FDU 神经网络 6. 生成对抗网络

本文参考以下教材:

- 神经网络与深度学习 (邱锡鹏) 第 13 章

欢迎批评指正!

6.1 概率生成模型

概率生成模型 (Probabilistic Generative Model)，简称生成模型，是用于随机生成可观测数据的模型.
假设在一个连续或离散的高维空间 $\mathcal{X}$ 中，存在一个随机向量 X 服从一个未知的数据分布 $p_r(x)$ $(x \in \mathcal{X})$.
生成模型是根据一些可观测的样本 $x^{(1)}, \dots, x^{(N)}$ 学习一个参数化的模型 $p_\theta(x)$ 来近似真实分布 $p_r(x)$,
并可以用这个模型来生成一些样本，使得 "生成" 的样本和 "真实" 的样本尽可能地相似.

生成模型通常包含两个基本功能: 概率密度估计和生成样本 (即采样).
下图以手写体数字图像为例给出了生成模型的两个功能示例，
其中左图表示手写体数字图像的真实分布 $p_r(x)$ 以及从中采样的一些 "真实" 样本，
右图表示估计出了分布 $p_\theta(x)$ 以及从中采样的 "生成" 样本.

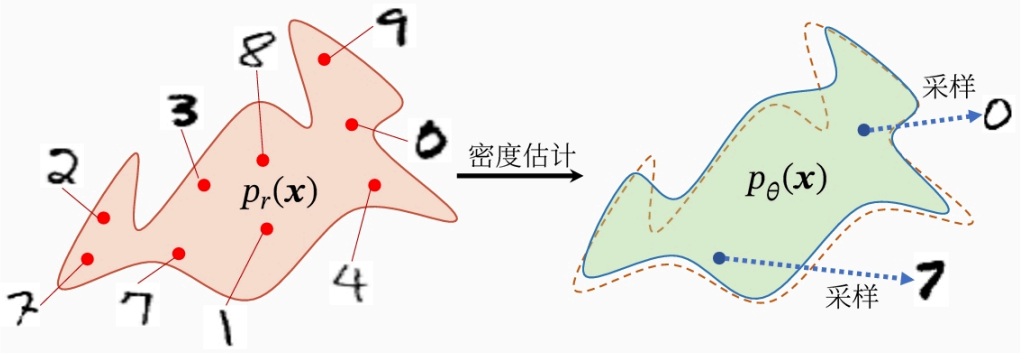

图 13.1　生成模型的两个功能

深度生成模型就是使用深度神经网络来建模一个复杂分布 $p_r(x)$ 或直接生成符合分布 $p_r(x)$ 的样本.

- **概率密度估计:**
  给定一组数据 $\mathcal{D} = \{x^{(n)}\}_{n=1}^N$，假设它们都是独立地从概率密度函数为 $p_r(x)$ 的未知分布中产生的.
  概率密度估计便是根据数据集 $\mathcal{D}$ 来估计其概率密度函数 $p_\theta(x)$.

  直接建模 $p_r(x)$ 比较困难，我们通常通过引入隐变量 z 来简化模型，
  这样密度估计问题可以转换为估计变量 x, z 的两个局部条件概率 $p_\theta(z)$ 和 $p_\theta(x|z)$.
  一般为了简化模型，假设隐变量 z 的先验分布为标准高斯分布 $N(0, I)$.
  在这个假设下，先验分布 $p_\theta(z)$ 中没有参数.
  因此密度估计的重点是估计条件分布 $p_\theta(x|z)$.
  当这两个分布比较复杂时，我们可以利用神经网络来进行建模，这就是**变分自编码器**的思想.

- **生成样本:**
  生成样本就是给定一个概率密度函数为 $p_\theta(x)$ 的分布，生成一些服从这个分布的样本.
  在得到两个变量的局部条件概率 $p_\theta(z)$ 和 $p_\theta(x|z)$ 之后，我们就可以生成数据 x:
  ① 根据隐变量的先验分布 $p_\theta(z)$ 进行采样，得到样本 z.
  ② 根据条件分布 $p_\theta(x|z)$ 进行采样，得到样本 x.

为了便于采样，通常 $p_\theta(x|z)$ 不能太过复杂.
因此另一种生成样本的思想是从一个简单分布 $p(z)$ (例如标准正态分布) 中采集一个样本 z，
并利用一个深度神经网络 $g: \mathcal{Z} \mapsto \mathcal{X}$ 使得 $g(z)$ 服从 $p_r(x)$.
这样我们就可以避免密度估计问题，并有效降低生成样本的难度，这正是**生成对抗网络**的思想.

6.2 变分自编码器

(神经网络与深度学习, 第 13.2 节, 待整理)

6.3 生成对抗网络

6.3.1 隐式密度模型

变分自编码器显式地构建出样本的密度函数 $p_\theta(x) = p_\theta(x|z)p_\theta(z)$，
并通过最大似然估计来求解参数 θ，称为**显式密度模型** (Explicit Density Model).
虽然它使用了神经网络来估计 $p_\theta(x|z)$，但是我们依然假设 $p_\theta(x|z)$ 为一个参数分布族，
而神经网络只是用来预测这个参数分布族的参数.
这在某种程度上限制了神经网络的能力.

如果只是希望有一个模型能生成符合数据分布 $p_r(x)$ 的样本，
那么可以不显示地估计出数据分布的密度函数.
假设在低维空间 $\mathcal{Z}$ 中有一个简单容易采样的分布 $p(z)$ (通常为标准多元正态分布 $N(0, I)$).
我们用神经网络构建一个映射函数 $G: \mathcal{Z} \mapsto \mathcal{X}$，称为**生成网络**.
利用神经网络强大的拟合能力，使得 $G(z;\theta)$ 服从数据分布 $p_r(x)$.
这种模型就称为**隐式密度模型** (Implicit Density Model).
所谓 "隐式" 就是指并不显式地建模 $p_r(x)$，而是建模生成过程.

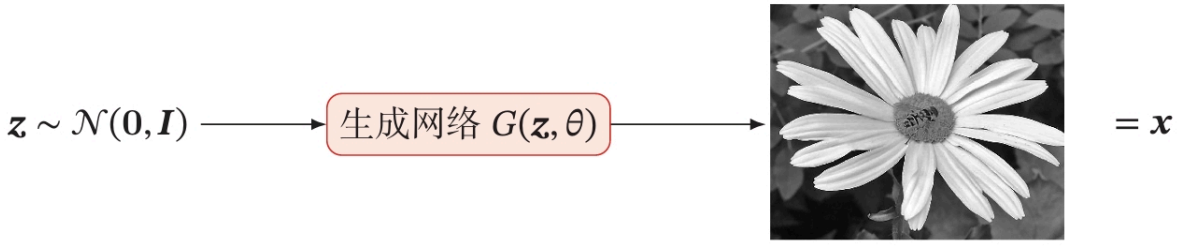

图 13.7　隐式模型生成样本的过程

6.3.2 网络分解

隐式密度模型的关键是确保生成网络产生的样本近似服从真实的数据分布.
由于我们没有显式构建密度函数，故无法通过最大似然估计等方法来训练.
生成对抗网络 (Generative Adversarial Networks, GAN)
通过对抗训练的方式来使得生成网络产生的样本服从真实数据分布.
在生成对抗网络中，有两个网络进行对抗训练.
一个是判别网络，目标是尽量准确地判断一个样本是来自于真实数据还是由生成网络产生;
另一个是生成网络，目标是尽量生成判别网络无法区分来源的样本.
这两个目标相反的网络不断地进行交替训练.
当最后收敛时，如果判别网络再也无法判断出一个样本的来源，

那么也就等价于生成网络可以生成符合真实数据分布的样本.
生成对抗网络的流程如图所示:

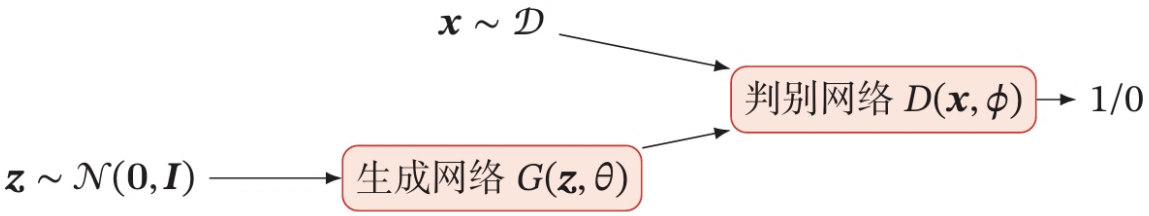

图 13.8　生成对抗网络的流程图

- **判别网络 (Discriminator Network):**
  判别网络 $D(x;\phi)$ 的目标是区分出一个样本 x 是来自于真实分布 $p_r(x)$ 还是来自于生成模型 $p_\theta(x)$,
  因此判别网络实际上是一个二分类的分类器.
  若用标签 $y=1$ 来表示样本来自真实分布，$y=0$ 表示样本来自生成模型，
  则判别网络 $D(x;\phi)$ 的输出为 x 属于真实数据分布的概率，即:

  $$\mathrm{P}\{y=1|x\} = D(x;\phi)$$

  其目标函数为交叉熵损失函数:

  $$\min_{\phi} \left\{ -\mathbb{E}_{x\sim p(x)} \left[y \cdot \log(\mathrm{P}\{y=1|x\}) + (1-y) \cdot \log(\mathrm{P}\{y=0|x\}) \right] \right\}$$
  $$\Updownarrow$$
  $$\min_{\phi} \left\{ -\mathbb{E}_{x\sim p(x)} \left[y \cdot \log(D(x;\phi)) + (1-y) \cdot \log(1-D(x;\phi)) \right] \right\}$$

  其中 $p(x)$ 是真实分布 $p_r(x)$ 和近似分布 $p_\theta(x)$ 按一定比例混合而成的分布，
  而 θ 和 ϕ 分别是生成网络和判别网络的参数.

- **生成网络 (Generator Network):**
  生成网络的目标刚好和判别网络相反，即让判别网络将自己生成的样本判别为真实样本.

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

生成对抗网络是指一类采用对抗训练方式来进行学习的深度生成模型，
其包含的判别网络和生成网络都可以根据不同的生成任务使用不同的网络结构.
一个具体的例子:
深度卷积生成对抗网络 (Deep Convolutional Generative Adversarial Network, DCGAN)

在 DCGAN 中，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.
第一层是全连接层，输入是从均匀分布中随机采样的 100 维向量 z,
输出是 $4 \times 4 \times 1024$ 的向量，重塑为 $4 \times 4 \times 1024$ 的张量，
然后是四层的微步卷积，没有汇聚层.

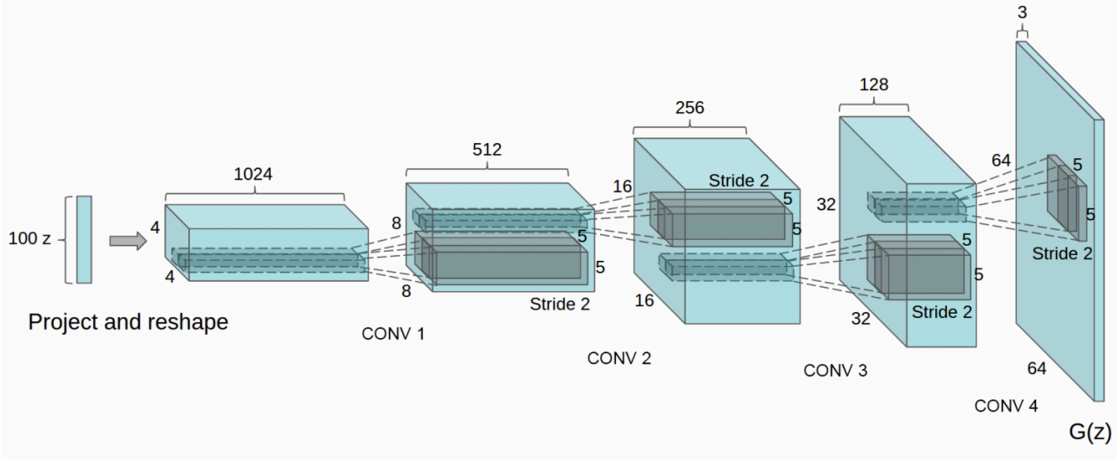

图 13.9　DCGAN 中的生成网络（图片来源：[Radford et al., 2016]）

DCGAN 的主要优点是通过一些经验性的网络结构设计使得对抗训练更加稳定.

- ① 使用带步长的卷积 (在判别网络中) 和微步卷积 (在生成网络中) 来代替汇聚操作，以免损失信息.
- ② 使用批量归一化 (Batch Normalization).
- ③ 去除卷积层之后的全连接层.
- ④ 在生成网络中，除了最后一层使用 $\tanh$ 激活函数外，其余层都使用 ReLU 函数.
- ⑤ 在判别网络中，都使用 Leaky ReLU 激活函数.

6.3.3 训练过程

和单目标的优化任务相比，生成对抗网络的两个子网络的优化目标刚好相反，训练往往不太稳定.
一般情况下，需要平衡两个网络的能力.
对于判别网络来说，一开始的判别能力不能太强，否则难以提升生成网络的能力.
但判别网络的判别能力也不能太弱，否则针对它训练的生成网络也不会太好.
在训练时需要使用一些技巧，使得在每次迭代中，判别网络比生成网络的能力强一些，但又不能强太多.

每次迭代时，判别网络更新 K 次而生成网络更新一次，
即首先要保证判别网络足够强才能开始训练生成网络.
其中 K 是一个超参数，其取值一般取决于具体任务.

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 $\{\boldsymbol{x}^{(m)}\}, 1 \leqslant m \leqslant M$;

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

　　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 $\{\boldsymbol{z}^{(m)}\}, 1 \leqslant m \leqslant M$;

　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 $G(\boldsymbol{z}; \theta)$

6.3.4 理论分析

我们把判别网络和生成网络合并为一个整体，
将整个生成对抗网络的目标函数看作**最小化最大化游戏** (minimax game):

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

$$\Updownarrow$$

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

注意到在 $(0, 1)$ 区间上，$\log(\cdot)$ 的梯度随着自变量增大而减小.
当判别网络 D 以很高的概率认为生成网络 G 产生的样本是 "假" 样本，即 $(1 - D(G(z;\theta);\phi)) \to 1$ 时，
这时目标函数关于 θ 的梯度很小，不利于优化.
所以上述最小化最大化形式的目标函数一般用来进行理论分析，并不是实际训练时的目标函数.

当 $p_r(x)$ 和 $p_\theta(x)$ 已知时，最优的判别网络为:

$$D_\star(x) := D(x;\phi_\star) = \frac{p_r(x)}{p_r(x) + p_\theta(x)}$$

将其代入目标函数，可得:

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

其中 "平均" 分布 $p_a(x) = \frac{1}{2}(p_r(x) + p_\theta(x))$,
$\mathrm{KL}(\cdot)$ 代表 Kullback–Leibler 散度，$\mathrm{JS}(\cdot)$ 代表 Jensen–Shannon 散度.

在生成对抗网络中，当判别网络为最优时，
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 $G_\star(\cdot)$ 对应的损失为 $\mathcal{L}(G_\star|D_\star) = -2\log(2)$.

(1) 训练稳定性

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，即 $\frac{\partial}{\partial\theta}\mathcal{L}(G|D_\star) = 0$.

下图给出了生成对抗网格中的梯度消失问题的示例.
当真实分布 $p_r(x)$ 和模型分布 $p_\theta(x)$ 没有重叠时，
最优的判别器 $D_\star$ 对所有生成数据的输出都为 0，即 $D(G(z;\theta);\phi_\star) = 0\ (\forall\ z \in \mathcal{Z})$,
从而造成生成网络的梯度消失.

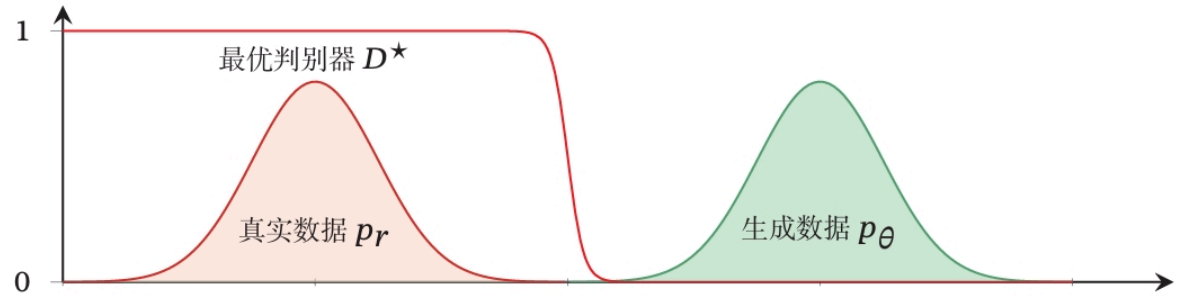

图 13.10　生成对抗网络中的梯度消失问题

因此在实际训练生成对抗网络时，一般不会将判别网络训练到最优，
只进行一步或多步梯度下降，使得生成网络的梯度依然存在.
另外，判别网络也不能太差，否则生成网络的梯度为错误的梯度.
但如何在梯度消失和梯度错误之间取得平衡并不是一件容易的事，
这个问题使得生成对抗网络在训练时稳定性比较差.

(2) 模型坍塌

当 $p_r(x)$ 和 $p_\theta(x)$ 已知时，最优的判别网络为:

$$D_\star(x) := D(x;\phi_\star) = \frac{p_r(x)}{p_r(x)+p_\theta(x)}$$

将其代入生成网络的目标函数，可得:

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

注意到 $\mathbb{E}_{x\sim p_r(x)}[\log(D(x;\phi_\star))]$ 与生成网络无关，于是我们有:

$$
\arg\max_\theta \mathcal{L}(G|D_\star) = \arg\min_\theta \mathrm{KL}(p_\theta, p_r) - 2\mathrm{JS}(p_r, p_\theta)
$$

其中 JS 散度 $\mathrm{JS}(p_\theta, p_r) \in [0, \log(2)]$ 为有界函数，
因此生成网络的目标更多的是受逆向 KL 散度 $\mathrm{KL}(p_\theta, p_r)$ 影响，
使得生成网络更倾向于生成一些更 "安全" 的样本，从而造成**模型坍塌** (model collapse) 问题.

我们定义:

- 前向 KL 散度 $\mathrm{KL}(p_r, p_\theta) = \mathbb{E}_{x\sim p_\theta(x)}\left[\log(\frac{p_r(x)}{p_\theta(x)})\right] = \int p_r(x)\log(\frac{p_r(x)}{p_\theta(x)})\mathrm{d}x$
  当 $p_r(x)\to 0$ 而 $p_\theta(x) > 0$ 时，$p_r(x)\log(\frac{p_r(x)}{p_\theta(x)}) \to 0$，前向 KL 散度会变得很小.
  当 $p_r(x) > 0$ 而 $p_\theta(x)\to 0$ 时，$p_r(x)\log(\frac{p_r(x)}{p_\theta(x)}) \to 0$，后向 KL 散度会变得很大.
  因此前向 KL 散度会鼓励模型分布 $p_\theta(x)$ 尽可能覆盖所有真实分布 $p_r(x) > 0$ 的点，
  而不用回避真实分布 $p_r(x) \approx 0$ 的点.
- 后向 KL 散度 $\mathrm{KL}(p_\theta, p_r) = \mathbb{E}_{x\sim p_r(x)}\left[\log(\frac{p_\theta(x)}{p_r(x)})\right] = \int p_\theta(x)\log(\frac{p_\theta(x)}{p_r(x)})\mathrm{d}x$
  当 $p_r(x)\to 0$ 而 $p_\theta(x) > 0$ 时，$p_\theta(x)\log(\frac{p_\theta(x)}{p_r(x)}) \to \infty$，后向 KL 散度会变得很大.
  当 $p_r(x) > 0$ 而 $p_\theta(x)\to 0$ 时，$p_\theta(x)\log(\frac{p_\theta(x)}{p_r(x)}) \to 0$，后向 KL 散度会变得很小.
  因此后向 KL 散度会鼓励模型分布 $p_\theta(x)$ 尽可能避开所有真实分布 $p_r(x) \approx 0$ 的点，
  而不用考虑是否覆盖所有真实分布 $p_r(x) > 0$ 的点.

下图给出了使用前向 KL 散度和后向 KL 散度进行优化得到结果的示意图

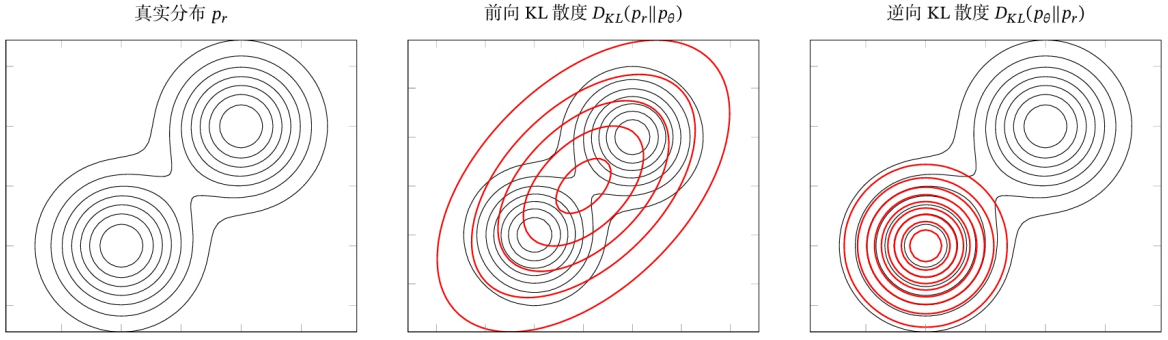

图 13.11　前向和逆向 KL 散度

6.3.5 改进模型

在生成对抗网络中，JS 散度不适合衡量生成数据分布和真实数据分布的距离.
由于通过优化交叉熵 (JS 散度) 训练生成对抗网络会导致训练稳定性和模型坍塌问题，
因此要改进生成对抗网络，就需要改变其损失函数.

$\mathrm{W\text{-}GAN}$ 是一种通过用 **Wasserstein 距离**替代 JS 散度来优化训练的生成对抗网络.
对于真实分布 $p_r(x)$ 和模型分布 $p_\theta(x)$，它们的**第一 Wasserstein 距离**为:

$$W^1(p_r, p_\theta) = \inf_{u \sim U(p_r, p_\theta)} \mathbb{E}_{(x,y)\sim u}[\|x - y\|]$$

其中 $U(p_r, p_\theta)$ 是边际分布为 p_r 和 p_θ 的所有可能的联合分布集合.

当两个分布没有重叠或者重叠非常少时，
它们之间的 KL 散度为 $+\infty$，JS 散度为 $\log(2)$，并不随着两个分布之间的距离而变化.
而第一 Wasserstein 距离依然可以衡量两个没有重叠分布之间的距离.

The End