《神经网络与深度学习》

深度生成模型
Deep Generative Models

https://nndl.github.io/

参考

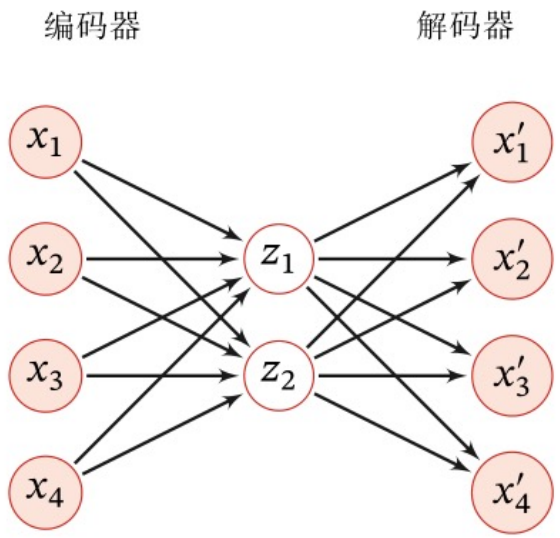
- 《神经网络与深度学习》- 深度生成模型
  - https://nndl.github.io/
- 一些例子来自于李宏毅《Introduction of Generative Adversarial Network (GAN)》

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]

自编码器（ Auto-Encoder ）

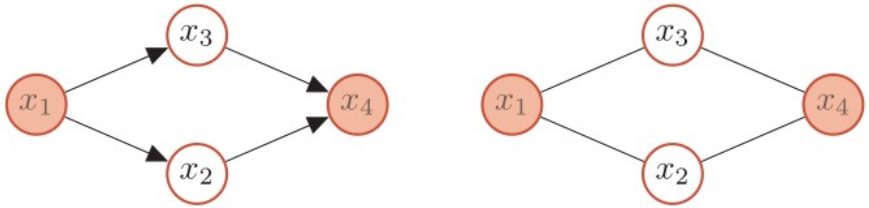
- 编码器（Encoder） $f: \mathbb{R}^D \rightarrow \mathbb{R}^M$
- 解码器（Decoder） $g: \mathbb{R}^M \rightarrow \mathbb{R}^D$



概率图的参数估计

概率图模型

- 概率图模型是指一种用图结构来描述多元随机变量之间条件独立关系的概率模型。

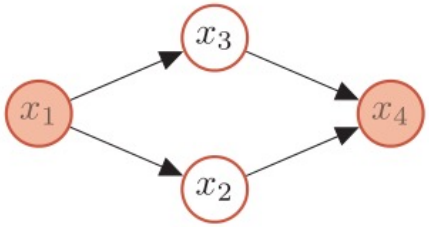


(a) 有向图：贝叶斯网络　　(b) 无向图：马尔可夫随机场

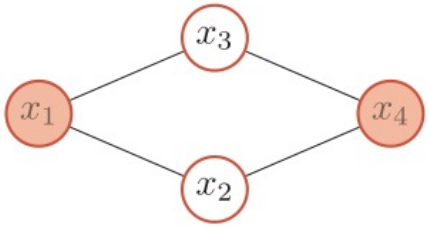
- 每个节点都对应一个随机变量，可以是观察变量，隐变量或是未知参数等；
- 每个连接表示两个随机变量之间具有依赖关系。

概率图模型

- 模型表示（图结构）
  - 有向图
  - 无向图



(a) 有向图：贝叶斯网络



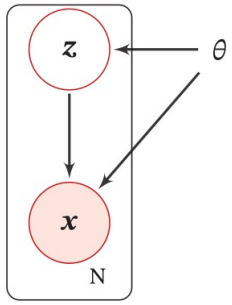
(b) 无向图：马尔可夫随机场

- 推断（Inference）
  - 给定部分变量，推断另一部分变量的后验概率。

- 学习（Learning）
  - 参数学习：给定一组训练样本，求解模型参数

含隐变量的参数学习

- 隐变量即变量是不可观测的



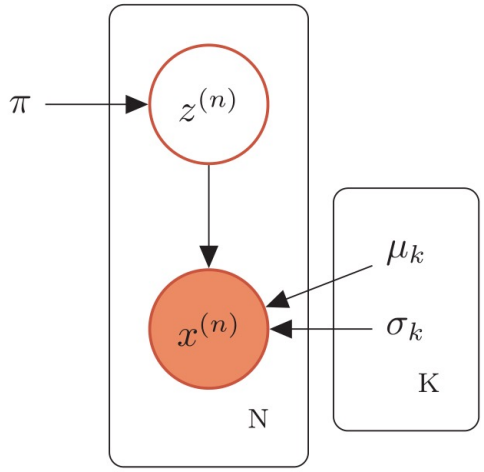
- 边际似然函数（Marginal Likelihood）

$$p(\boldsymbol{x};\theta)=\sum_{\boldsymbol{z}} p(\boldsymbol{x},\boldsymbol{z};\theta)$$

  - 需要用EM算法进行参数估计

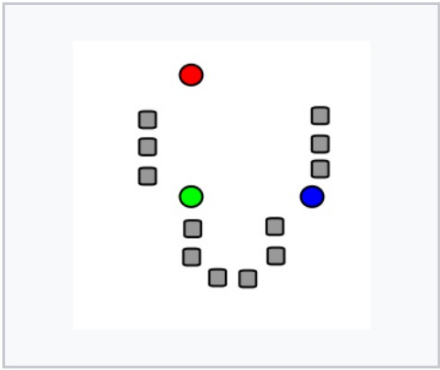
高斯混合模型

- 图模型表示

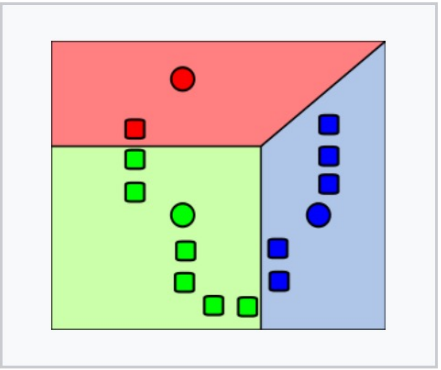


$$\mathcal{N}(x|\mu_k,\sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

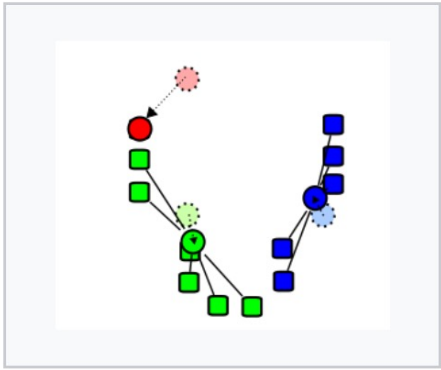
一个简单的解法：K-means



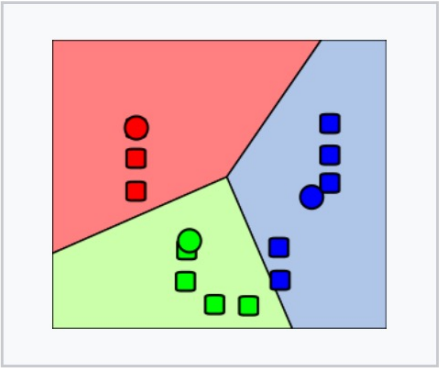
1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3. The centroid of each of the *k* clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

K-means算法

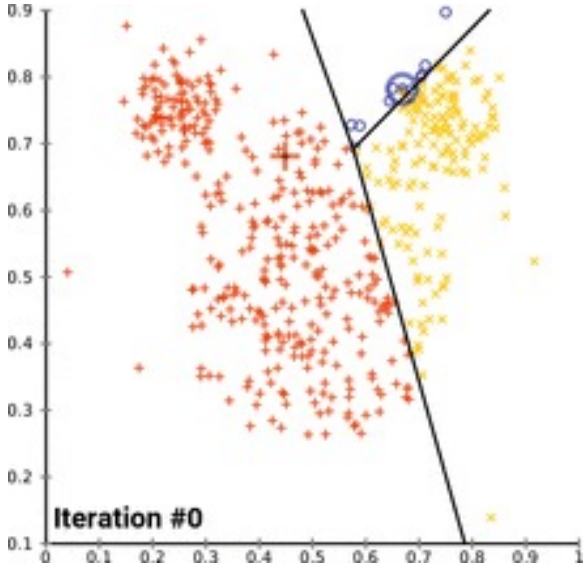
- 初始化中心点 $m_1^{(1)},\ldots,m_k^{(1)}$
- 迭代执行下面两步
  - 分配步（ Assignment step ）：

$$S_i^{(t)} = \left\{x_p : \left\|x_p - m_i^{(t)}\right\|^2 \le \left\|x_p - m_j^{(t)}\right\|^2 \ \forall j, 1 \le j \le k\right\},$$

  - 更新步（Update step）

$$m_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j$$

K-means算法



期望最大化（Expectation-Maximum，EM ）算法

- 假设有一组变量，有部分变量是不可观测的，如何进行参数估计呢?

D.2.7.1 Jensen不等式

如果X是随机变量，g是凸函数，则

$$g(\mathbb{E}[X]) \leq \mathbb{E}[g(X)]. \tag{D.41}$$

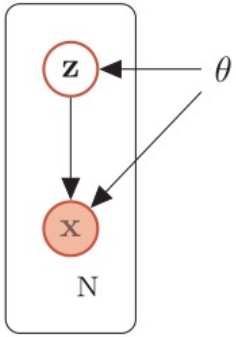
等式当且仅当X是一个常数或g是线性时成立，这个性质称为Jensen不等式.

特别地，对于凸函数g定义域上的任意两点x_1、x_2和一个标量$\lambda \in [0,1]$，有

$$g(\lambda x_1 + (1-\lambda)x_2) \leq \lambda g(x_1) + (1-\lambda)g(x_2), \tag{D.42}$$

即凸函数g上的任意两点的连线位于这两点之间函数曲线的上方.

变分



对数边际似然函数

$$\begin{aligned}\log p(\boldsymbol{x};\theta) &= \log \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \frac{p(\boldsymbol{x},\boldsymbol{z};\theta)}{q(\boldsymbol{z})} \\ &\geq \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \log \frac{p(\boldsymbol{x},\boldsymbol{z};\theta)}{q(\boldsymbol{z})} \\ &\triangleq ELBO(q,\boldsymbol{x};\theta),\end{aligned}$$

证据下界

期望最大化（Expectation-Maximum，EM ）算法

- 假设有一组变量，有部分变量是不可观测的，如何进行参数估计呢?

对数边际似然函数

$$\log \mathrm{p}(\mathrm{x};\theta) = \log E_{z\sim q(z)}\left(\frac{p(z,x;\theta)}{q(z)}\right)$$

$$\geq E_{z\sim q(z)} \log\left(\frac{p(z,x;\theta)}{q(z)}\right)$$

$$= \mathrm{ELBO}$$

另外一种推导

$$
\begin{aligned}
\log p(\boldsymbol{x};\theta) &= \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \log p(\boldsymbol{x};\theta) \\
&= \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \Big(\log p(\boldsymbol{x},\boldsymbol{z};\theta) - \log p(\boldsymbol{z}|\boldsymbol{x};\theta) \Big) \\
&= \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \log \frac{p(\boldsymbol{x},\boldsymbol{z};\theta)}{q(\boldsymbol{z})} - \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \log \frac{p(\boldsymbol{z}|\boldsymbol{x};\theta)}{q(\boldsymbol{z})} \\
&= ELBO(q,\boldsymbol{x};\theta) + \mathrm{KL}(q(\boldsymbol{z}) || p(\boldsymbol{z}|\boldsymbol{x};\theta)),
\end{aligned}
$$

EM算法

$$ELBO(q, \boldsymbol{x}; \theta) + \mathrm{KL}(q(\boldsymbol{z}) || p(\boldsymbol{z}|\boldsymbol{x}; \theta))$$

- E步

$$q_{t+1}(\mathbf{z}) = \arg\max_{q} ELBO(q, \boldsymbol{x}; \theta) + \mathrm{KL}(q(\boldsymbol{z}) || p(\boldsymbol{z}|\boldsymbol{x}; \theta))$$

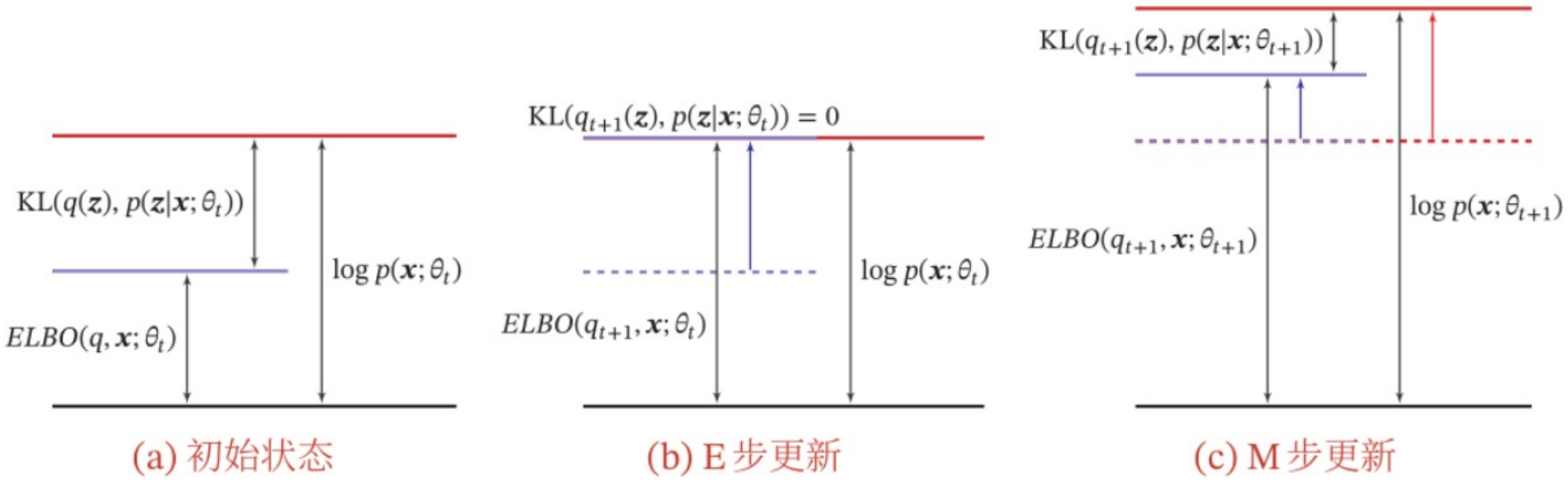
- M步

$$\theta_{t+1} = \arg\max_{\theta} ELBO(q, \boldsymbol{x}; \theta) + \mathrm{KL}(q(\boldsymbol{z}) || p(\boldsymbol{z}|\boldsymbol{x}; \theta))$$

收敛性

$$\log p(\boldsymbol{x};\theta_{t+1}) \geq ELBO(q_{t+1},\boldsymbol{x};\theta_{t+1}) \geq ELBO(q_{t+1},\boldsymbol{x};\theta_t) = \log p(\boldsymbol{x};\theta_t)$$

M步　　E步



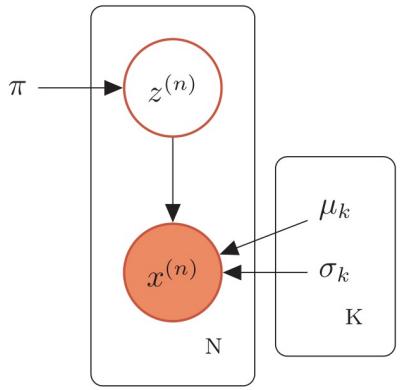
(a) 初始状态　(b) E步更新　(c) M步更新

GMM Revisit

E步 先固定参数μ, σ，计算后验分布$p(z^{(n)}|x^{(n)})$

$$\begin{aligned}\gamma_{nk} &\triangleq p(z^{(n)}=k|x^{(n)}) \\ &= \frac{p(z^{(n)})p(x^{(n)}|z^{(n)})}{p(x^{(n)})} \\ &= \frac{\pi_k \mathcal{N}(x^{(n)}|\mu_k, \sigma_k)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x^{(n)}|\mu_k, \sigma_k)},\end{aligned}$$

其中γ_{nk}定义了样本$x^{(n)}$属于第k个高斯分布的后验概率。



$$\mathcal{N}(x|\mu_k, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

GMM Revisit

M步 令$q(z=k)=\gamma_{nk}$，训练集$\mathcal{D}$的证据下界为

$$
\begin{aligned}
ELBO(\gamma,\mathcal{D}|\pi,\mu,\sigma) &= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\log\frac{p(x^{(n)},z^{(n)}=k)}{\gamma_{nk}} \\
&= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\left(\log\mathcal{N}(x^{(n)}|\mu_k,\sigma_k)+\log\frac{\pi_k}{\gamma_{nk}}\right) \\
&= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\left(\frac{-(x-\mu_k)^2}{2\sigma_k^2}-\log\sigma_k+\log\pi_k\right)+C,
\end{aligned}
$$

其中C为和参数无关的常数。

$$
\text{s.t.}\quad \sum_{k=1}^{K}\pi_k=1.
$$

$$
\mathcal{N}(x|\mu_k,\sigma_k)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)
$$

$$
p(x^n,z^n=k)=\pi_k N(x^n|\mu_k,\sigma_k)
$$

$$
\log\left(\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)\right)=\frac{-(x-\mu_k)^2}{2\sigma_k^2}-\log\sigma_k-\log\sqrt{2\pi}
$$

$$
C=\sum_{n=1}^{N}\sum_{k=1}^{K}(-log\gamma_{n_k}-\log\sqrt{2\pi})*\gamma_{nk}
$$

GMM Revisit

M 步 令 $q(z=k)=\gamma_{nk}$，训练集 $\mathcal{D}$ 的证据下界为

$$
\begin{aligned}
ELBO(\gamma,\mathcal{D}|\pi,\mu,\sigma) &= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\log\frac{p(x^{(n)},z^{(n)}=k)}{\gamma_{nk}} \\
&= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\left(\log\mathcal{N}(x^{(n)}|\mu_k,\sigma_k)+\log\frac{\pi_k}{\gamma_{nk}}\right) \\
&= \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\left(\frac{-(x-\mu_k)^2}{2\sigma_k^2}-\log\sigma_k+\log\pi_k\right)+C,
\end{aligned}
$$

其中 C 为和参数无关的常数。

$$
\text{s.t.}\quad \sum_{k=1}^{K}\pi_k=1.
$$

$$
ELBO(\gamma,D|\pi,\mu,\sigma)+\lambda\left(\sum_{k=1}^{K}\pi_k-1\right)
$$

$$
\sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{nk}\left(\frac{-(x-\mu_k)^2}{2\sigma_k^2}-\log\sigma_k+\log\pi_k\right)+\lambda\left(\sum_{k=1}^{K}\pi_k-1\right)+C
$$

$$
\frac{\partial L}{\partial\pi_k}=\sum_{n=1}^{N}\gamma_{nk}\frac{1}{\pi_k}+\lambda=0
$$

$$
\frac{\partial L}{\partial\mu_k}=\sum_{n=1}^{N}\gamma_{nk}\frac{(x^n-\mu_k)}{\sigma_k^2}=0
$$

$$
\frac{\partial L}{\partial\sigma_k}=\sum_{n=1}^{N}\gamma_{nk}\left(\frac{(x-\mu_k)^2}{\sigma_k^3}-\frac{1}{\sigma_k}\right)=0
$$

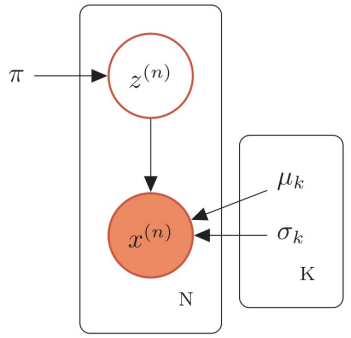
GMM Revisit

$$N_k = \sum_{n=1}^{N} \gamma_{nk}.$$

$$\pi_k = \frac{N_k}{N},$$

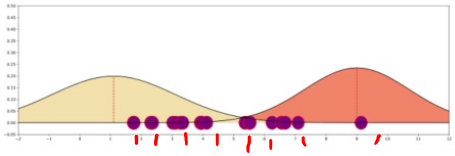
$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} x^{(n)},$$

$$\sigma_k^2 = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} (x^{(n)} - \mu_k)^2,$$

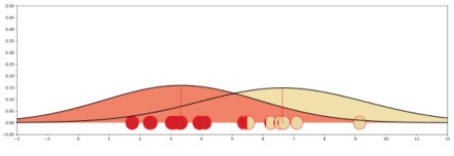


$$\mathcal{N}(x|\mu_k, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\Big(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\Big)$$

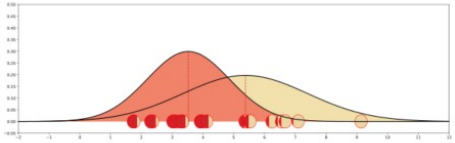
GMM的参数学习



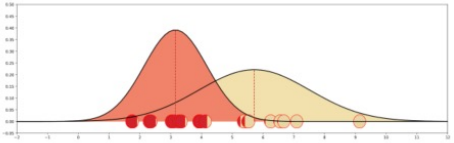
(a) 初始化



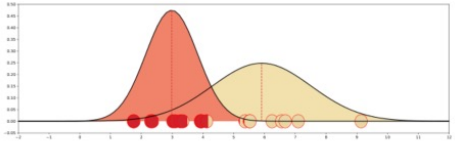
(b) 第 1 次迭代



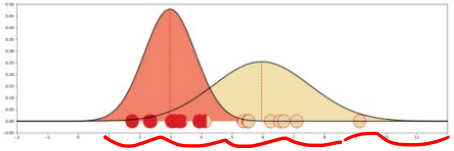
(c) 第 4 次迭代



(d) 第 8 次迭代

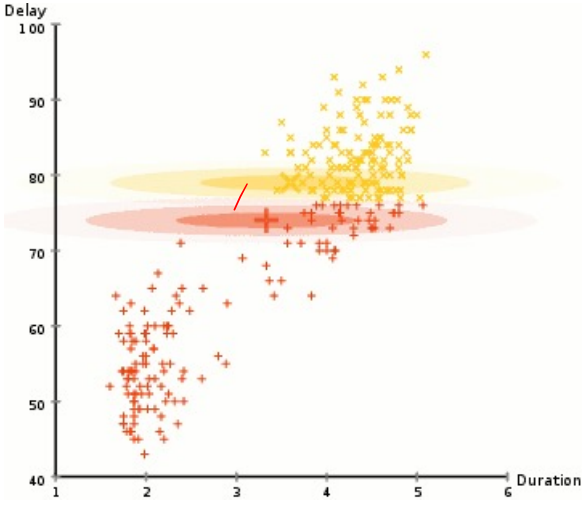


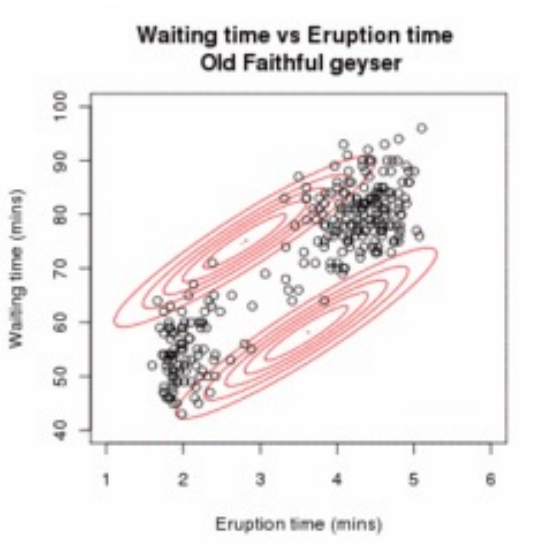
(e) 第 12 次迭代



(f) 第 16 次迭代

GMM的参数学习


Delay
100
90
80
70
60
50
40
1
2
3
4
5
6
Duration

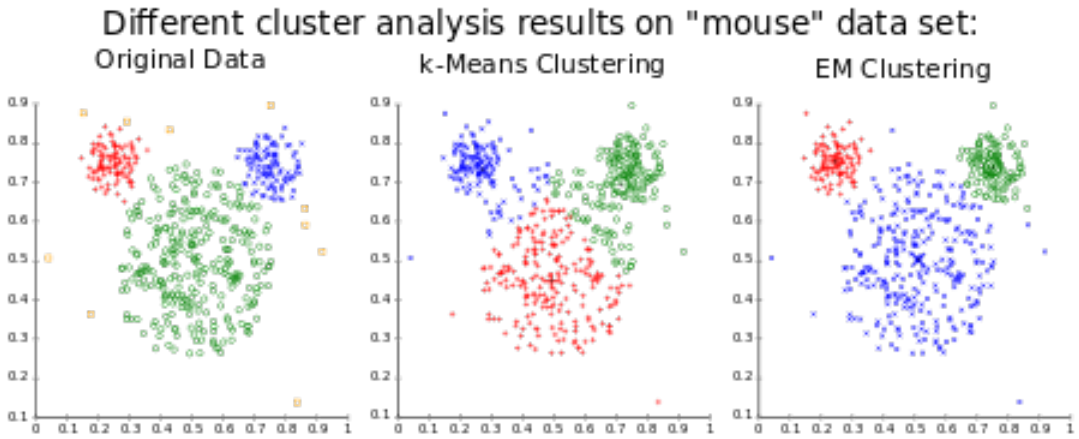

Waiting time vs Eruption time
Old Faithful geyser
Waiting time (mins)
100
90
80
70
60
50
40
1
2
3
4
5
6
Eruption time (mins)

k-means clustering vs. EM clustering

- 随机确定k中心点
- 计算每个样本点到中心点的距离
- 把每个样本点分配到离它最近的中心点
- 对每一个中心点，根据分配到的样本点重新计算中心点位置
- 迭代直到收敛

- 随机初始化k个高斯分布的均值和方差
- 计算每个样本点属于每个高斯分布的后验概率
- 用后验概率重新估计每个高斯分布的均值和方差
- 迭代直到收敛

k-means clustering vs. EM clustering

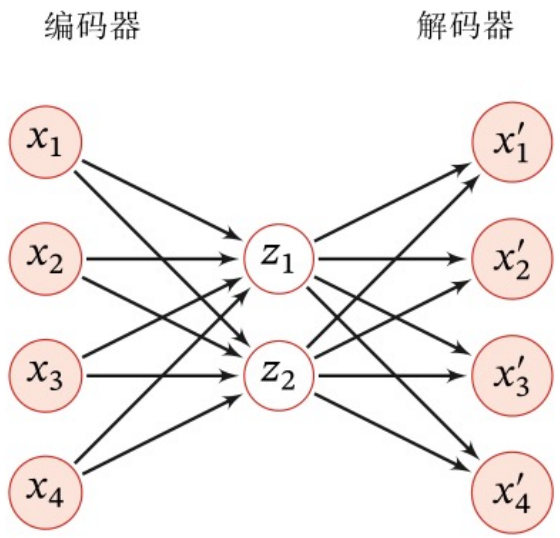


图模型与神经网络的关系

1. 图模型的节点是随机变量，图结构主要描述变量之间的依赖关系；而神经网络中的节点是神经元，是一个计算节点。
2. 图模型中每个变量一般有着明确的解释，变量之间依赖关系一般是人工来定义；而神经网络中单个神经元没有直观的解释。
3. 神经网络是判别式模型，直接用来分类；而图模型既可以是判别式模型，也可以是生成式模型。
4. 神经网络和概率图的结合越来越紧密。

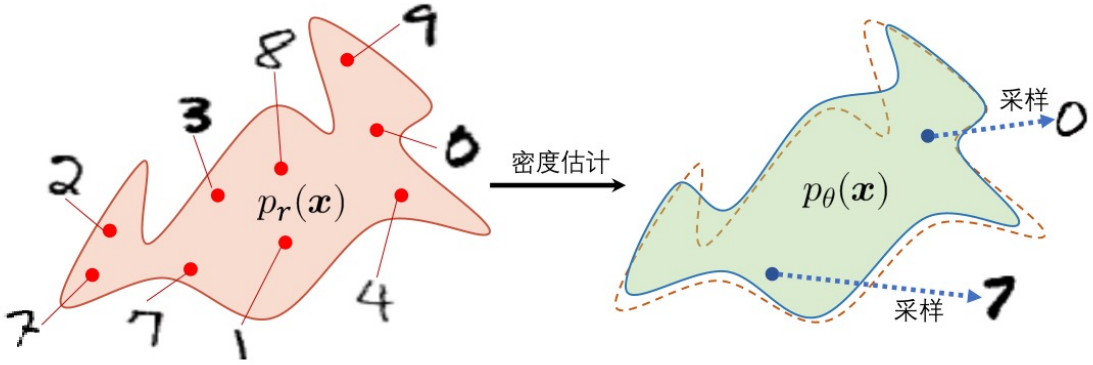
自编码器（ Auto-Encoder ）

- 编码器（Encoder） $f: \mathbb{R}^D \rightarrow \mathbb{R}^M$
- 解码器（Decoder） $g: \mathbb{R}^M \rightarrow \mathbb{R}^D$

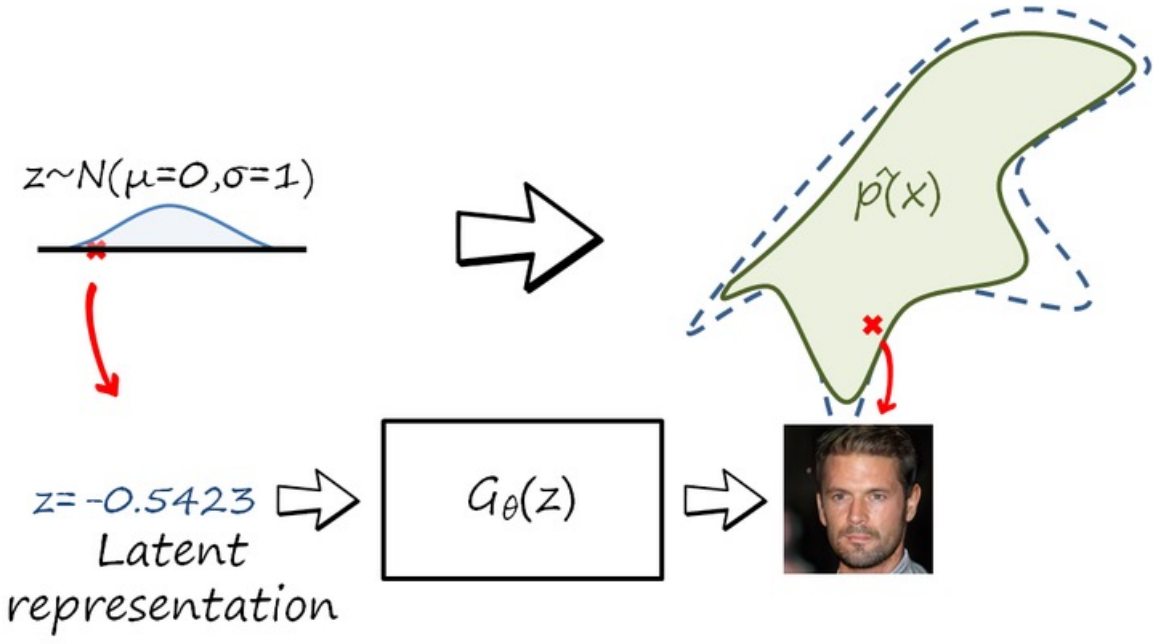


生成模型：一系列用于随机生成可观测数据的模型

- 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

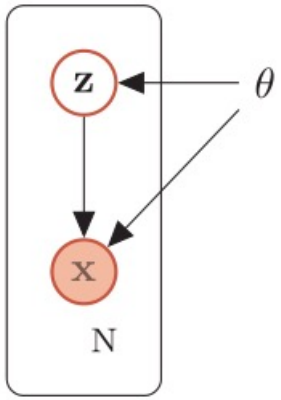


变分自编码器
Variational Autoencoder，VAE

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x，其对数边际似然log p(x|θ)可以分解为

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

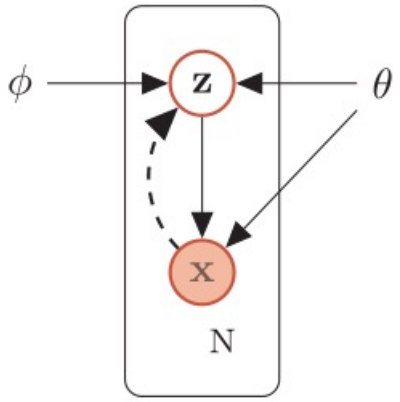
$$\log p(\mathbf{x}|\theta) = ELBO(q,\mathbf{x}|\theta,\phi) + D_{\mathrm{KL}}(q(\mathbf{z}|\phi)\|p(\mathbf{z}|\mathbf{x},\theta))$$

M step　　E step

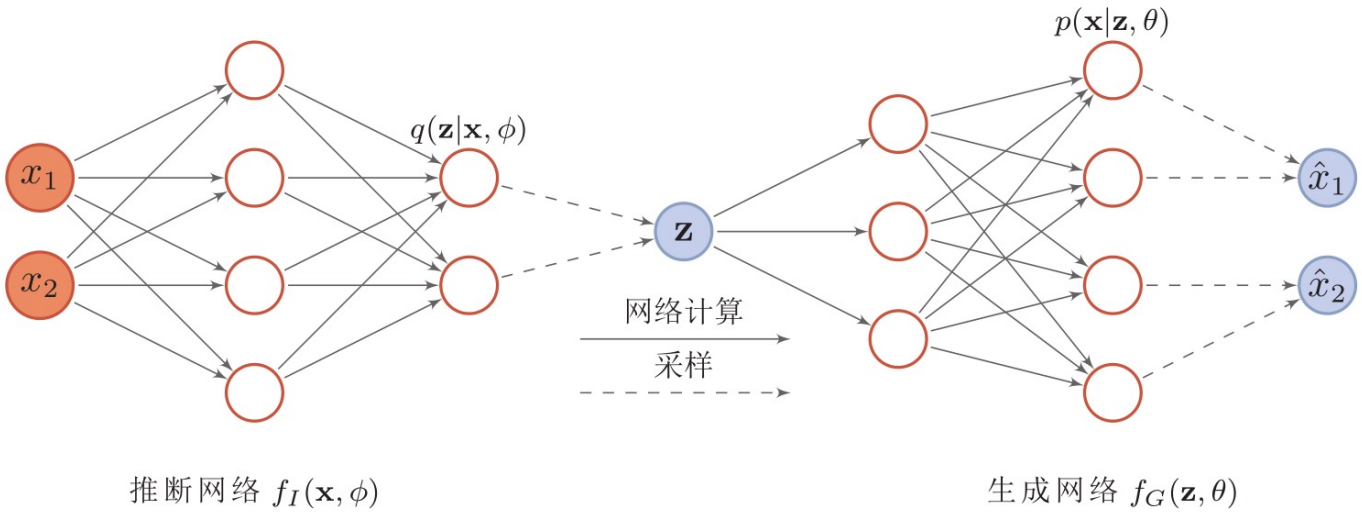
变分自编码器(VAE)

- 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$p(z|x;\theta)$
  - 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


$p(\mathbf{x}|\mathbf{z},\theta)$
$q(\mathbf{z}|\mathbf{x},\phi)$
x_1
x_2
$\mathbf{z}$
$\hat{x}_1$
$\hat{x}_2$
网络计算
采样
推断网络 $f_I(\mathbf{x},\phi)$
生成网络 $f_G(\mathbf{z},\theta)$

变分自编码器 vs. 自编码器

- 变分自编码器的名字来自于其整个网络结构和自编码器比较类似。
  - 推断网络可看作“编码器”，将可观测变量映射为隐变量
  - 生成网络看作“解码器”，将隐变量映射为可观测变量
- 变分自编码器的原理和自编码器原理不同
  - 变分自编码器中的编码器和解码器的输出为分布（或分布的参数），而不是确定的编码。

推断网络

- 推断网络

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

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

$$h = ReLU(W^1 x + b^1)$$
$$\mu_I = W^2 x + b^2$$
$$\sigma_I = W^3 x + b^3$$

- E步目标

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

生成网络

- 先验分布p(z|θ)
  - 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,θ)
  - 假设p(x|z,θ)服从对角化协方差的高斯分布

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

- M步目标

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

模型汇总

$$\max_{\theta,\phi} ELBO(q,x|\theta,\phi) = \max_{\theta,\phi} E_{z\sim q(z|x;\phi)}[\log \frac{p(x,z|\theta)}{q(z|x;\phi)}] = \max_{\theta,\phi} E_{z\sim q(z|x;\phi)}[\log \frac{p(x|z;\theta)p(z|\theta)}{q(z|x;\phi)}]$$

$$= \int q(z|x;\phi) \left[\log \frac{p(x|z;\theta)p(z|\theta)}{q(z|x;\phi)}\right] dz$$

$$= \int q(z|x;\phi)[\log p(x|z;\theta)]dz + \int q(z|x;\phi) \left[\log \frac{p(z|\theta)}{q(z|x;\phi)}\right] dz$$

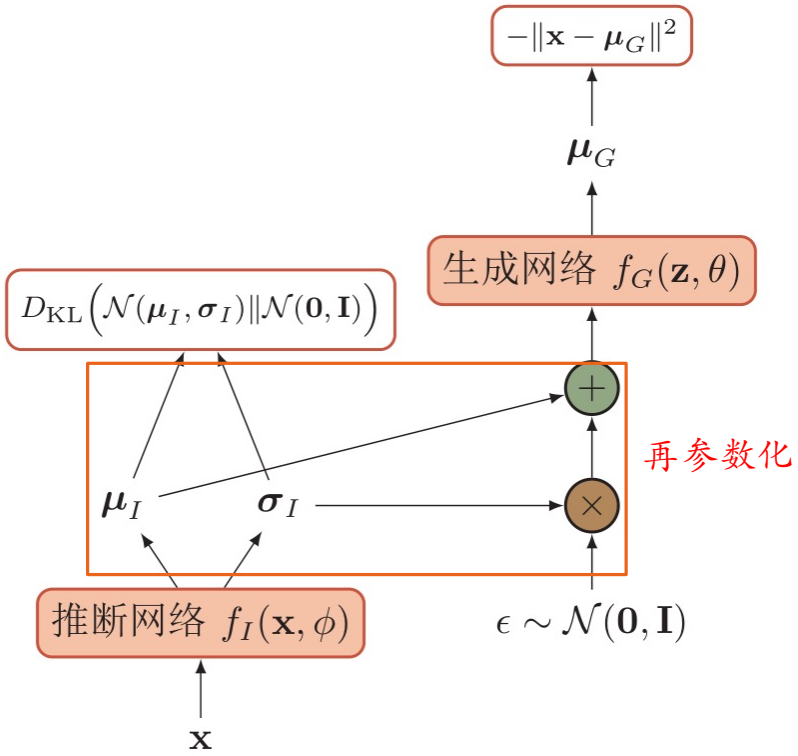
$$\max_{\theta,\phi} \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x},\phi)} \Big[\log p(\mathbf{x}|\mathbf{z},\theta) \Big] - D_{\mathrm{KL}} \Big(q(\mathbf{z}|\mathbf{x},\phi) || p(\mathbf{z}|\theta) \Big)$$

模型汇总

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

最大化图像重构概率　　最小化后验概率分布$q(z|x,\phi)$和高斯分布$p(z|\theta)$的差异

变分自编码器的训练过程



$D_{kl}\left(N(\mu_I, \sigma_I) \mid N(0, I)\right)$推导

$$N(\mu \ \ , \sigma \ \ ) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

$$N(0,1) = \frac{1}{\sqrt{2\pi}} e^{-(x)^2/2}$$

$$D_{kl}\left(N(\mu \ \ , \sigma \ \ ) \mid N(0,1)\right) = \int \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2} \log \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}/\sqrt{2\pi}\sigma}{e^{-\frac{x^2}{2}}/\sqrt{2\pi}} dx$$

$D_{kl}\left(N(\mu_I,\sigma_I)\mid N(0,I)\right)$推导

$$D_{kl}\left(N(\mu\ \ ,\sigma\ \ )\mid N(0,1)\right)=\int\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/2\sigma^2}\log\frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}/\sqrt{2\pi}\sigma}{e^{-\frac{x^2}{2}}/\sqrt{2\pi}}dx$$

$$=\int\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/2\sigma^2}\log\frac{1}{\sigma}e^{\frac{1}{2}[x^2-\frac{(x-\mu)^2}{\sigma^2}]}dx$$

$$=\int\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/2\sigma^2}[\log(\sigma^2)^{-\frac{1}{2}}+\frac{1}{2}(x^2-\frac{(x-\mu)^2}{\sigma^2})]dx$$

$$=\frac{1}{2}*\int\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}[-\log(\sigma^2)\ \ +x^2-\frac{(x-\mu)^2}{\sigma^2}]dx$$

$$=\frac{1}{2}*(-\log\sigma^2+\mu^2+\sigma^2-1)$$

μ 表示 x 在高斯分布下的平均值。相似的，二阶矩为（the second order moment）：

$$E[x^2] = \int_{-\infty}^{\infty} N(x|\mu, \sigma^2)x^2 dx = \mu^2 + \sigma^2$$

$$var[x] = E[x^2] - E[x]^2 = \sigma^2$$

证明上式：令 $z = x - \mu$，则

$$E[x^2] = \int_{-\infty}^{\infty} N(x|\mu, \sigma^2)x^2 dx$$

$$= \int_{-\infty}^{\infty} \frac{1}{(2\pi\sigma^2)^{1/2}} exp\{-\frac{1}{2\sigma^2}z^2\}(z+\mu)^2 dz$$

$$= \int_{-\infty}^{\infty} \frac{1}{(2\pi\sigma^2)^{1/2}} exp\{-\frac{1}{2\sigma^2}z^2\}z^2 dz + 2\mu\int_{-\infty}^{\infty} \frac{1}{(2\pi\sigma^2)^{1/2}} exp\{-\frac{1}{2\sigma^2}z^2\}z dz$$

$$+\mu^2 \int_{-\infty}^{\infty} \frac{1}{(2\pi\sigma^2)^{1/2}} exp\{-\frac{1}{2\sigma^2}z^2\}dz$$

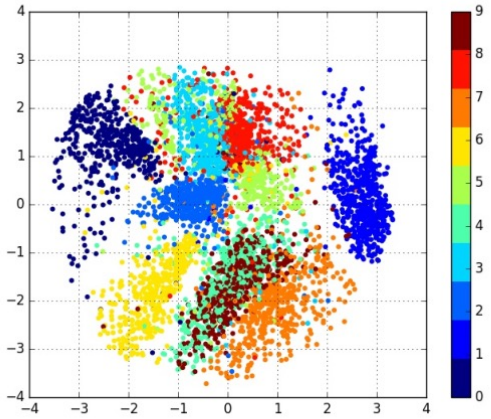
式中第二项为奇函数，且在$(-\infty, \infty)$内积分，所以为0；第三项积分部分为1，所以第三项为 μ^2，因此现在只需要求出第一项即可。在第一项中 $z = x - \mu$，所以第一项等于 $E[z^2] = E[(x-\mu)^2]$:

$$E[(x-\mu)^2] = \{\sum_{i=1}^{N}(x_i - \mu)^2 + ... + (x_\infty - \mu)^2\}/N = \sigma^2 \quad , \ N \to \infty$$

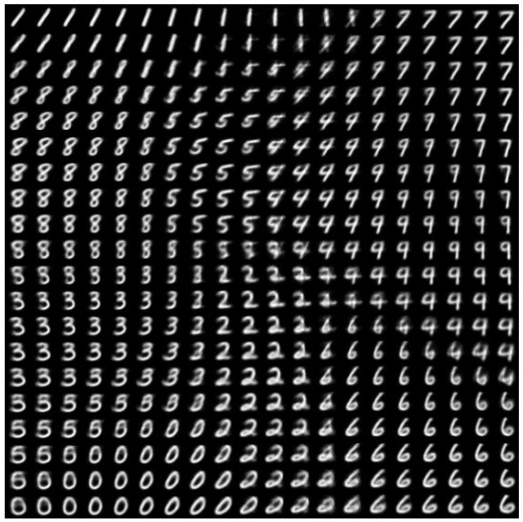
所以 $E[(x-\mu)^2] = \sigma^2$，即

$$E[x^2] = \sigma^2 + 0 + \mu^2 = \sigma^2 + \mu^2$$

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 $\mathbf{z}$ 在图像空间的投影。

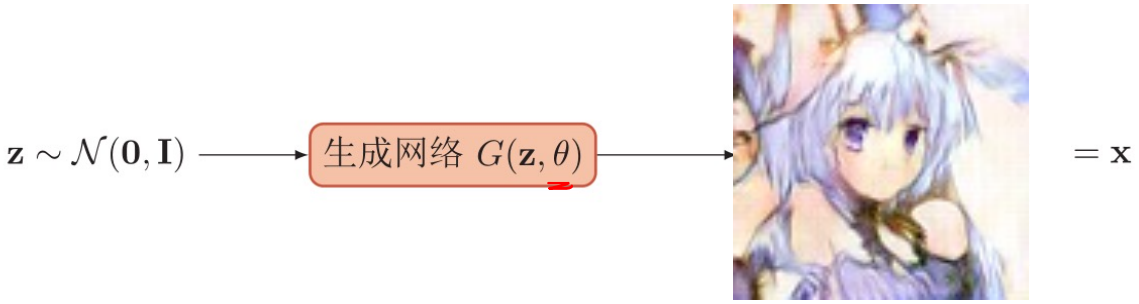
生成对抗网络
Generative Adversarial Network（GAN）

显式密度模型和隐式密度模型

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

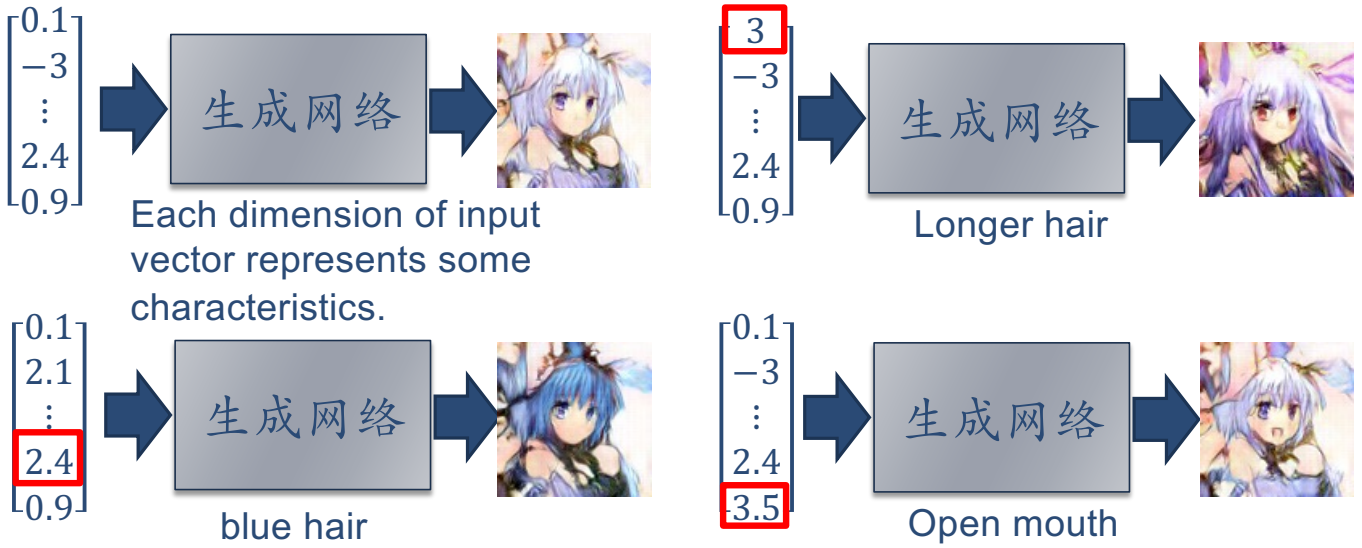
生成网络

- 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。



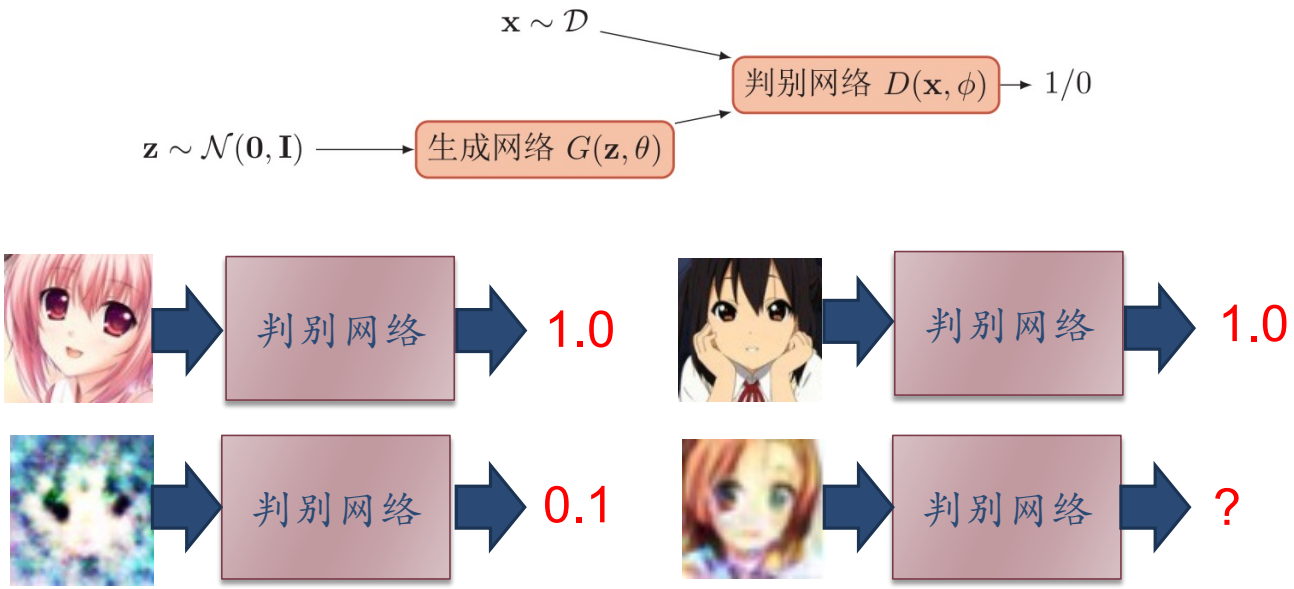
如何学习生成网络?

生成网络示例


0.1
−3
2.4
0.9
生成网络
Each dimension of input vector represents some characteristics.
3
−3
2.4
0.9
生成网络
Longer hair
0.1
2.1
2.4
0.9
生成网络
blue hair
0.1
−3
2.4
3.5
生成网络
Open mouth

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。

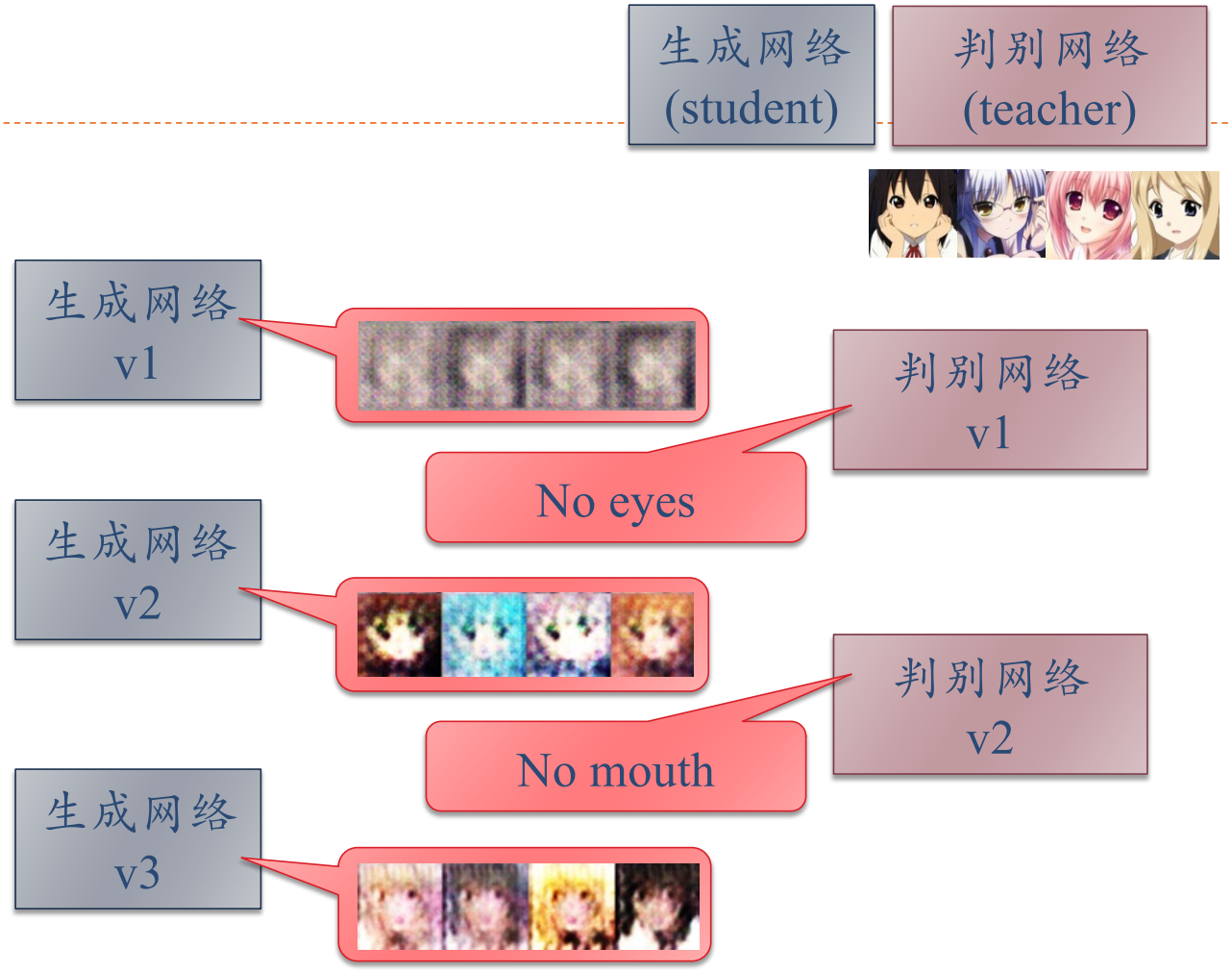


MinMax Game

- 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)
生成网络
v1
判别网络
v1
No eyes
生成网络
v2
判别网络
v2
No mouth
生成网络
v3

MinMax Game

- 判别网络

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

- 生成网络

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

- Minimax Game

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

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

随机初始化 θ, ϕ;

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

  // 训练判别网络 $D(\boldsymbol{x}; \phi)$

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

    // 采集小批量训练样本

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

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

    使用随机梯度上升更新 ϕ，梯度为

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

  end

  // 训练生成网络 $G(\boldsymbol{z}; \theta)$

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

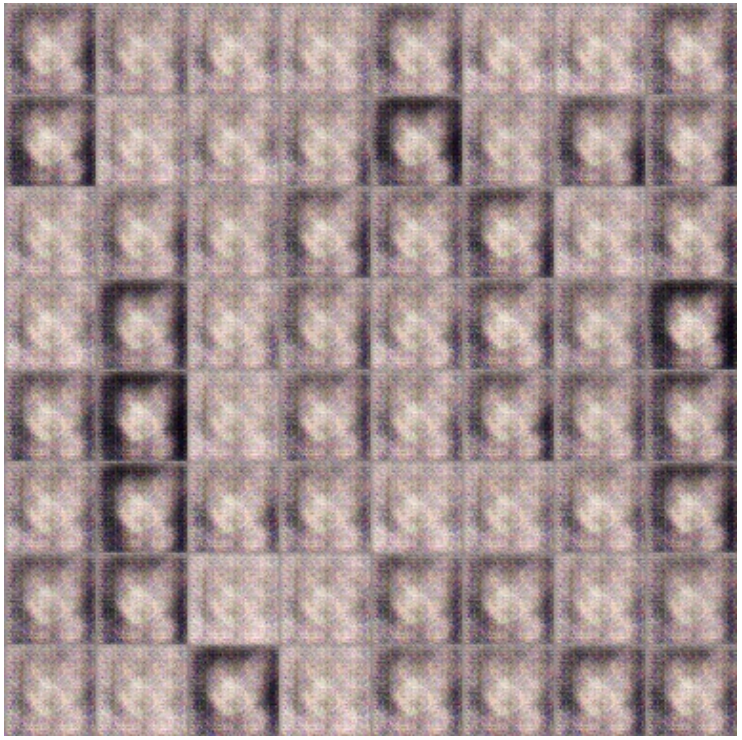
  使用随机梯度上升更新 θ，梯度为

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

end

输出: 生成网络 $G(\boldsymbol{z}; \theta)$

Anime Face Generation



100 updates

1000 updates

Anime Face Generation

2000 updates

5000 updates

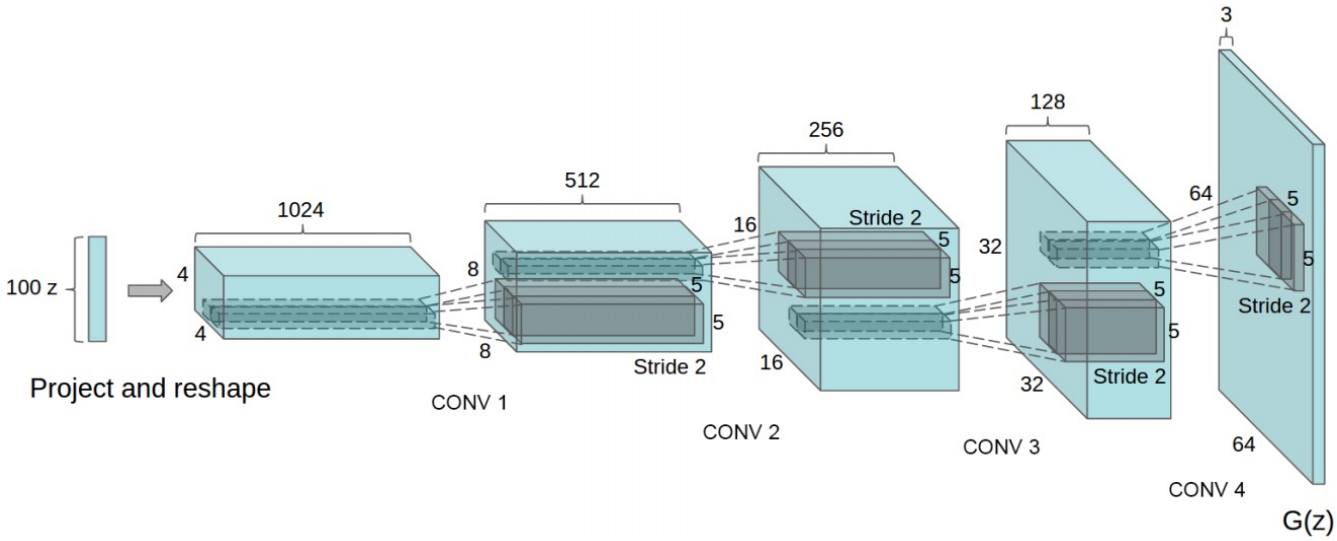
Anime Face Generation

10,000
updates

50,000
updates

一个具体的模型：DCGANs

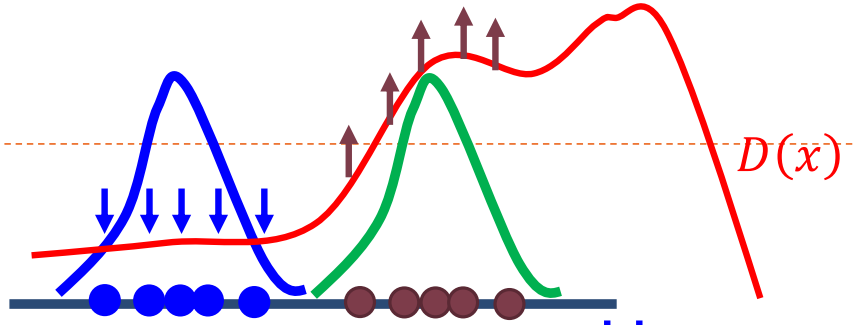
- 判别网络是一个传统的深度卷积网络，但使用了带步长的卷积来实现下采样操作，不用最大汇聚（pooling）操作。
- 生成网络使用一个特殊的深度卷积网络来实现使用微步卷积来生成64×63大小的图像。

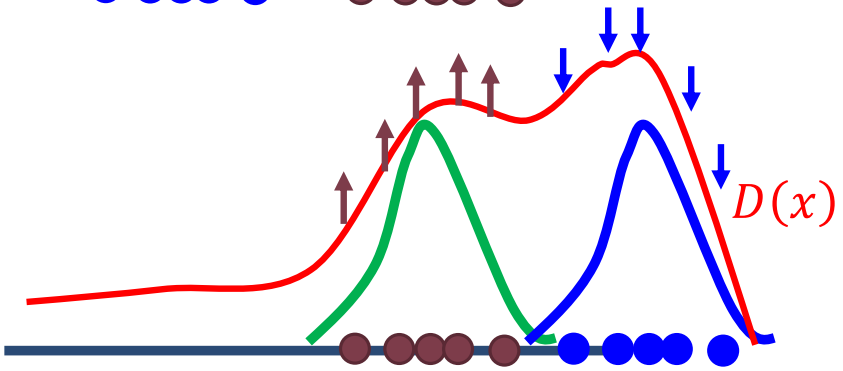


DCGANs

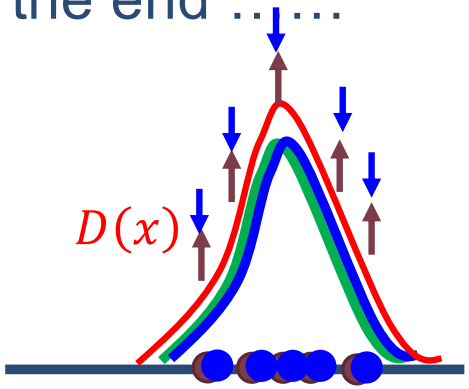
模型分析

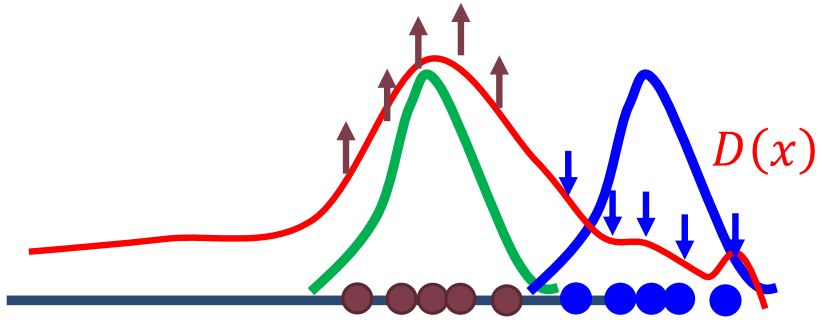
数据分布





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

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

$$f(y) = a\log y + b\log(1-y)$$

$$f'(y) = 0 \Rightarrow \frac{a}{y} - \frac{b}{1-y} = 0 \Rightarrow y = \frac{a}{a+b}$$

$$D^\star(\boldsymbol{x}) = \frac{p_r(\boldsymbol{x})}{p_r(\boldsymbol{x}) + p_\theta(\boldsymbol{x})}$$

通过在其驻点进行二阶导求解可得：

$$f''\left(\frac{a}{a+b}\right) = -\frac{a}{\left(\frac{a}{a+b}\right)^2} - \frac{b}{1-\left(\frac{a}{a+b}\right)^2} < 0$$

模型分析

- 假设$p_r(x)$和$p_\theta(x)$已知，则最优的判别器为

$$D^\star(\boldsymbol{x}) = \frac{p_r(\boldsymbol{x})}{p_r(\boldsymbol{x}) + p_\theta(\boldsymbol{x})}$$

- 目标函数变为

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

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

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

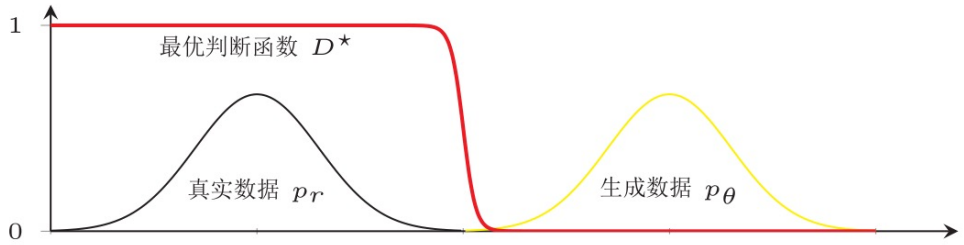
不稳定性：生成网络的梯度消失

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

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-2log2。

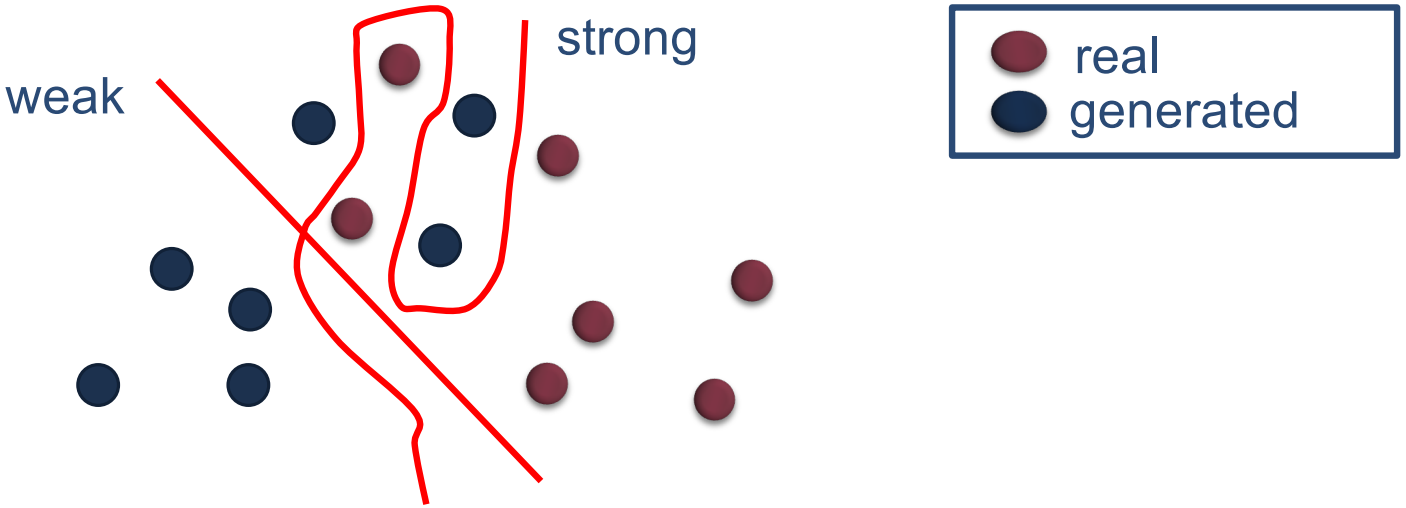
$$\frac{\partial\mathcal{L}(G|D^\star)}{\partial\theta}=0.$$

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log2。对生成网络来说，目标函数关于参数的梯度为0。



改进

- 弱化判别器
- 使用更好的损失函数



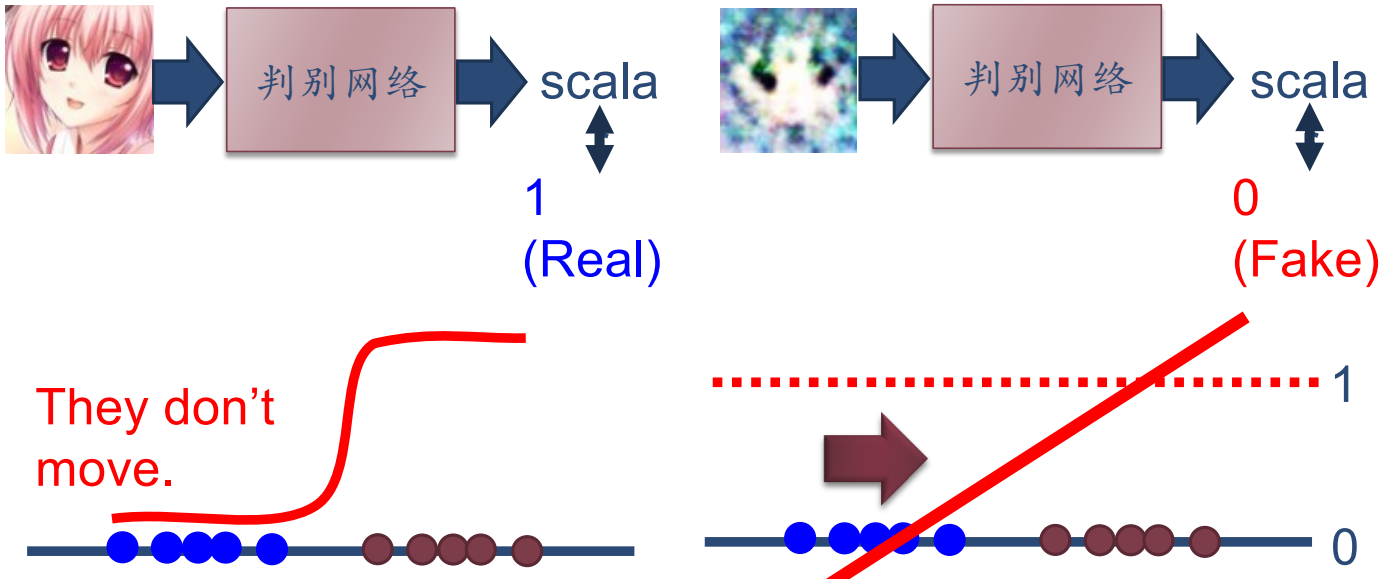
Least Square GAN (LSGAN)

real
generated

- Replace sigmoid with linear (replace classification with regression)

$$\min_D V_{\text{LSGAN}}(D) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x})-b)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z}))-a)^2\right]$$

$$\min_G V_{\text{LSGAN}}(G) = \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z}))-c)^2\right],$$



Wasserstein距离

- Wasserstein距离用于衡量两个分布之间的距离。

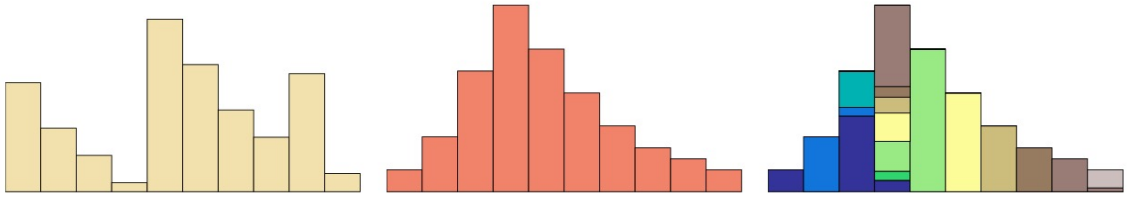
$$W_p(q_1, q_2) = \left(\inf_{\gamma(x,y) \in \Gamma(q_1, q_2)} \mathbb{E}_{(x,y)\sim\gamma(x,y)}[d(x,y)^p] \right)^{\frac{1}{p}}$$

  - 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。

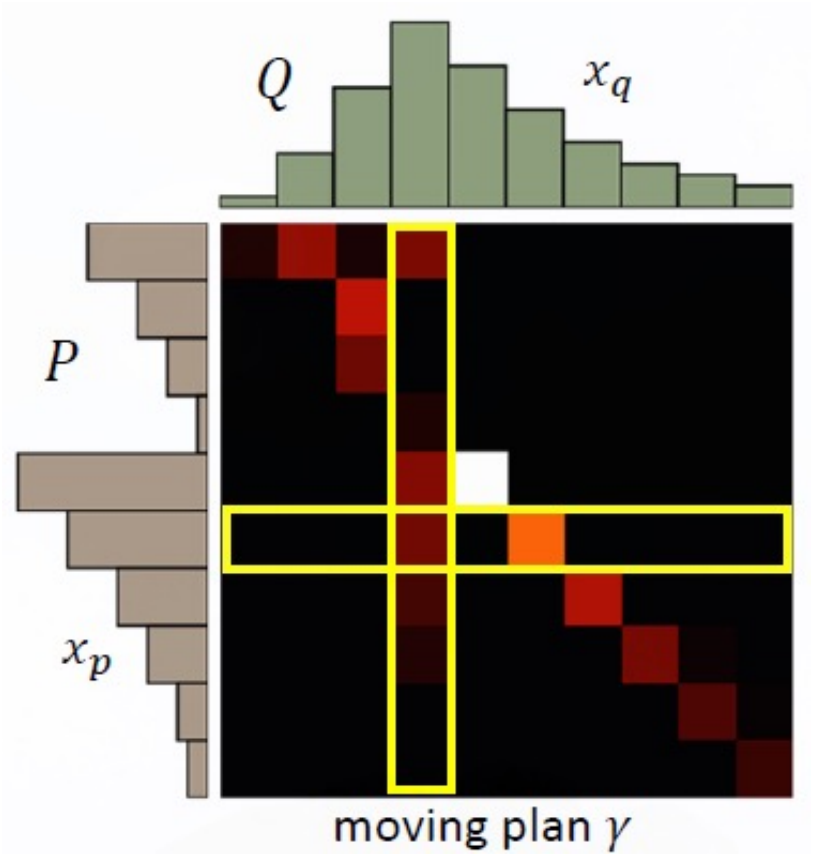
Wasserstein距离

- 如果将两个分布看作是两个土堆，联合分布γ(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）。



(a) $q_1(x)$　(b) $q_2(x)$　(c) q_1 到 q_2 最优的运输方案

Wasserstein距离



DCGAN | LSGAN | Original WGAN | Improved WGAN

G: CNN, D: CNN

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

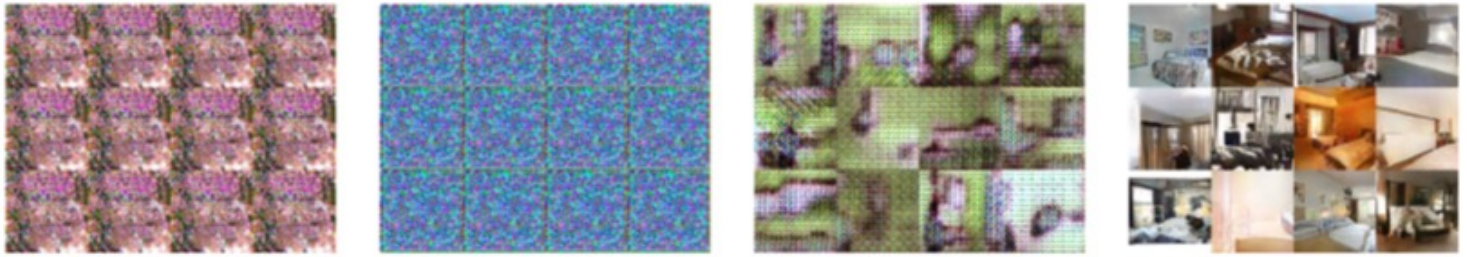
G: CNN (tanh), D: CNN(tanh)

DCGAN | LSGAN | Original WGAN | Improved WGAN

G: MLP, D: CNN

G: CNN (bad structure), D: CNN

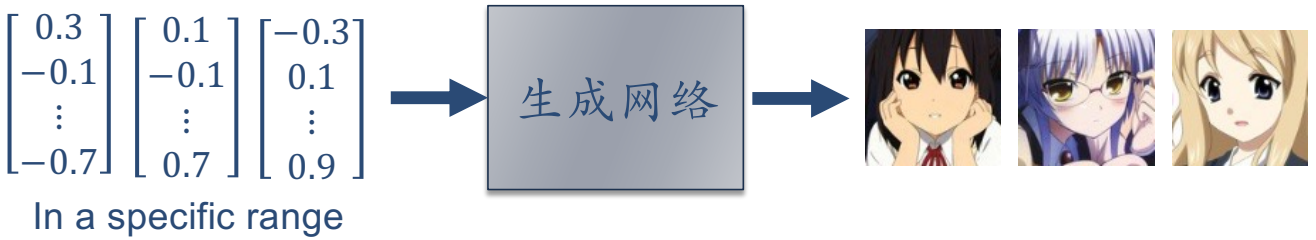
G: 101 layer, D: 101 layer



GAN的扩展

条件生成

- 根据条件针对性的生成数据



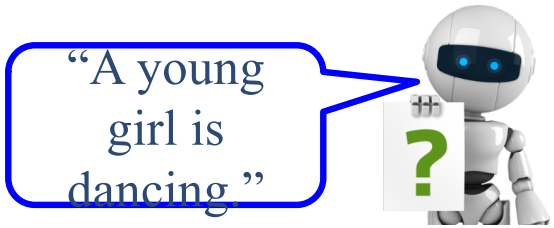
“Girl with red hair and red eyes”

“Girl with yellow ribbon”

条件生成

Caption Generation

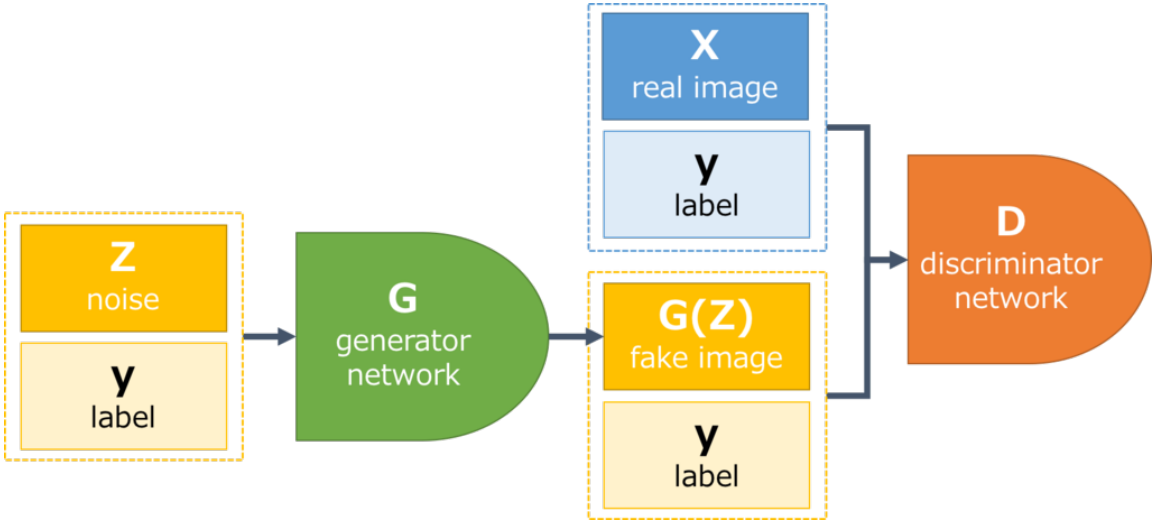
Given condition:

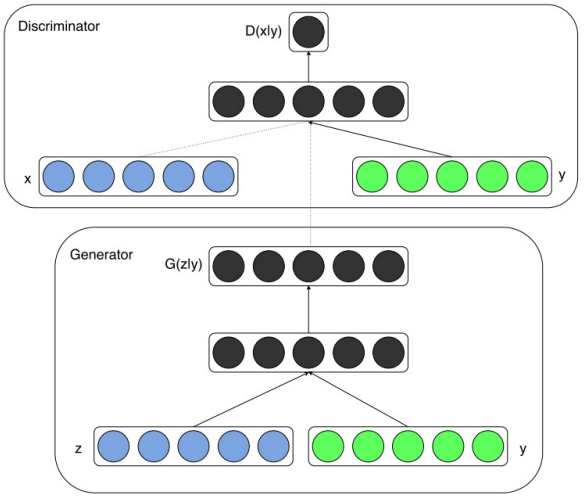


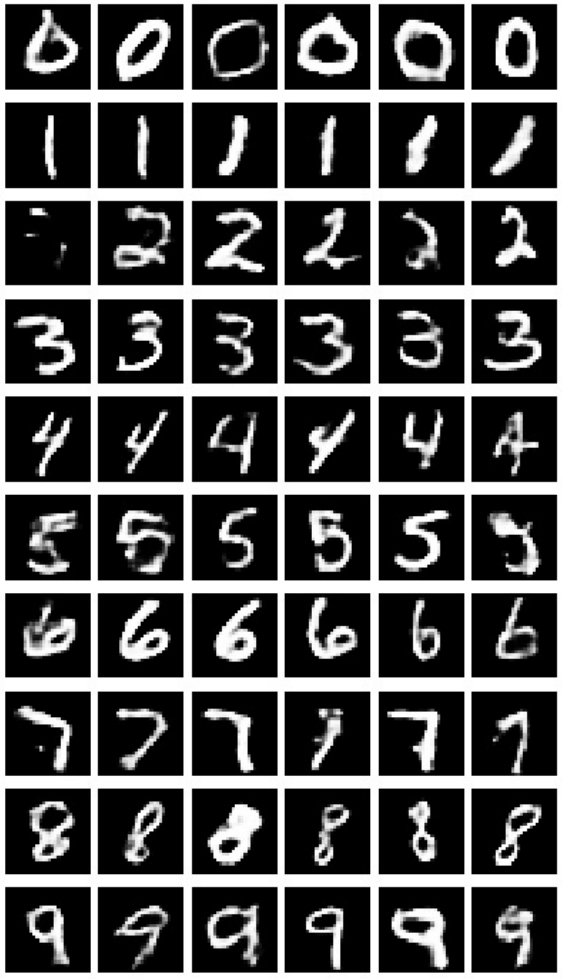
Chat-bot

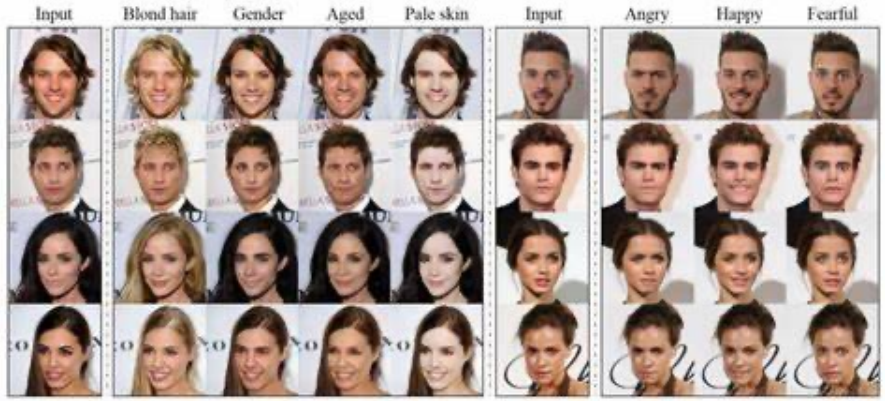
Given condition:

Conditional GAN







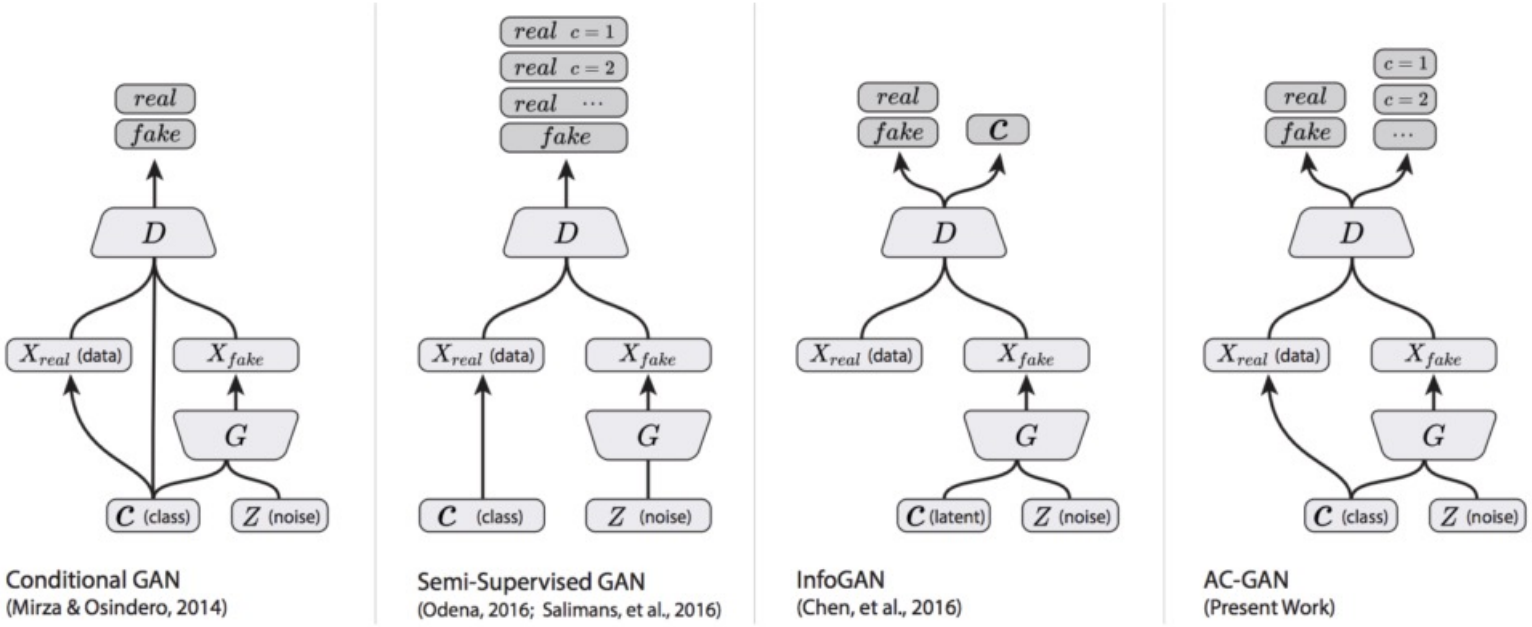

Input
Blond hair
Gender
Aged
Pale skin
Input
Angry
Happy
Fearful

(a) Rotation (b) Width

(c) Lighting (d) Wide or Narrow

https://arxiv.org/abs/1606.03657

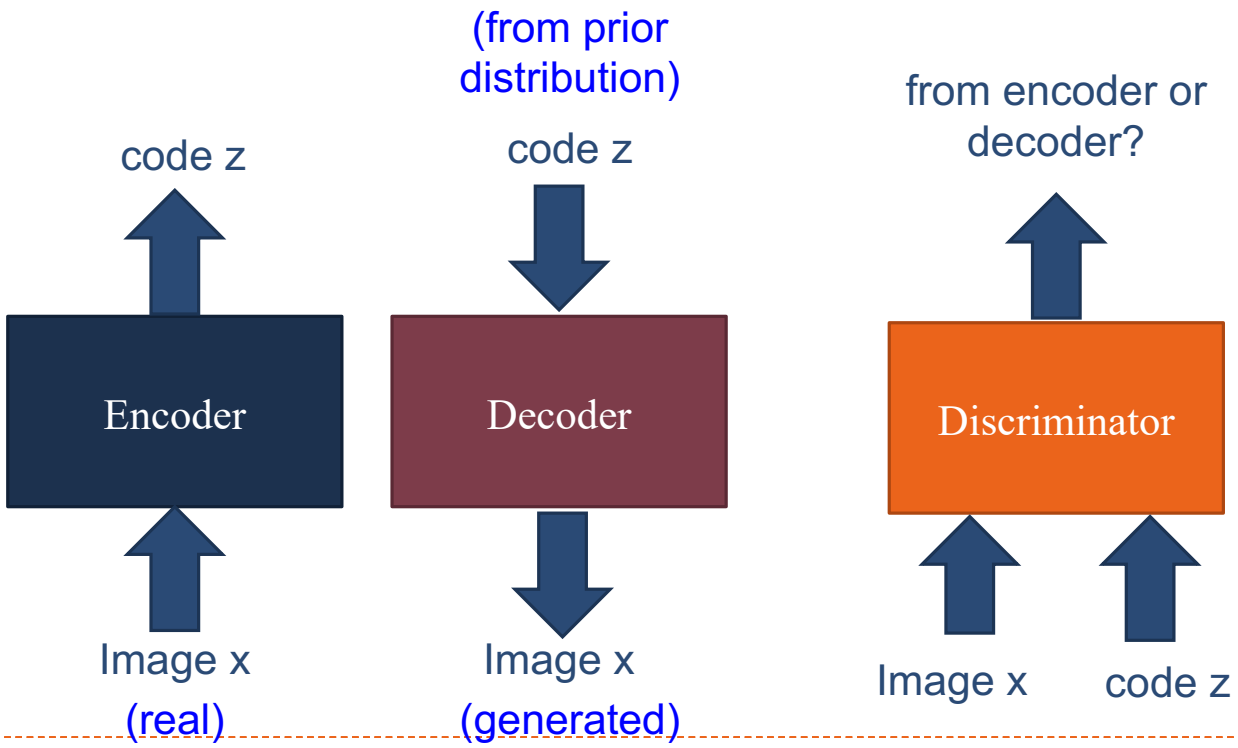
AC-GAN


real
fake
D
X_{real} (data)
X_{fake}
G
C (class)
Z (noise)
Conditional GAN
(Mirza & Osindero, 2014)
real c = 1
real c = 2
real ⋯
fake
D
X_{real} (data)
X_{fake}
G
C (class)
Z (noise)
Semi-Supervised GAN
(Odena, 2016; Salimans, et al., 2016)
real
fake
C
D
X_{real} (data)
X_{fake}
G
C (latent)
Z (noise)
InfoGAN
(Chen, et al., 2016)
c = 1
c = 2
⋯
real
fake
D
X_{real} (data)
X_{fake}
G
C (class)
Z (noise)
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/