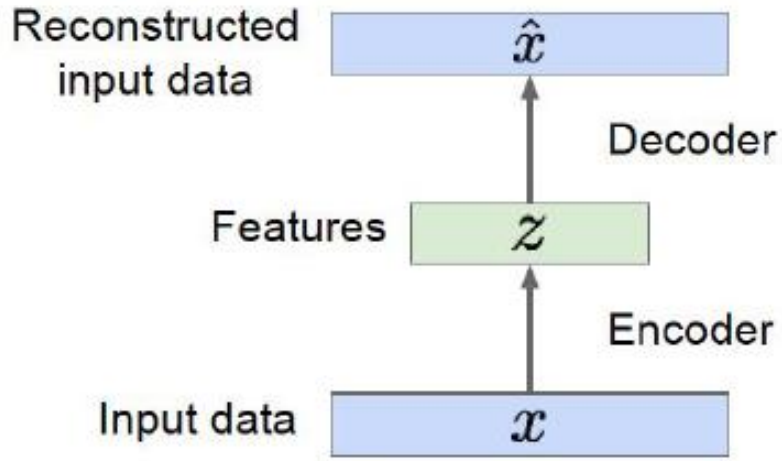
Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN



Autoencoder

- Feature representation learning

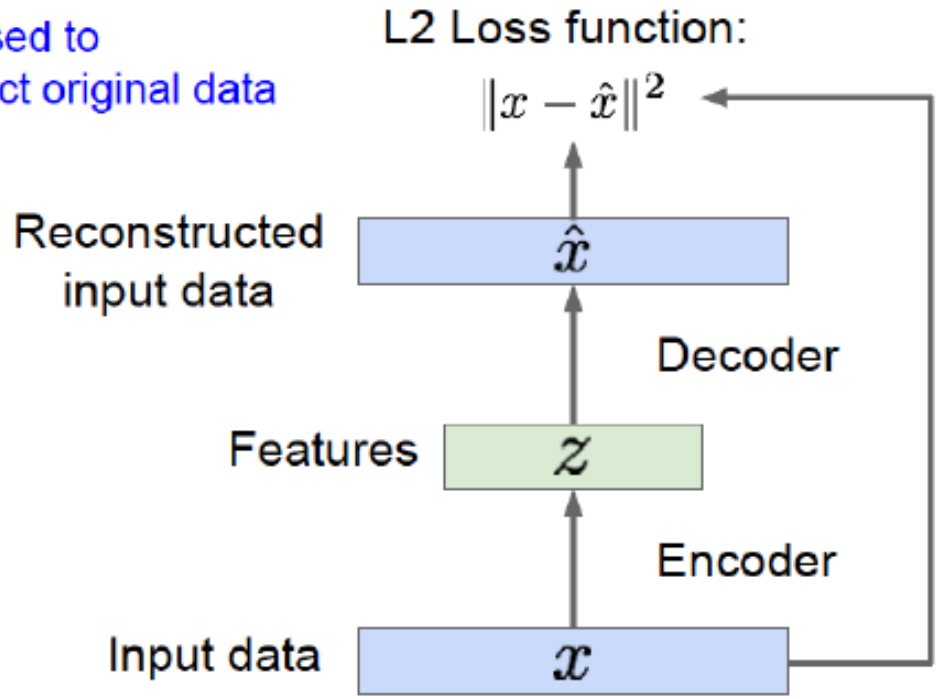
How to learn this feature representation?
Train such that features can be used to reconstruct original data
"Autoencoding" - encoding itself



Autoencoder

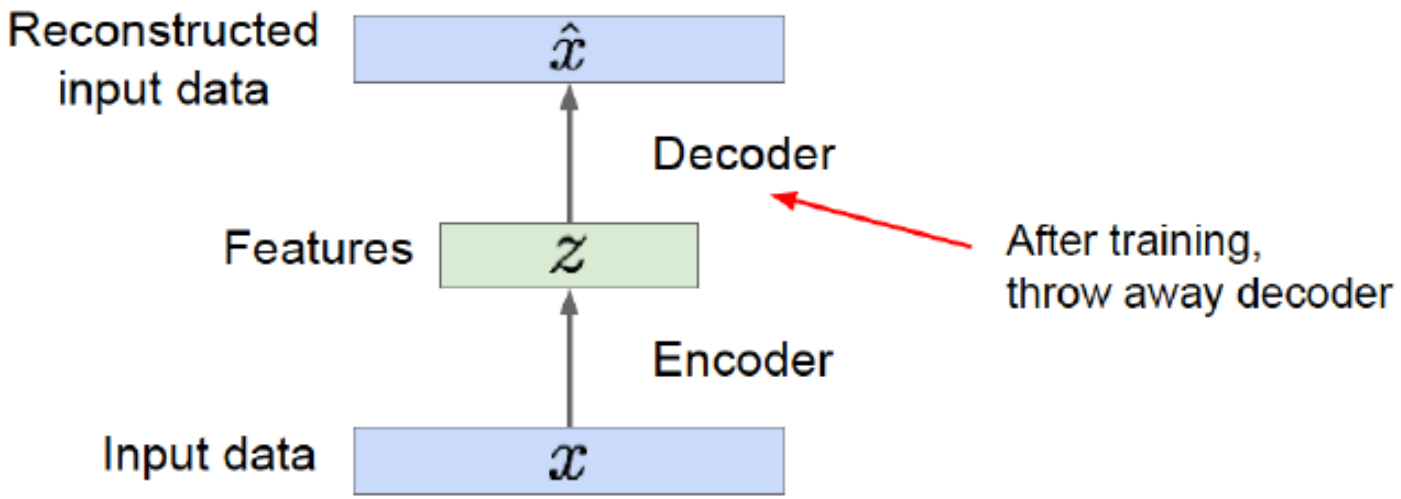
- Feature representation learning

Train such that features can be used to reconstruct original data



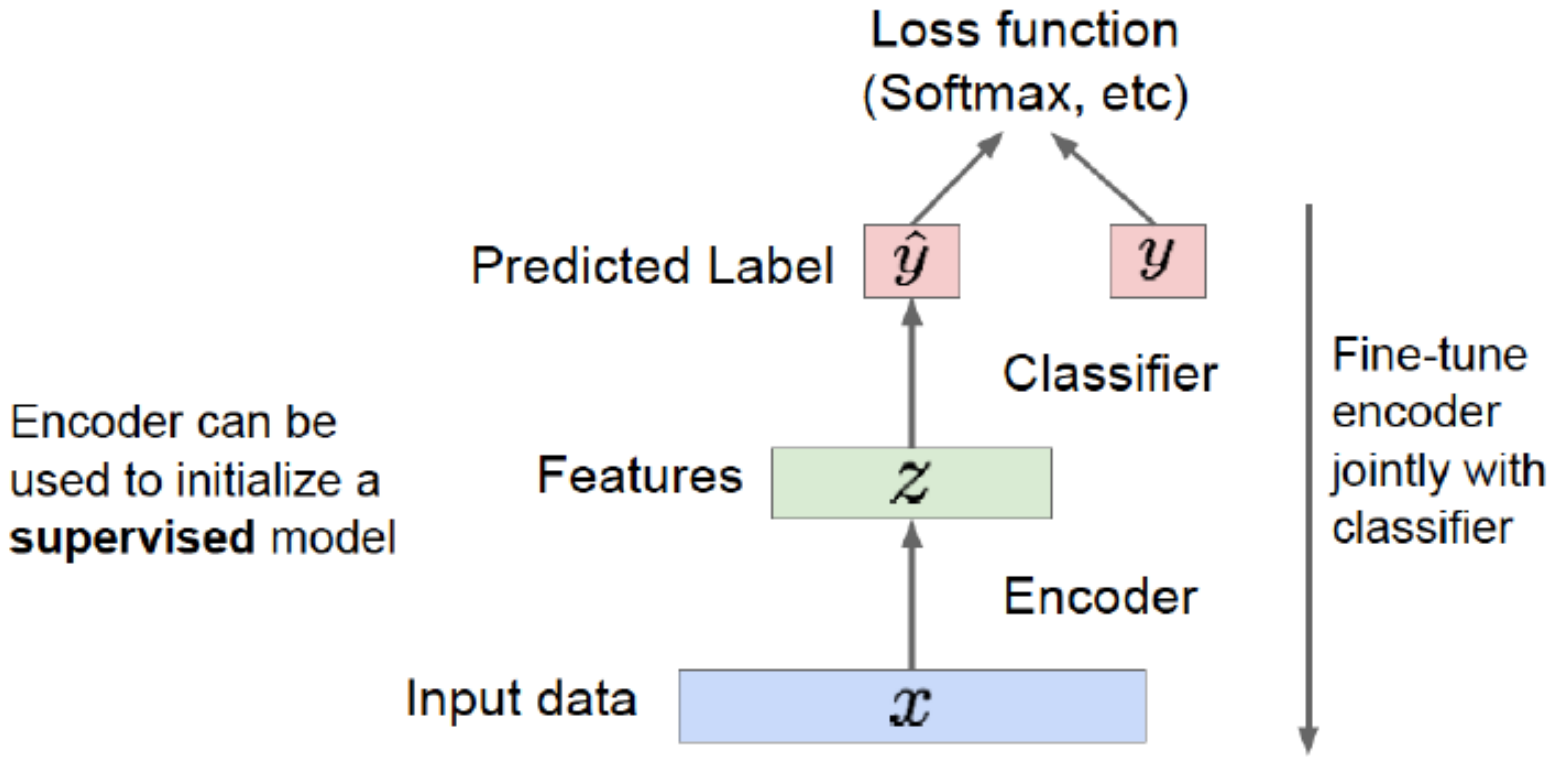
Autoencoder

- Feature representation learning



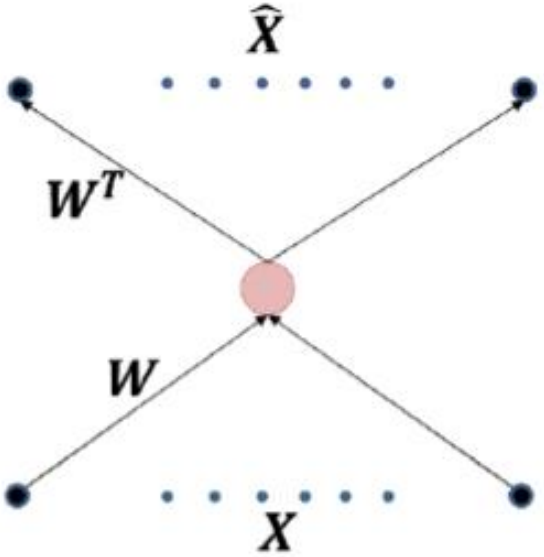
Autoencoder

- Feature representation learning



Autoencoder

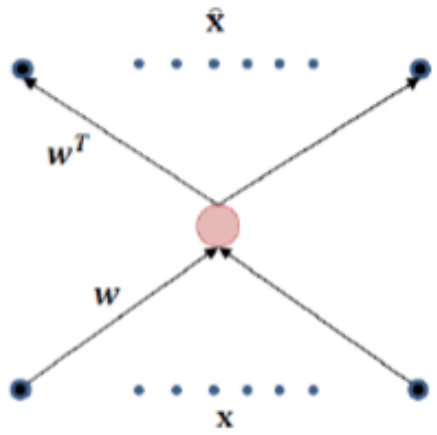
- Linear hidden layer example



- A single hidden unit
- Hidden unit has *linear* activation
- What will this learn?

Autoencoder

- Linear hidden layer example



Training: Learning W by minimizing L2 divergence

$\hat{\mathrm{x}} = w^T w\mathrm{x}$ → ww^Tx（保证矩阵乘法）

$$div(\hat{\mathrm{x}}, \mathrm{x}) = \|\mathrm{x} - \hat{\mathrm{x}}\|^2 = \|\mathrm{x} - \mathrm{w}^T w\mathrm{x}\|^2$$

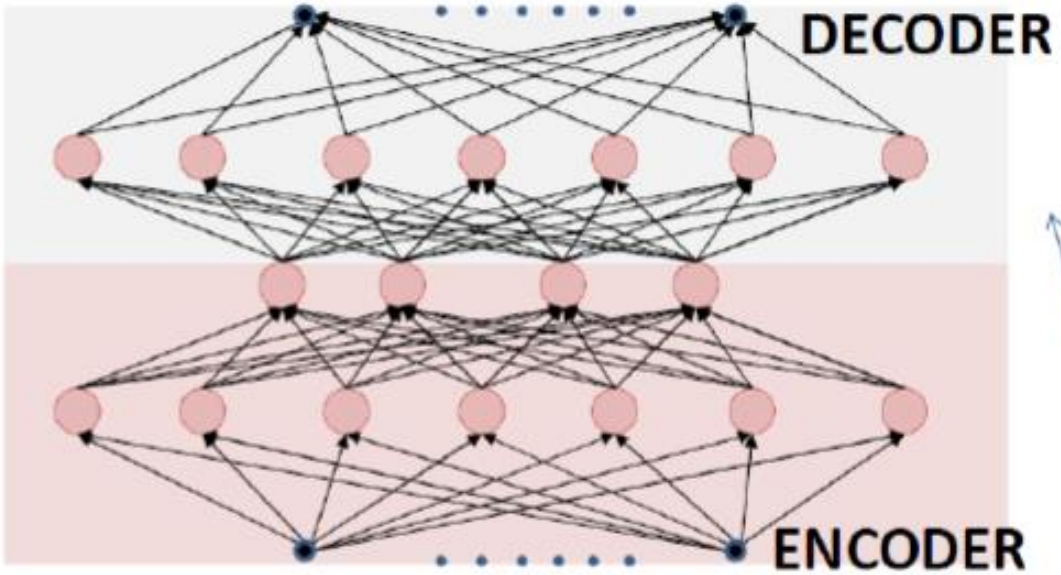
$$\widehat{W} = \underset{W}{\operatorname{argmin}}\, E[div(\hat{\mathrm{x}}, \mathrm{x})]$$

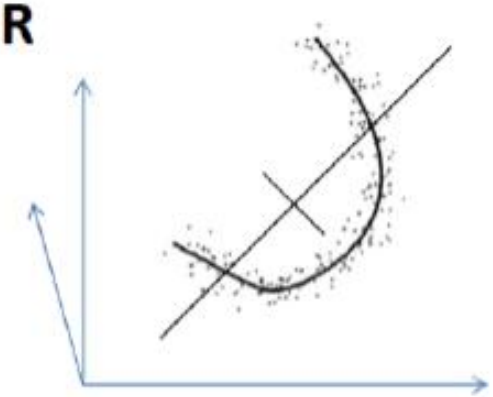
$$\widehat{W} = \underset{W}{\operatorname{argmin}}\, E[\|\mathrm{x} - \mathrm{w}^T w\mathrm{x}\|^2]$$

- This is just PCA! ⇒ 单隐层且只有一个隐层单元时就是一个线性PCA.

Autoencoder

- Nonlinear hidden layer





- With non-linearity
  - "Non linear" PCA
  - Deeper networks can capture more complicated manifolds
    - "Deep" autoencoders

多层会导致计算慢并可能陷入局部最优
(初始化困难)

选两层 → 使用 RBM (受限玻尔兹曼机) 来初始化

$$P_w(v,h) = \frac{\exp(-v^T W h)}{Z(w)}$$

变分自编码器　　$x=W_2z+\varepsilon$，若 $\varepsilon\sim\mathcal{N}(0,\sigma^2)$ 则 $P(x|z)=\mathcal{N}(W_2z,\sigma^2)$

拓展：$P(x|z)=\mathcal{N}(\mu_\theta(z),\sigma^2_\theta(z))$　　变成函数

给定 x 去估计 θ，但目前未知 z

类似 EM 算法：{估计 $q(z)$ / 重算 $\log P_\theta(x)$

$$\max_\theta \log P_\theta(x)=\log\int P_\theta(x,z)\,dz=\log\int q(z)\frac{P_\theta(x,z)}{q(z)}\,dz$$

$$=\log E_{q(z)}\left[\frac{P_\theta(x,z)}{q(z)}\right]\geq E_{q(z)}\left[\log\frac{P_\theta(x,z)}{q(z)}\right]$$

琴生不等式：$\log$ 是 concave

$$=E_{q_\phi(z|x)}\left[\log\frac{P_\theta(x,z)}{q_\phi(z|x)}\right]$$

VAE：去做一个概率的自编码器，使用 $q_\phi(z|x)$ 拟合 $q(z)$

(后验) $q_\phi(z|x)\sim\mathcal{N}(\mu_\phi(x),\sigma^2_\phi(x))$

$$\Rightarrow\theta=\arg\max_\theta E_{q_\phi(z|x)}\left[\log\frac{P_\theta(x,z)}{q_\phi(z|x)}\right]=-KL(q_\phi(z|x)\|P_\theta(x,z))$$

$$=\int q_\phi(z|x)\log\frac{P_\theta(x,z)}{q_\phi(z|x)}\,dz$$

$P_\theta(x|z)$　$P_\theta(z)$　一般是给一个较简单的先验，如：$z\sim\mathcal{N}(0,I)$，是隐空间的初始采样，通过采样去拟合 $q(z)$

$$=\int q_\phi(z|x)\left(\log\frac{q(z)}{q_\phi(z|x)}+\log P_\theta(x|z)\right)dz$$

$$=\int q_\phi(z|x)\log\frac{P_\theta(z)}{q_\phi(z|x)}\,dz+\int q_\phi(z|x)\log P_\theta(x|z)\,dz$$

$P_\theta(z)\sim\mathcal{N}(\mu_\theta(z),\sigma^2_\theta(z))$

$$=-KL(q_\phi(z|x)\|P_\theta(z))+E_{q_\phi(z|x)}[\log P_\theta(x|z)]$$

$$ELBO=\frac{1}{k}\sum_{i=1}^{k}\log\frac{P_\theta(x,z_i)}{q_\phi(z_i|x)}=\frac{1}{k}\sum_{i=1}^{k}\log\frac{P_\theta(x,\mu_\phi(x)+\sigma_\phi(x)\odot\varepsilon_i)}{q_\phi(\mu_\phi(x)+\sigma_\phi(x)\odot\varepsilon_i|x)}$$

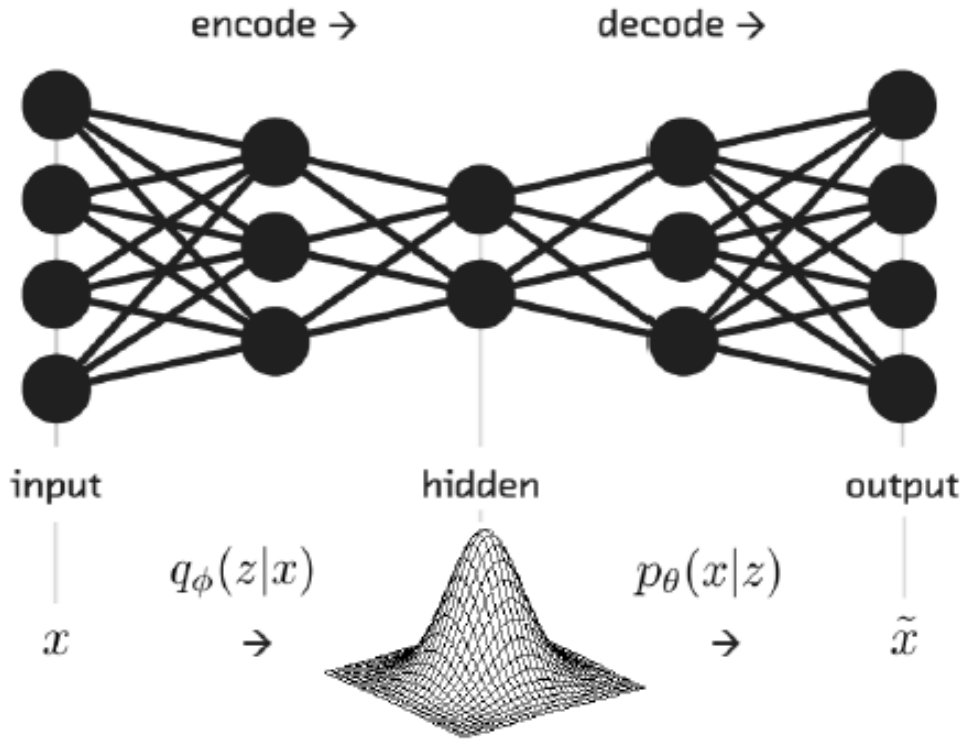
$z_i\sim q_\phi(z|x)$，$z_i=\mu_\phi(x)+\sigma_\phi(x)\odot\varepsilon_i$，$\varepsilon_i\sim\mathcal{N}(0,I)$

通过梯度下降计算：$\theta\leftarrow\theta-\eta\frac{\partial L}{\partial\theta}$，$\phi\leftarrow\phi-\eta\frac{\partial L}{\partial\phi}$

Variational Autoencoder (VAE)

- Objective $\mathcal{L}(x, \phi, \theta) = -D_{KL}(q_\phi(z|x)||p_\theta(z)) + E_{q_\phi(z|x)}[\log p_\theta(x|z)]$

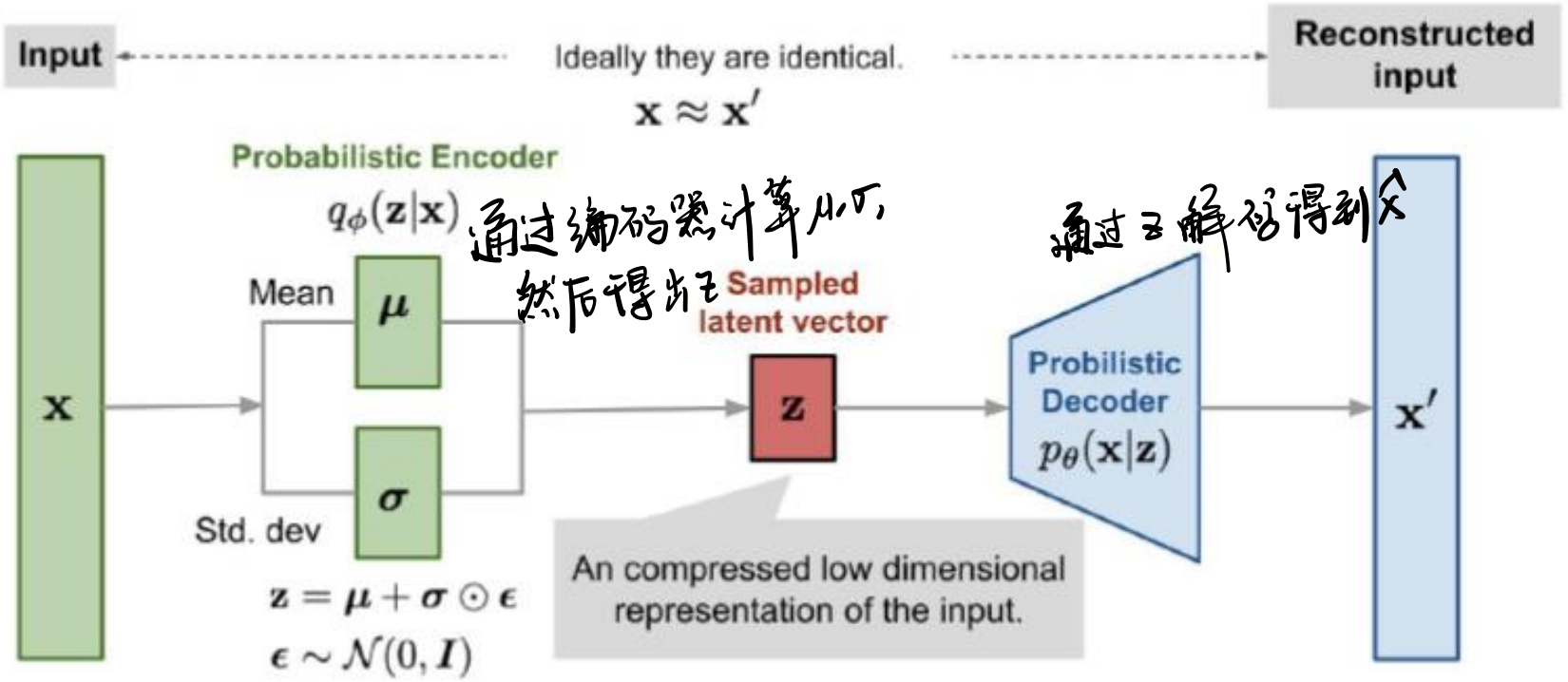
Regularization term Reconstruction term



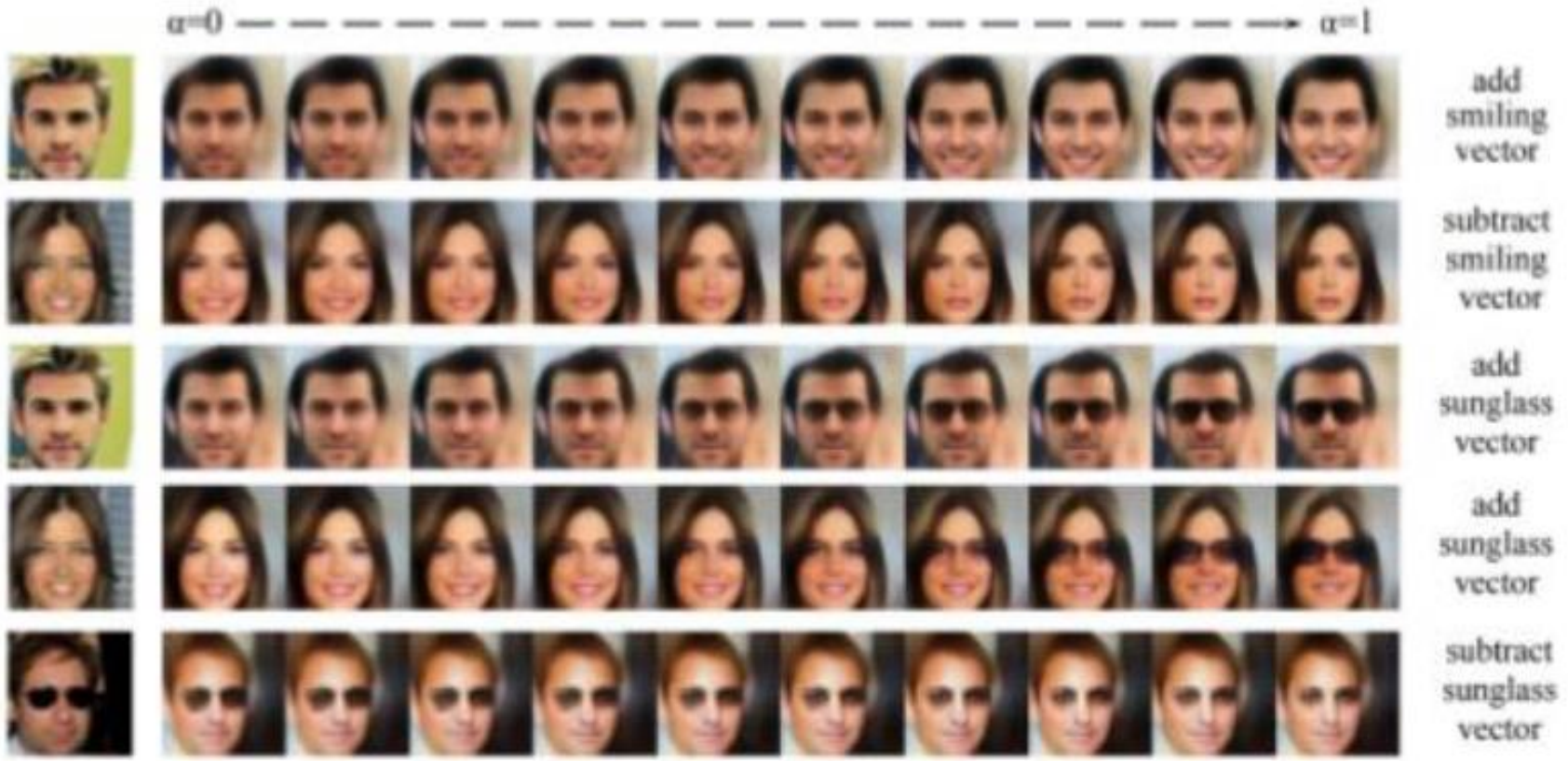
Variational Autoencoder (VAE)

- Objective $\mathcal{L}(x, \phi, \theta) = -D_{KL}(q_\phi(z|x)||p_\theta(z)) + E_{q_\phi(z|x)}[\log p_\theta(x|z)]$

Regularization term Reconstruction term



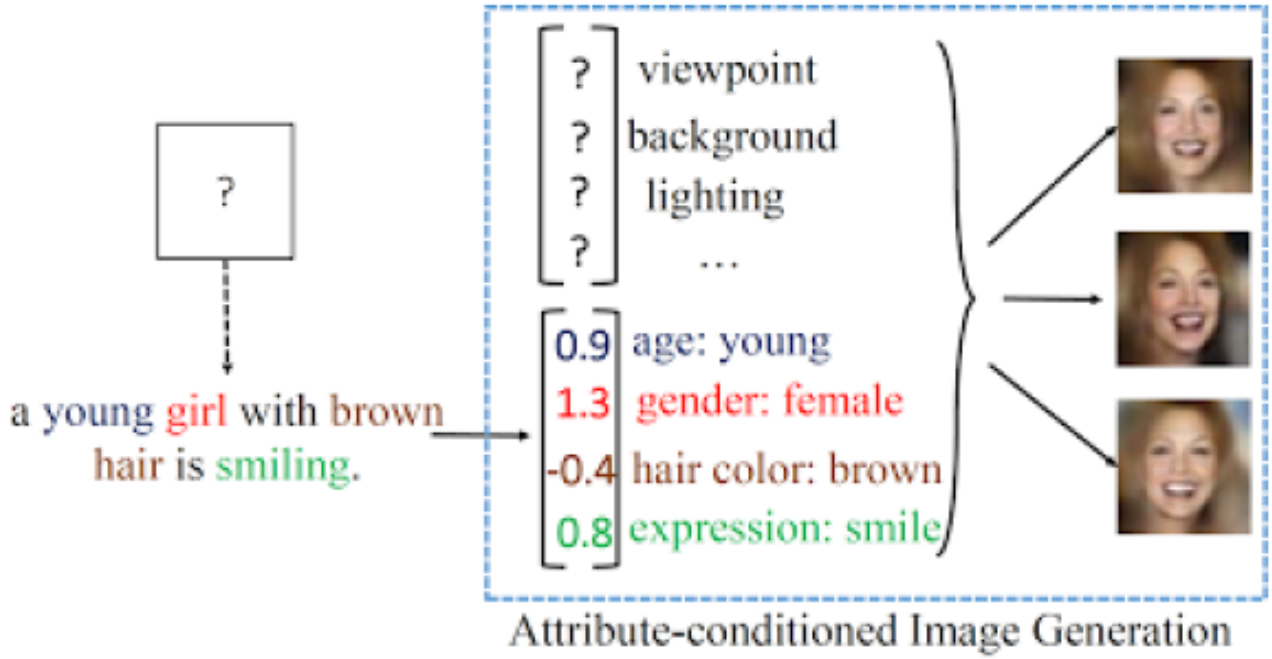
Interpreting the Latent Space

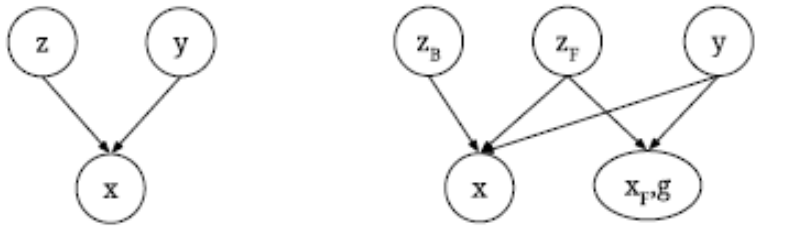


https://arxiv.org/pdf/1610.00291.pdf

Example: Attribute2Image

- Main ideas

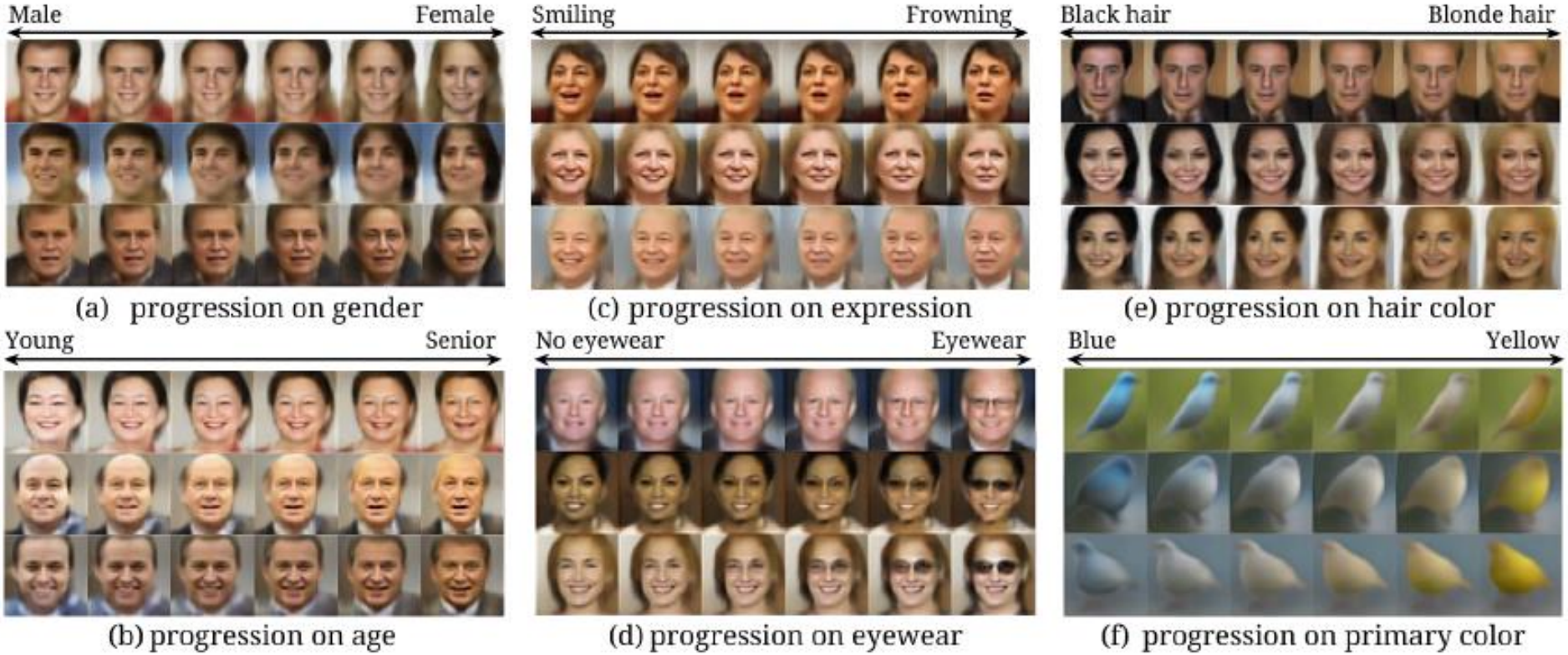




(a) CVAE: $p_\theta(x|y,z)$ (b) disCVAE: $p_\theta(x, x_F, g|y, z_F, z_B)$

Example: Attribute2Image

- Results



Problem of VAE

- Blurry images

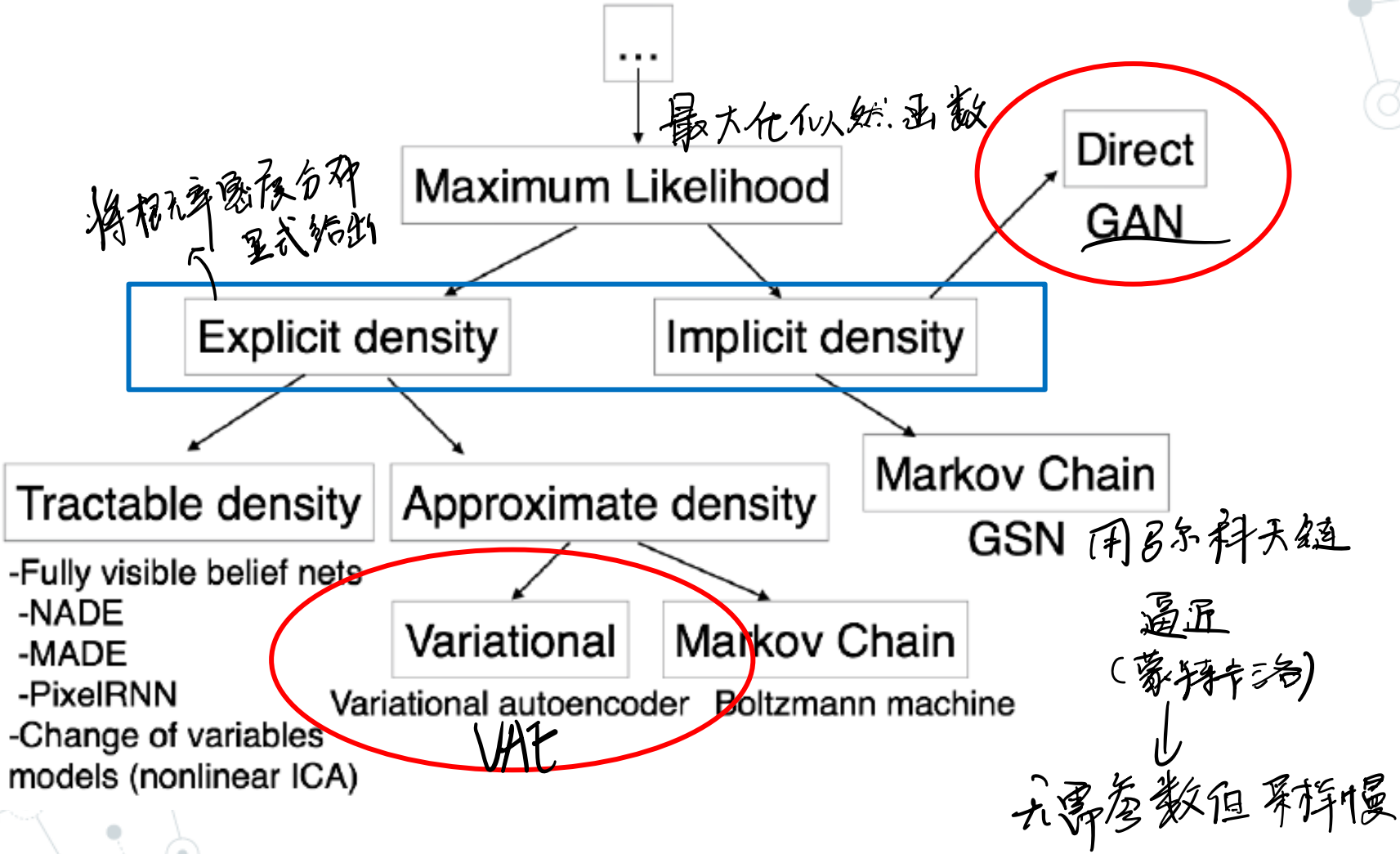
https://blog.openai.com/generative-models/

Deep Generative Networks-DGM

- Overview
- Representation Learning with Autoencoder
- **Generative Adversarial Network (GAN)**
- Applications of GANs

生成对抗网络

Taxonomy of Generative Models



Implicit Generative Models

更加复杂

- Working with explicit model p(x) could be expensive
  - Variational Autoencoder (variational inference)
  - Boltzmann Machines (MCMC)

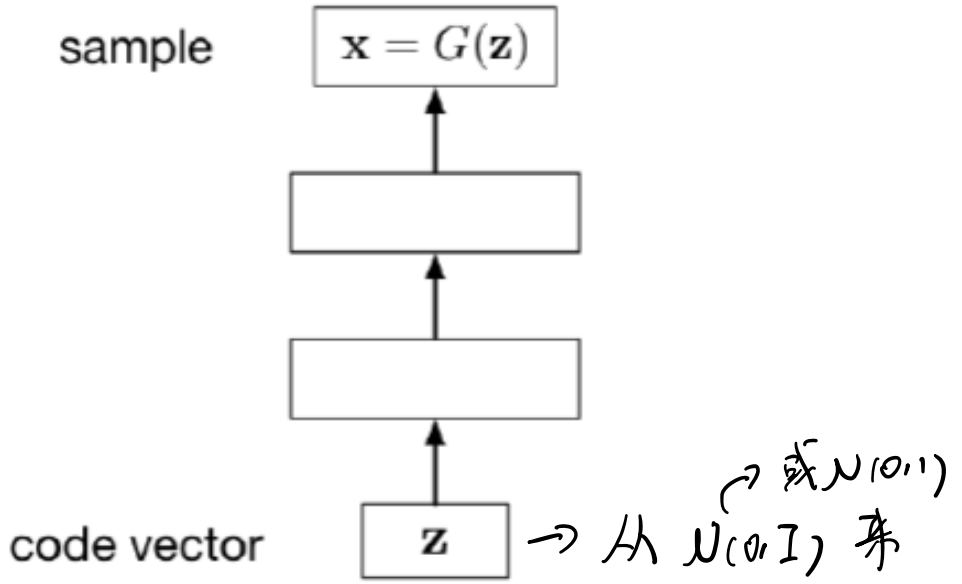
- Representation learning may not require p(x)
  - Sometimes we are more interested in taking samples from p(x) instead of p itself

不需要知道样本点的概率分布

只用知到样本本身

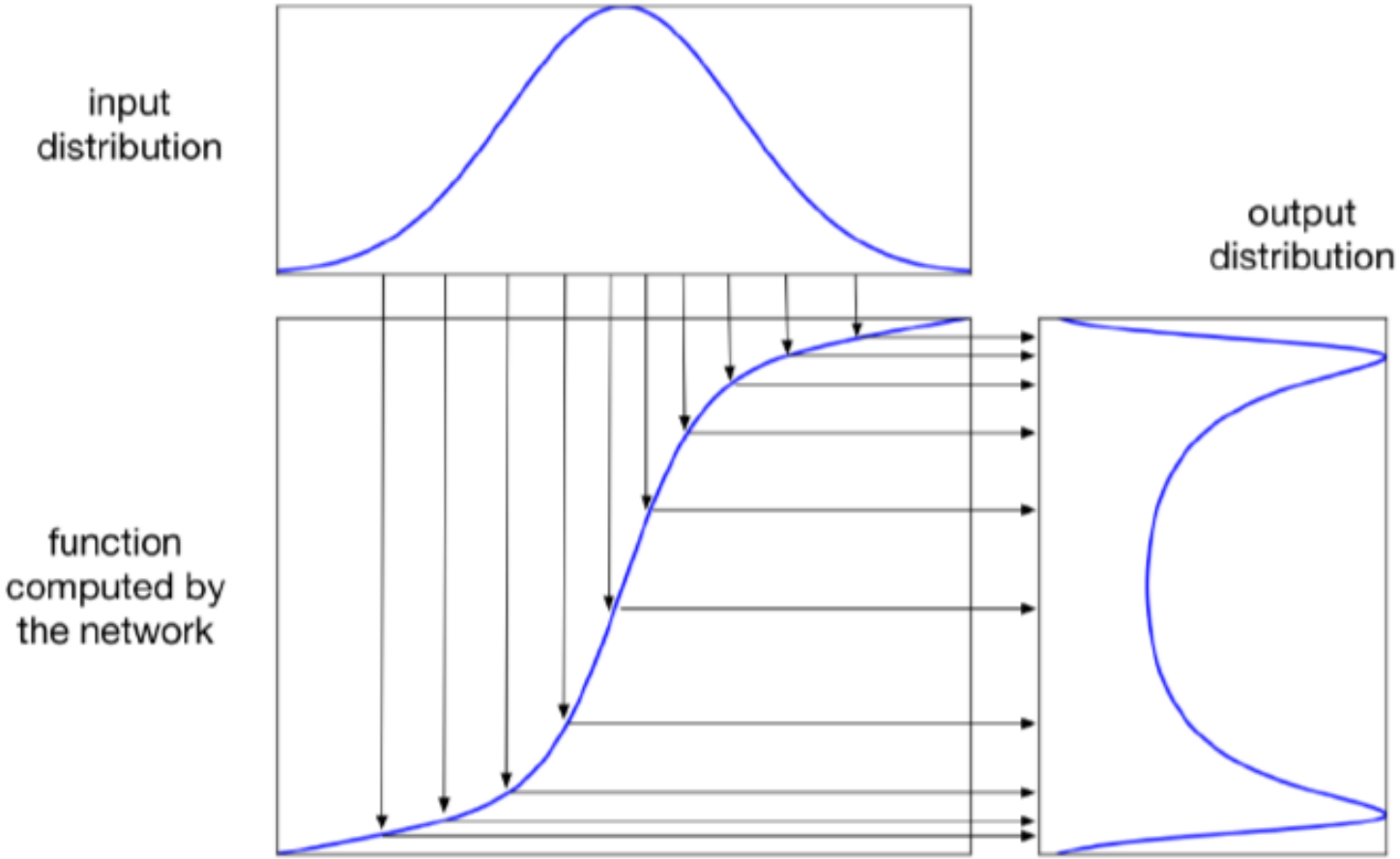
Implicit Generative Models

- Implicitly define a probability distribution
- Start by sampling the code vector z from a fixed, simple distribution
- A generator network computes a differentiable function G mapping z to an x in data space



Implicit Generative Models

- Intuition: 1D example

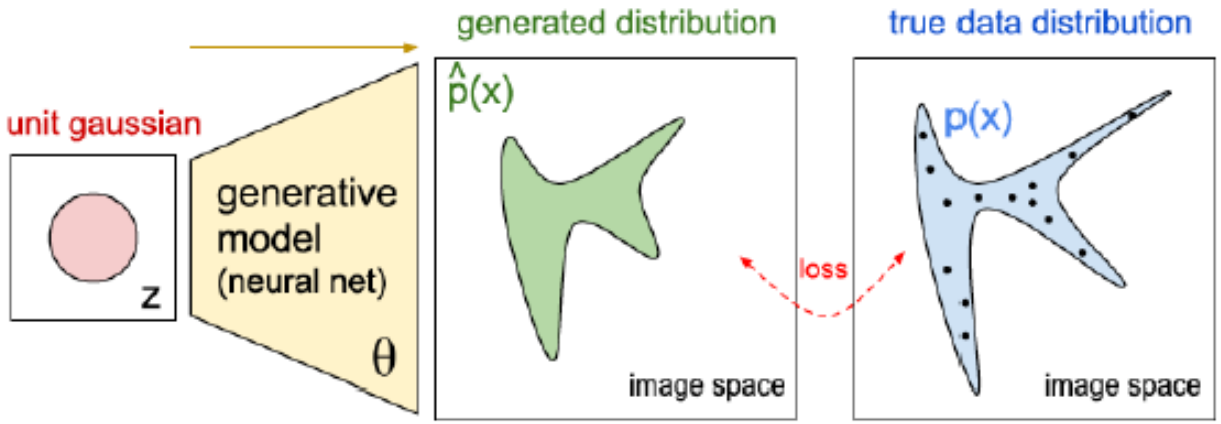


可以用NN模拟任何分布

足够复杂则可以生成图象.

Implicit Generative Models

- Intuition

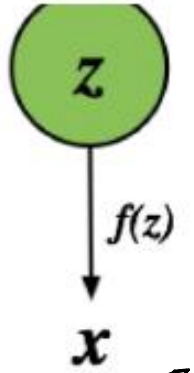


advocate/penalize samples within the blue/white region.

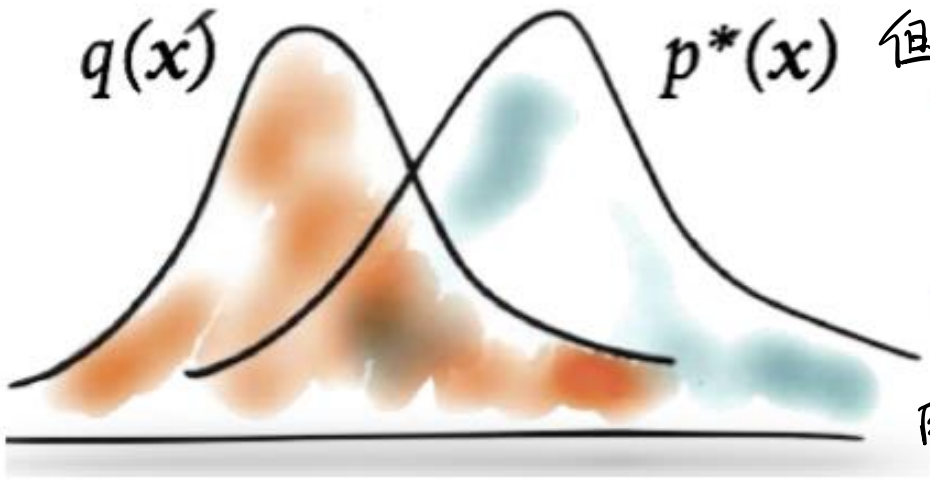
Learning by Comparison

- Basic idea

For some models, we only have access to an unnormalised probability, partial knowledge of the distribution, or a simulator of data.



通过惩罚来使 $q(x)$ 趋向于 $p^*(x)$



但 $q(x)$ 未知，所以无法用以散度

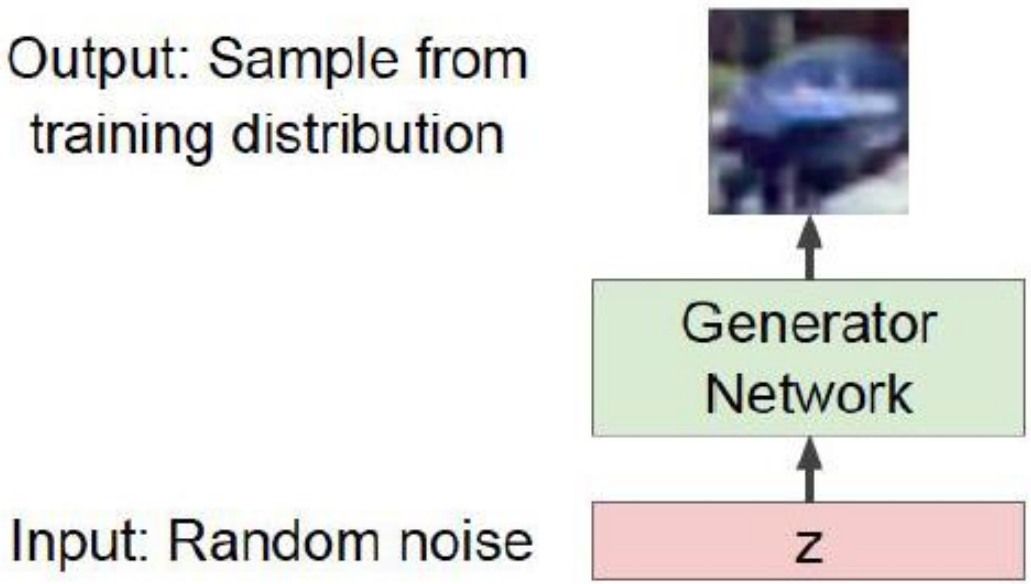
We compare the estimated distribution q(x) to the true distribution p*(x) using samples.

用生成器生成图片，给判别器判别器也在同时训练

通过判别器的反馈惩罚 $q(x)$

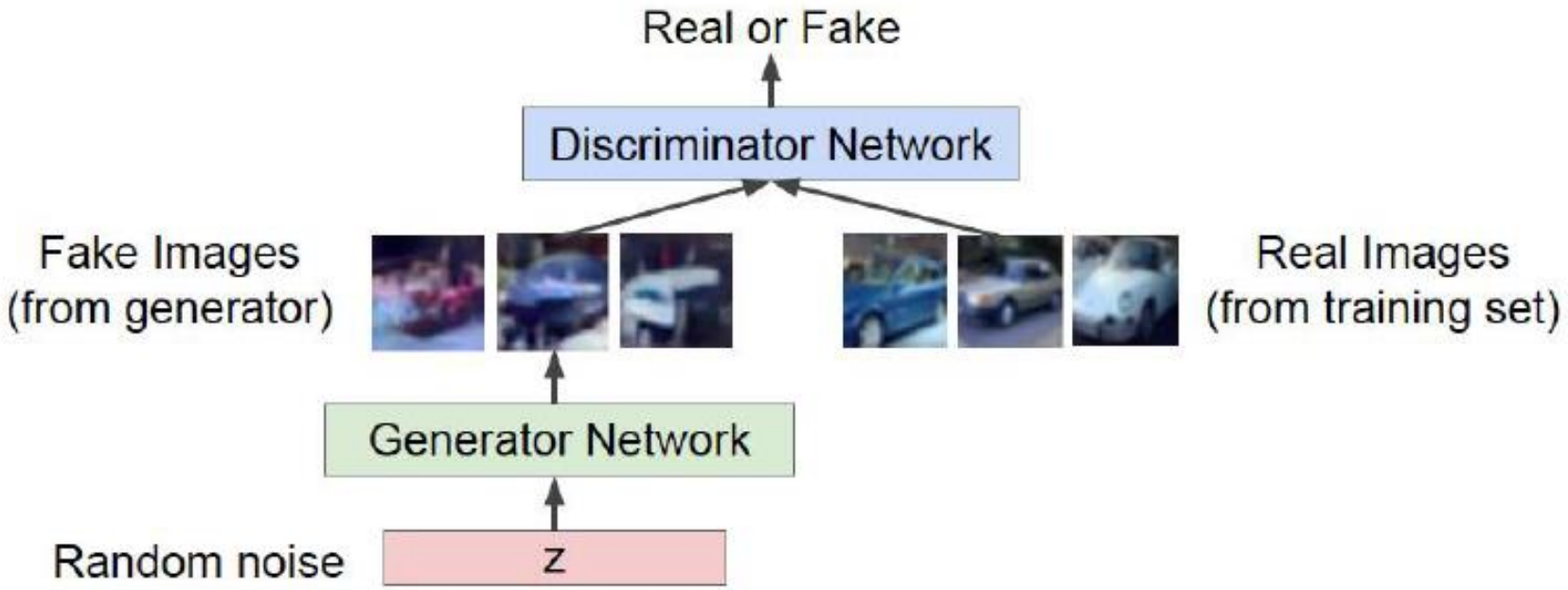
Generative Adversarial Networks (GAN)

- Using a neural network to generate data



Generative Adversarial Networks (GAN)

- Using another neural network to determine if the data is real or not

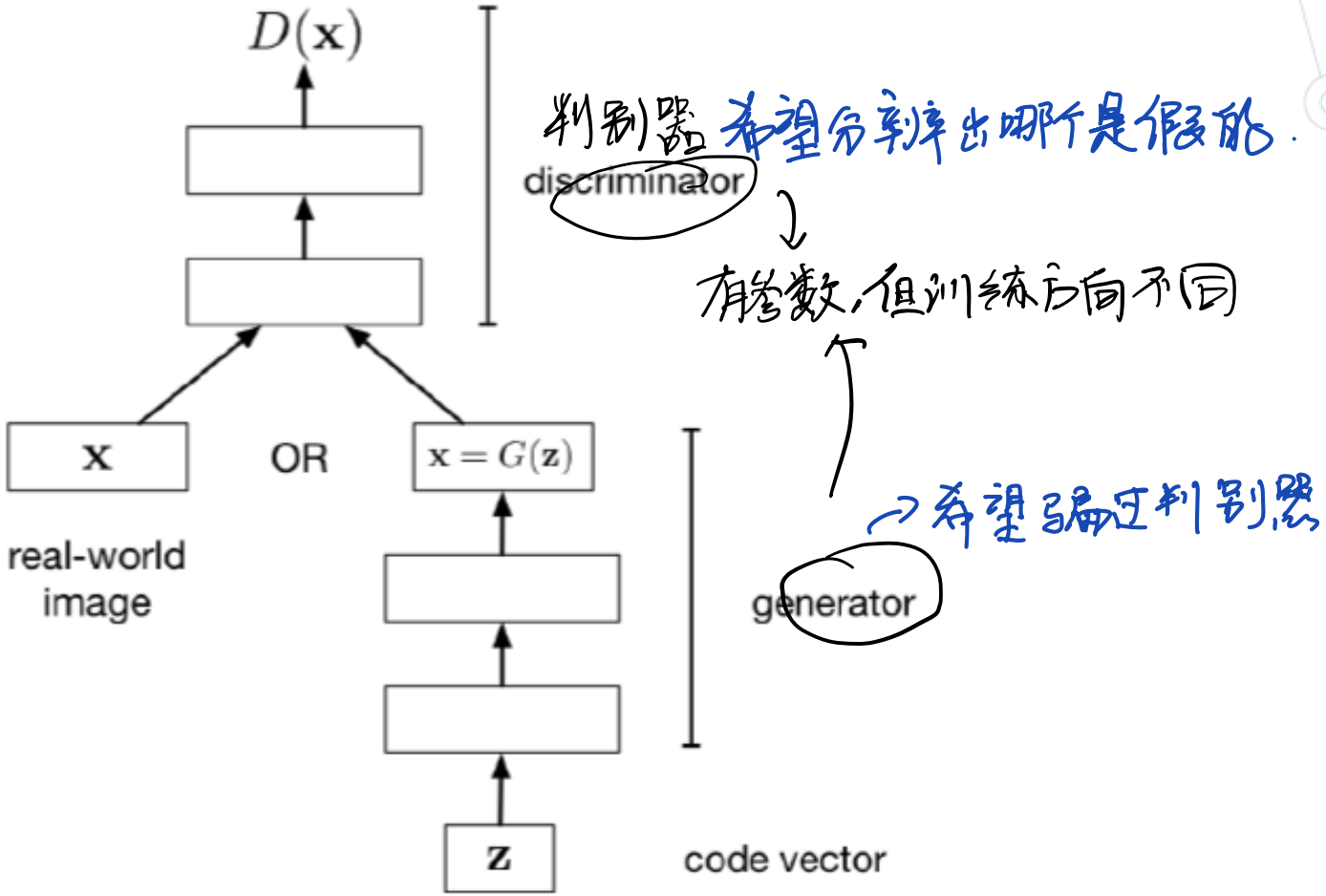


Adversarial Learning

- GAN objective for the generator is some complicated objective function defined by a neural network.
  - This means a new way of thinking about "distance".
  - We are training networks to minimize the "distance" or "divergence" between generated images and real images.
  - Instead of some hand-crafted distance metric like L1 or L2, we can make something completely new.
  - A neural network, with the right architecture, is arguably the definition of perceptual similarity (assuming our visual system is some sort of neural network).

Adversarial Learning

- Adversarial loss



Logistic 分布：

取值	1	0
概率	$\sigma(\beta^T x)$	$1-\sigma(\beta^T x)$

$$p(y_i|x_i,\beta) = \sigma^{y_i}(\beta^T x)\,(1-\sigma(\beta^T x))^{1-y_i}$$

$$\Rightarrow \max_\beta \sum_{i=1}^n \log p(y_i|x_i,\beta) = \max_\beta \sum_{i=1}^n [\, y_i \log \sigma(\beta^T x) + (1-y_i)\log(1-\sigma(\beta^T x))\,]$$

cross entropy – loss

Adversarial Learning

- Let D denote the discriminator's predicted probability of being real data
- Discriminator's cost function: cross-entropy loss for task of classifying real vs. fake images

判别器的损失

$$J_D = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[-\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}}[-\log(1-D(G(\mathbf{z})))]$$

- One possible cost function for the generator: the opposite of the discriminator's

生成器损失

$$J_G = -J_D$$
$$= \text{const} + \mathbb{E}_{\mathbf{z}}[\log(1-D(G(\mathbf{z})))]$$

因为参数不在D中，只在G中

Two-Player Game

- Minimax formulation
  - The generator and discriminator are playing a zero-sum game against each other 最大最小化问题

$$\max_G \min_D \mathcal{J}_D$$

  - Using parametric models

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data x

Discriminator output for generated fake data G(z)

注意有一个负号，所以 max 和 min 需要换一下.

Learning Procedure

- Minimax objective function

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

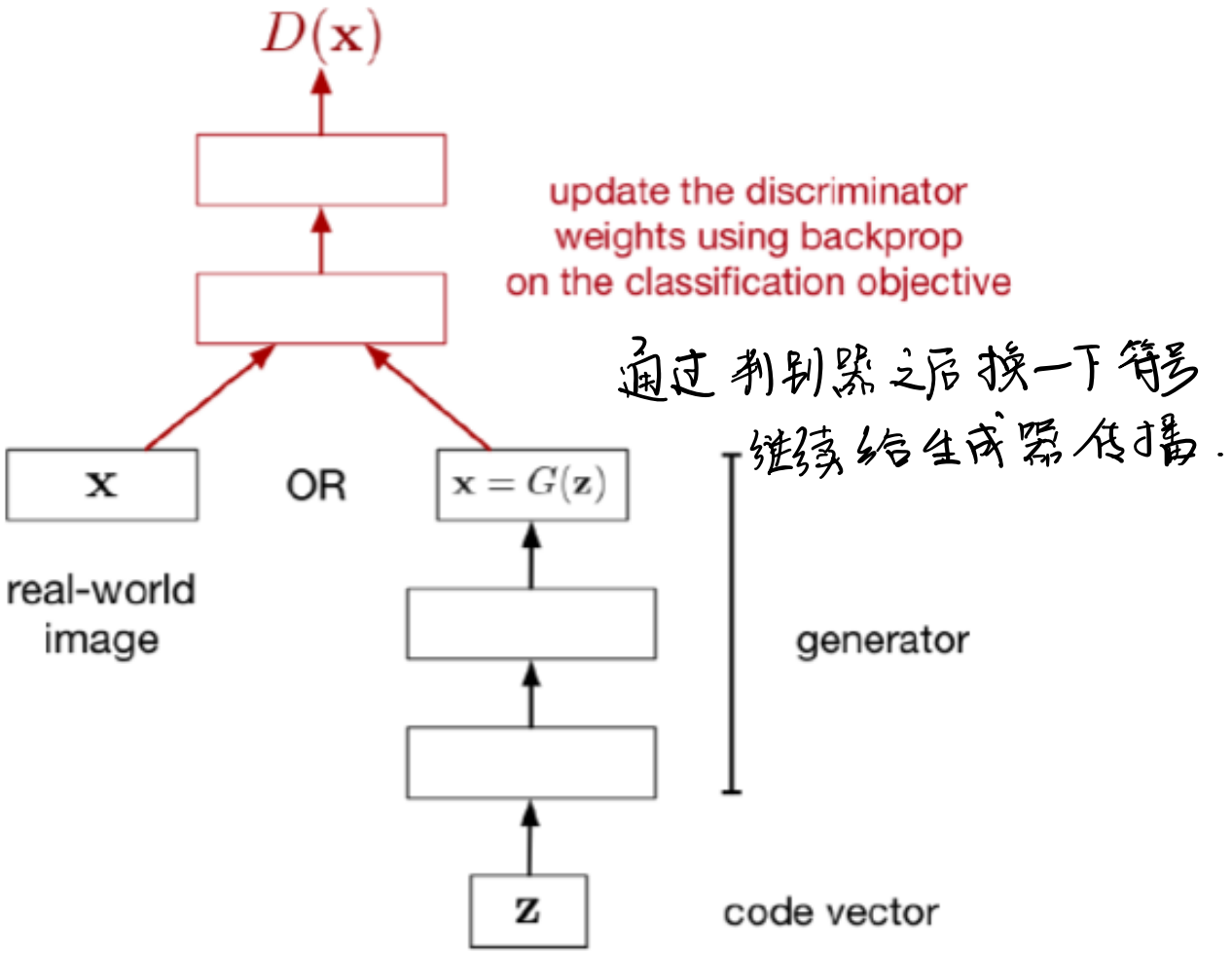
$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

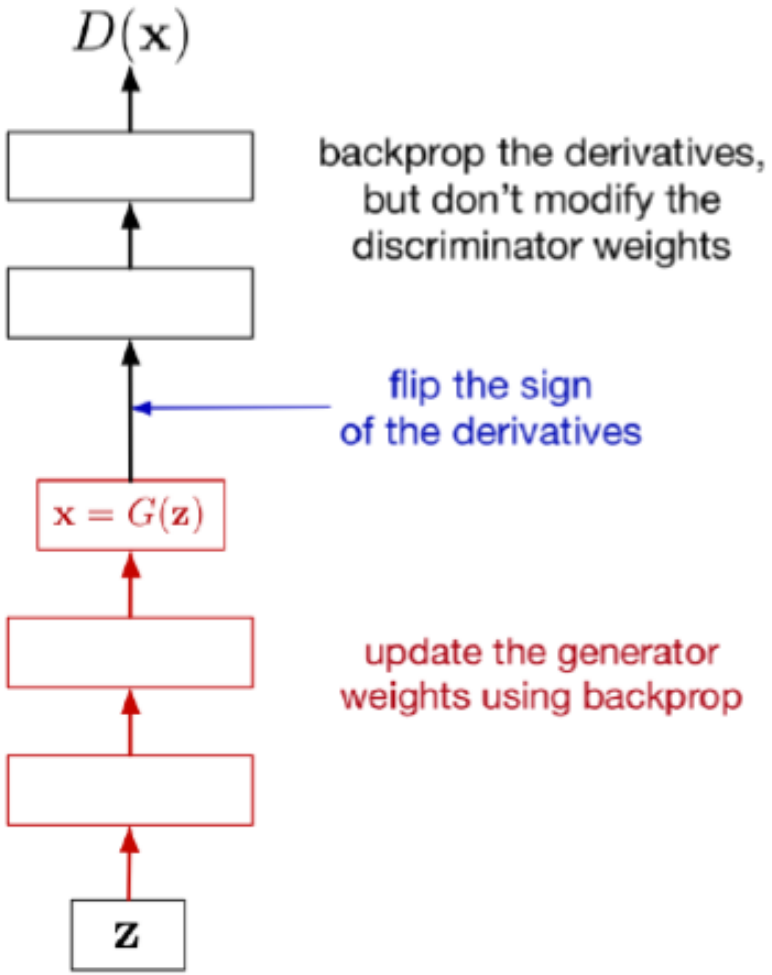
Learning Procedure

- Updating the discriminator

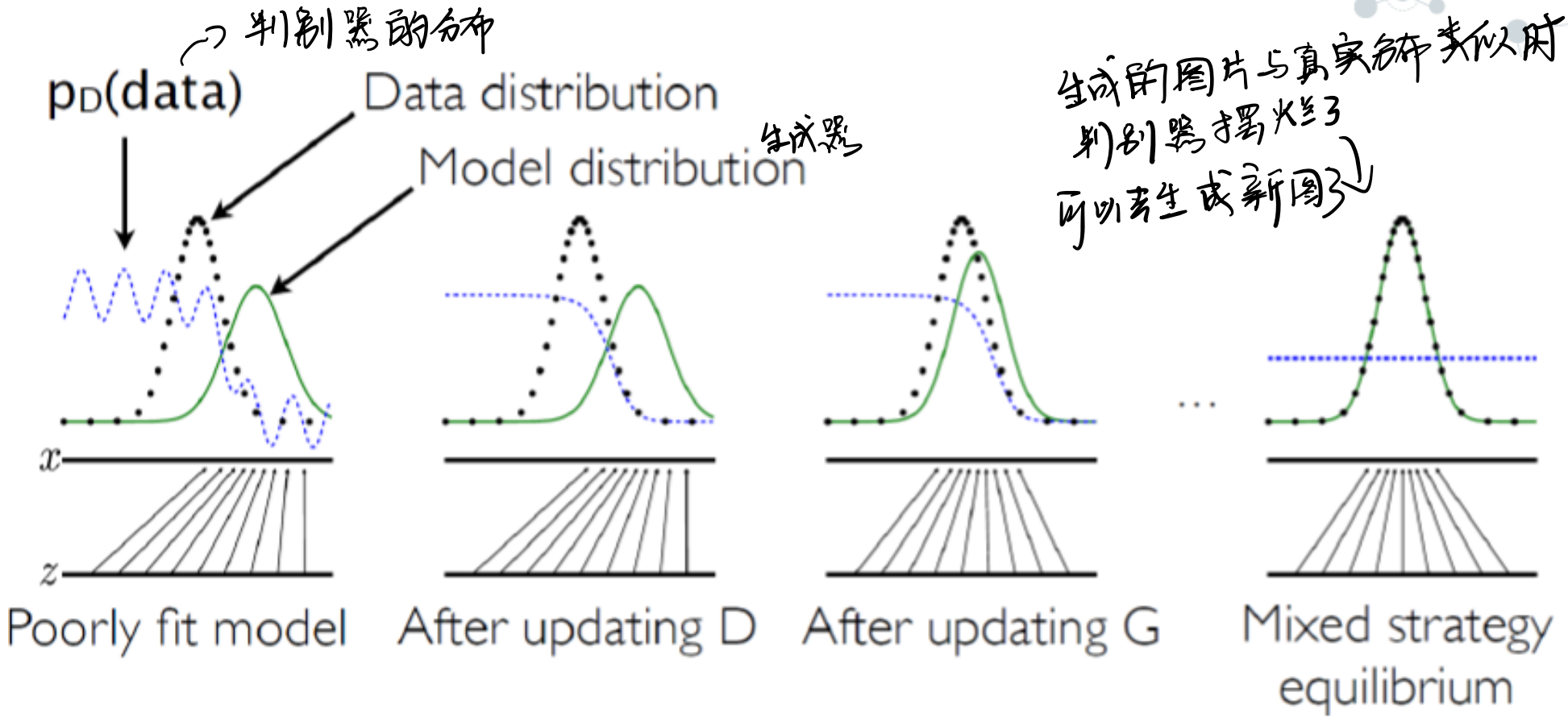


Learning Procedure

- Updating the generator

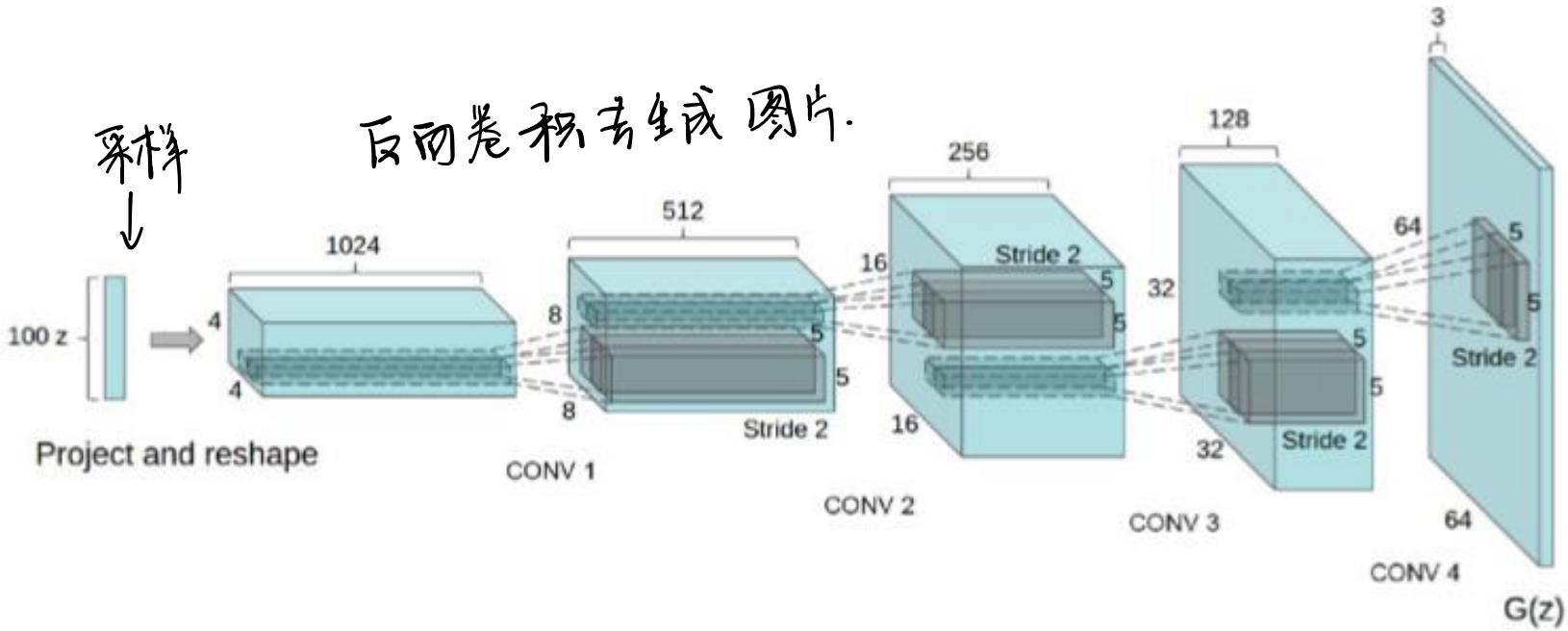


Training GANs



Typical Generator Architecture

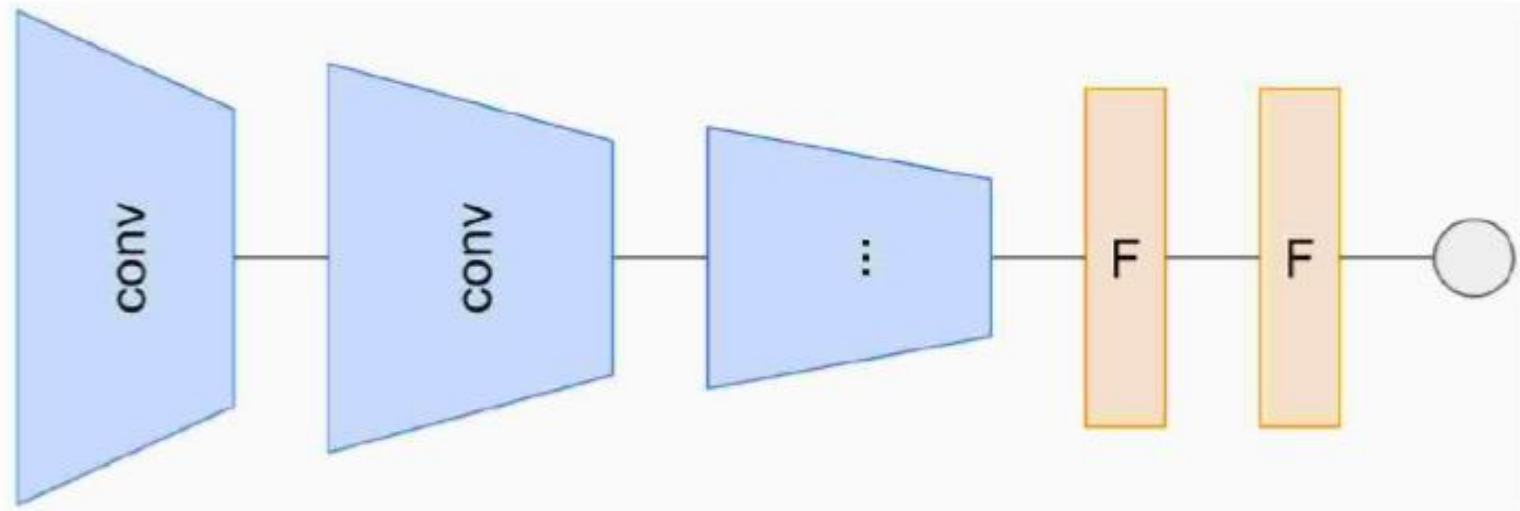
- For images



- Unit Gaussian distribution on z, typically 10-100 dim.
- Up-convolutional deep network (reverse recognition CNN)

Typical Discriminator Architecture

- For images



- Recognition CNN model
- Binary classification output: real / synthetic

Training GANs

- Since GANs were introduced in 2014, there have been hundreds of papers introducing various architectures and training methods
- GAN Zoo: https://github.com/hindupuravinash/the-gan-zoo
- In general, training a GAN is tricky and unstable
- Many tricks:
  - S. Chintala, How to train a GAN, ICCV 2017 tutorial
  - https://github.com/soumith/talks/blob/master/2017-ICCV_Venice/How_To_Train_a_GAN.pdf

非常不稳定

但一旦有个好模型

图片质量会极高.

Generative Samples

Celebrities:

Karras et al., 2017. Progressive growing of GANs for improved quality, stability, and variation

Generative Samples

Objects:

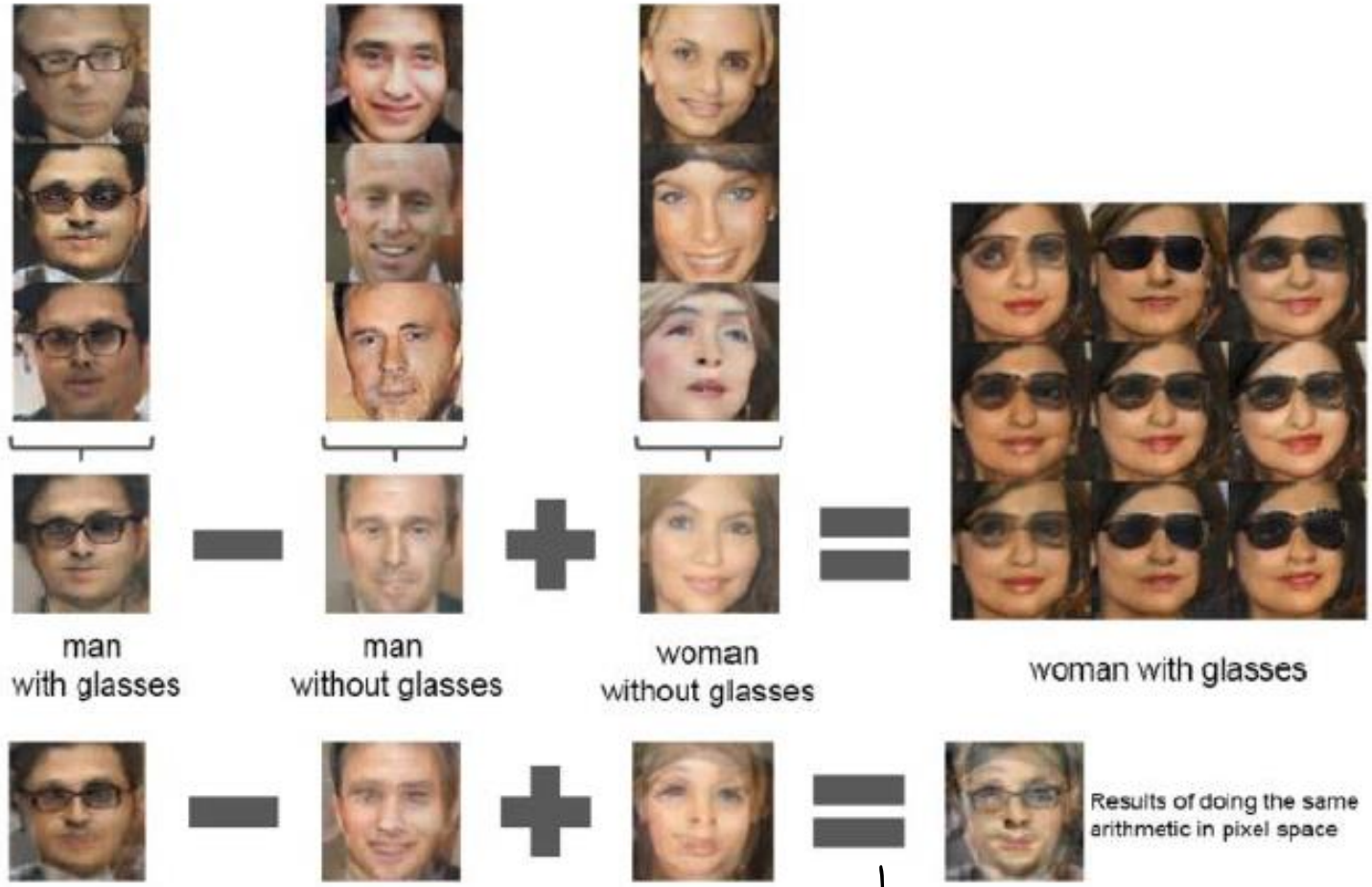
Walk Around Data Manifold

Interpolating between random points in laten space

Radford et al, ICLR 2016

Walk Around Data Manifold

- Vector Arithmetic



Deep Generative Networks-DGM

- Overview
- Representation Learning with Autoencoder
- Generative Adversarial Network (GAN)
- **Applications of GANs**

Conditional GANs

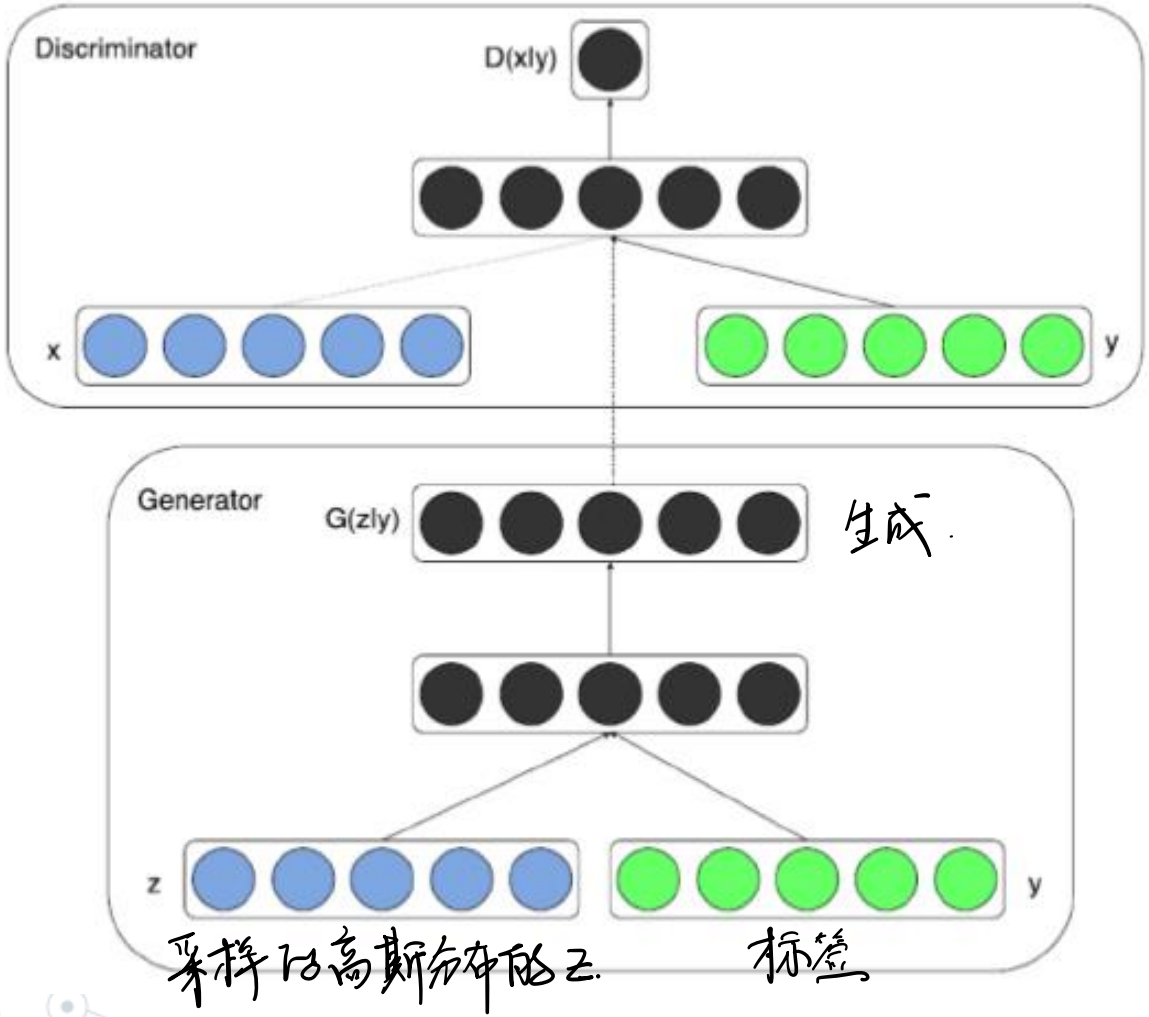
- Conditional GANs include a label and learn P(X|Y)
  - Add conditional variable y into G and D
  - Objective function

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

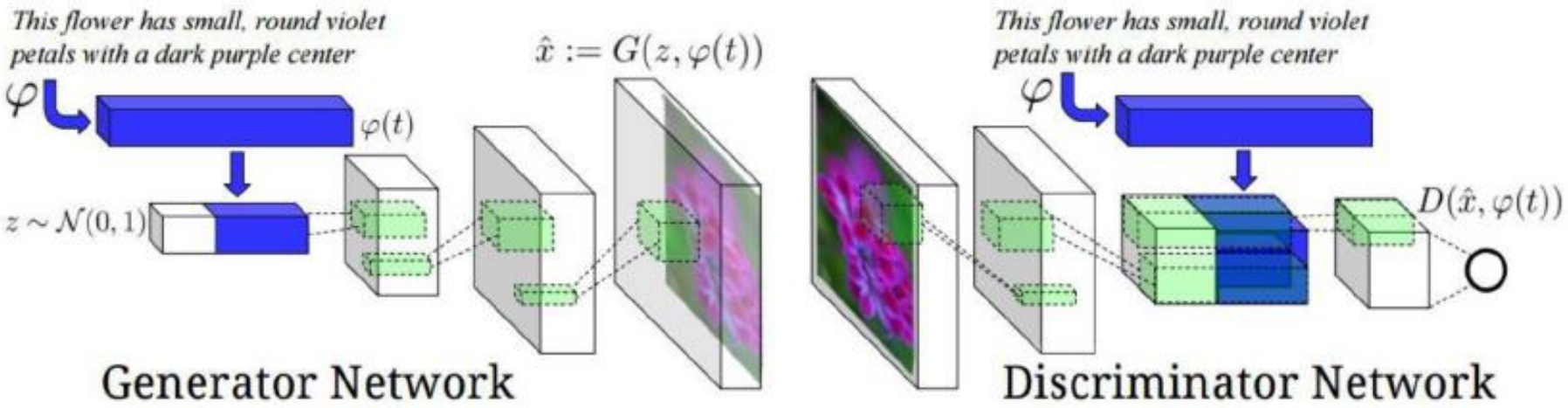
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))].$$

Conditional GANs

- Model architecture



Conditional GANs



this small bird has a pink breast and crown, and black primaries and secondaries.

this magnificent fellow is almost all black with a red crest, and white cheek patch.

Reed et al 2015

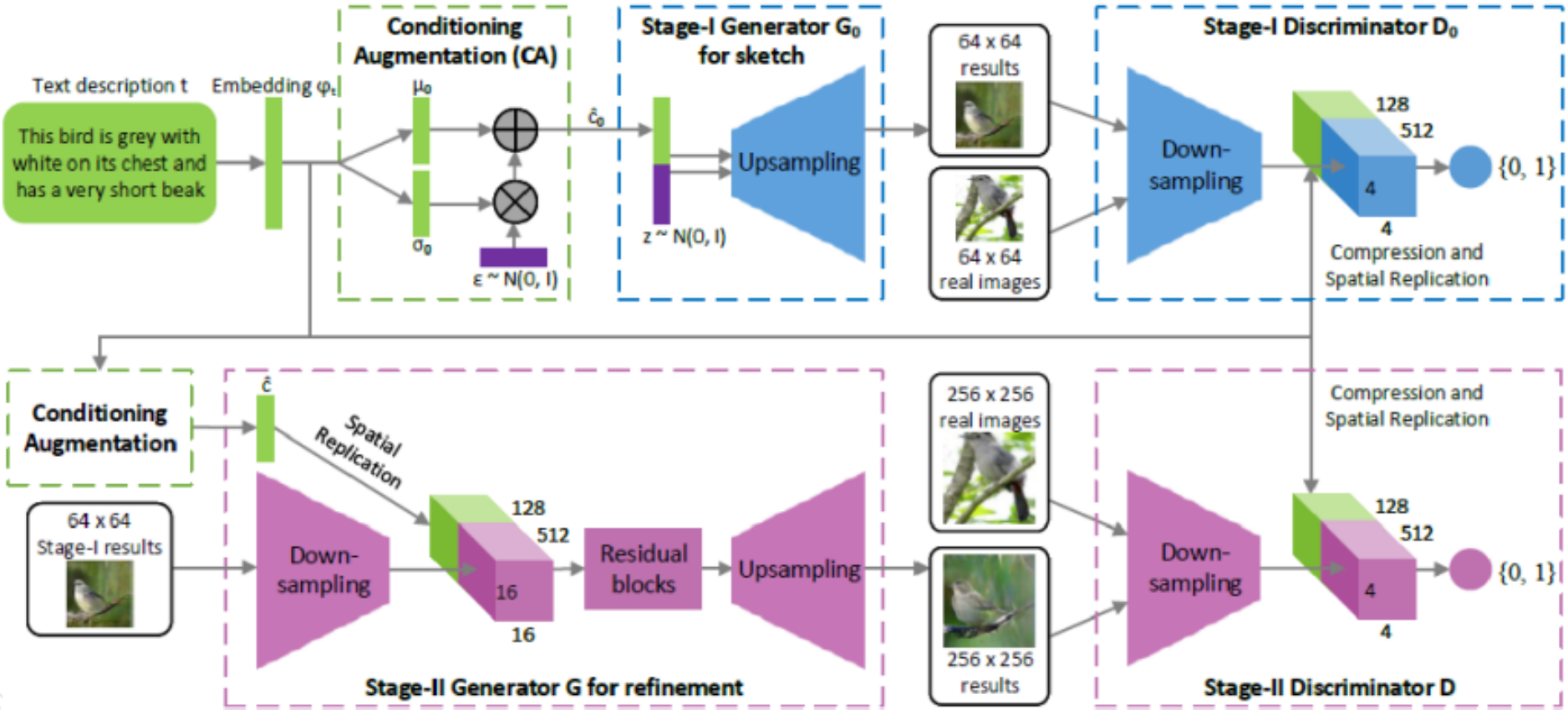
StackGAN

- A coarse-to-fine manner

Zhang et al. 2016

StackGAN

- Use stacked GAN structure



More StackGAN results

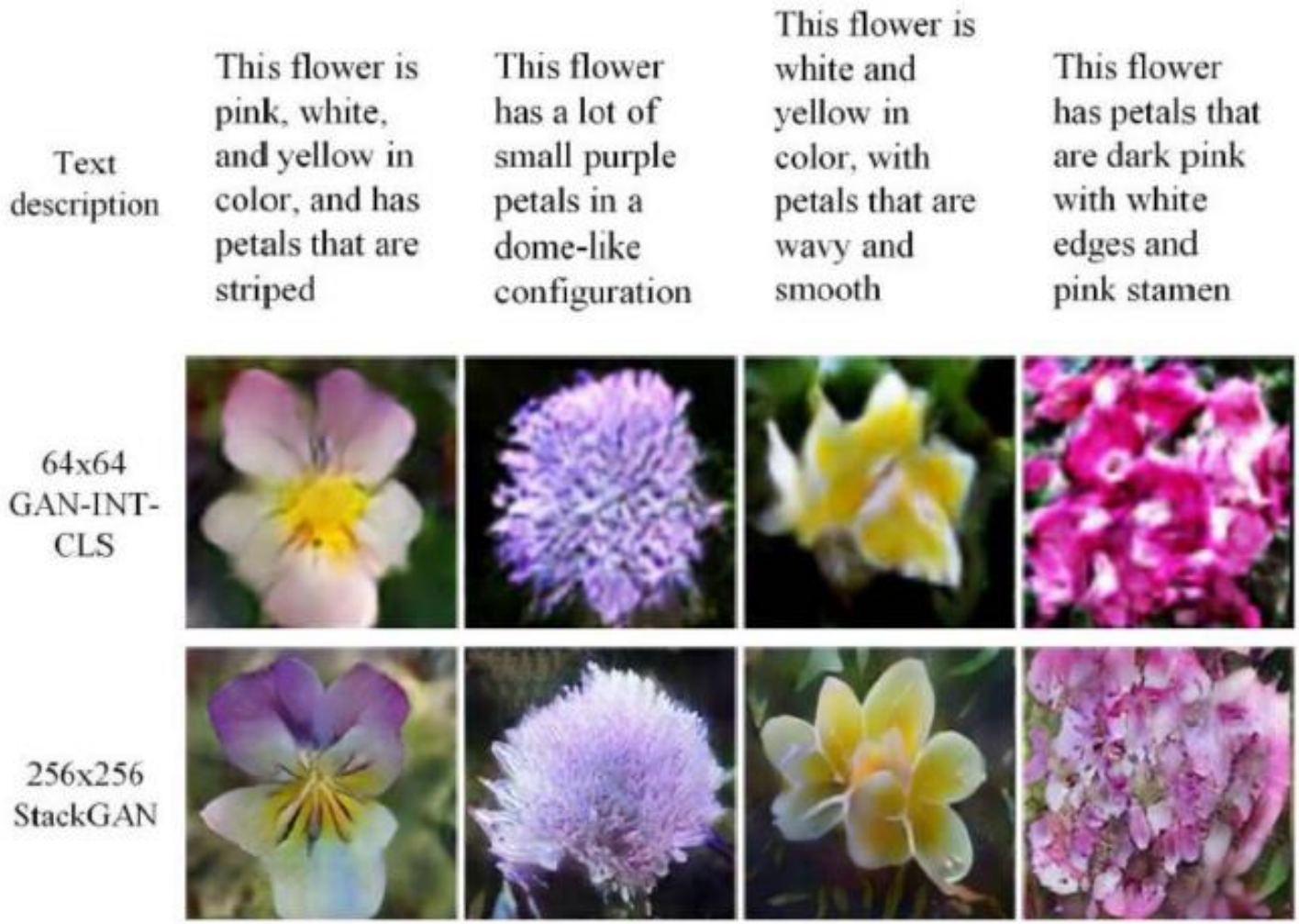


Image-to-Image Translation

- One-to-many or many-to-one mapping [Isola et al., 2016]

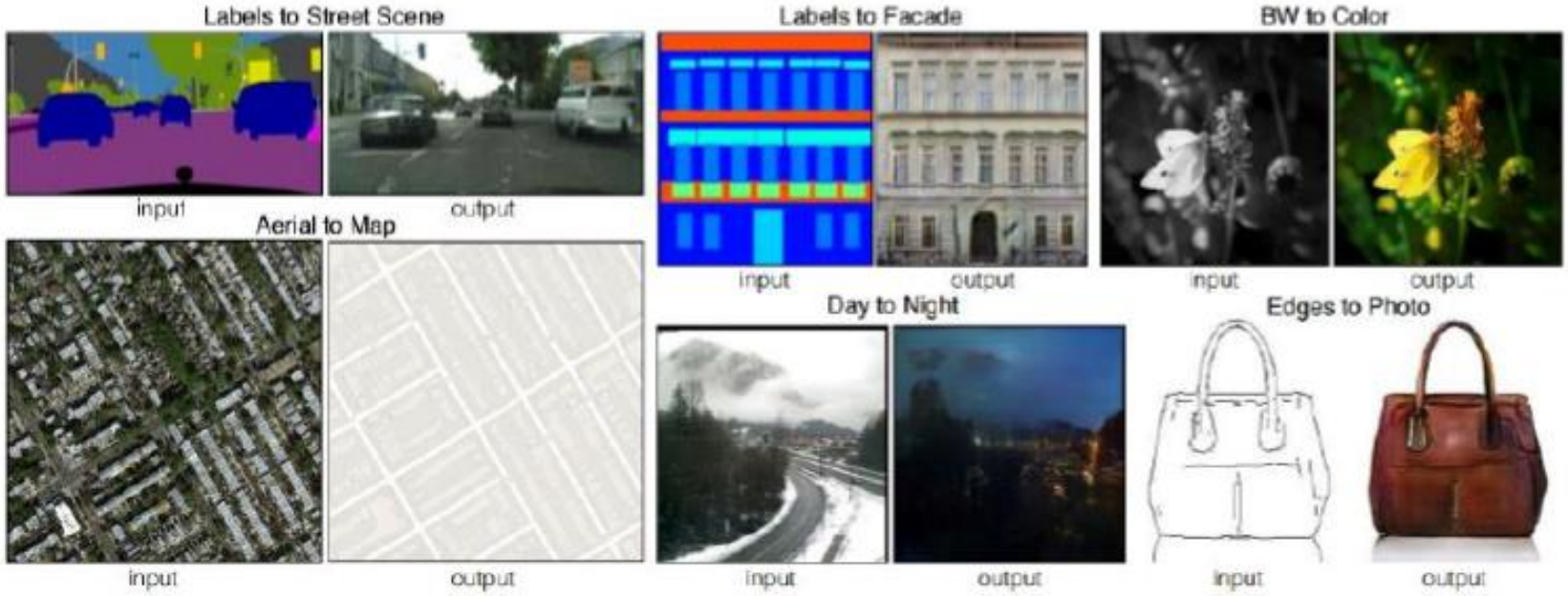
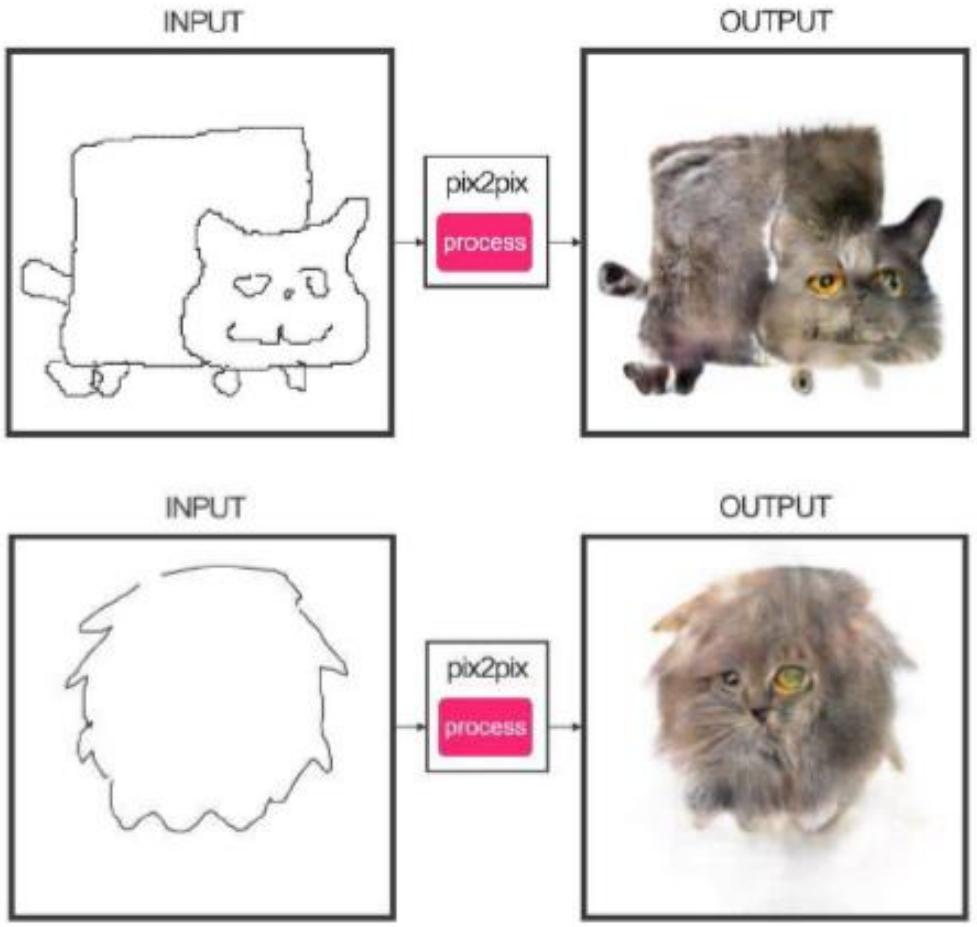


Image-to-Image Translation

- More results

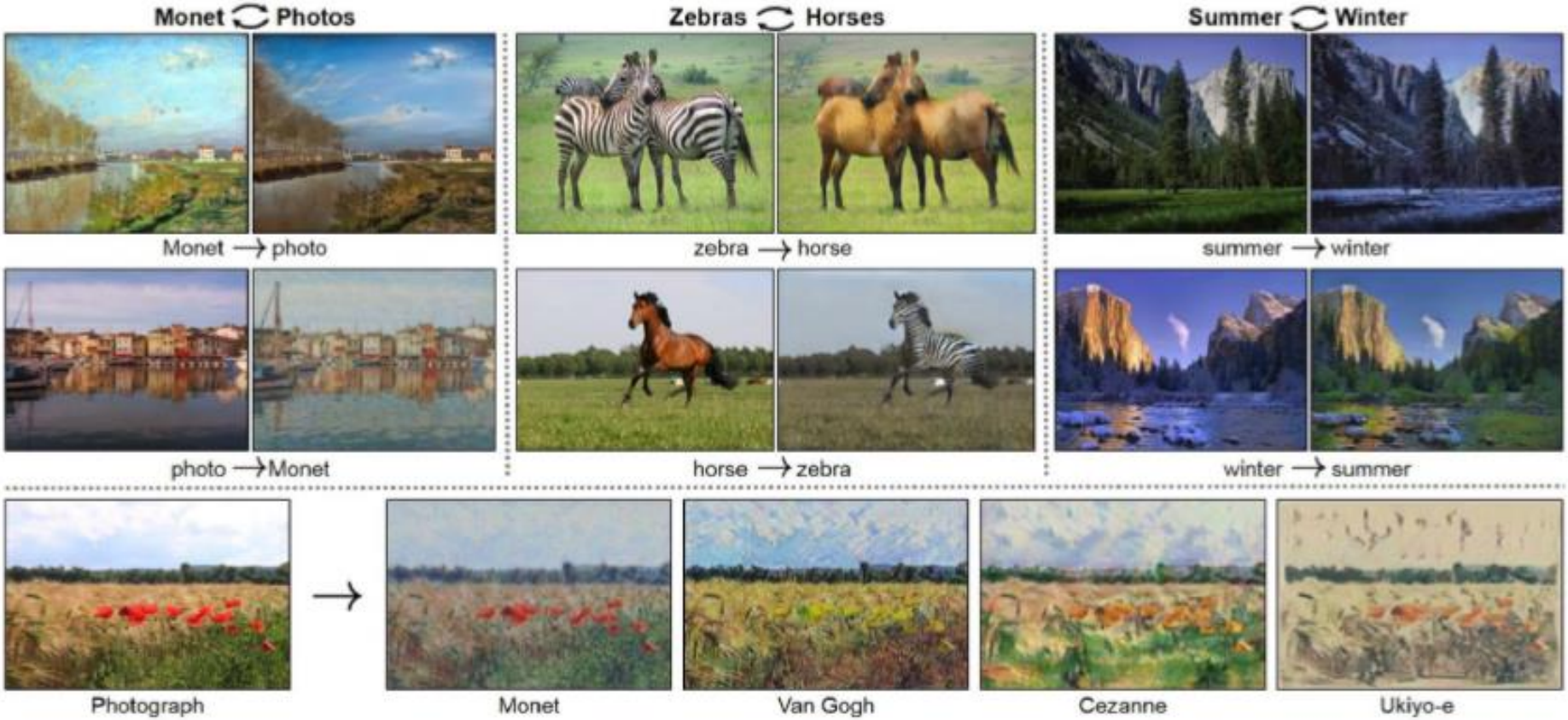
Image-to-Image Translation

- More results



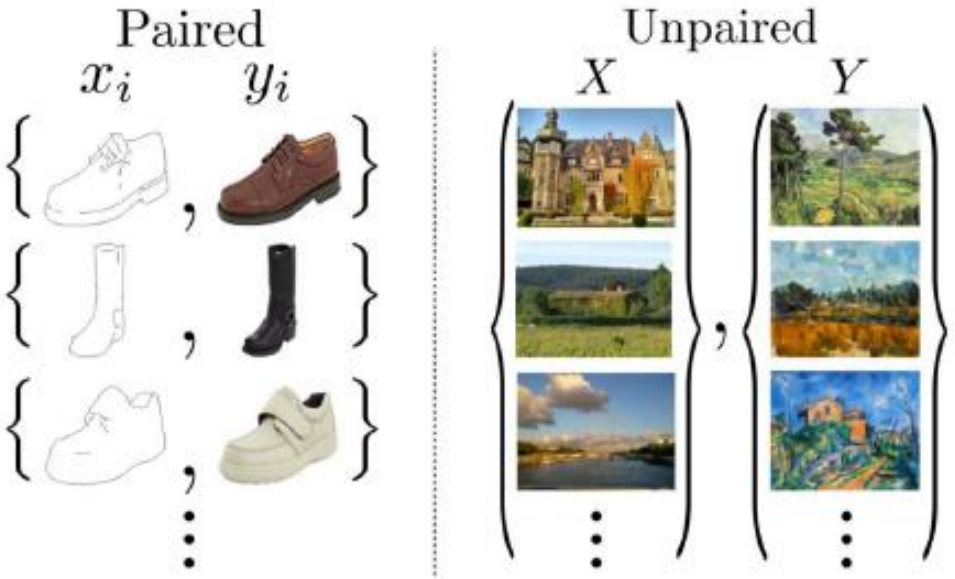
CycleGAN

- Image-to-image translation without paired data



CycleGAN

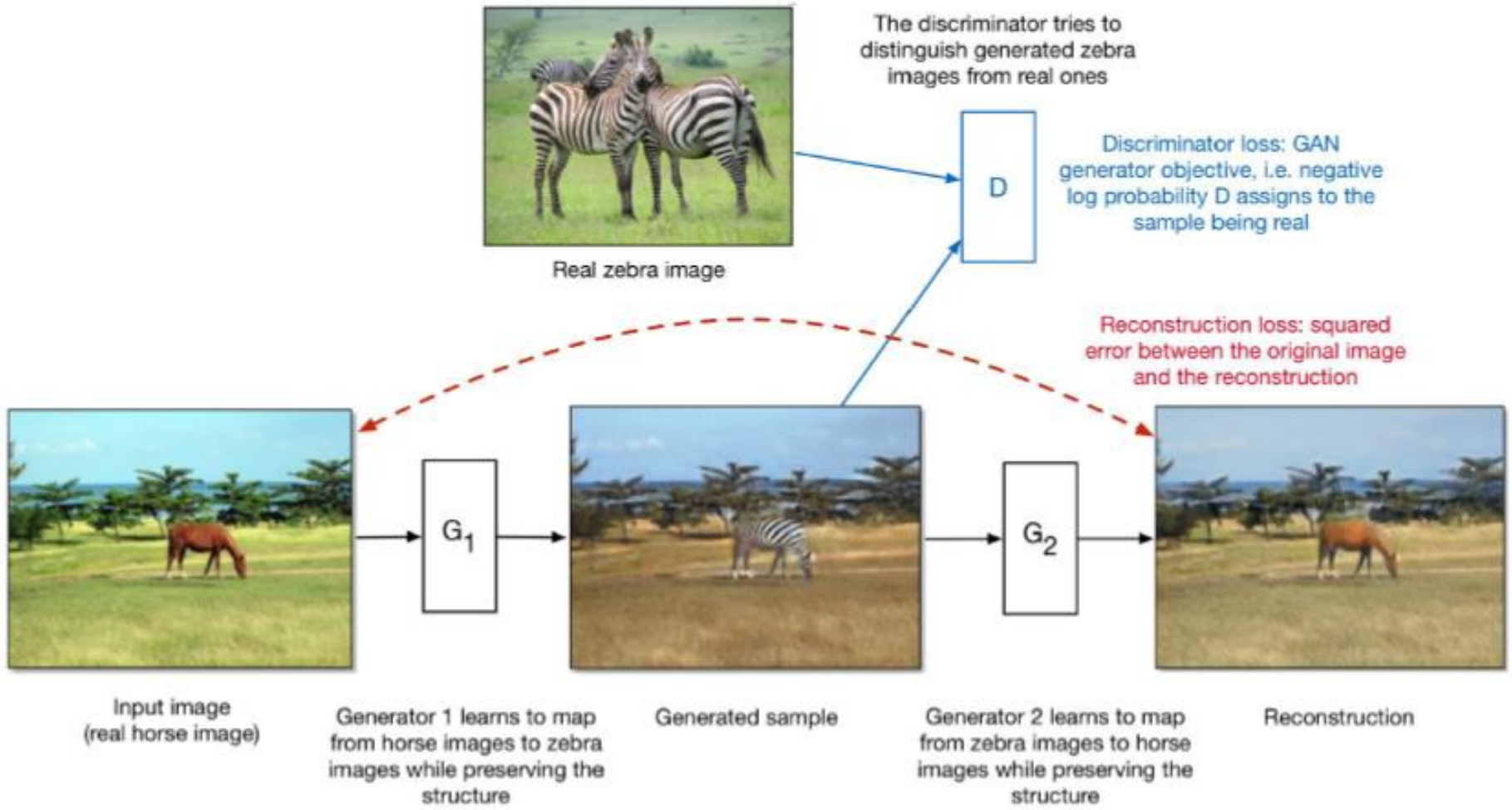
- If we had paired data (same content in both styles), this would be a supervised learning problem. But this is hard to find.



CycleGAN

- If we had paired data (same content in both styles), this would be a supervised learning problem. But this is hard to find.
- The CycleGAN architecture learns to do it from unpaired data.
  - Train two different generator nets to go from style 1 to style 2, and vice versa.
  - Make sure the generated samples of style 2 are indistinguishable from real images by a discriminator net.
  - Make sure the generators are **cycle-consistent**: mapping from style 1 to style 2 and back again should give you almost the original image.

CycleGAN



Total loss = discriminator loss + reconstruction loss

CycleGAN

- Results

- More details

https://hardikbansal.github.io/CycleGANBlog/