

《神经网络与深度学习》



深度生成模型 Deep Generative Models

<https://nndl.github.io/>

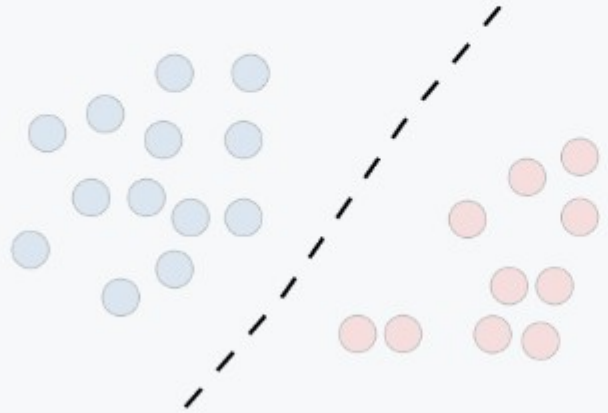
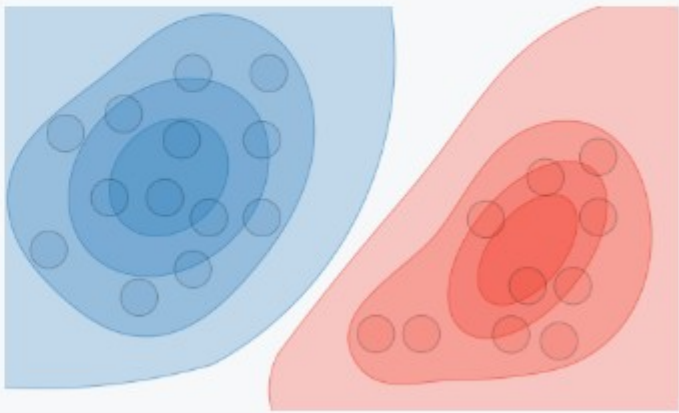
参考

- ▶ 《神经网络与深度学习》 - 深度生成模型
 - ▶ <https://nndl.github.io/>
- ▶ 一些例子来自于李宏毅 《Introduction of Generative Adversarial Network (GAN) 》



生成模型 Generative Models

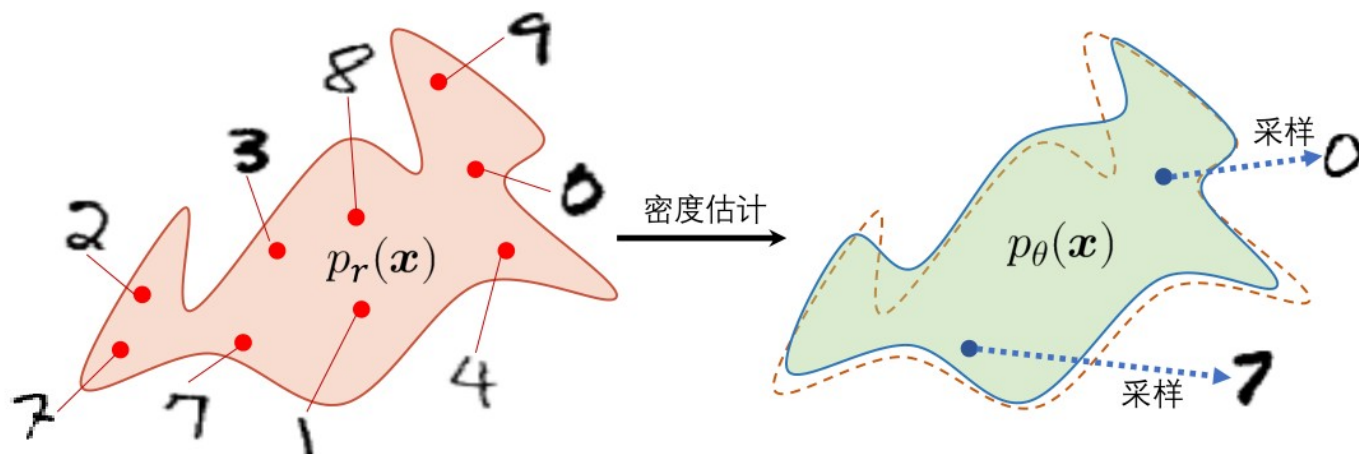
机器学习的两种范式

	Discriminative model	Generative model
Goal	Directly estimate $P(y x)$	Estimate $P(x y)$ to then deduce $P(y x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes

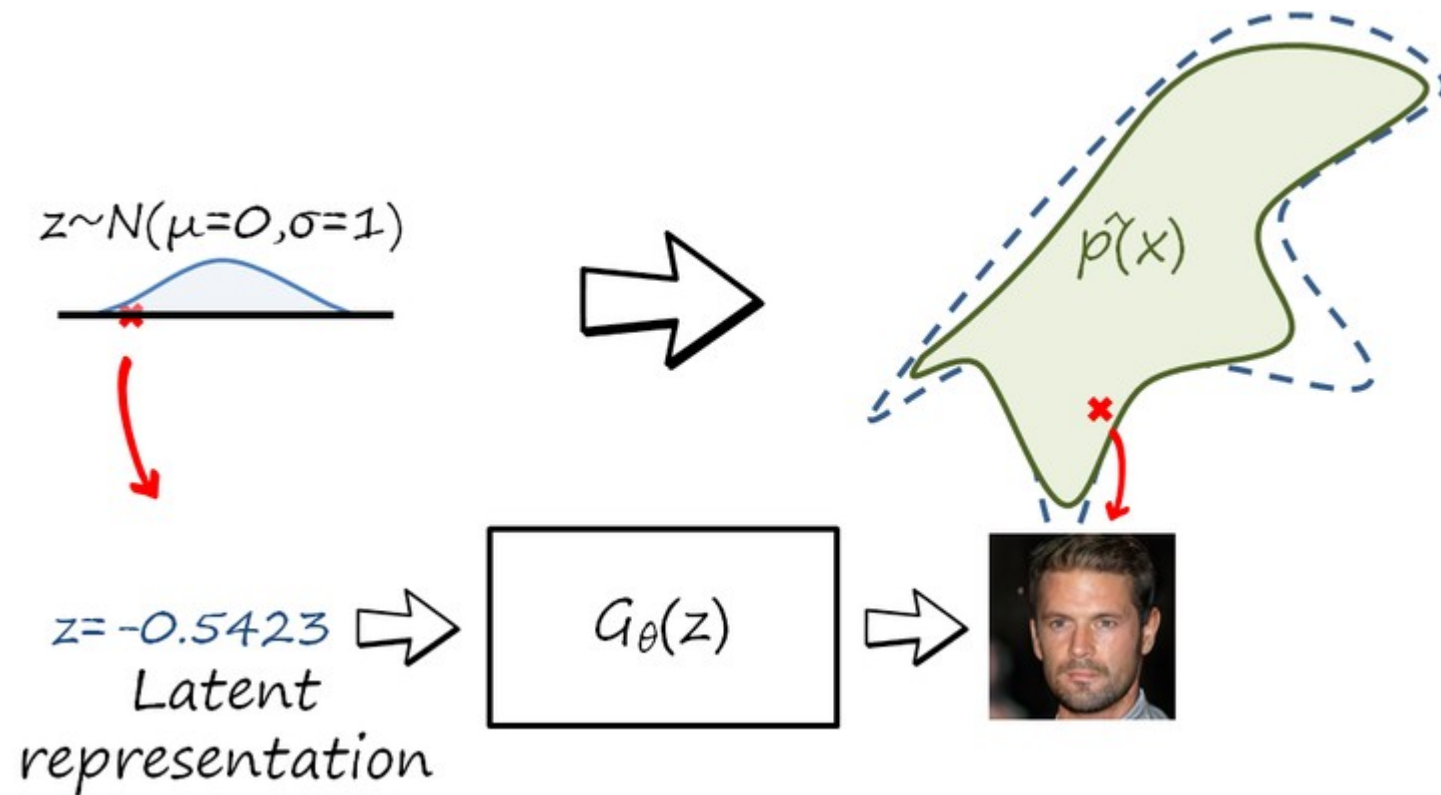
生成模型：一系列用于随机生成可观测数据的模型

▶ 生成模型包含两个步骤：

- ▶ 密度估计
- ▶ 采样



生成数据的另一种思路



深度生成模型

- ▶ 深度生成模型就是利用神经网络构建生成模型。
 - ▶ 变分自编码器（ Variational Autoencoder ， VAE ）
 - ▶ [Kingma and Welling, 2013, Rezende et al., 2014]
 - ▶ 生成对抗网络（ Generative Adversarial Network ， GAN ）
 - ▶ [Goodfellow et al., 2014]

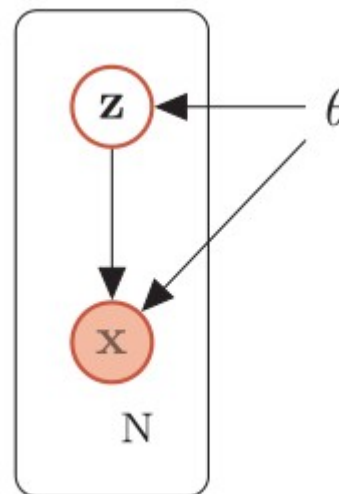


变分自编码器

Variational Autoencoder , VAE

概率生成模型

- ▶ 含隐变量的概率图模型
- ▶ 生成数据 x 的过程可以分为两步进行：
 - ▶ 根据隐变量的先验分布 $p(z;\theta)$ 采样得到样本 z ；
 - ▶ 根据条件分布 $p(x|z;\theta)$ 采样得到 x 。



EM 算法回顾

- 给定一个样本 $\mathbf{x}$ ，其对数边际似然 $\log p(\mathbf{x}|\theta)$ 可以分解为

$$\log p(\mathbf{x}|\theta) = \boxed{ELBO(q, \mathbf{x}|\theta, \phi)} + \boxed{D_{\text{KL}}(q(\mathbf{z}|\phi) \| p(\mathbf{z}|\mathbf{x}, \theta))}$$

$\mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\phi)} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\phi)} \right]$

M step E step

变分自编码器 (VAE)

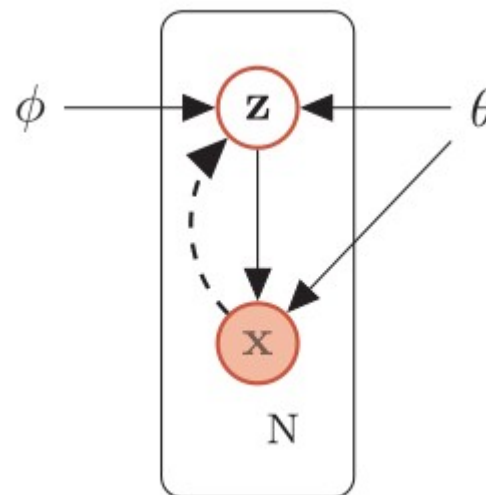
► 变分自编码器的模型结构可以分为两个部分：

► 寻找后验分布 $p(z|x;\theta)$ 的变分近似 $q(z|x;\phi)$ ；

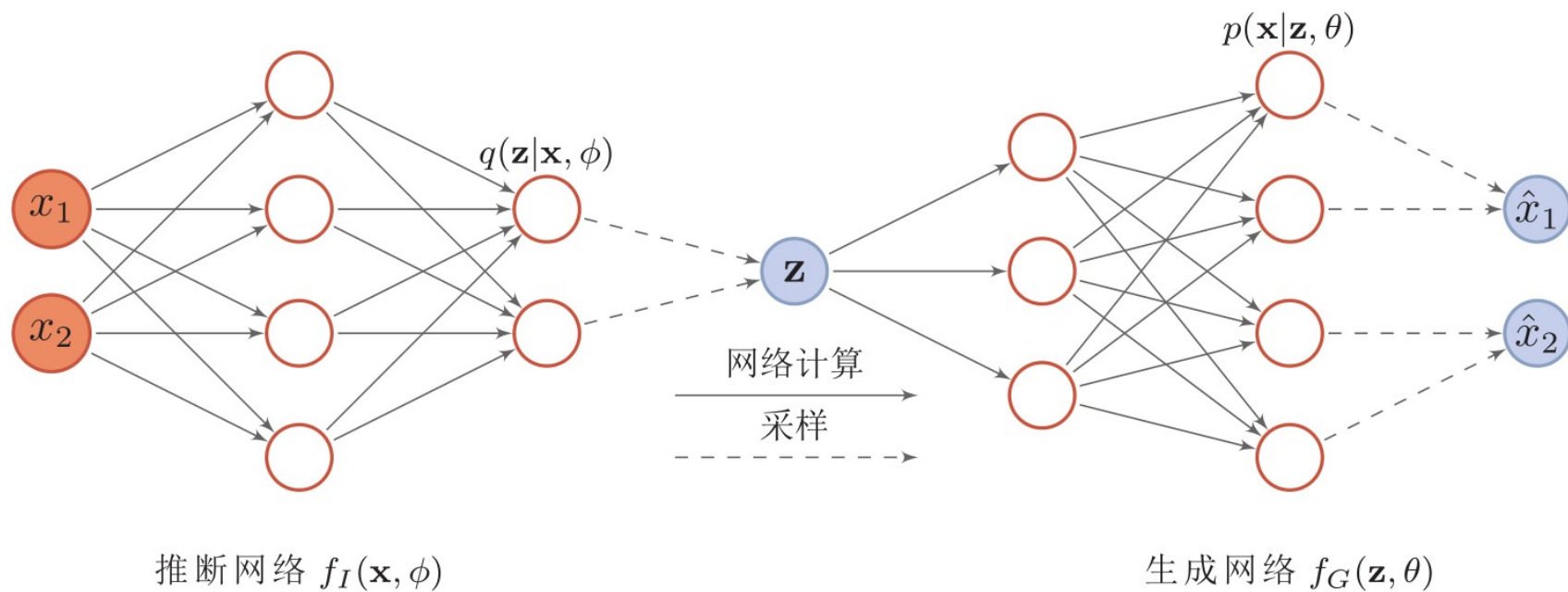
► 变分推断：用简单的分布 q 去近似复杂的分布 $p(z|x;\theta)$

► 在已知 $q(z|x;\phi)$ 的情况下，估计更好的分布 $p(x|z;\theta)$ 。

用神经网络来替代



变分自编码器



推断网络

► 推断网络

$$q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_I, \boldsymbol{\sigma}_I^2 I)$$

$$\begin{bmatrix} \boldsymbol{\mu}_I \\ \boldsymbol{\sigma}_I \end{bmatrix} = f_I(\mathbf{x}, \phi)$$

► 目标

$$\begin{aligned} \phi^* &= \arg \min_{\phi} D_{\text{KL}}(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z}|\mathbf{x}, \theta)) \\ &= \arg \min_{\phi} \log p(\mathbf{x}|\theta) - ELBO(q, \mathbf{x}|\theta, \phi) \\ &= \arg \max_{\phi} ELBO(q, \mathbf{x}|\theta, \phi). \end{aligned}$$

生成网络

▶ 先验分布 $p(z|\theta)$

- ▶ 一般假设隐变量 z 的先验分布为各向同性的标准高斯分布 $N(z|0, I)$

▶ 条件概率分布 $p(x|z, \theta)$

- ▶ 假设 $p(x|z, \theta)$ 服从对角化协方差的高斯分布

$$p(\mathbf{x}|\mathbf{z}, \theta) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_G, \boldsymbol{\sigma}_G^2 I)$$

▶ 目标

$$\theta^* = \arg \max_{\theta} ELBO(q, \mathbf{x}|\theta, \phi)$$

模型汇总

$$\max_{\theta, \phi} ELBO(q, \mathbf{x}|\theta, \phi) = \max_{\theta, \phi} \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\phi)} \left[\log \frac{p(\mathbf{x}|\mathbf{z}, \theta)p(\mathbf{z}|\theta)}{q(\mathbf{z}|\phi)} \right]$$



$$\max_{\theta, \phi} \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}, \phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \theta) \right] - D_{\text{KL}} \left(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z}|\theta) \right)$$

再参数化



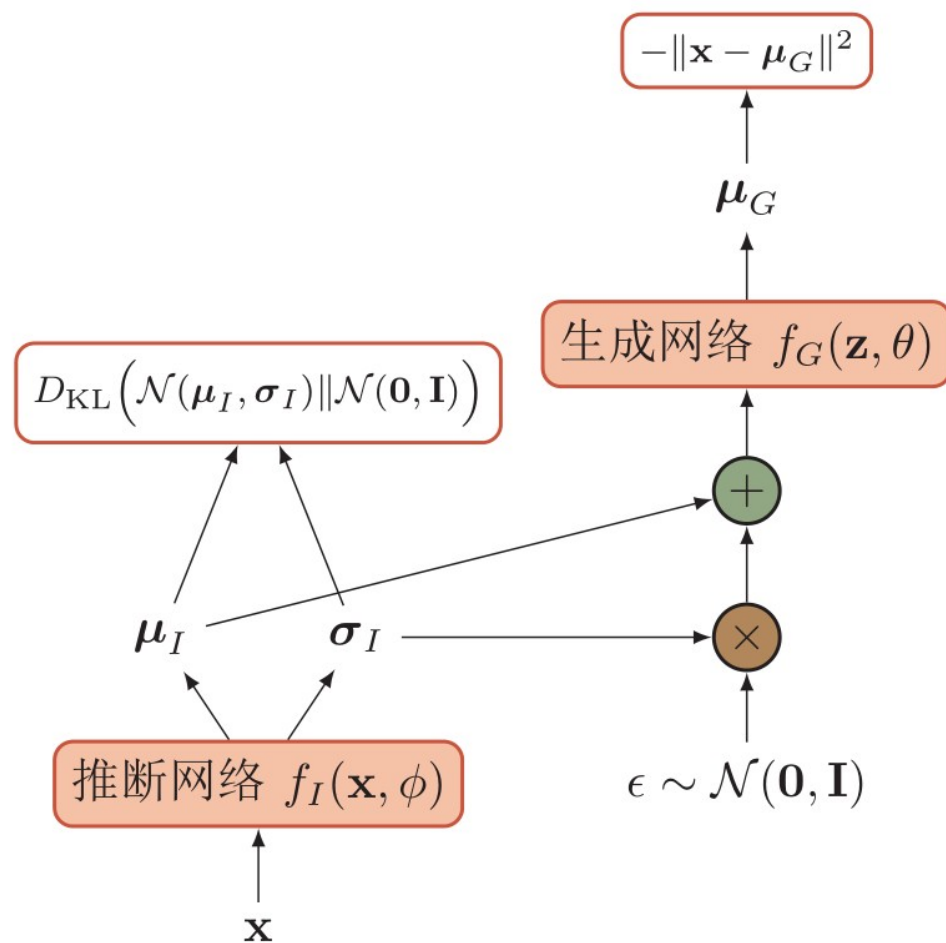
$$\sum_{n=1}^N \left(\frac{1}{M} \sum_{m=1}^M \log p(\mathbf{x}^{(n)} | \mathbf{z}^{(n,m)}, \theta) - D_{\text{KL}} \left(q(\mathbf{z}|\mathbf{x}^{(n)}, \phi) || \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \right) \right)$$

再参数化

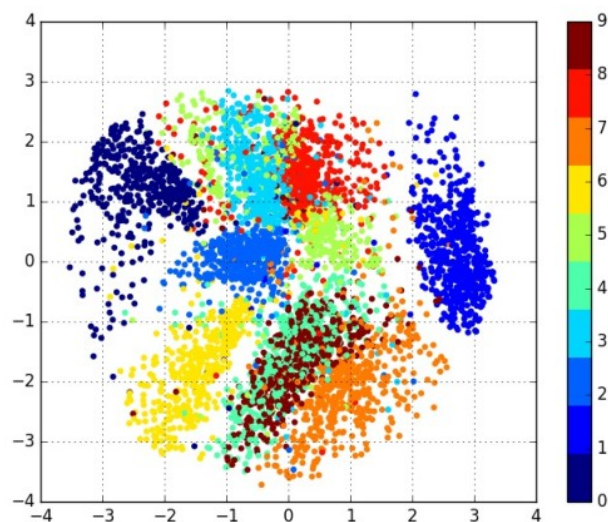
- ▶ 分布 $q(z|x, \phi)$ 依赖于参数 ϕ
- ▶ 再参数化（reparameterization）是实现通过随机变量实现反向传播的一种重要手段

$$z \sim N(\mu_I, \sigma_I^2 I) \quad \Rightarrow \quad \begin{aligned} \epsilon &\sim N(0, I) \\ z &= \mu_I + \sigma_I \odot \epsilon, \end{aligned}$$

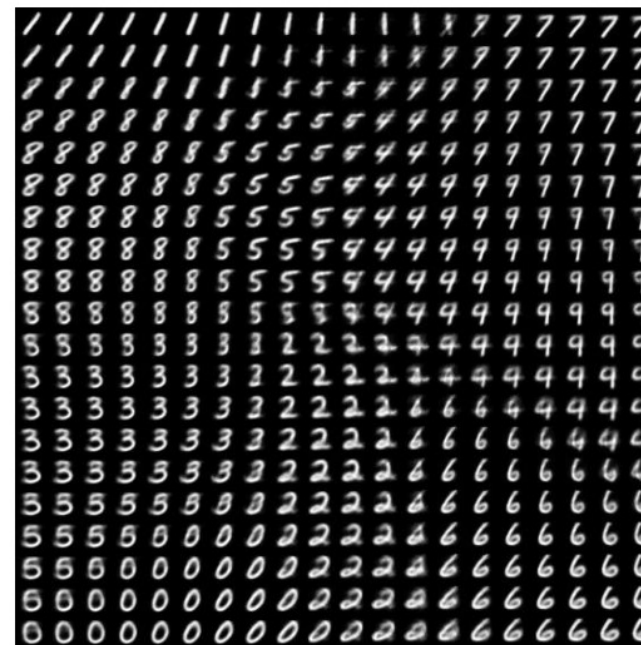
变分自编码器的训练过程



变分自编码器学习到的隐变量流形



(a) 训练集上所有样本在隐空间上的投影。



(b) 隐变量 z 在图像空间的投影。



生成对抗网络

Generative Adversarial Network (GAN)

显式密度模型和隐式密度模型

▶ 显式密度模型

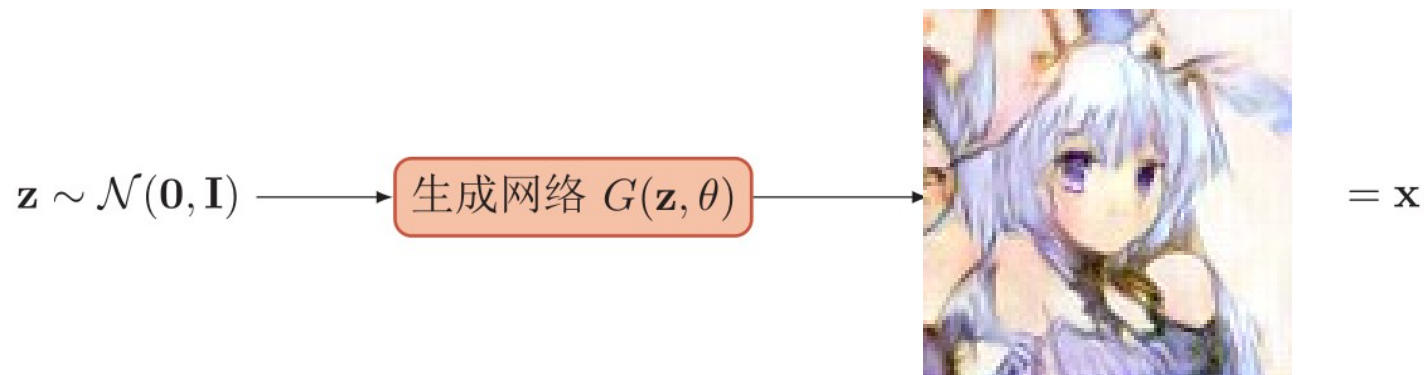
- ▶ 显式地构建出样本的密度函数 $p(x|\theta)$ ，并通过最大似然估计来求解参数；
- ▶ 变分自编码器、深度信念网络

▶ 隐式密度模型

- ▶ 不显式地估计出数据分布的密度函数
- ▶ 但能生成符合数据分布 $p_{\text{data}}(x)$ 的样本
- ▶ 无法用最大似然估计

生成网络

- 生成网络从隐空间（latent space）中随机采样作为输入，其输出结果需要尽量模仿训练集中的真实样本。



如何学习生成网络？

生成网络示例

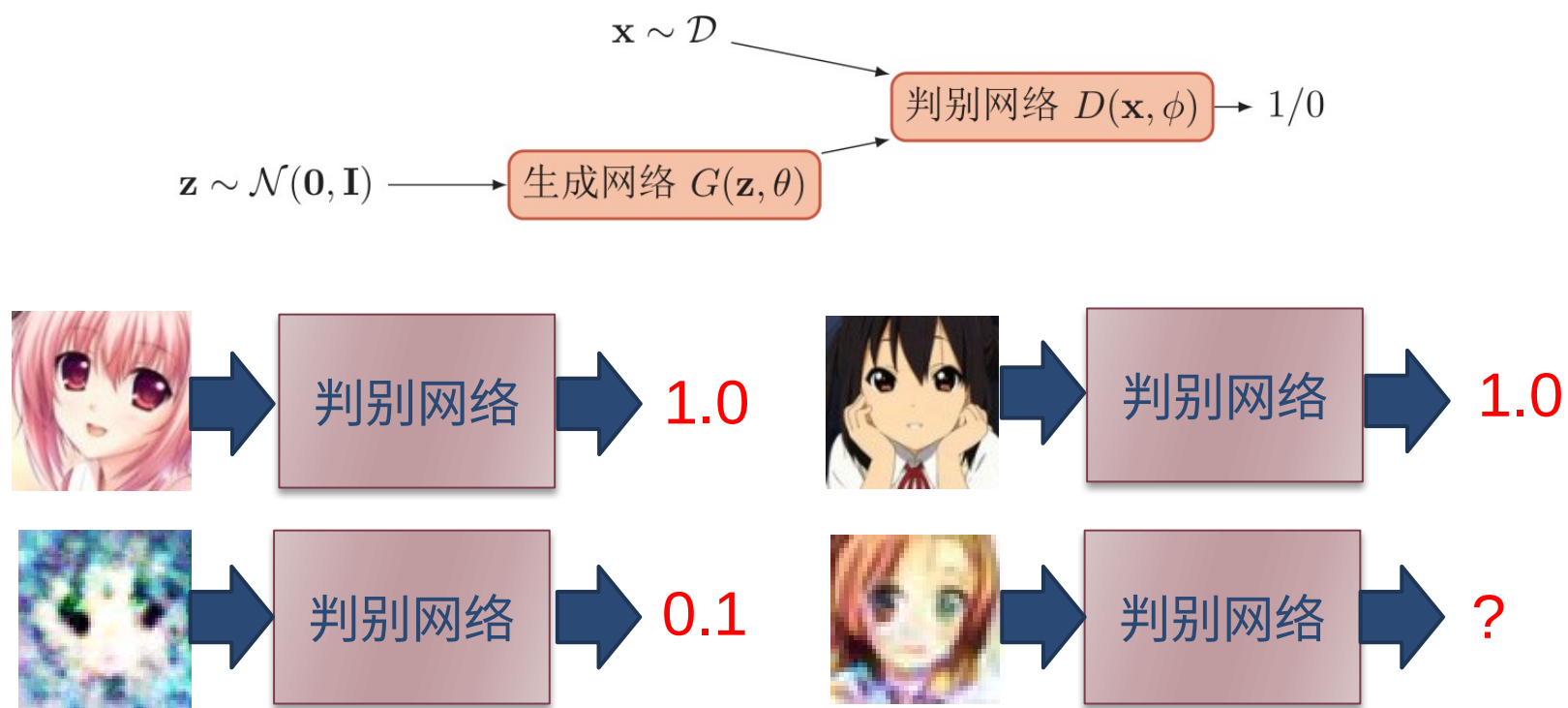


Each dimension of input vector represents some characteristics.



判别网络

- 判别网络的输入则为真实样本或生成网络的输出，其目的是将生成网络的输出从真实样本中尽可能分辨出来。



MinMax Game

▶ 对抗训练

- ▶ 生成网络要尽可能地欺骗判别网络。
 - ▶ 判别网络将生成网络生成的样本与真实样本中尽可能区分出来。
-
- ▶ 两个网络相互对抗、不断调整参数，最终目的是使判别网络无法判断生成网络的输出结果是否真实。

对抗过程

生成网络
(student)

判别网络
(teacher)



MinMax Game

► 判别网络

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

► 生成网络

$$\max_{\theta} \left(\mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \left[\log D(G(\mathbf{z}; \theta); \phi) \right] \right)$$

► Minimax Game

$$\min_{\theta} \max_{\phi} \left(\mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})} \left[\log D(\mathbf{x}; \phi) \right] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \left[\log(1 - D(G(\mathbf{z}; \theta); \phi)) \right] \right)$$

训练过程

算法 13.1 生成对抗网络的训练过程

输入: 训练集 $\mathcal{D}$, 对抗训练迭代次数 T , 每次判别网络的训练迭代次数 K , 小批量样本数量 M

1 随机初始化 θ, ϕ ;

2 **for** $t \leftarrow 1$ **to** T **do**

 // 训练判别网络 $D(\mathbf{x}; \phi)$

3 **for** $k \leftarrow 1$ **to** K **do**

 // 采集小批量训练样本

4 从训练集 $\mathcal{D}$ 中采集 M 个样本 $\{\mathbf{x}^{(m)}\}, 1 \leq m \leq M$;

5 从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 M 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

6 使用随机梯度上升更新 ϕ , 梯度为

$$\frac{\partial}{\partial \phi} \left[\frac{1}{M} \sum_{m=1}^M \left(\log D(\mathbf{x}^{(m)}; \phi) + \log (1 - D(G(\mathbf{z}^{(m)}; \theta); \phi)) \right) \right];$$

7 **end**

 // 训练生成网络 $G(\mathbf{z}; \theta)$

8 从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 M 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

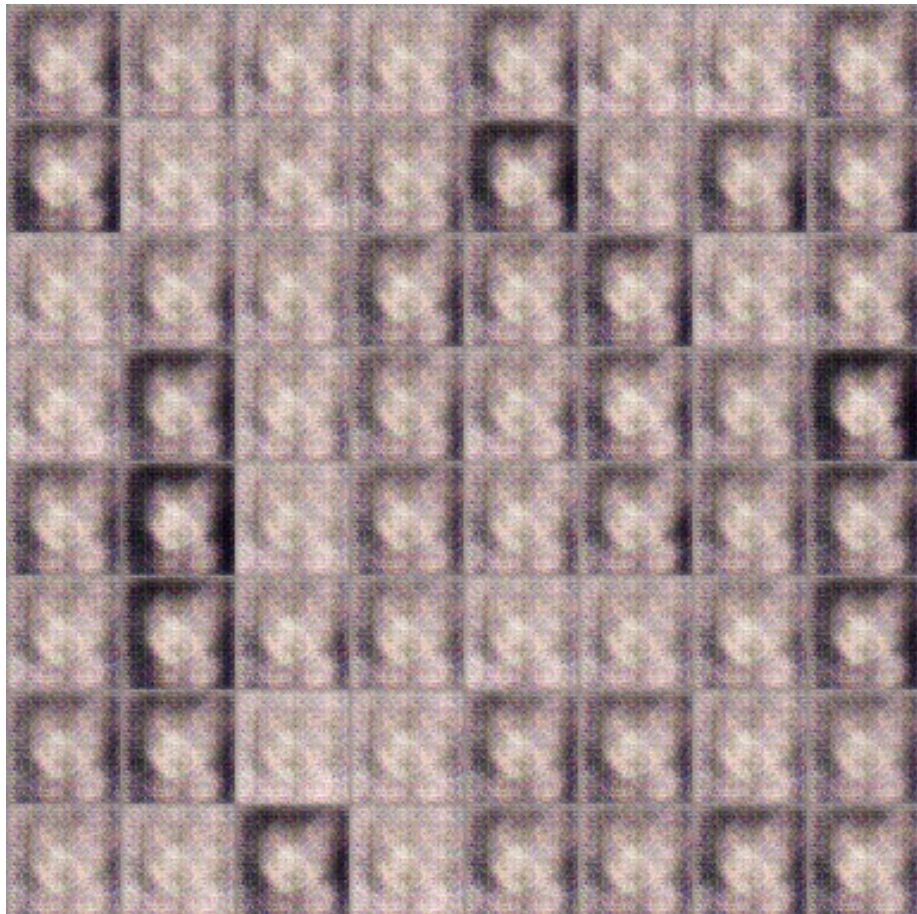
9 使用随机梯度上升更新 θ , 梯度为

$$\frac{\partial}{\partial \theta} \left[\frac{1}{M} \sum_{m=1}^M D(G(\mathbf{z}^{(m)}; \theta), \phi) \right];$$

10 **end**

输出: 生成网络 $G(\mathbf{z}; \theta)$

Anime Face Generation



100 updates



1000 updates

Anime Face Generation



2000 updates

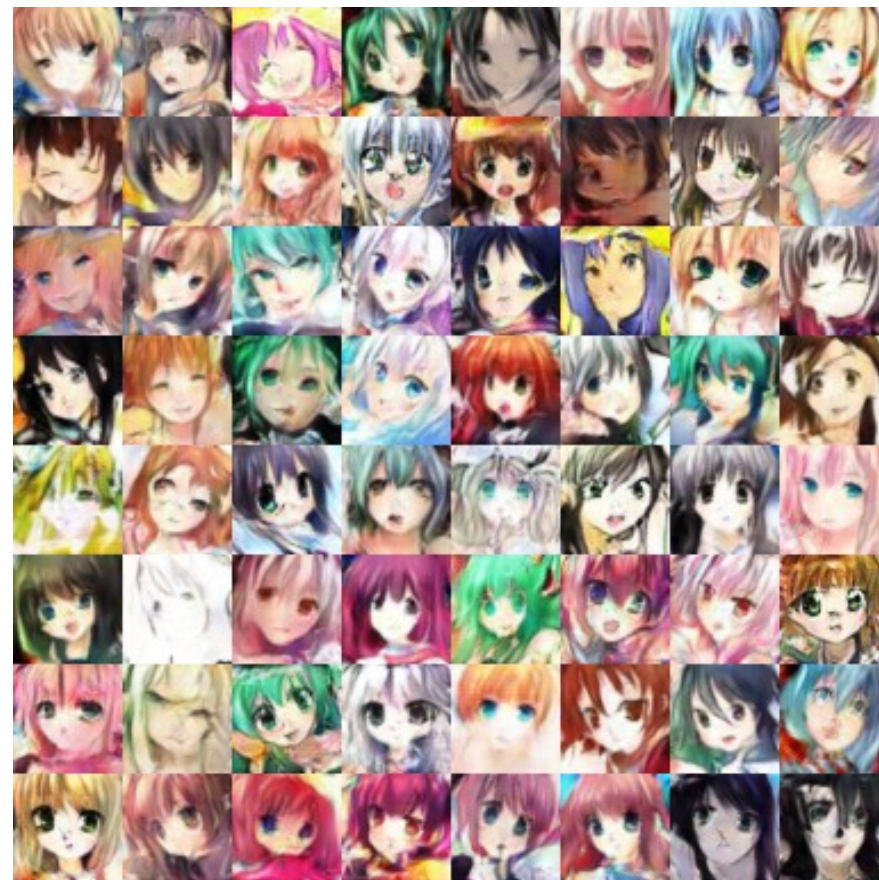


5000 updates

Anime Face Generation



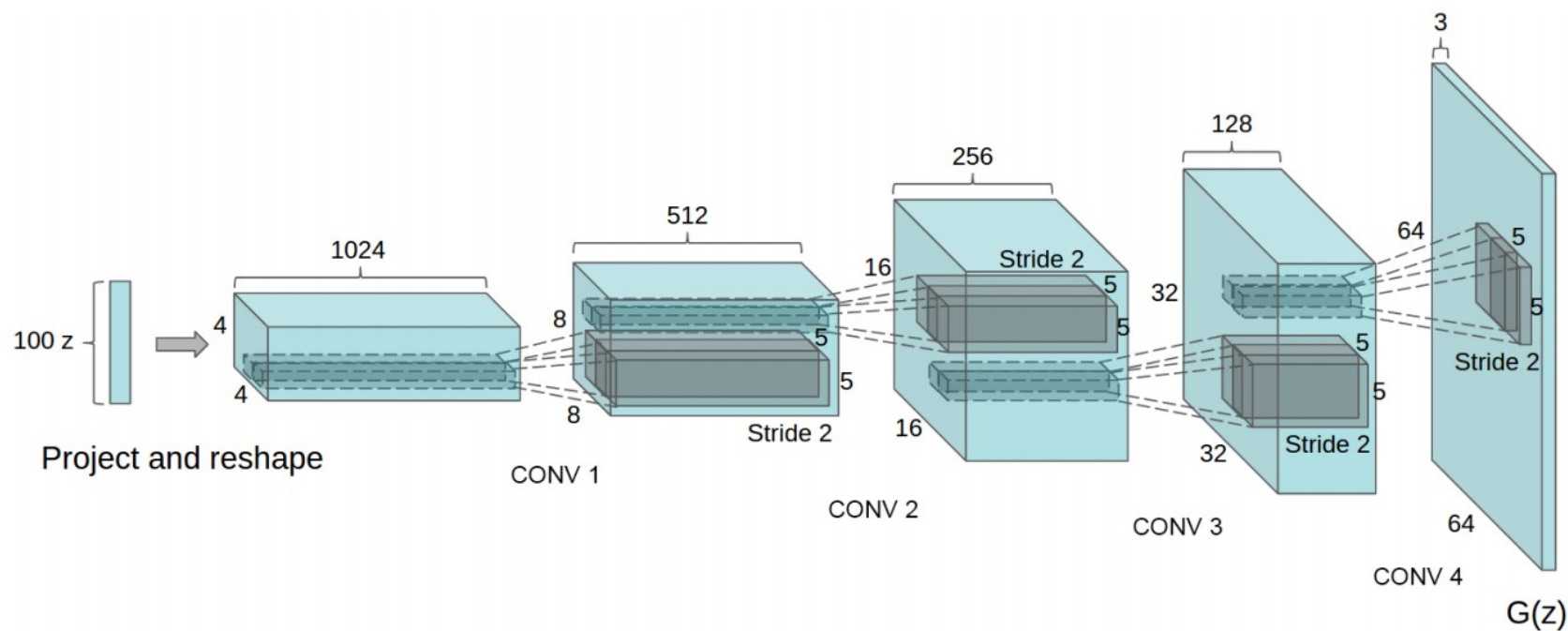
10,000
updates



50,000
updates

一个具体的模型： DCGANs

- ▶ 判别网络是一个传统的深度卷积网络，但使用了带步长的卷积来实现下采样操作，不用最大汇聚（pooling）操作。
- ▶ 生成网络使用一个特殊的深度卷积网络来实现使用微步卷积来生成 64×64 大小的图像。

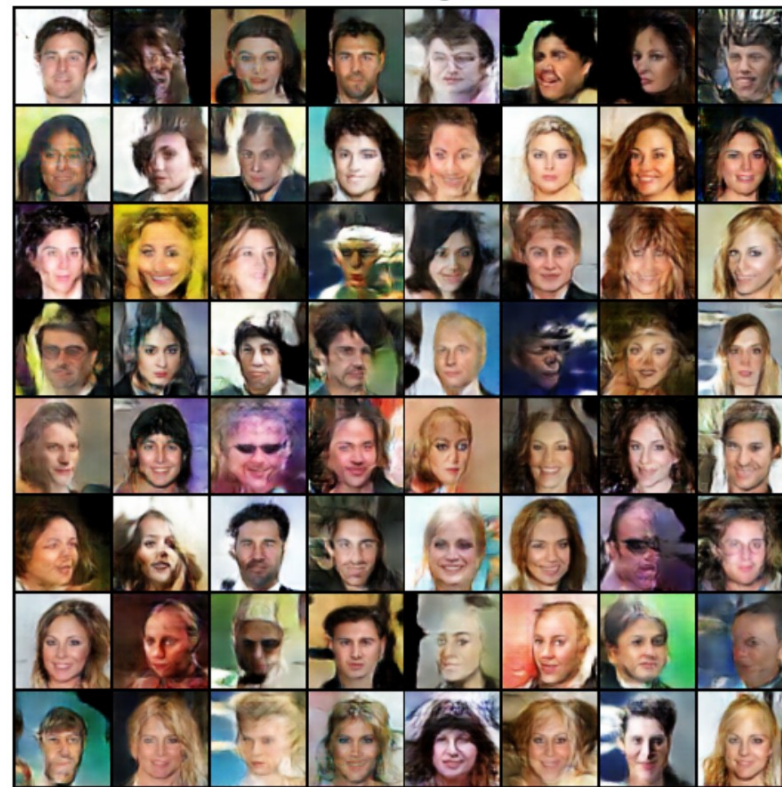


DCGANs

Real Images



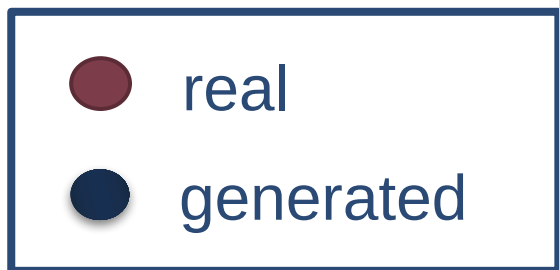
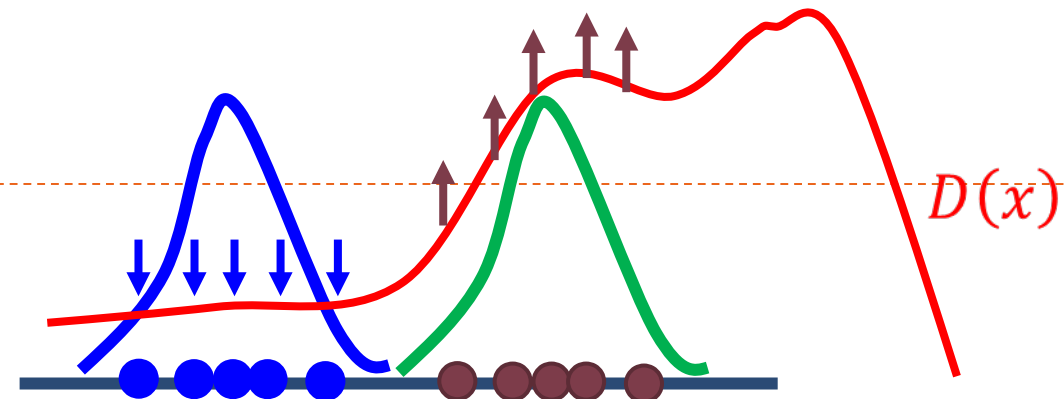
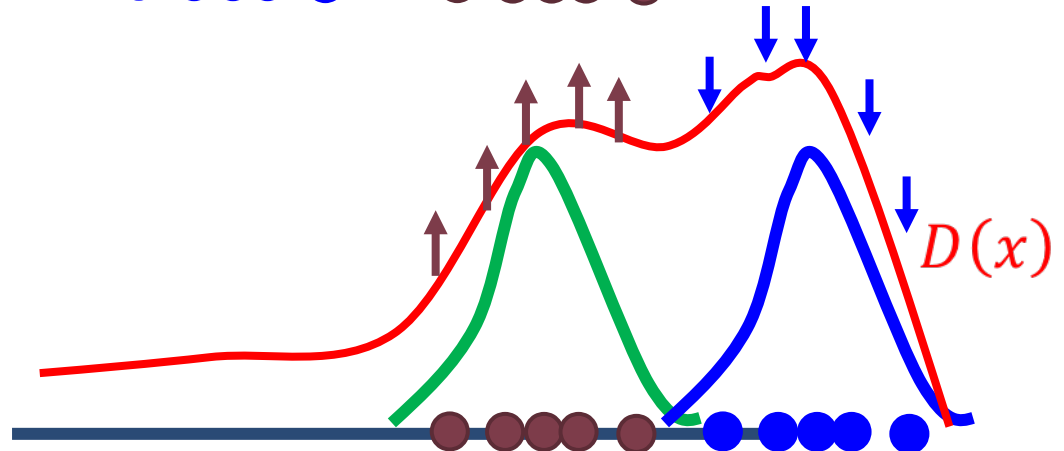
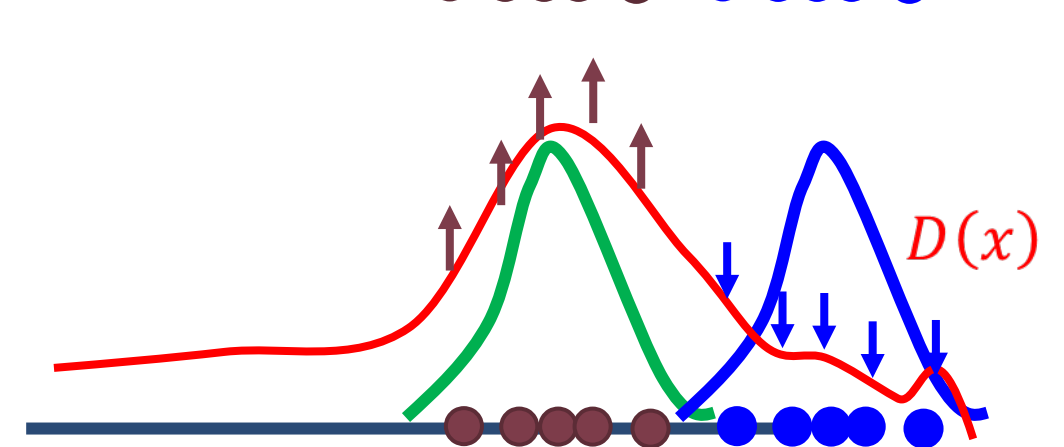
Fake Images



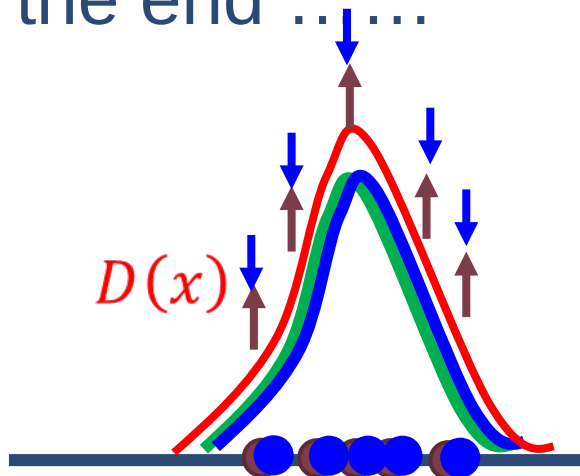


模型分析

数据分布



In the end



模型分析

► 假设 $p_r(x)$ 和 $p_\theta(x)$ 已知，则最优的判别器为

$$D^*(\mathbf{x}) = \frac{p_r(\mathbf{x})}{p_r(\mathbf{x}) + p_\theta(\mathbf{x})}$$

► 目标函数变为

$$\begin{aligned}\mathcal{L}(G|D^*) &= \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})} \left[\log D^*(\mathbf{x}) \right] + \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log(1 - D^*(\mathbf{x})) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})} \left[\log \frac{p_r(\mathbf{x})}{p_r(\mathbf{x}) + p_\theta(\mathbf{x})} \right] + \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x})}{p_r(\mathbf{x}) + p_\theta(\mathbf{x})} \right] \\ &= D_{\text{KL}}(p_r \| p_a) + D_{\text{KL}}(p_\theta \| p_a) - 2 \log 2 \\ &= 2D_{\text{JS}}(p_r \| p_\theta) - 2 \log 2,\end{aligned}$$

$$D_{\text{JS}}(p \| q) = \frac{1}{2} D_{\text{KL}}(p \| m) + \frac{1}{2} D_{\text{KL}}(q \| m)$$

$$m = \frac{1}{2}(p + q)$$

不稳定性：生成网络的梯度消失

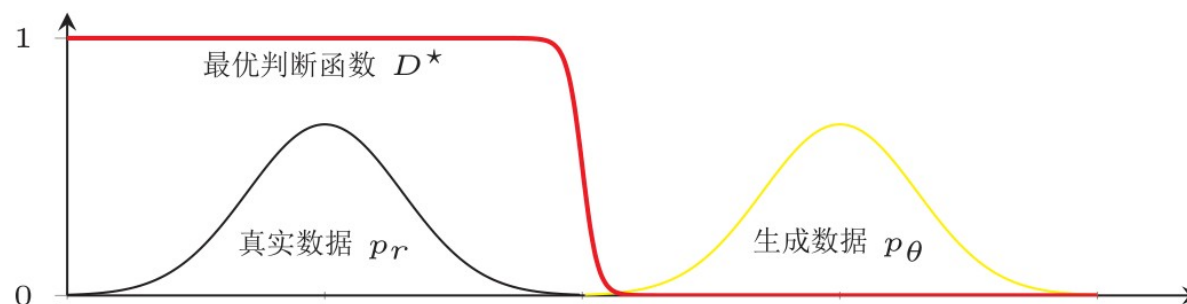
$$\min_{\theta} \max_{\phi} \left(\mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} \left[\log D(\mathbf{x}, \phi) \right] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \left[\log(1 - D(G(\mathbf{z}, \theta), \phi)) \right] \right)$$

在生成对抗网络中，当判断网络为最优时，生成网络的优化目标是最小化真实分布 $p_r(x)$ 和模型分布 $p_{\theta}(x)$ 之间的JS散度。

当两个分布相同时，JS散度为0，最优生成网络对应的损失为 $-2\log 2$ 。

使用JS散度来训练生成对抗网络的一个问题是当两个分布没有重叠时，它们之间的JS散度恒等于常数 $\log 2$ 。对生成网络来说，目标函数关于参数的梯度为0。

$$\frac{\partial \mathcal{L}(G|D^*)}{\partial \theta} = 0.$$



模型坍塌：生成网络的“错误”目标

► 生成网络的目标函数

$$\begin{aligned}\mathcal{L}'(G|D^*) &= \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log D^*(\mathbf{x}) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log \frac{p_r(\mathbf{x})}{p_r(\mathbf{x}) + p_\theta(\mathbf{x})} \cdot \frac{p_\theta(\mathbf{x})}{p_\theta(\mathbf{x})} \right] \\ &= -\mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x})}{p_r(\mathbf{x})} \right] + \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x})}{p_r(\mathbf{x}) + p_\theta(\mathbf{x})} \right] \\ &= -D_{\text{KL}}(p_\theta \| p_r) + \mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x})} \left[\log (1 - D^*(\mathbf{x})) \right] \\ &= -D_{\text{KL}}(p_\theta \| p_r) + 2D_{\text{JS}}(p_r \| p_\theta) - 2\log 2 - \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})} \left[\log D^*(\mathbf{x}) \right]\end{aligned}$$

► 其中后两项和生成网络无关，因此

$$\arg \max_{\theta} \mathcal{L}'(G|D^*) = \arg \min_{\theta} D_{\text{KL}}(p_\theta \| p_r) - 2D_{\text{JS}}(p_r \| p_\theta)$$

前向和逆向 KL 散度

▶ KL散度是一种非对称的散度，在计算真实分布 $p_r(x)$ 和模型分布 $p_\theta(x)$ 之间的KL散度时，按照顺序不同，有两种KL散度：

前向 KL 散度

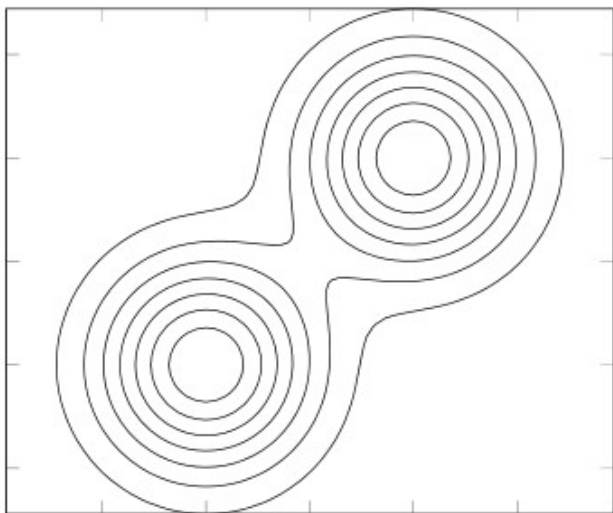
$$D_{\text{KL}}(p_r \| p_\theta) = \int p_r(\mathbf{x}) \log \frac{p_r(\mathbf{x})}{p_\theta(\mathbf{x})} d\mathbf{x}$$

逆向 KL 散度

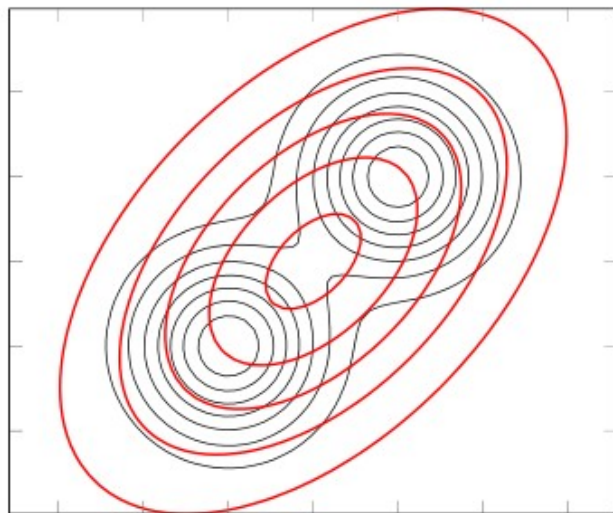
$$D_{\text{KL}}(p_\theta \| p_r) = \int p_\theta(\mathbf{x}) \log \frac{p_\theta(\mathbf{x})}{p_r(\mathbf{x})} d\mathbf{x}$$

前向和逆向 KL 散度

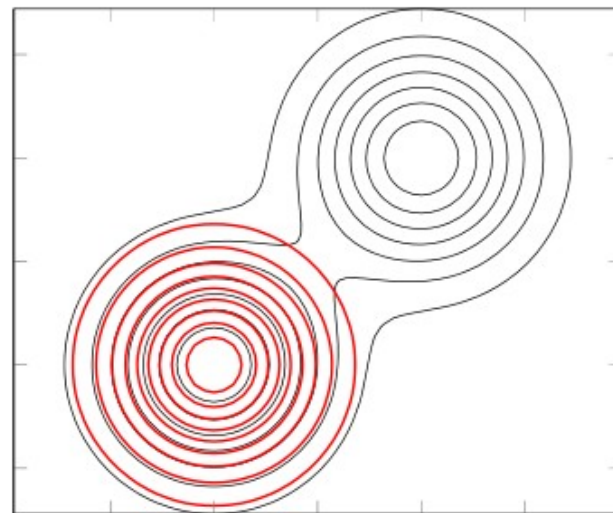
真实分布 p_r



前向 KL 散度 $D_{KL}(p_r||p_\theta)$



逆向 KL 散度 $D_{KL}(p_\theta||p_r)$

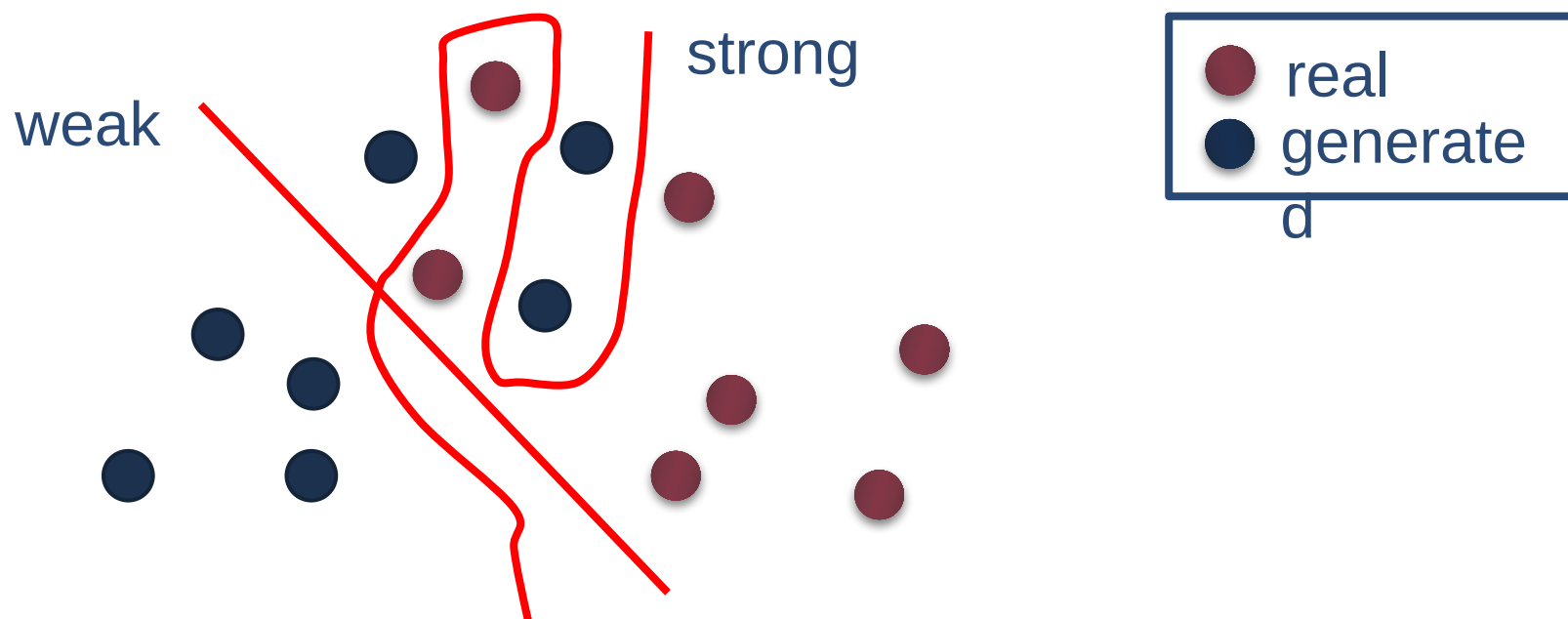


$$D_{KL}(p_r||p_\theta) = \int p_r(\mathbf{x}) \log \frac{p_r(\mathbf{x})}{p_\theta(\mathbf{x})} d\mathbf{x}$$

$$D_{KL}(p_\theta||p_r) = \int p_\theta(\mathbf{x}) \log \frac{p_\theta(\mathbf{x})}{p_r(\mathbf{x})} d\mathbf{x}$$

改进

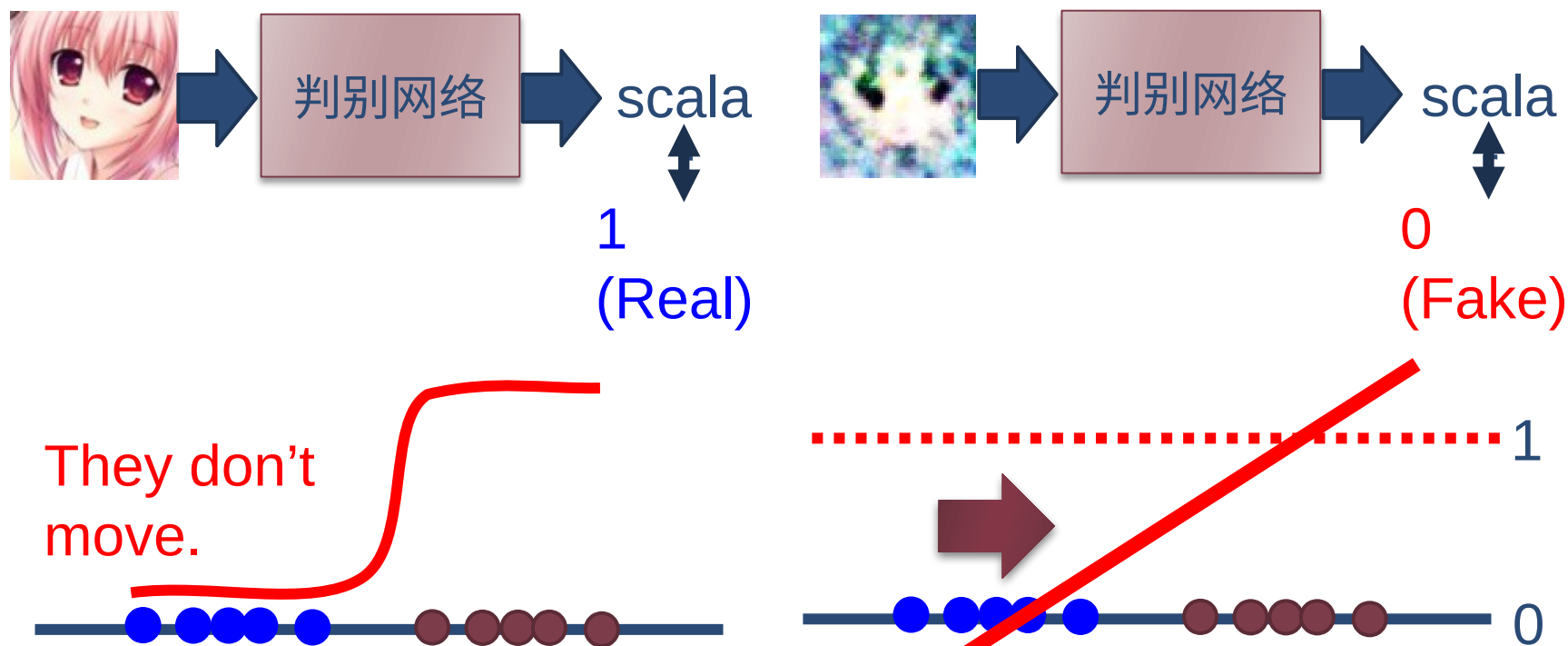
- ▶ 弱化判别器
- ▶ 使用更好的损失函数



Least Square GAN (LSGAN)



- Replace sigmoid with linear (replace classification with regression)



f-GAN

► f-divergences $D_f(P||Q)$

Name	$D_f(P Q)$	Generator $f(u)$	$T^*(x)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} dx$	$u \log u$	$1 + \log \frac{p(x)}{q(x)}$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} dx$	$-\log u$	$-\frac{q(x)}{p(x)}$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} dx$	$(u-1)^2$	$2(\frac{p(x)}{q(x)} - 1)$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)} \right)^2 dx$	$(\sqrt{u} - 1)^2$	$(\sqrt{\frac{p(x)}{q(x)}} - 1) \cdot \sqrt{\frac{q(x)}{p(x)}}$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx$	$-(u+1) \log \frac{1+u}{2} + u \log u$	$\log \frac{2p(x)}{p(x)+q(x)}$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx - \log(4)$	$u \log u - (u+1) \log(u+1)$	$\log \frac{p(x)}{p(x)+q(x)}$

f-GAN

► 目标函数

$$F(\theta, \omega) = \mathbb{E}_{x \sim P} [T_\omega(x)] - \mathbb{E}_{x \sim Q_\theta} [f^*(T_\omega(x))]$$

► f-divergences

Name	Output activation g_f	dom_{f^*}	Conjugate $f^*(t)$	$f'(1)$
Kullback-Leibler (KL)	v	$\mathbb{R}$	$\exp(t - 1)$	1
Reverse KL	$-\exp(-v)$	$\mathbb{R}_-$	$-1 - \log(-t)$	-1
Pearson χ^2	v	$\mathbb{R}$	$\frac{1}{4}t^2 + t$	0
Squared Hellinger	$1 - \exp(-v)$	$t < 1$	$\frac{t}{1-t}$	0
Jensen-Shannon	$\log(2) - \log(1 + \exp(-v))$	$t < \log(2)$	$-\log(2 - \exp(t))$	0
GAN	$-\log(1 + \exp(-v))$	$\mathbb{R}_-$	$-\log(1 - \exp(t))$	$-\log(2)$



Wasserstein GAN

Wasserstein 距离

▶ Wasserstein距离用于衡量两个分布之间的距离。

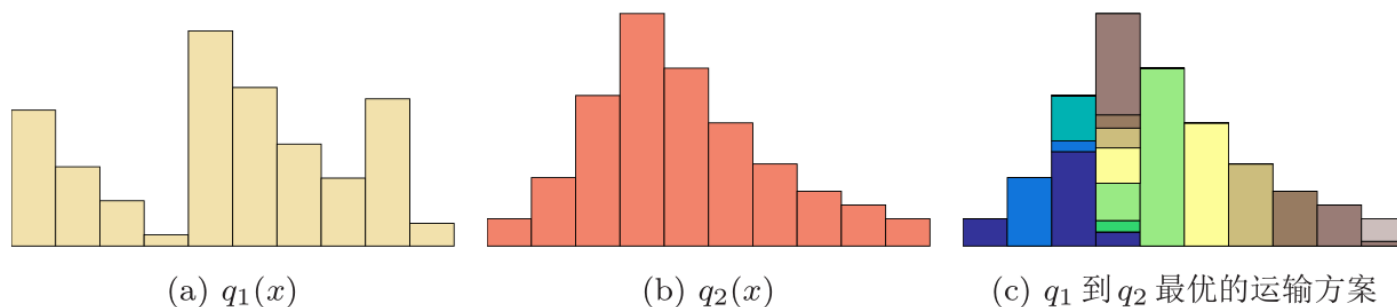
$$W_p(q_1, q_2) = \left(\inf_{\gamma(x,y) \in \Gamma(q_1, q_2)} \mathbb{E}_{(x,y) \sim \gamma(x,y)} [d(x,y)^p] \right)^{\frac{1}{p}}$$

- ▶ 其中 $\Gamma(q_1, q_2)$ 是边际分布为 q_1, q_2 的所有可能的联合分布集合， $d(x,y)$ 为 x 和 y 的距离，比如 ℓ_p 距离等。
- ▶ Wasserstein距离相比KL散度和JS散度的优势在于：即使两个分布没有重叠或者重叠非常少，Wasserstein距离仍然能反映两个分布的远近。

Wasserstein 距离

► 如果将两个分布看作是两个土堆，联合分布 $\gamma(x,y)$ 看作是从土堆 q_1 的位置 x 到土堆 q_2 的位置 y 的搬运土的数量。

Wasserstein距离可以理解为搬运土堆的最小工作量，也称为推土机距离（Earth-Mover's Distance, EMD）。



Kantorovich- Rubinstein 对偶定理

This formula is highly intractable

$$\text{EMD}(P_r, P_\theta) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \sum_{x, y} \|x - y\| \gamma(x, y) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \mathbb{E}_{(x, y) \sim \gamma} \|x - y\|$$

Kantorovich-Rubinstein Duality



$$\text{EMD}(P_r, P_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_r} f(x) - \mathbb{E}_{x \sim P_\theta} f(x).$$

1-Lipschitz Constraint

Lipschitz 连续函数

数学小知识 | Lipschitz 连续函数

在数学中，对于一个实数函数 $f: \mathbb{R} \rightarrow \mathbb{R}$ ，如果满足函数曲线上任意两点连线的斜率一致有界，即任意两点的斜率都小于常数 $K > 0$ ，

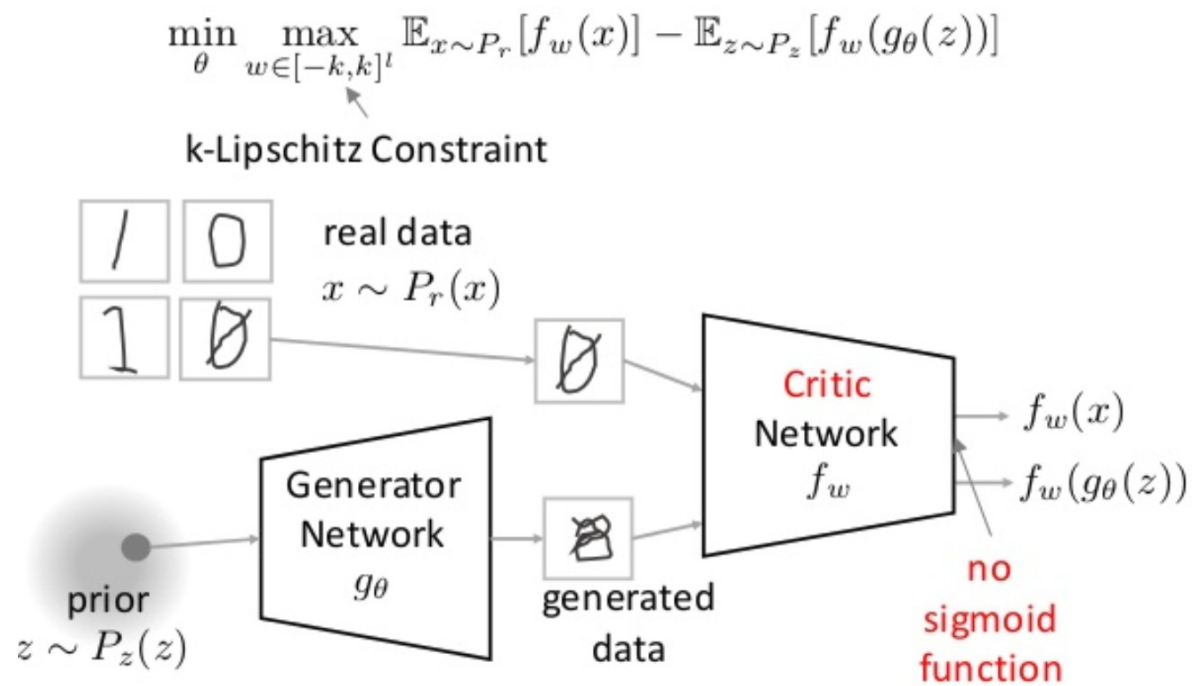
$$|f(x_1) - f(x_2)| \leq K|x_1 - x_2|, \quad (13.54)$$

则函数 f 就称为 K -Lipschitz 连续函数， K 称为 Lipschitz 常数。

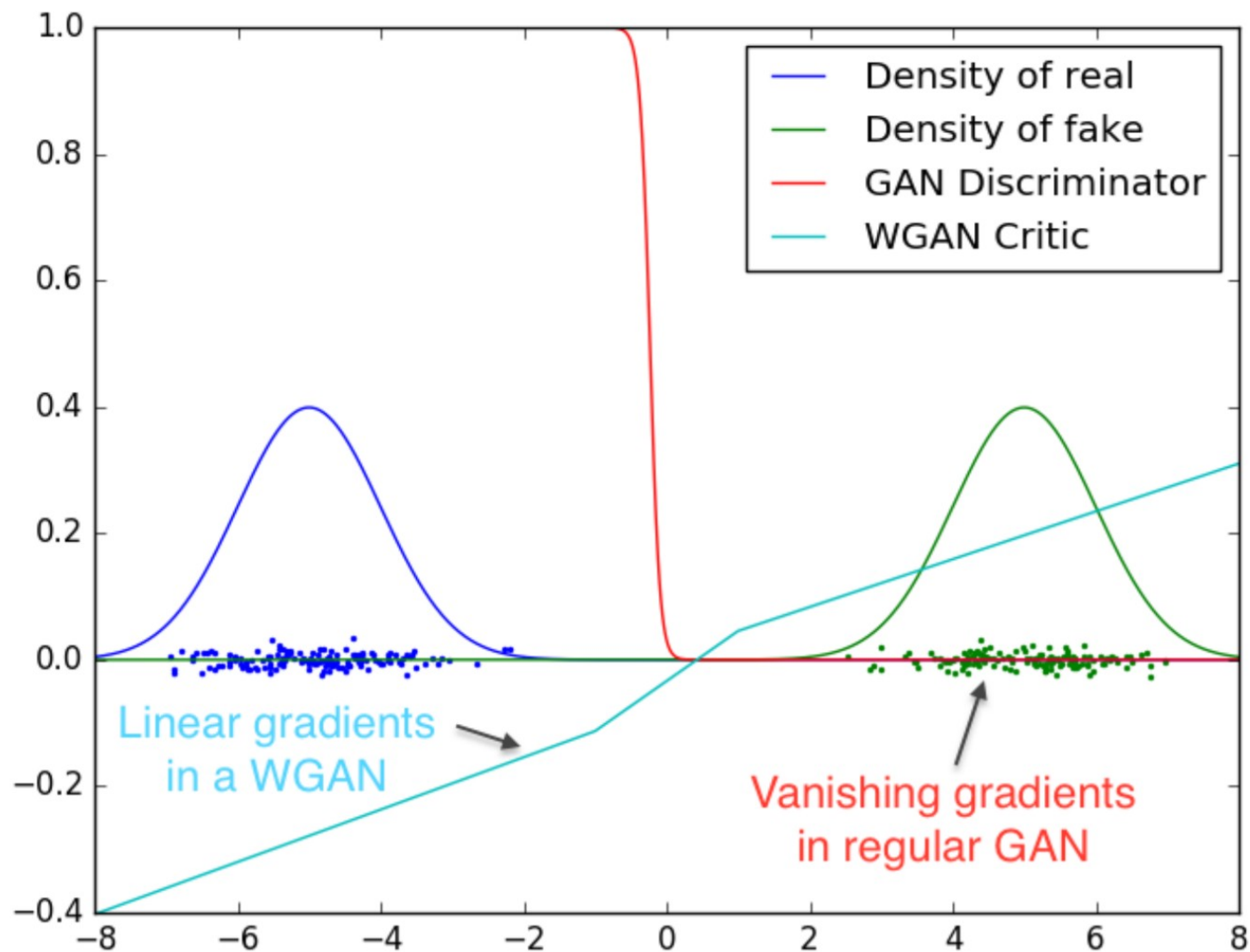
Lipschitz 连续要求函数在无限的区间上不能有超过线性的增长。如果一个函数可导，并满足 Lipschitz 连续，那么导数有界。如果一个函数可导，并且导数有界，那么函数为 Lipschitz 连续。

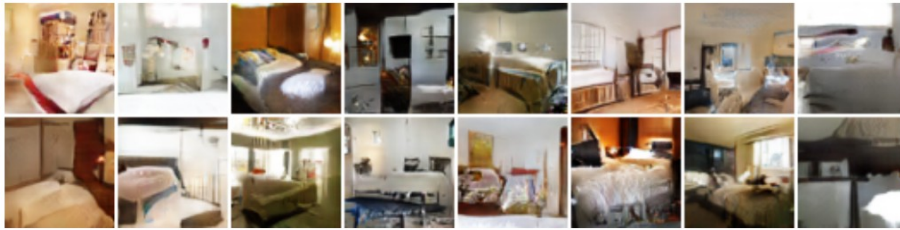


Wasserstein GAN



梯度问题





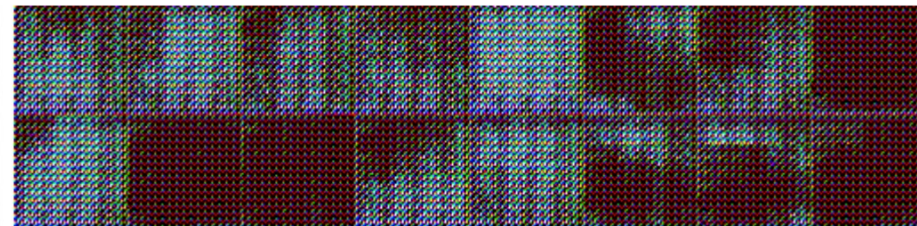
WGAN



DCGAN



~~batch normalization
constant number of filters at every layer~~



DCGAN

LSGAN

Original
WGAN

Improved
WGAN

G: CNN, D: CNN



G: CNN (no normalization), D: CNN (no normalization)



G: CNN (tanh), D: CNN(tanh)



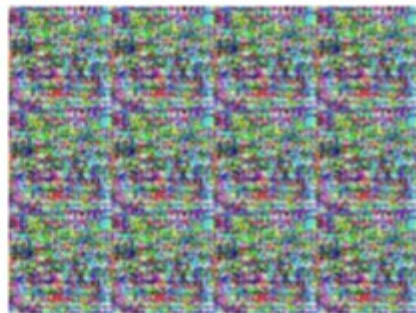
DCGAN

LSGAN

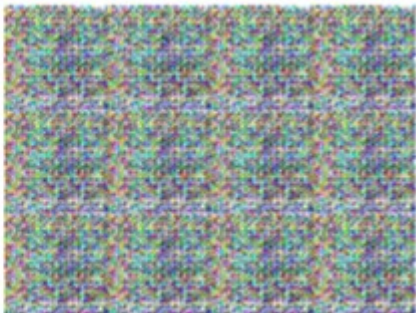
Original
WGAN

Improved
WGAN

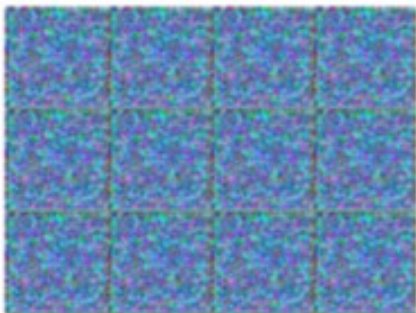
G: MLP, D: CNN



G: CNN (bad structure), D: CNN



G: 101 layer, D: 101 layer





GAN 的扩展

条件生成

► 根据条件针对性的生成数据



“Girl with red hair and red eyes”

“Girl with yellow ribbon”



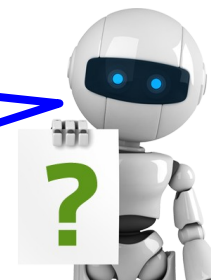
条件生成

Caption Generation

Given
condition:



“A young
girl is
dancing”

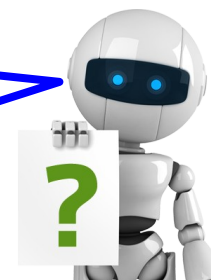


Chat-bot

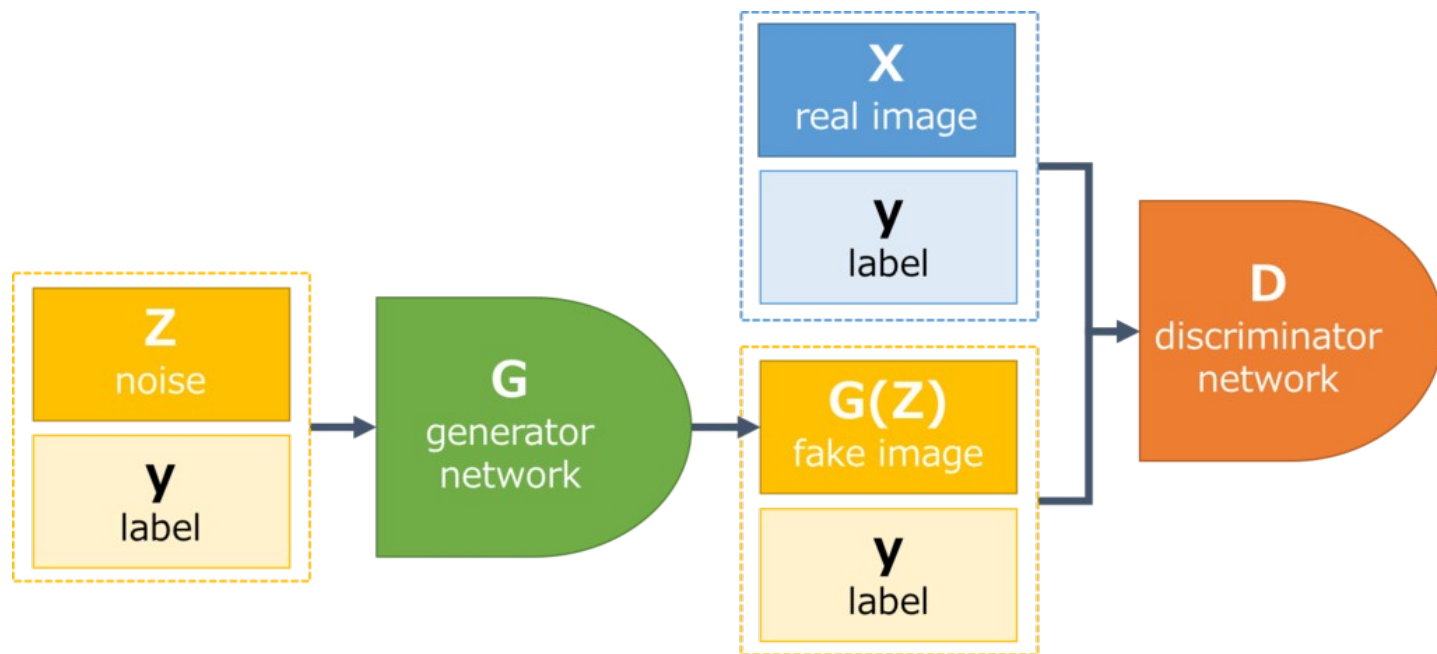
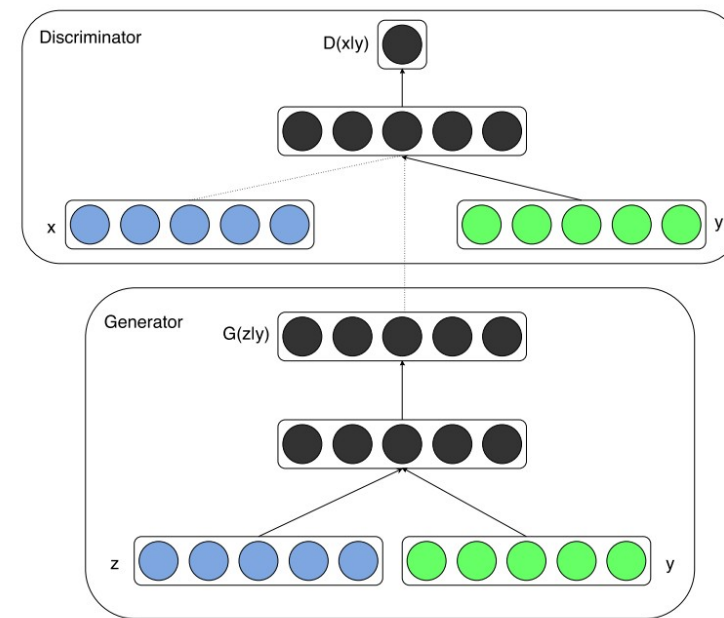
Given
condition:

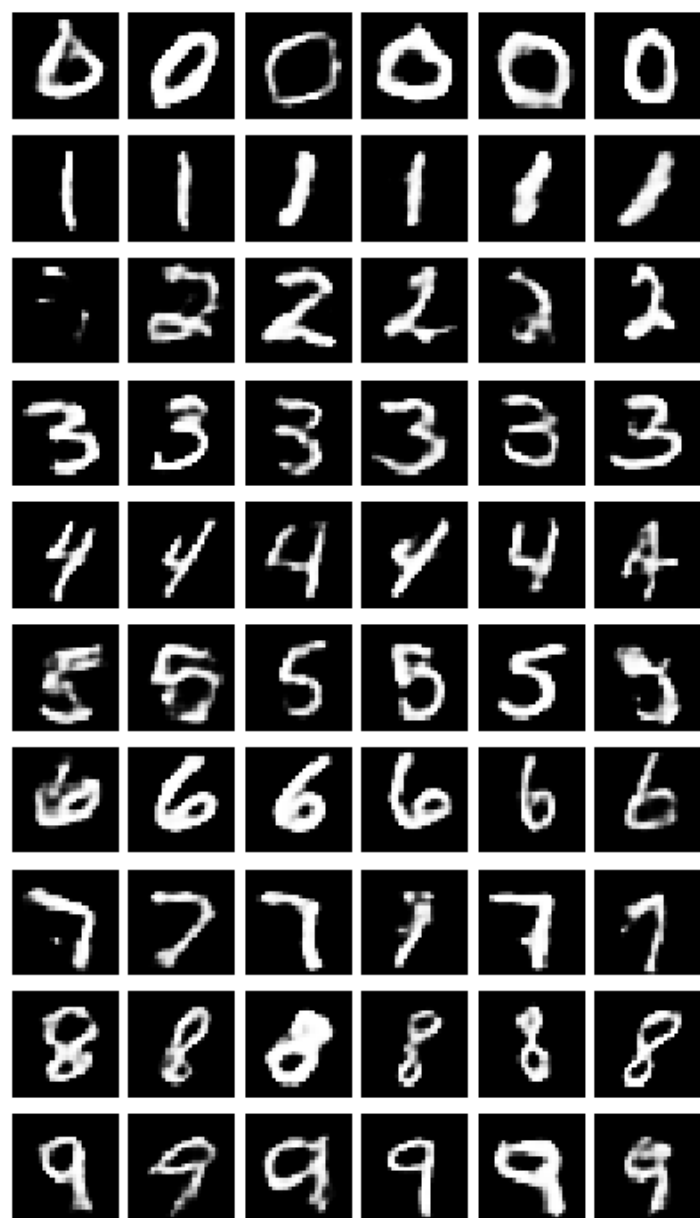


“Hello. Nice
to see you.”

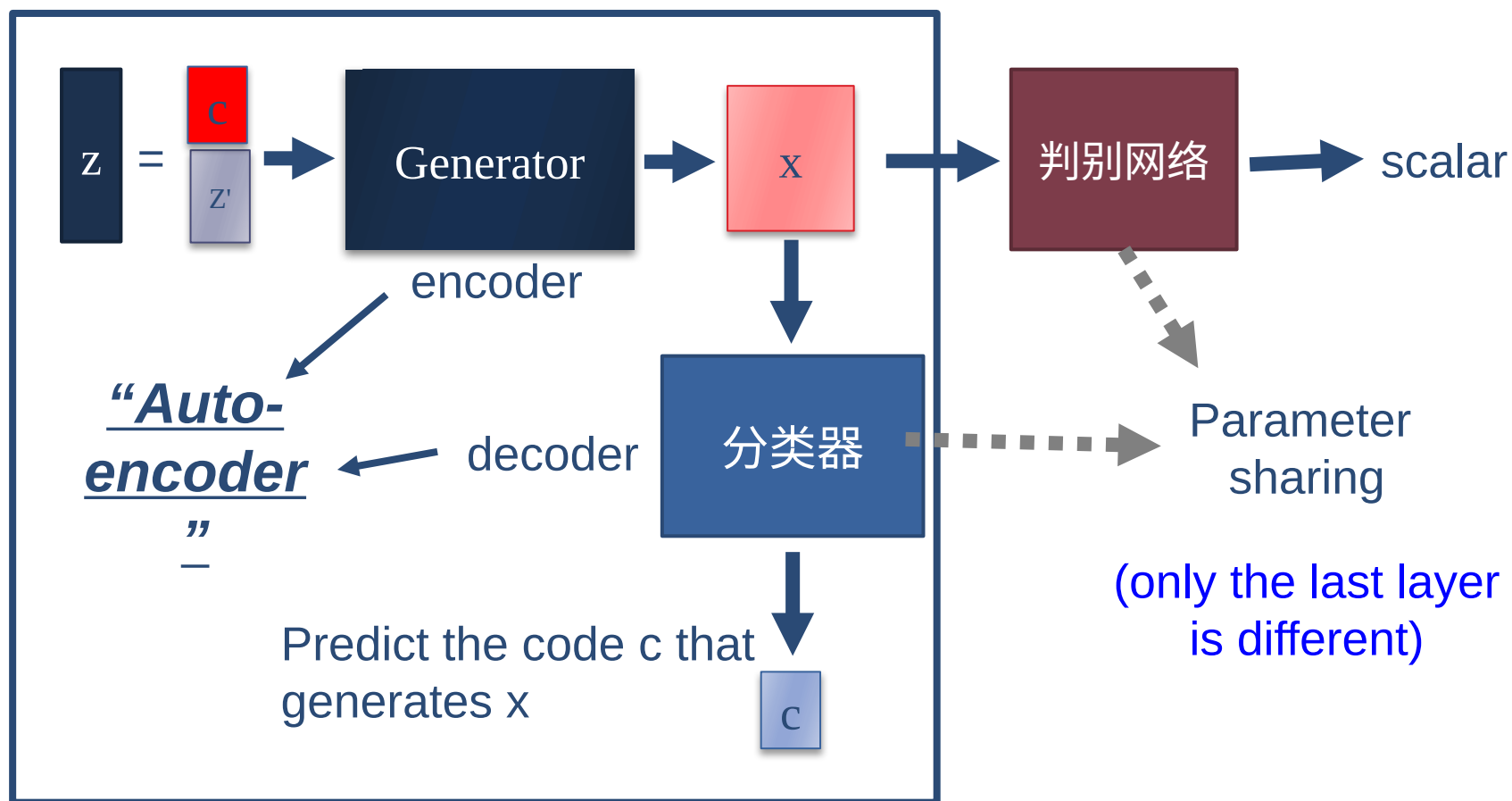


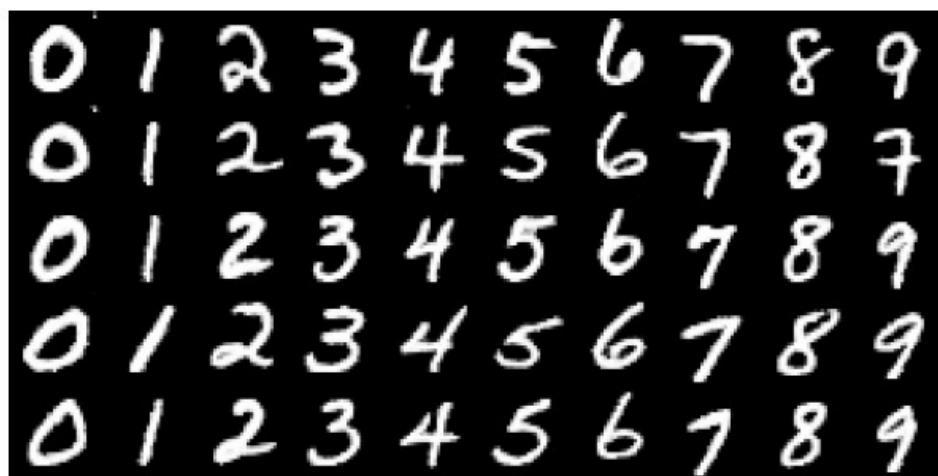
Conditional GAN



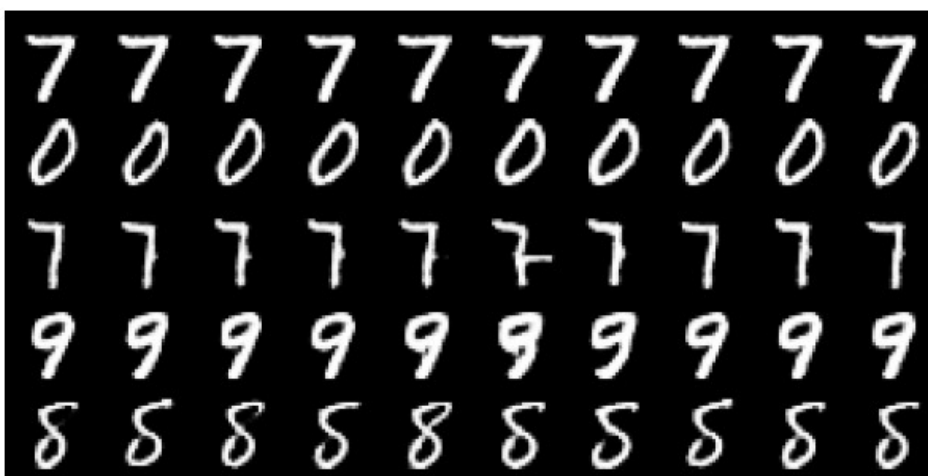


InfoGAN

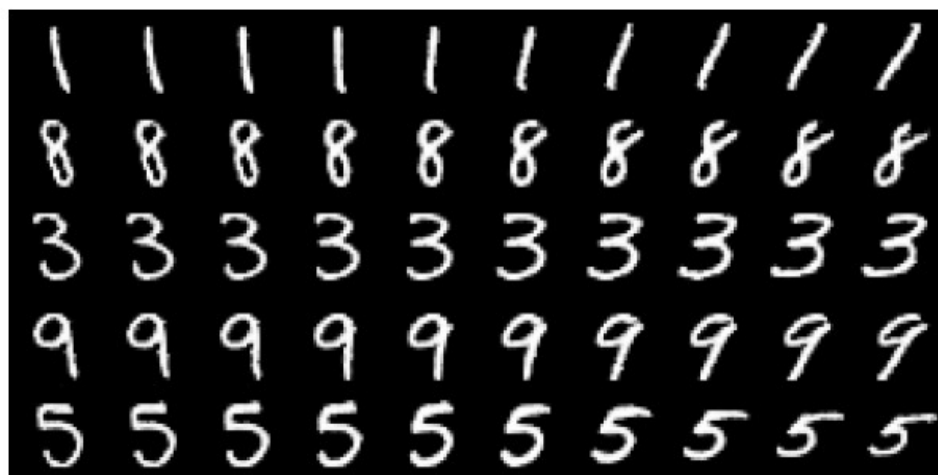




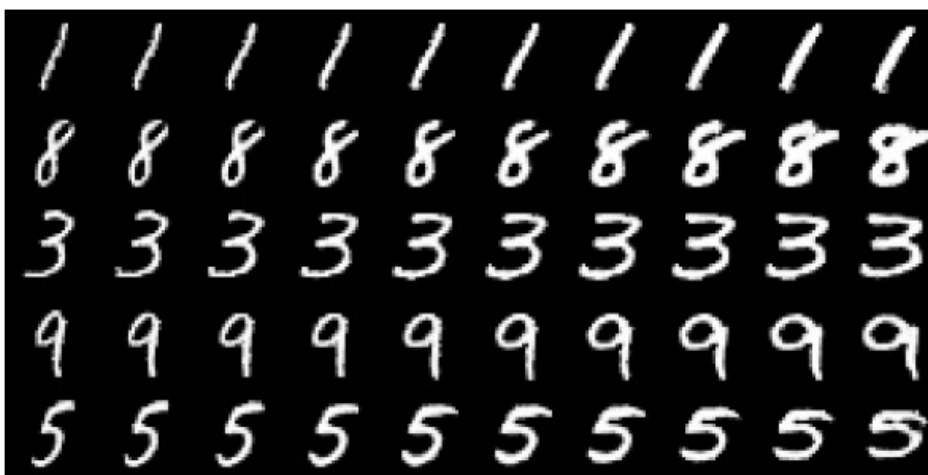
(a) Varying c_1 on InfoGAN (Digit type)



(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)



(a) Rotation

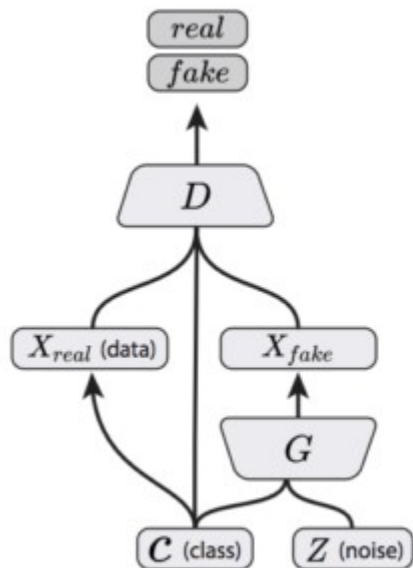
(b) Width



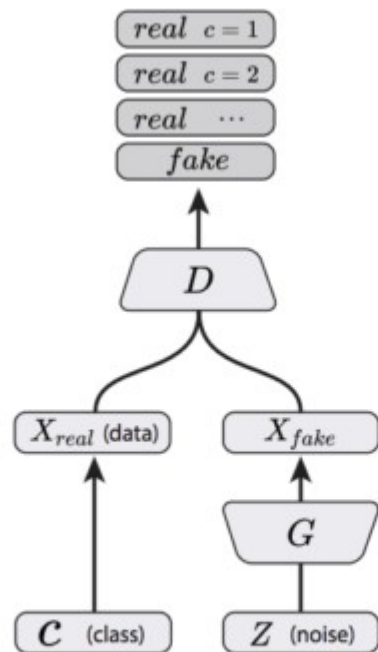
(c) Lighting

(d) Wide or Narrow

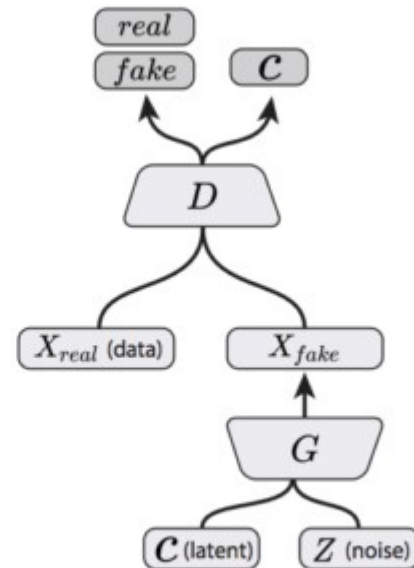
AC-GAN



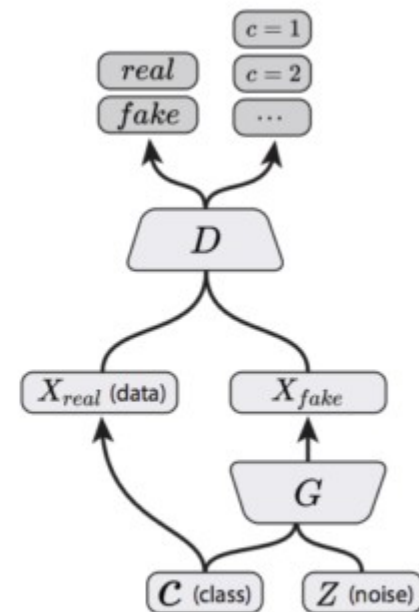
Conditional GAN
(Mirza & Osindero, 2014)



Semi-Supervised GAN
(Odena, 2016; Salimans, et al., 2016)



InfoGAN
(Chen, et al., 2016)

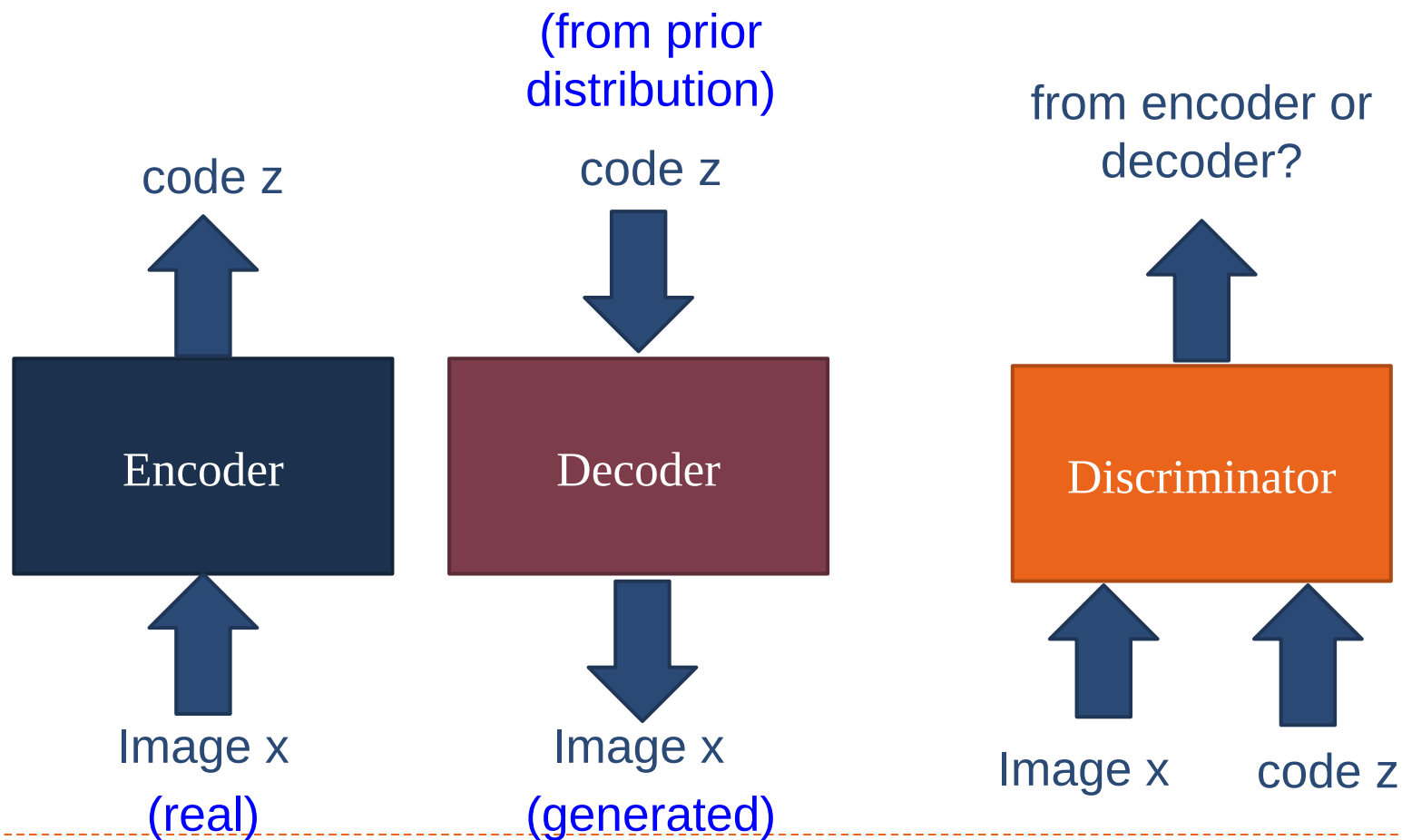


AC-GAN
(Present Work)

BiGAN

Jeff Donahue, Philipp Krähenbühl, Trevor Darrell,
“Adversarial Feature Learning”, ICLR, 2017

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier
Mastropietro, Alex Lamb, Martin Arjovsky, Aaron Courville,
“Adversarially Learned Inference”, ICLR, 2017





谢谢

<https://nndl.github.io/>