Generative Models: Fundamentals and Applications

Lecture 7: Generative Adversarial Networks

Shuigeng Zhou, Yuxi Mi
College of CSAI

November 17, 2025

目录

- 无似然学习（Likelihood-Free Learning）
- 生成对抗网络
- 经典模型

回顾

- 生成模型：极大似然估计MLE（近似）
  - 朴素贝叶斯

$$p(x, y|\theta) = p(y|\pi) \sum_{i=1}^{D} p(x_i|y, \theta_i)$$

  - 主题模型

$$p(\theta, \varphi, \boldsymbol{z}|\boldsymbol{w}, \alpha, \beta) = \frac{p(\boldsymbol{w}, \boldsymbol{z}, \theta, \varphi|\alpha, \beta)}{p(\boldsymbol{w}|\alpha, \beta)}$$

  - 高斯混合模型

$$p(x|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

  - 隐马尔科夫模型
    - 给定观测序列$\mathbf{x}_{1:T}$，学习模型参数θ，使得生成该观测序列的概率最大$\max p(\mathbf{x}_{1:T}|\theta)$

回顾

- 生成模型：极大似然估计MLE（近似）
  - 自回归模型

$$p(x|\theta) = \prod_{i=1}^{D} p_\theta(x_i|x_{<i})$$

  - 能量模型

$$p_\theta(x) = \frac{1}{Z(\theta)} \exp(f_\theta(x))$$

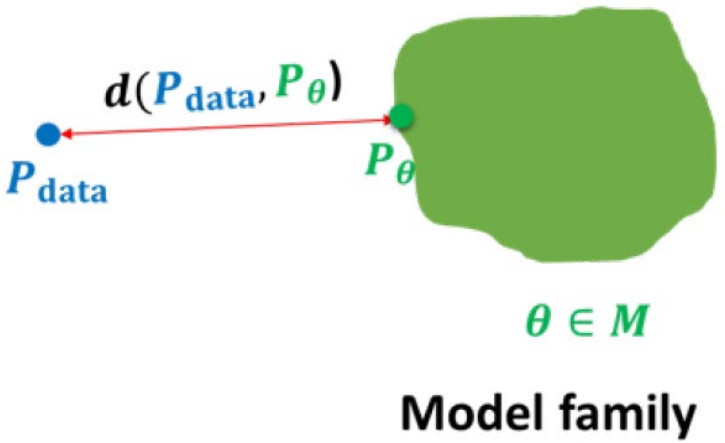
回顾

- 给定n个独立同分布的数据

$$x_1, x_2, \ldots, x_n \sim p_{data}(x)$$

- 令$p_\theta(x)$是估计的概率模型

$x_i \sim P_{\text{data}}$
$i = 1, 2, \ldots, n$



回顾

- 最小化两个分布间的距离

$$\min D_{KL}(p_{data}(x) \| p_\theta(x)) = \mathbb{E}_{p_{data}}\left[\log \frac{p_{data}(x)}{p_\theta(x)}\right]$$

$$= \mathbb{E}_{p_{data}} \log p_{data}(x) - \mathbb{E}_{p_{data}} p_\theta(x)$$

$$= constant - L(\theta)$$

  - 等价于极大似然估计

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(x_i) \rightarrow \mathbb{E}_{p_{\text{data}}}\left[\log p_\theta(x)\right]$$

为什么需要无似然学习？

- 问：高似然函数值=高质量生成样本？
  - 高似然函数值，低质量生成样本
    - E.g., 离散噪声混合模型

低质量噪声样本

$$p_\theta(\mathbf{x}) = 0.01 p_{\text{data}}(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})$$

    - 对数似然函数下界

$$\begin{aligned}\log p_\theta(\mathbf{x}) &= \log[0.01 p_{\text{data}}(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})] \\ &\geq \log 0.01 p_{\text{data}}(\mathbf{x}) \\ &= \log p_{\text{data}}(\mathbf{x}) - \log 100\end{aligned}$$

为什么需要无似然学习？

- 问：高似然函数值=高质量生成样本？
  - 高似然函数值，低质量生成样本
    - 期望对数似然函数下界

$$E_{p_{\text{data}}}[\log p_\theta(\mathbf{x})] \geq E_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})] - \log 100$$

    - 期望对数似然函数上界（$KL(p_{\text{data}}, p_\theta) \geq 0$）

$$E_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x}))] \geq E_{p_{\text{data}}}[\log p_\theta(\mathbf{x})]$$

    - 当x的维度增加时，$|\log p_{\text{data}}(\mathrm{x})|$逐渐增大

高维时非常大
导致上下界差距小

$$E_{p_{\text{data}}}[\log p_\theta(\mathbf{x})] \approx E_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})]$$

为什么需要无似然学习？

- 问：高似然函数值=高质量生成样本 ？
  - 低似然函数值，高质量生成样本
    - E.g., 记忆训练集（Memorizing training set）

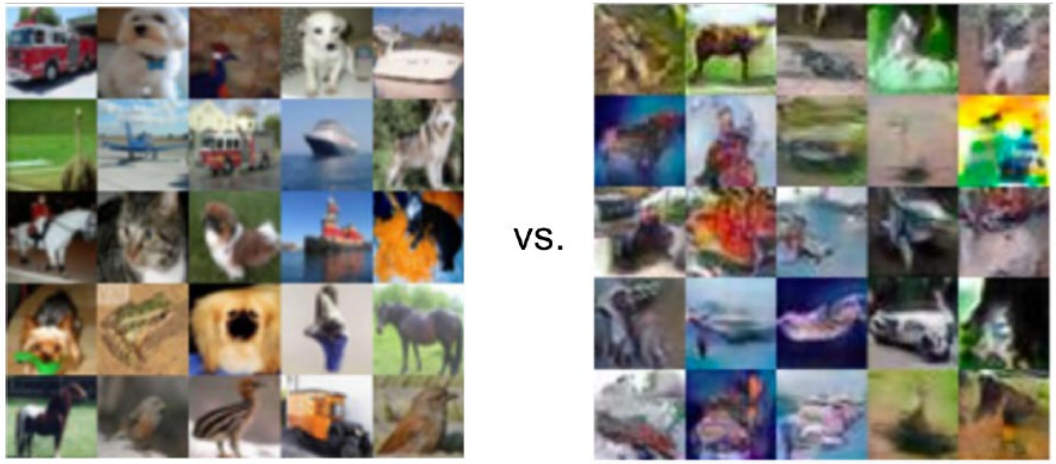
$$q(\mathbf{x}) = \frac{1}{N}\sum_{n}\mathcal{N}(\mathbf{x};\mathbf{x}_n,\varepsilon^2\mathbf{I}),$$

      - 生成样本和训练样本看起来一模一样
      - $\varepsilon \to 0$，在训练样本点取得高概率密度，在未见样本上概率几乎为0
        此时对数似然很低

- 结论：生成样本质量 $\not\Leftrightarrow$ 似然函数值

无似然学习学习

- 研究不直接依赖于似然函数的目标函数
  - 双样本检验
    - 给定两个样本集合$S_1 = \{x \sim P\}$和$S_2 = \{x \sim Q\}$，判断这些样本是否服从同一个分布（i.e. $P = Q$）

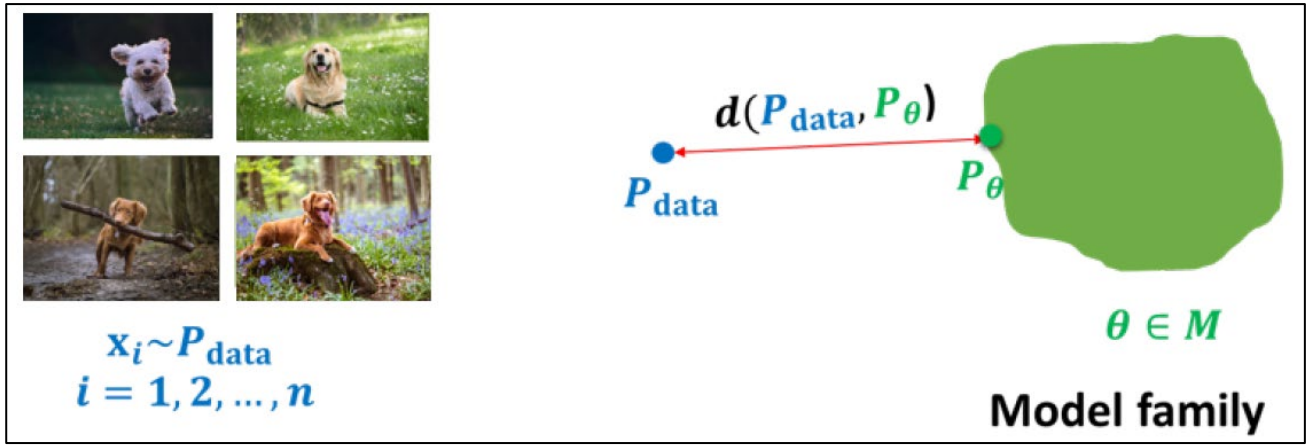


$S_1 = \{\mathbf{x} \sim P\}$　　$S_2 = \{\mathbf{x} \sim Q\}$

双样本检验

- 双样本检验
  - 考虑以下两个假设
    - 原假设 H_0： $P = Q$
    - 备择假设 H_1： $P \neq Q$
  - 检验统计量 T，例如
    - 均值的差：$\mu_1 - \mu_2$；标准差的差： $\sigma_1 - \sigma_2$
  - 如果$|T| < \alpha$，那么接受H_0，否则拒绝H_0
- 双样本检验天然是无似然的：
  - 检验统计量 T 只和给定样本有关
  - 不需要知道概率密度函数的形式，与似然函数无关

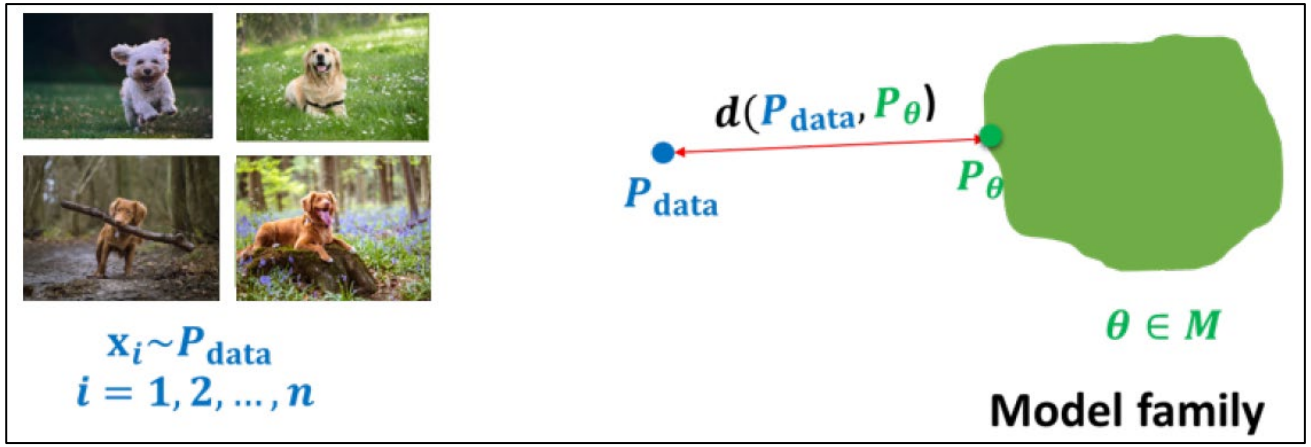
双样本检验 vs. 生成模型



- 选择$S_1 = \mathcal{D} = \{x \sim p_{\text{data}}\}$
- 选择$S_2 = \{x \sim p_\theta\}$
- 目标函数：最小化双样本检验的检验统计量 $d(p_{\text{data}}, p_\theta)$

$$\min_\theta d(p_{data}, p_\theta)$$

双样本检验 via 判别模型



- 检验统计量 $d(p_{\text{data}}, p_\theta)$ 怎么求？
  - 判别器

$$y = \begin{cases} 1, & \text{if } x \in S_1 \\ 0, & \text{if } x \in S_2 \end{cases}$$

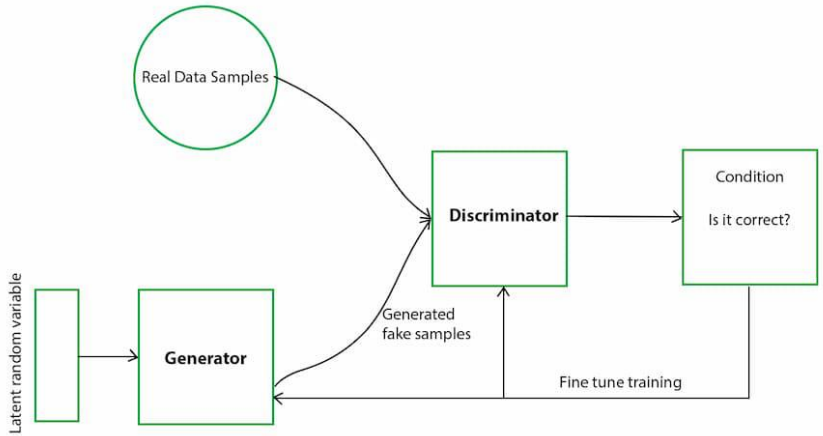
  - 等价于学习一个能最大化样本集 S_1, S_2 间的距离的最优检验统计量

目录

- 无似然学习（Likelihood-Free Learning）
- 生成对抗网络
- 经典模型

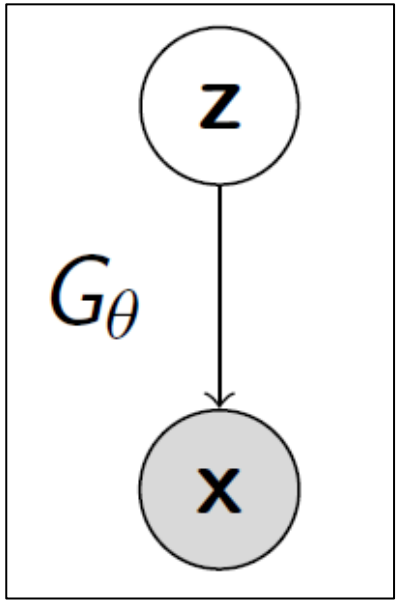
生成对抗网络

- 生成对抗网络 (Generative Adversarial Network, GAN)
  - 整体：两层 minmax 博弈结构
  - 生成器
    - 生成样本，使其和真实样本尽可能接近
  - 判别器
    - 区分真实样本和生成样本



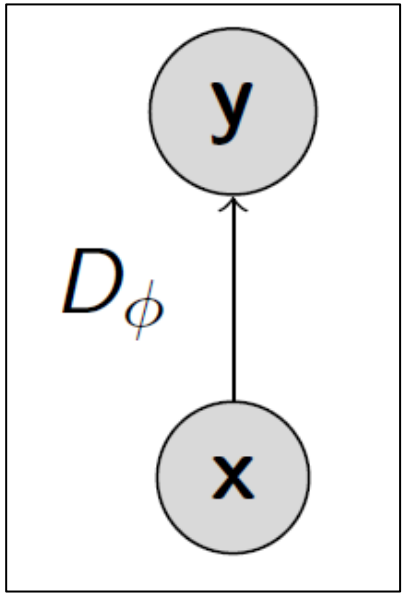
生成对抗网络

- 生成器
  - 有向、隐变量模型
    - 从简单隐变量 z 出发
    - 通过网络 G_θ 映射生成样本 x
  - 目标：最小化双样本检验的统计量
    - 支持原假设 $p_{\text{data}} = p_\theta$



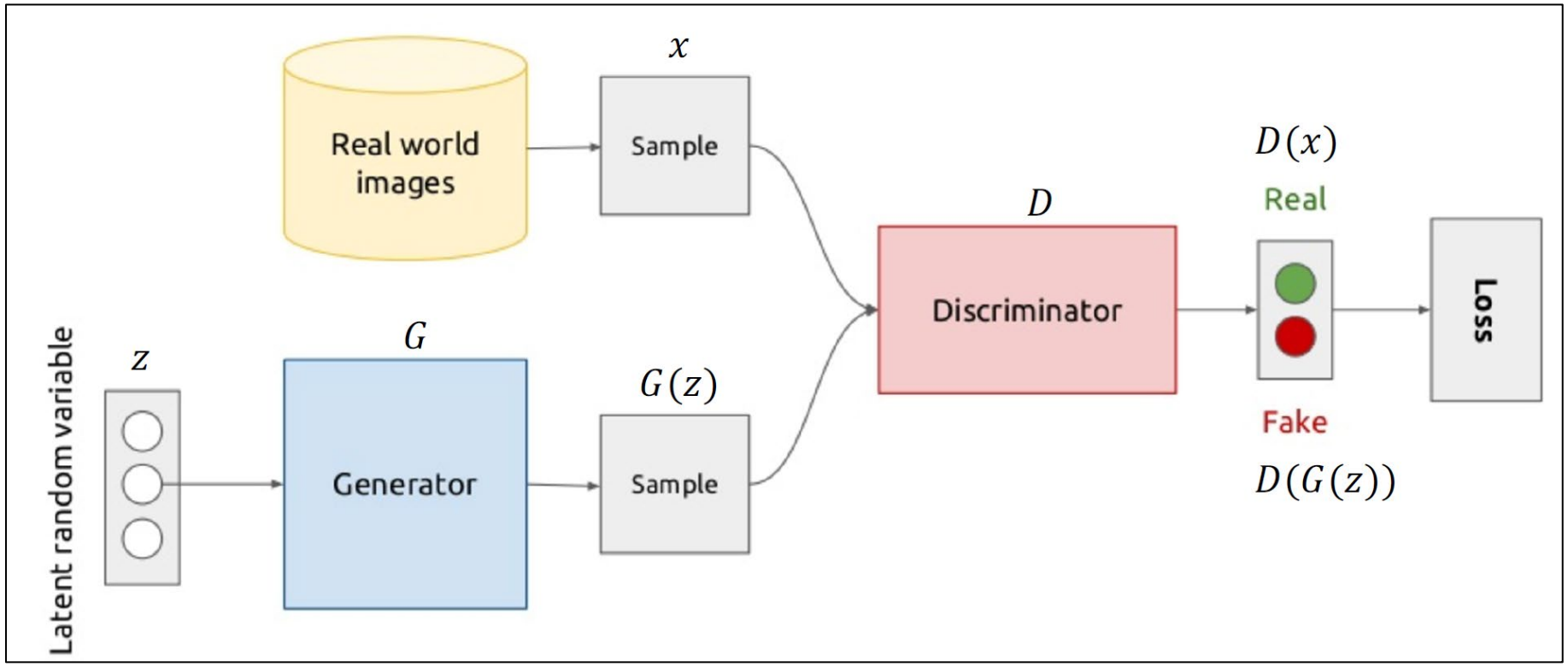
生成对抗网络

- 判别器
  - 有向、可测变量模型
    - 接收样本 x，输出可观测的分数或概率 y
  - 目标：最大化双样本检验的统计量
    - 支持备择假设 $p_{\text{data}} \neq p_\theta$



生成对抗网络

- 整体流程


Real world images
x
Sample
$D(x)$
Real
D
Discriminator
Loss
Latent random variable
z
G
Generator
$G(z)$
Sample
Fake
$D(G(z))$

生成对抗网络

- 目标函数：min-max 优化问题

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

  - $D(x)$：x 是真实样本的概率
    - 对真实样本 $x\sim p_{\text{data}}$，$D(x)=1$；对生成样本 $x\sim p_G$，$D(x)=0$
  - 生成器/判别器的目标是最小/最大化：
    - 真实样本被正确识别为真的概率 $\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})]$
    - 和，生成样本被正确识别为假的概率 $\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]$
    - 等价于二分类交叉熵损失的负值

生成对抗网络：判别器

- 判别器的学习
  - 固定生成器 G，判别器 D 的目标函数为

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

  - 最大化负交叉熵损失
    - 最大化训练样本被标记为正类的对数似然函数
    - 最大化生成样本被标记为负类的对数似然函数

生成对抗网络：判别器

- 最优判别器

$$\nabla_D V(G,D) = \nabla_D \int_x p_{\text{data}}(x)\log D(x)\ dx + \nabla_D \int_z p_z(z)\log(1-D(G(z)))\ dz$$

真实样本的偏导项　　生成样本的偏导项

$$= \nabla_D \int_x p_{\text{data}}(x)\log D(x)\ dx + \nabla_D \int_x p_G(x)\log(1-D(x))\ dx$$

$$= \int_x \nabla_D \left[p_{\text{data}}(x)\log D(x) + p_G(x)\log(1-D(x))\right]dx$$

$$= \int_x \left[\frac{p_{\text{data}}(x)}{D(x)} - \frac{p_G(x)}{1-D(x)}\right]dx = 0$$

真实、生成样本具有相同的“判断强度”

$$\Rightarrow \quad D_G^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}$$

  - $p_{\text{data}}(x) + p_G(x)$ 是归一化因子

生成对抗网络：生成器

- 生成器的学习
  - 固定判别器 D，生成器 G 的目标函数为

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

$$\Longleftrightarrow \quad \min_G \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]$$

  - 最小化负交叉熵损失
    - 最小化生成样本被标记为负类的对数似然函数

生成对抗网络：生成器

- 将最优判别器D_G^*带入目标函数

$$C(G) = \max_D V(G,D) = V(G, D_G^*(\mathbf{x}))$$

$$= E_{\mathbf{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}\right] + E_{\mathbf{x}\sim p_G}\left[\log \frac{p_G(\mathbf{x})}{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}\right]$$

$$= E_{\mathbf{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right] + E_{\mathbf{x}\sim p_G}\left[\log \frac{p_G(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right] - \log 4$$

$$= D_{KL}\left[p_{\text{data}}, \frac{p_{\text{data}}+p_G}{2}\right] + D_{KL}\left[p_G, \frac{p_{\text{data}}+p_G}{2}\right] - \log 4$$

两个KL散度的平均

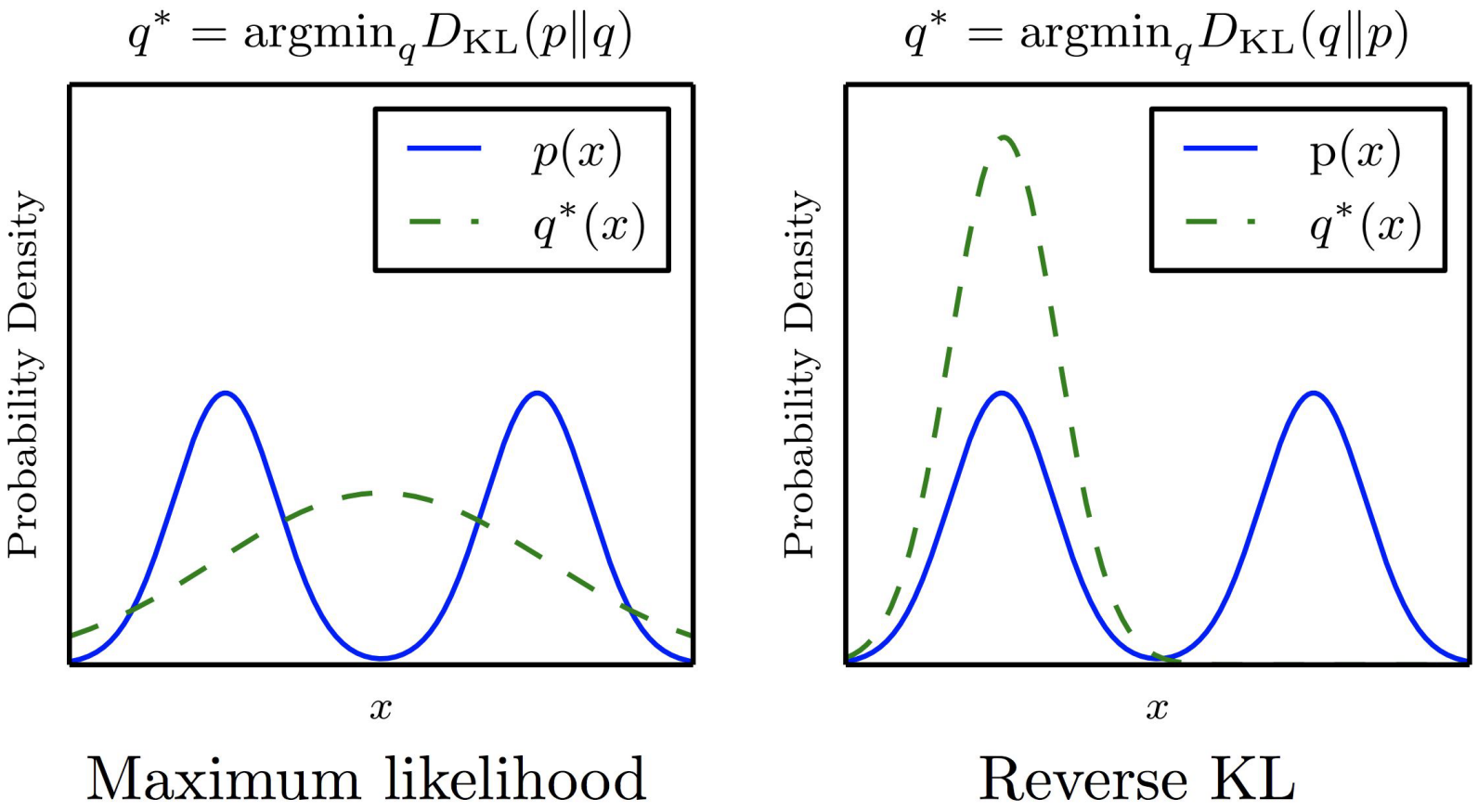
Jenson-Shannon散度

- Jenson-Shannon (JS) 散度：亦称为对称KL散度

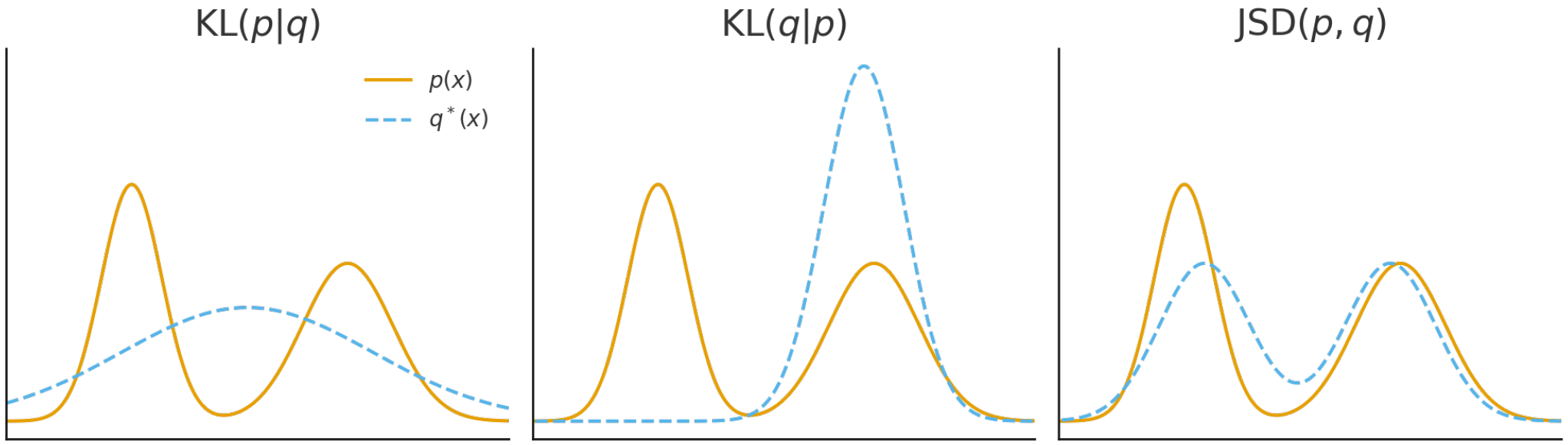
$$D_{JSD}[p,q] = \frac{1}{2}\left(D_{KL}\left[p, \frac{p+q}{2}\right] + D_{KL}\left[q, \frac{p+q}{2}\right]\right)$$

  - 性质
    - $D_{JSD}[p,q] \geq 0$
    - $D_{JSD}[p,q] = 0$ 当且仅当 p=q
    - $D_{JSD}[p,q] = D_{JSD}[q,p]$

KL散度方向的影响


$q^* = \text{argmin}_q D_{\text{KL}}(p\|q)$
$p(x)$
$q^*(x)$
Probability Density
x
Maximum likelihood
$q^* = \text{argmin}_q D_{\text{KL}}(q\|p)$
p(x)
$q^*(x)$
Probability Density
x
Reverse KL

Jenson-Shannon散度



生成对抗网络

- 将最优判别器D_G^*带入目标函数

$$C(G) = \max_D V(G, D) = V(G, D_G^*(\mathbf{x}))$$
$$= 2D_{JSD}[p_{\text{data}}, p_G] - \log 4$$

- 最优生成器：$p_{\text{data}} = p_G$
  - 对于最优生成器G^*和最优判别器D_G^*

$$V(G^*, D_{G^*}^*(\mathbf{x})) = -\log 4$$

算法

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do** 多次更新判别器

- 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:

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

更新生成器

end for

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

算法

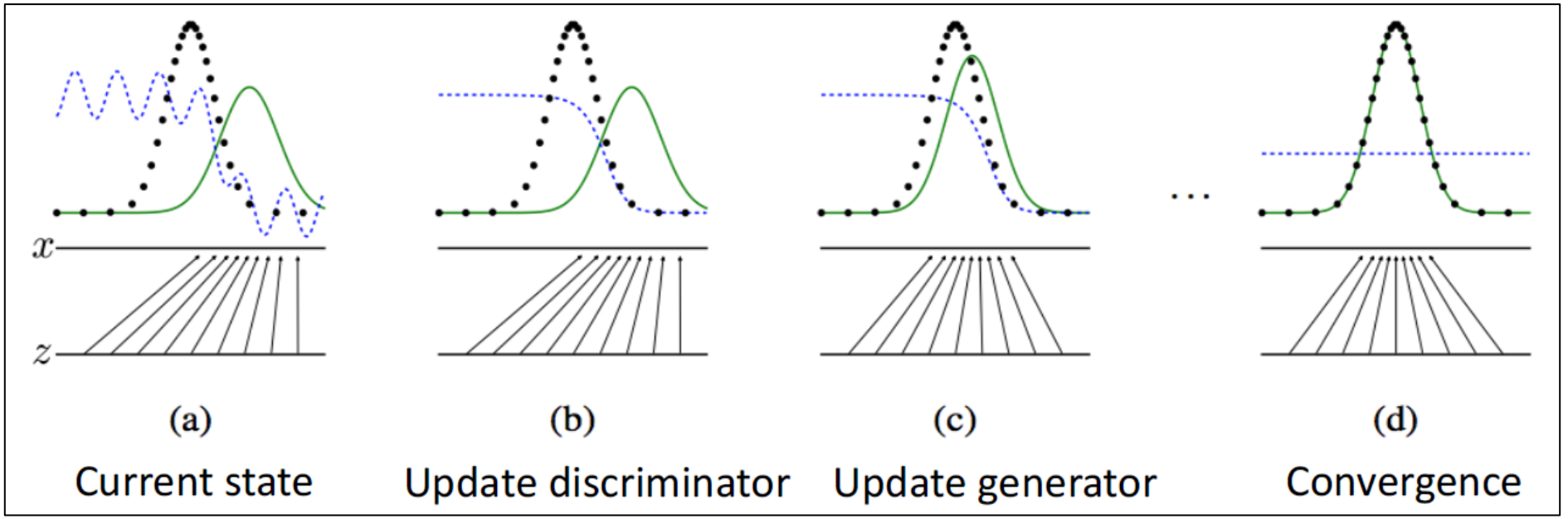
- 迭代计算G和D的模型参数

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

  - （理论）收敛点：$p_{\text{data}} = p_G$

$$\Rightarrow \quad D_G^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)} = \frac{1}{2}$$

算法



黑色：数据真实分布　　绿色：生成数据分布　　蓝色：判别器

损失函数

- 判别损失：交叉熵损失

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

- 生成损失
  - Min-max博弈：

$$J^{(G)} = -J^{(D)}$$

  - 启发式算法：实际使用

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

损失函数

- 生成损失
  - Minmax博弈

$$J^{(G)} = -J^{(D)}$$

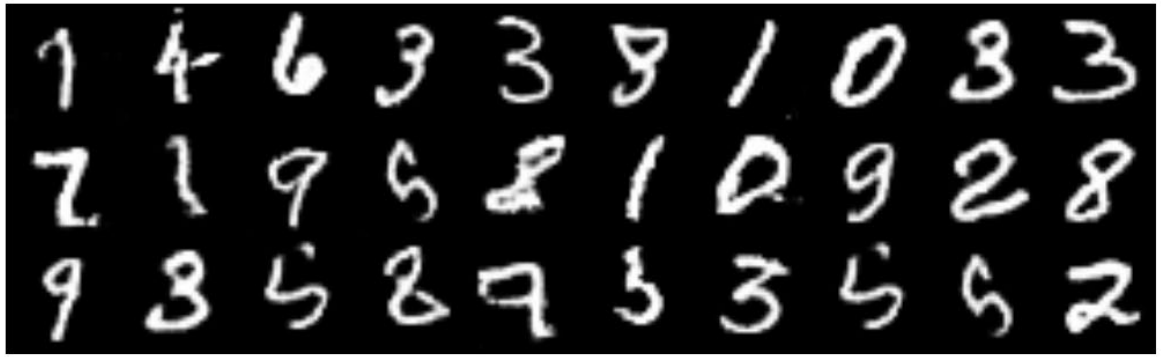
    - 生成器：生成数据尽可能近似真实数据
    - 判别器正确分类的概率达到最低
    - 判别损失最大
  - 启发式算法：

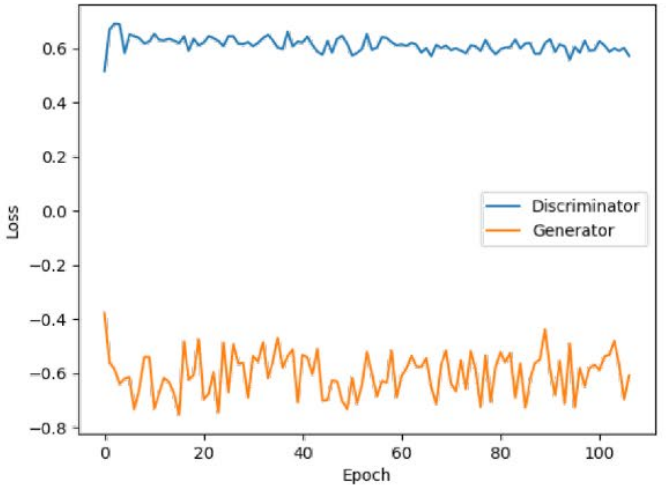
$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

    - 生成器：生成数据尽可能近似真实数据
    - 生成数据被判别为正类的概率尽可能大
    - 负对数似然函数最小

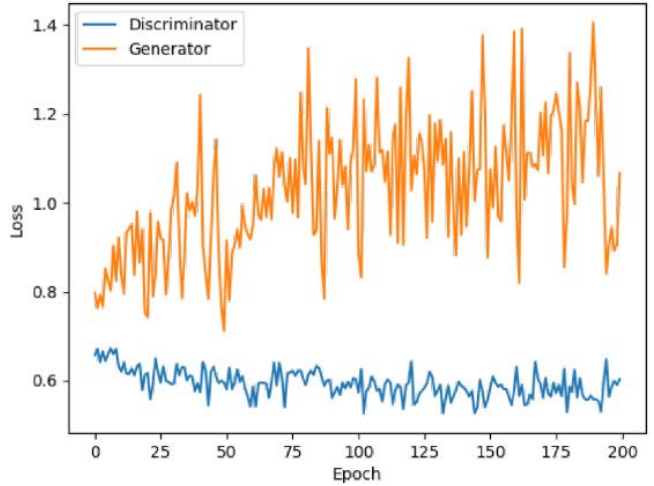
损失函数

Minimax



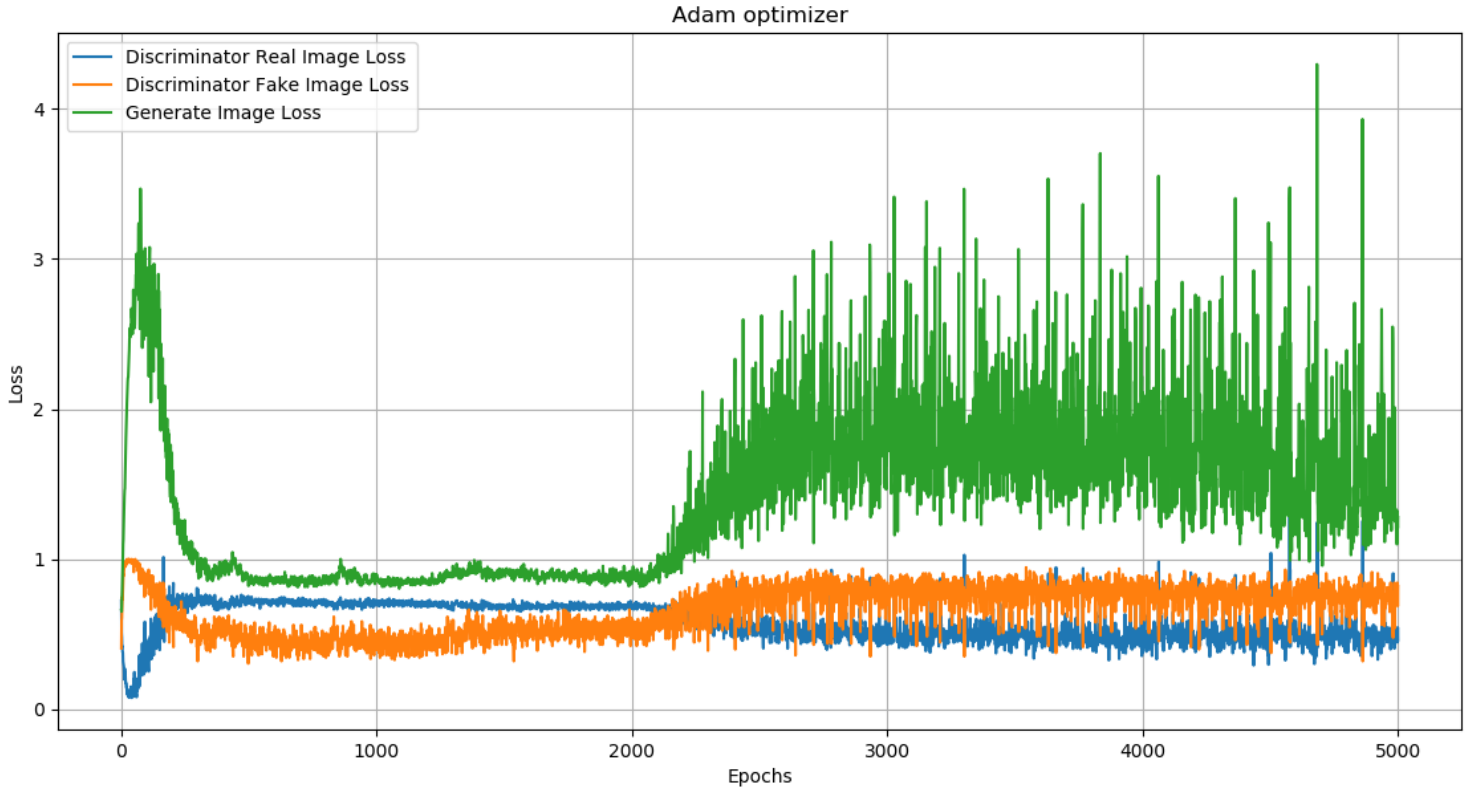


Heuristic



优化迭代

- 优化过程不稳定，没有鲁棒的迭代停止点



生成器梯度消失

- 判别器越好，生成器梯度消失越严重
  - 最优判别器

$$D_G^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}$$

  - 生成器损失

$$V(G, D_G^*(\mathbf{x})) = 2D_{JSD}[p_{\text{data}}, p_G] - \log 4$$

  - 如果判别器足够强：最小化生成器损失，等价于最小化真实、生成分布间的JS散度

生成器梯度消失

- 判别器越好，生成器梯度消失越严重
  - 如果真实分布和生成分布完全没有重叠，或者重叠部分可忽略
    - 四种可能情况

$$P_1(x) = 0 \text{且} P_2(x) = 0$$

$$P_1(x) \neq 0 \text{且} P_2(x) \neq 0$$

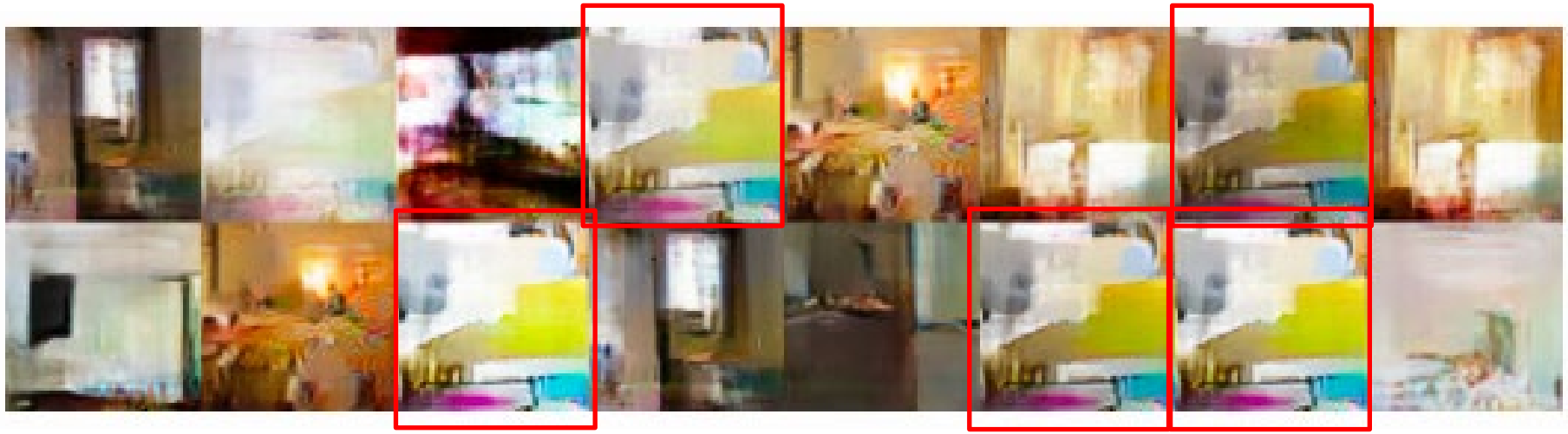
$$P_1(x) = 0 \text{且} P_2(x) \neq 0$$

$$P_1(x) \neq 0 \text{且} P_2(x) = 0$$

    - $D_{JSD}[p_{data}, p_G] = \log 2$
    - 生成器面临梯度消失的问题

模式崩溃

- 生成器产生单个或有限的模式

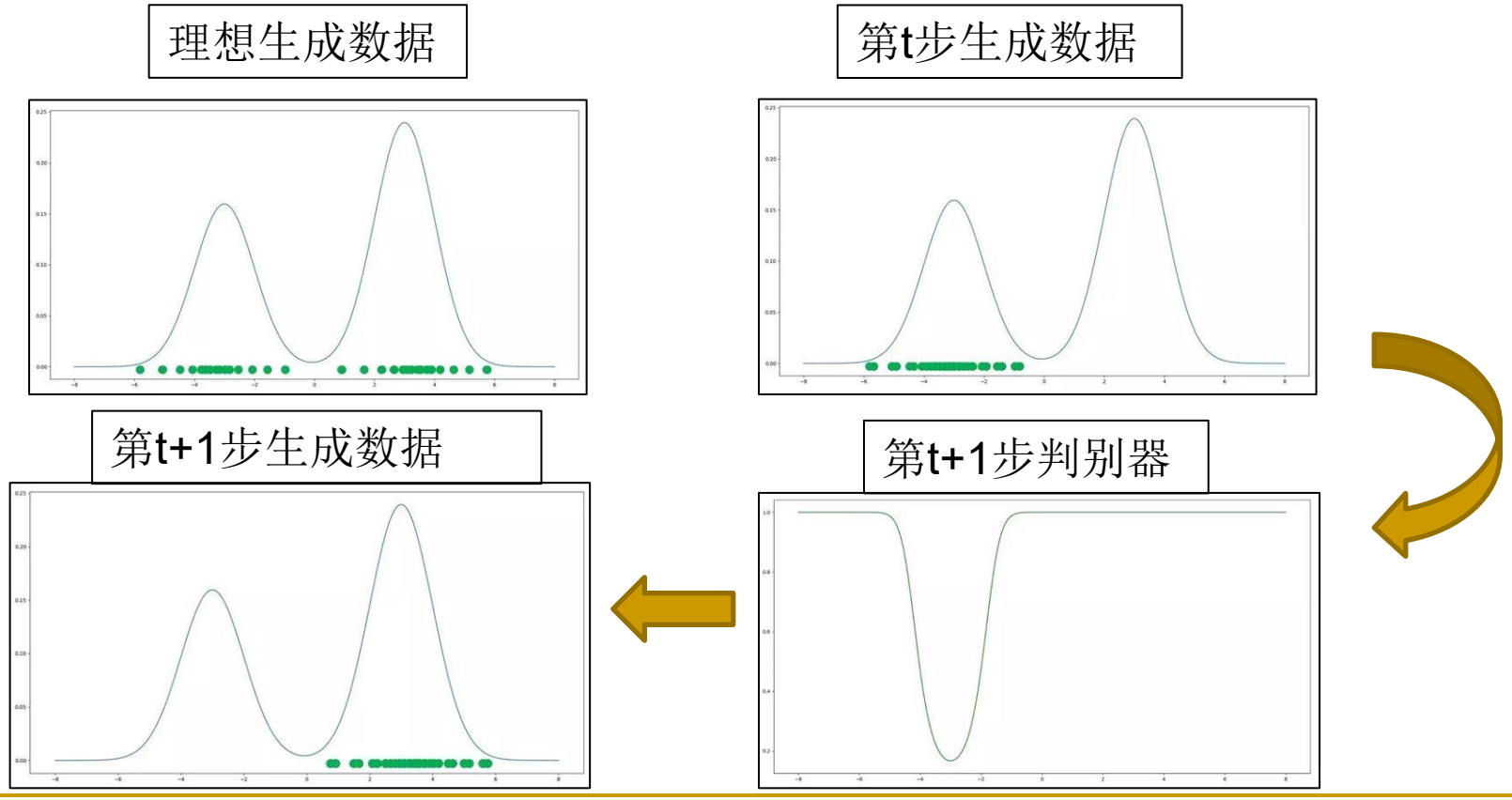


模式崩溃

- 生成器产生单个或有限的模式
  - Eg：真实分布为混合高斯模型

模式崩溃

- Eg: 真实分布为混合高斯模型



模式崩溃

- 针对目标函数的改进方法
  - Unrolled GAN
  - EBGAN
- 针对网络结构的改进方法
  - Multi agent diverse GAN(MAD-GAN)
    - 采用多个生成器，一个判别器以保障样本生成的多样性

模式崩溃

- Unrolled GAN
  - 修正生成器的优化目标

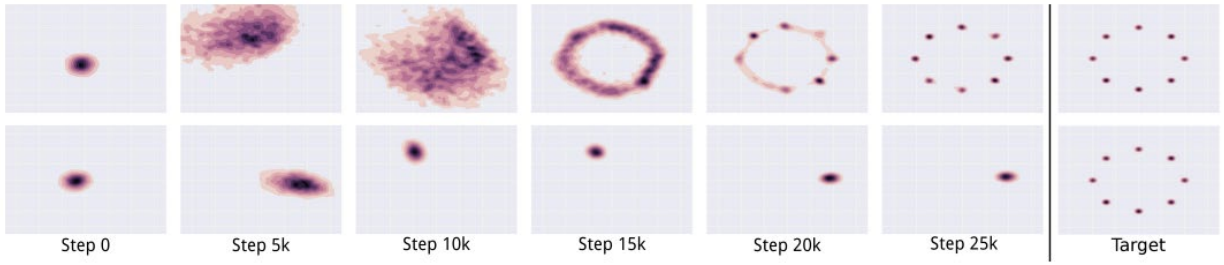
$$\theta_D^0 = \theta_D$$
$$\cdots\cdots$$
$$\theta_D^K = \theta_D^{K-1} + \eta \frac{\partial f(\theta_G, \theta_D^{K-1})}{\partial \theta_D^{K-1}}$$

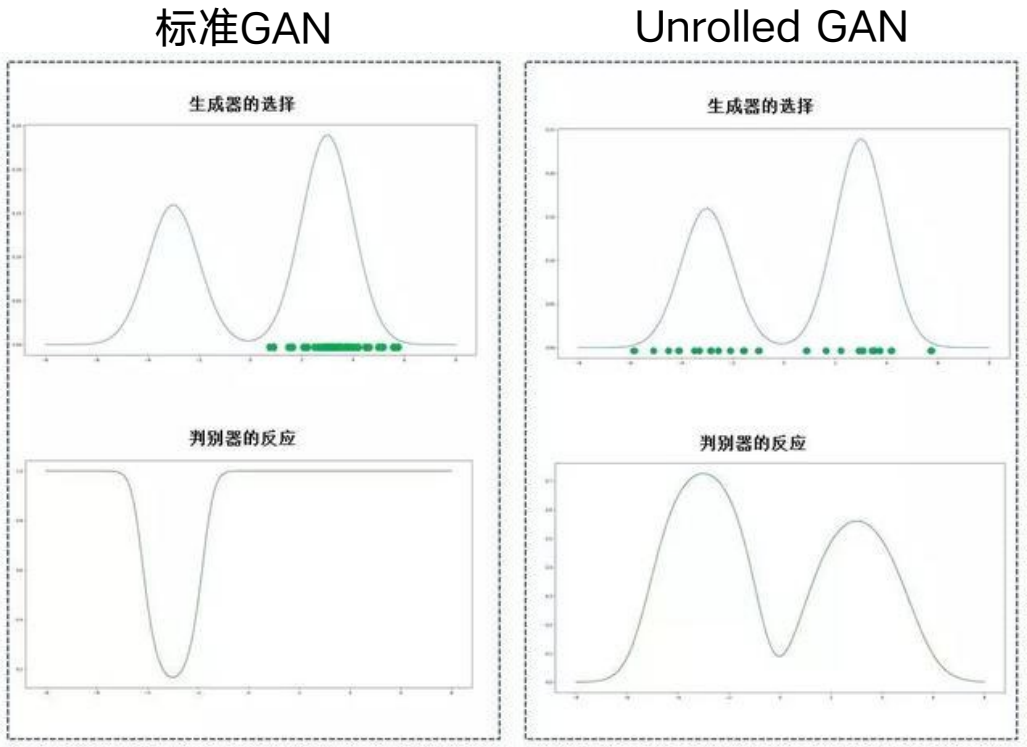
$$\frac{df_K(\theta_G, \theta_D)}{d\theta_G} = \frac{\partial f(\theta_G, \theta_D^K(\theta_G, \theta_D))}{\partial \theta_G} + \frac{\partial f(\theta_G, \theta_D^K(\theta_G, \theta_D))}{\partial \theta_D^K(\theta_G, \theta_D)} \frac{\partial \theta_D^K(\theta_G, \theta_D)}{\partial \theta_G}$$

    - 不仅仅考虑当前生成器的状态，还会额外考虑以当前状态为起始点，判别器更新K次后的状态，综合两个信息做出最优解

模式崩溃

- Unrolled GAN





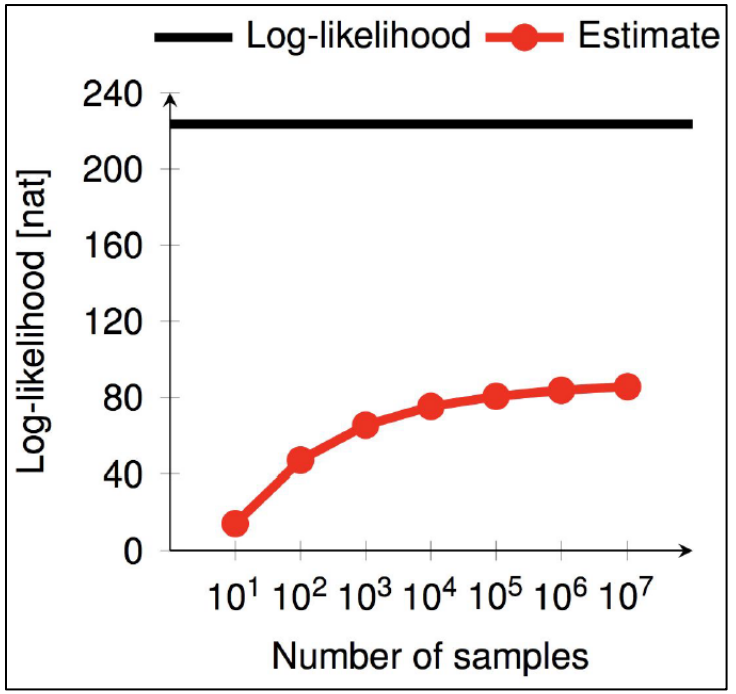
模型评估

- 生成对抗网络是Likelihood-free学习，不能直接估计密度函数

- 核密度估计（Parzen窗）
  - 高斯核函数

$$\hat{p_h}(x) = \frac{1}{nh} \sum_i K\left(\frac{x - x_i}{h}\right)$$

  - 对不同模型的密度估计值进行排序比较



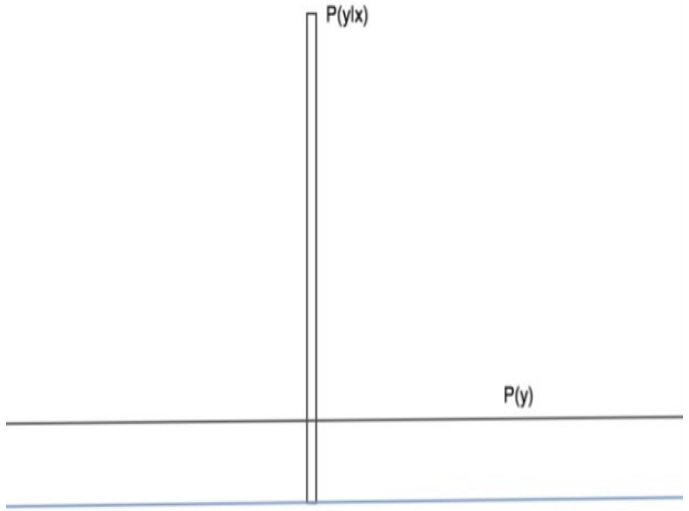
模型评估

- 模型性能
  - 生成数据的质量
  - 生成数据的多样性

- 评价指标
  - Inception Score
  - Fréchet Inception Distance
  - Maximum Mean Discrepancy

Inception Score

- 生成数据的质量
  - $p(y|x)$ 的熵尽可能小
    - 对于一张高质量图片，它属于某一类的概率应该非常大，而属于其它类的概率应该很小
- 生成数据的多样性
  - $p(y)$ 的熵尽可能大
    - 希望标签分布均匀，而不希望模型生成的都是某一类图片



Inception Score

- Inception Score公式

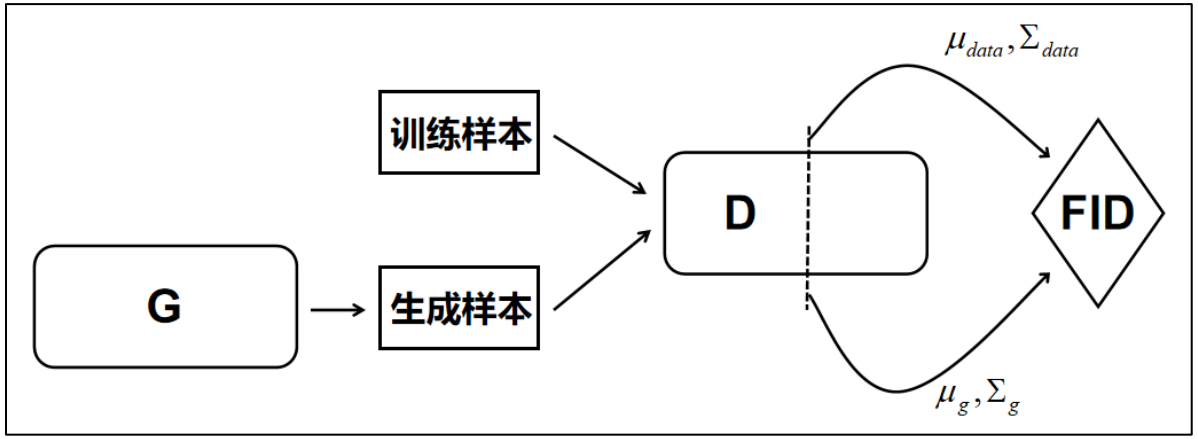
$$\begin{aligned}\mathrm{IS}(x) =& \exp\left(\mathbb{E}_{x\sim p_g}\left[D_{\mathrm{KL}}\left[p(y|x)\parallel p(y)\right]\right]\right)\\ =& \exp\left(\mathbb{E}_{x\sim p_g, y\sim p(y|x)}\left[\log p(y|x) - \log p(y)\right]\right)\\ =& \exp\left(H(y) - H(y|x)\right)\end{aligned}$$

  - Inception score (IS) 越大越好
  - 缺点：
    - 没有对真实数据和生成数据的统计量（如均值和方差）进行比较

Fréchet Inception Distance

- 计算真实样本和生成样本在特征空间的距离
  - 假设该抽象特征符合多元高斯分布
  - 定义两个高斯分布间的Fréchet距离

$$\| \mu_{data} - \mu_g \| + tr(\Sigma_{data} + \Sigma_g - 2(\Sigma_{data}\Sigma_g)^{\frac{1}{2}})$$



Fréchet Inception Distance

- 目标：FID 越小越好，表明生成图像的统计量与真实图像非常接近
- 优点：
  - 对噪声具有比较好的鲁棒性，能够对生成图像的质量有比较好的评价，其给出的分数与人类的视觉判断比较一致
  - FID的计算复杂度并不高
- 缺点：
  - 高斯分布的简化假设在实际中并不成立

Maximum Mean Discrepancy

- 在希尔伯特空间中度量两个分布的差异
  - 度量真实分布p_{data}和生成分布p_g的距离

$$\mathbb{E}_{x,x'\sim p_{data}}[k(x,x')] - 2\mathbb{E}_{x\sim p_{data},y\sim p_g}[k(x,y)] + \mathbb{E}_{y,y'\sim p_g}[k(y,y')]$$

  - 度量真实数据集$x_1,\dots,x_n$和生成数据集$y_1,\dots y_n$的距离

$$\frac{1}{C_n^2}\sum_{i\neq i'}k(x_i,x_i') - \frac{2}{C_n^2}\sum_{i\neq j}k(x_i,y_j) + \frac{1}{C_n^2}\sum_{j\neq j'}k(y_j,y_j')$$

GAN实际使用技巧

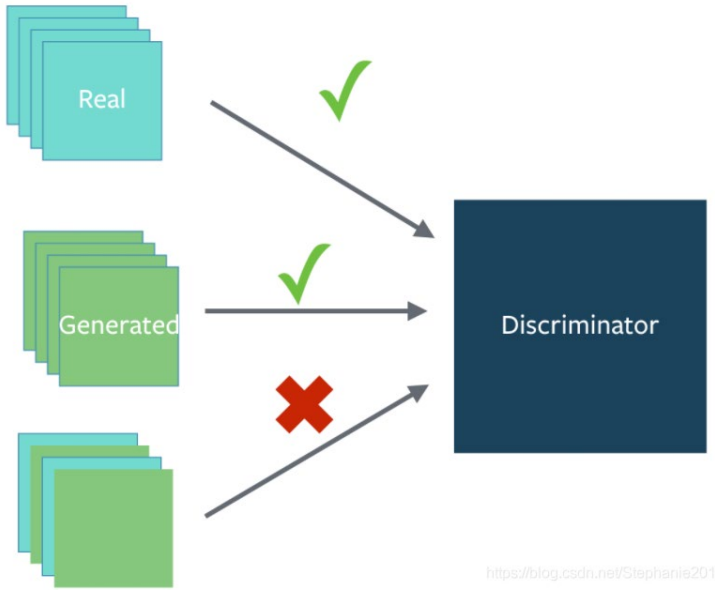
- 输入归一化
  - 把输入数据归一化到-1 到1
  - 生成器的最后一层输出用tanh
- 使用修饰过后的损失函数
  - 生成损失

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

- 采样z
  - 不从均匀分布中采样，是从高斯分布中采样

GAN实际使用技巧

- 特征标准化
  - 为真实数据和生成数据构造不同的mini batch，即每个mini batch仅需要包含所有真实图像或所有生成的图像
  - 若不使用batch norm，则使用实例标准化



GAN实际使用技巧

- 梯度计算
  - 判别器：随机梯度下降SGD
  - 生成器：Adam优化

- 目标函数中加入权衡参数λ
  - $\lambda < 1$，防止判别器向生成器提供过大的梯度信号

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\boldsymbol{\lambda}\log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}}\log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

GAN实际使用技巧

- 避免使用ReLU, MaxPool等操作引入稀疏梯度
  - GAN的稳定性会因为引入稀疏梯度受到很大影响
  - 在判别器和生成器中使用LeakyReLU激活函数
  - 下采样使用Average Pooling或者卷积+stride
  - 上采样使用PixelShuffle或者转置卷积+stride

目录

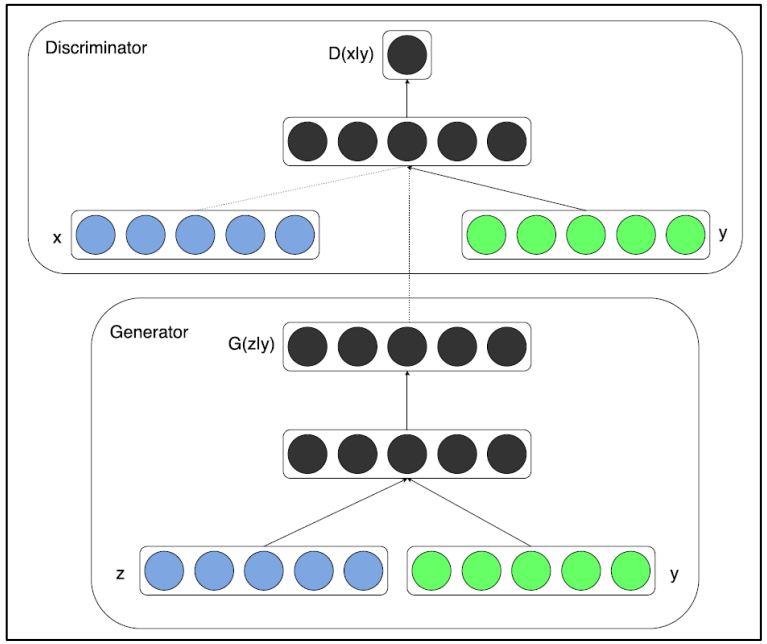
- 无似然学习（Likelihood-free learning）
- 生成对抗网络模型
- 最新进展
  - 条件生成对抗网络（ CGAN ）
  - 基于f散度的生成对抗网络（f-GAN）
  - 深度卷积生成对抗网络（ DCGAN ）
  - Wasserstein GAN （WGAN）

CGAN

- 目标函数

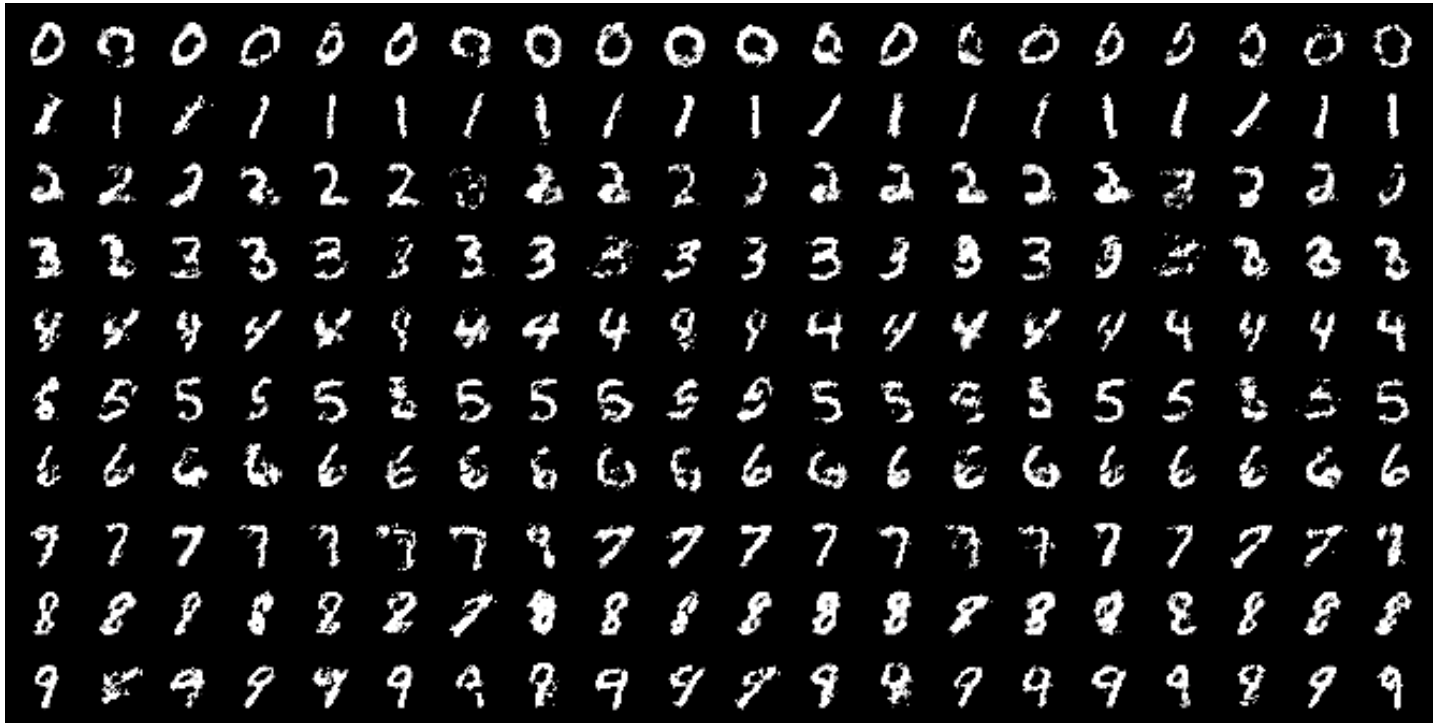
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

  - y: 额外信息，如tag
  - 把传统的无监督GAN
    改成了有监督的GAN



CGAN

- 生成样本多样性



CGAN

- 自动标注

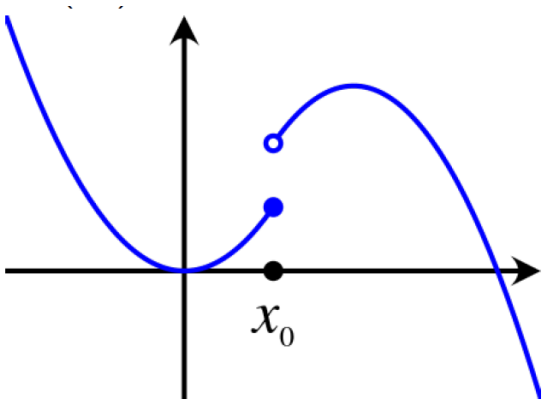
	User tags + annotations	Generated tags
	montanha, trem, inverno, frio, people, male, plant life, tree, structures, transport, car	taxi, passenger, line, transportation, railway station, passengers, railways, signals, rail, rails
	food, raspberry, delicious, homemade	chicken, fattening, cooked, peanut, cream, cookie, house made, bread, biscuit, bakes
	water, river	creek, lake, along, near, river, rocky, treeline, valley, woods, waters
	people, portrait, female, baby, indoor	love, people, posing, girl, young, strangers, pretty, women, happy, life

f-GAN

- f-散度

$$D_f(P\|Q) = \int_{\mathcal{X}} q(x) f\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x$$

  - 函数f是任意凸的、下半连续函数
  - $f(1) = 0$
  - Eg: KL散度
    - 令$f(u) = u\log u$，取 $u = \frac{p(x)}{q(x)}$



$$D_f(P||Q) = \int_x q(x) \frac{p(x)}{q(x)} \log \frac{p(x)}{q(x)}\, dx$$

f-GAN

Name	$D_f(P\|Q)$	Generator $f(u)$
Total variation	$\frac{1}{2}\int \lvert p(x)-q(x)\rvert \,\mathrm{d}x$	$\frac{1}{2}\lvert u-1\rvert$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$
Reverse Kullback-Leibler	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)}\,\mathrm{d}x$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$(\sqrt{u}-1)^2$
Jeffrey	$\int (p(x)-q(x))\log\left(\frac{p(x)}{q(x)}\right)\,\mathrm{d}x$	$(u-1)\log u$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$
Jensen-Shannon-weighted	$\int p(x)\pi\log\frac{p(x)}{\pi p(x)+(1-\pi)q(x)}+(1-\pi)q(x)\log\frac{q(x)}{\pi p(x)+(1-\pi)q(x)}\,\mathrm{d}x$	$\pi u\log u-(1-\pi+\pi u)\log(1-\pi+\pi u)$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$
α-divergence ($\alpha\notin\{0,1\}$)	$\frac{1}{\alpha(\alpha-1)}\int\left(p(x)\left[\left(\frac{q(x)}{p(x)}\right)^{\alpha}-1\right]-\alpha(q(x)-p(x))\right)\mathrm{d}x$	$\frac{1}{\alpha(\alpha-1)}\left(u^{\alpha}-1-\alpha(u-1)\right)$

Source: Nowozin et al., 2016

$$C(G)=\max_D V(G,D)=V(G,D_G^*(\mathbf{x}))$$

$$=E_{\mathbf{x}\sim p_{\text{data}}}\left[\log\frac{p_{\text{data}}(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right]+E_{\mathbf{x}\sim p_G}\left[\log\frac{p_G(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right]-\log 4$$

f-GAN

- f-散度：写成可学习下界

$$D_f(P\|Q) = \int_{\mathcal{X}} q(x) f\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x$$

$$\begin{aligned} D_f(P\|Q) &= \int_{\mathcal{X}} q(x) \sup_{t \in \mathrm{dom}_{f^*}} \left\{ t\frac{p(x)}{q(x)} - f^*(t) \right\} \mathrm{d}x \qquad \text{Fenchel共轭} \\ &\geq \sup_{T \in \mathcal{T}} \left(\int_{\mathcal{X}} p(x)\, T(x)\, \mathrm{d}x - \int_{\mathcal{X}} q(x)\, f^*(T(x))\, \mathrm{d}x \right) \\ &= \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{x \sim P}\left[T(x)\right] - \mathbb{E}_{x \sim Q}\left[f^*(T(x))\right] \right), \end{aligned}$$

f-GAN

- 令$P = p_{\text{data}}, Q = p_G$
- 目标函数

$$\min_{\theta} \max_{\phi} F(\theta, \phi) = E_{\mathbf{x} \sim p_{\text{data}}} [T_\phi(\mathbf{x})] - E_{\mathbf{x} \sim p_{G_\theta}} [f^*(T_\phi(\mathbf{x}))]$$

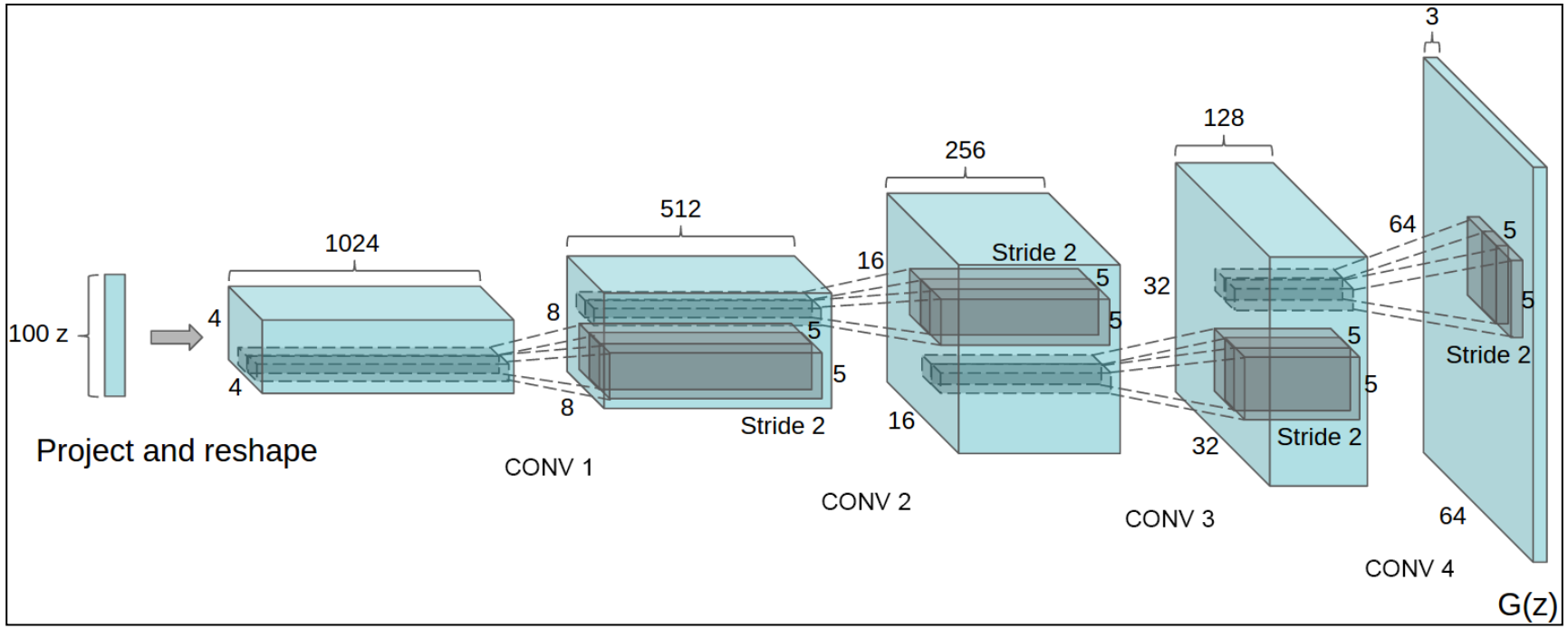
  - 判别器的目标是关于ϕ最大化
    - f-散度的紧致下界尽可能大
  - 生成器的目标是关于θ最小化
    - f-散度的紧致下界尽可能减小

f-GAN

- f-GAN是一个生产GAN模型的“工厂”
  - LSGAN：Pearson χ^2 散度
  - EBGAN：total variation 距离
  - WGAN：Wasserstein 距离

DCGAN

- 判别器和生成器都使用了卷积神经网络来替代GAN 中的多层感知机

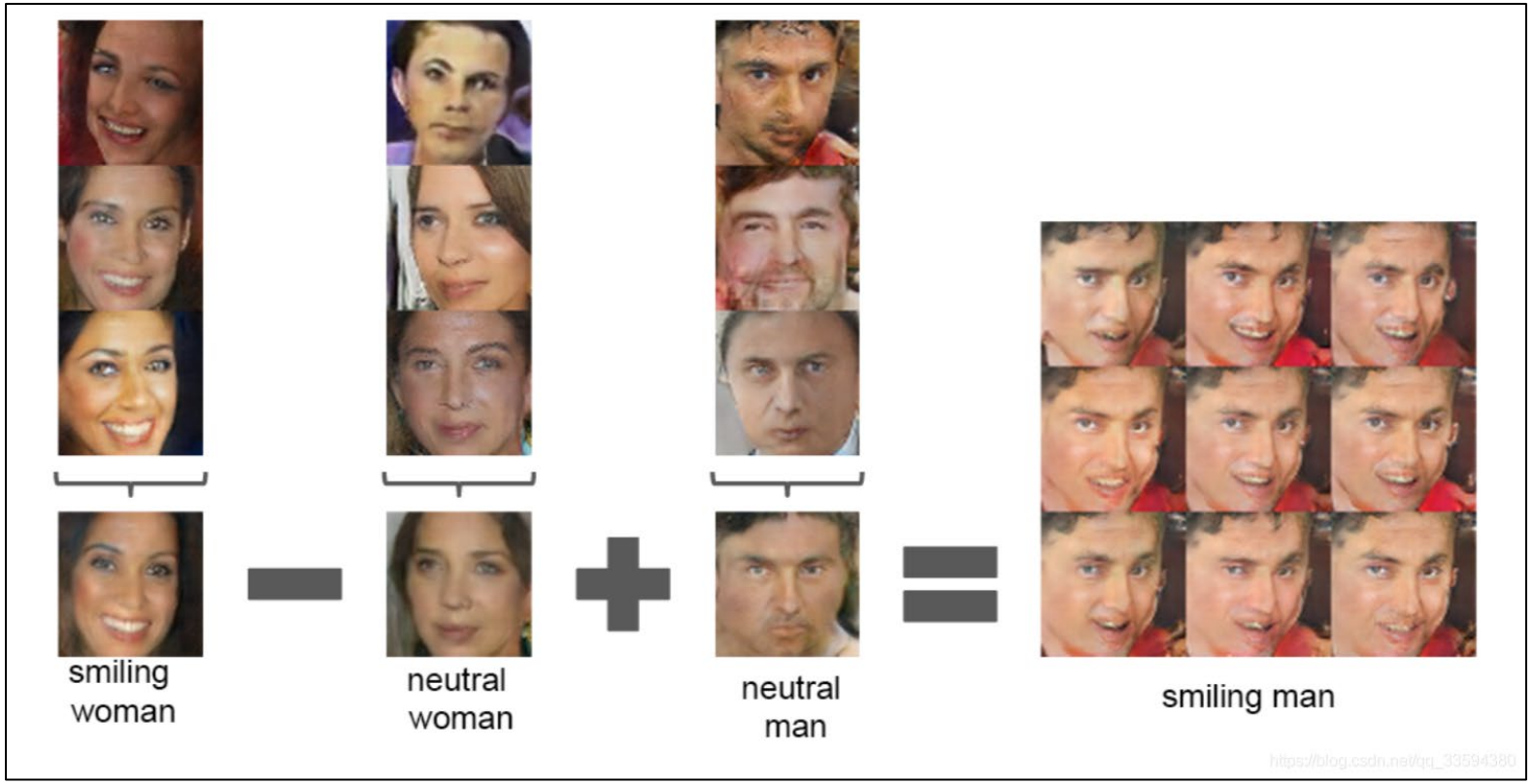


DCGAN

- 与GAN比较
  - 使用跨步卷积（判别器）和反卷积（生成器）代替池化层，使整个网络可微
  - 在生成器和判别器中都添加了批归一化操作，预防模式崩溃
  - 去掉了全连接层，减少计算量
  - 生成器的输出层使用Tanh 激活函数，其它层使用RELU
  - 判别器都使用LeakyReLU 激活函数

DCGAN

- 矢量算法

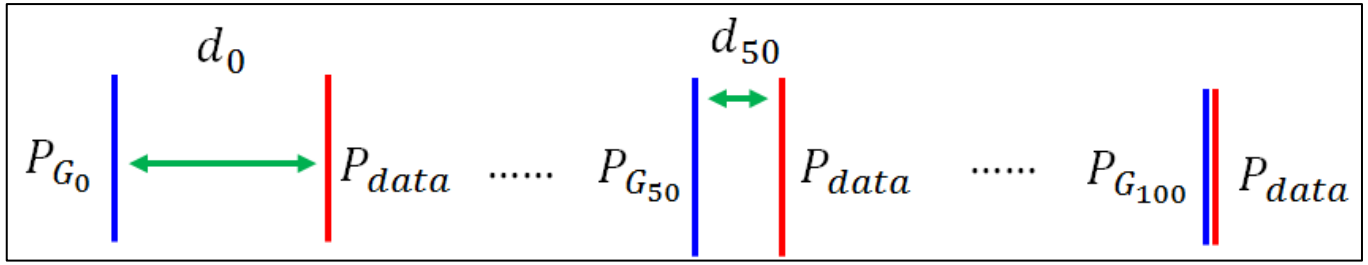


DCGAN

- 矢量算法

Wasserstein GAN

- 原始GAN
  - 关键：衡量两个分布之间的距离



$JS(P_{G_0}, P_{data}) = log2$

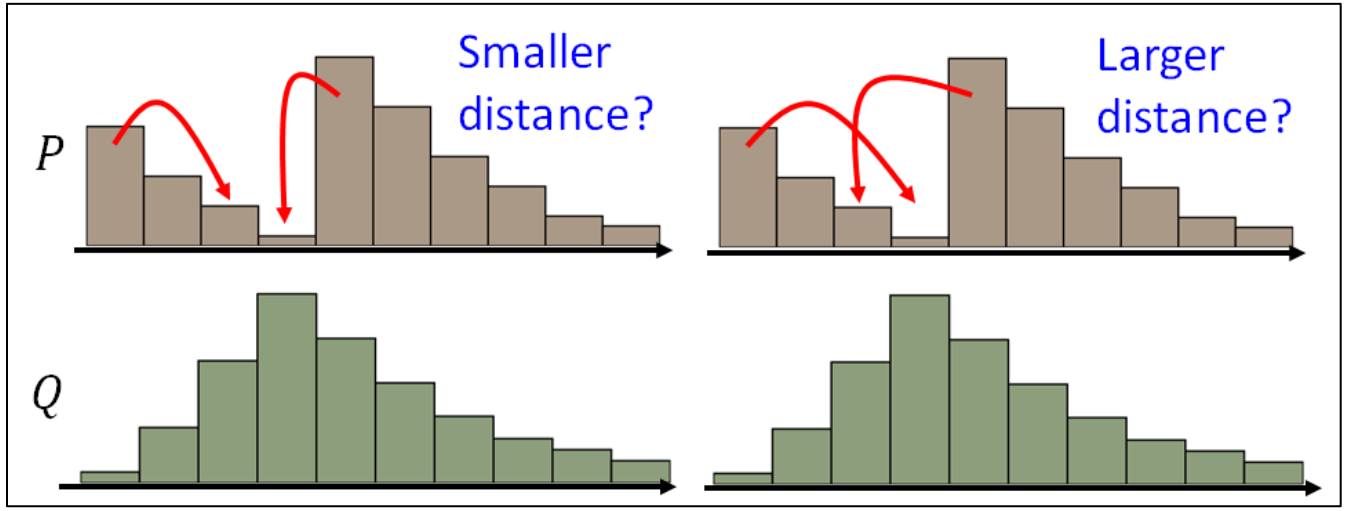
$JS(P_{G_{50}}, P_{data}) = log2$

$JS(P_{G_{100}}, P_{data}) = 0$

  - 缺点：
    - 等价优化的距离衡量（JS散度）不合理
    - 生成器随机初始化后的生成分布很难与真实分布有不可 忽略的重叠

Wasserstein GAN

- Wasserstein距离，亦称为推土机距离



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

  - 即便两个分布没有重叠，Wasserstein距离仍然能够反映它们的远近

Wasserstein GAN

- Wasserstein距离

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y)\sim\gamma} [||x - y||]$$

Kantorovich-Rubinstein 对偶

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

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2|$$

1-Lipschitz 条件

  - Wasserstein距离是平滑的，可以提供有意义的梯度

Wasserstein GAN

- 目标函数
  - 把f用一个带参数的神经网络来表示

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

    - 判别器的目标是关于ω最大化
    - 生成器的目标是关于θ最小化
  - 限制神经网络的所有参数w不超过某个范围$[-c, c]$
    - 满足Lipschitz连续条件：$\|f\|_L \leq 1$
  - 判别器网络f_w最后一层不是非线性激活层
    - 判别器做的是近似拟合Wasserstein距离，属于回归任务，所以要把最后一层的sigmoid拿掉

Wasserstein GAN

- GAN比较

	Discriminator/Critic	Generator
GAN	$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right]$	$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} -\log\left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$
WGAN	$\nabla_{w} \frac{1}{m} \sum_{i=1}^{m} \left[f\left(x^{(i)}\right) - f\left(G\left(z^{(i)}\right)\right) \right]$	$\nabla_{\theta} \frac{1}{m} \sum_{i=1}^{m} -f\left(G\left(z^{(i)}\right)\right)$

Wasserstein GAN

- GAN比较
  - 判别器最后一层去掉sigmoid
  - 每次更新判别器的参数之后把它们的绝对值截断到不超过一个固定常数c
  - 生成器和判别器的loss不取log
  - 不使用基于动量的优化算法（包括momentum和Adam），推荐RMSProp，SGD也行

Wasserstein GAN

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

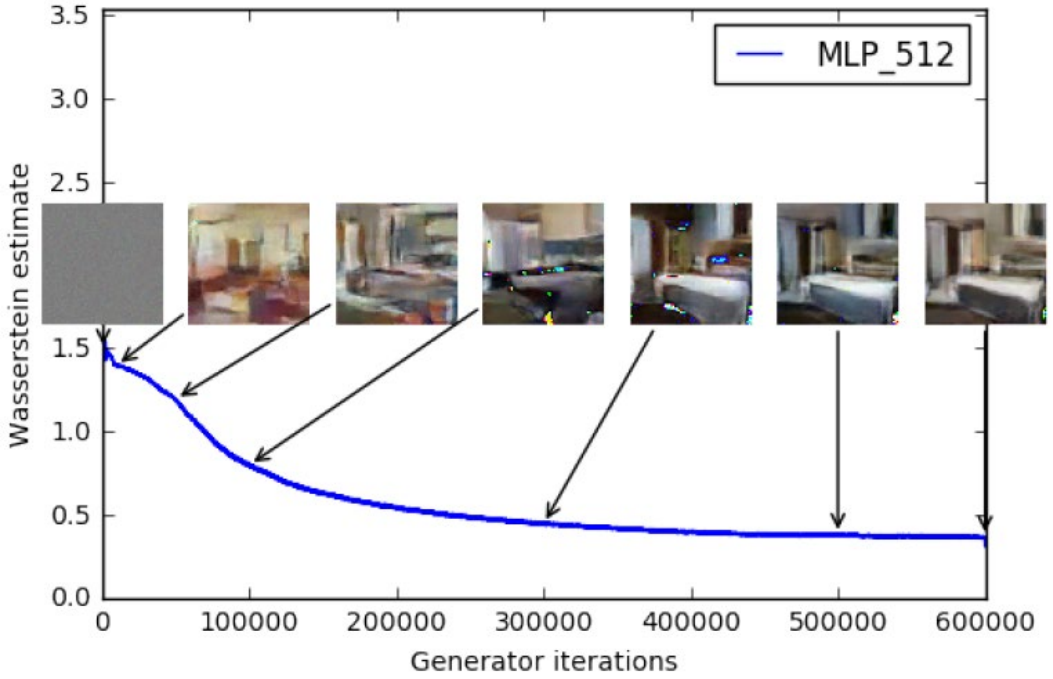
Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.

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

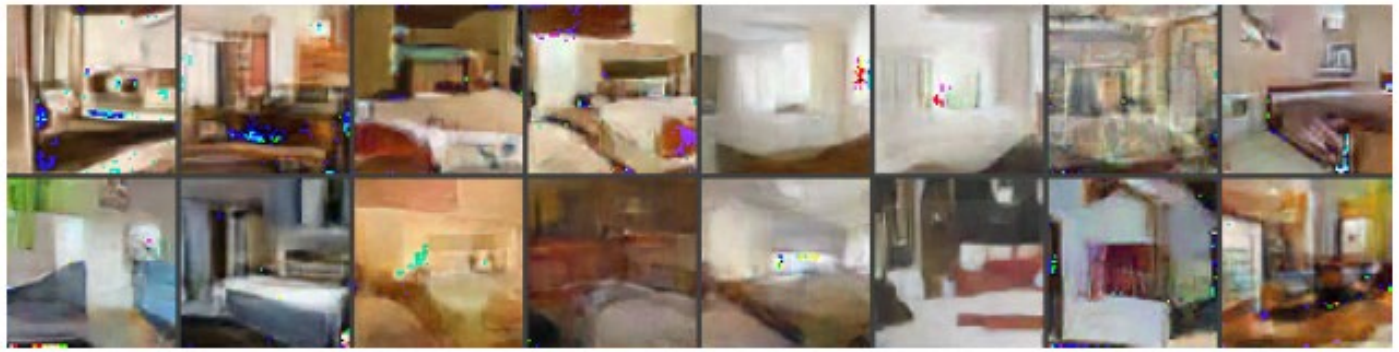
Wasserstein GAN

- 具有解释性：判别器所近似的Wasserstein距离与生成器的生成图片质量高度相关



Wasserstein GAN

- WGAN未观察到collapse mode

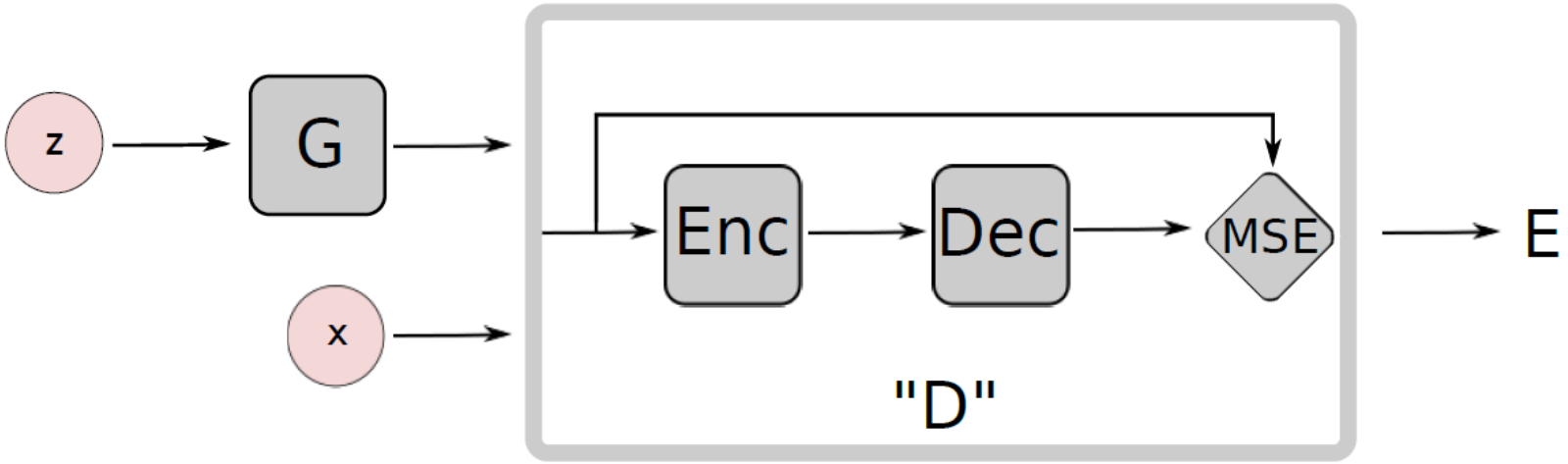


WGAN

原始GAN

EBGAN

- 将判别器视作一个能量函数，函数值越小代表生成数据越真实
- 能量：自编码器AE的重构损失



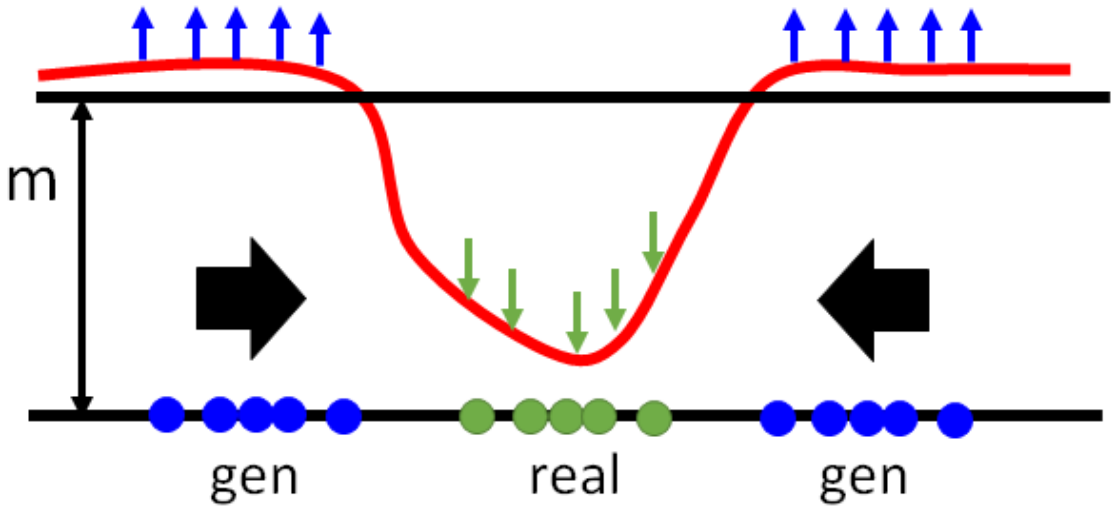
EBGAN

- 判别损失

$$\mathcal{L}_D(x,z) = D(x) + [m - D(G(z))]^+$$

- 生成损失

$$\mathcal{L}_G(z) = D(G(z))$$



EBGAN

- 比原始GAN网络更好，图像更稳定同时分辨率更高，对于整体起到了优化的作用

GAN

EBGAN

Thanks!

Questions?