第十一章　深度生成模型

生成模型（Generative Model）是概率统计和机器学习中的一个概念，指一系列用于随机生成可观测数据的模型。假设在一个连续的或离散的高维空间$\mathcal{X}$中，存在一个变量X服从一个未知分布$P_{data}(X)$。我们需要根据一些可观测的样本$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(N)}$来估计这个未知分布。生成模型就是建立一个分布模型$P_{model}(X)$来近似未知的数据分布$P_{data}(X)$，并可以用这个模型来生成一些样本，使得"生成"样本和"真实"样本尽可能地相似。

生成模型有两个基本功能：一是学习一个概率分布$P_{model}(X)$，即密度估计问题；二是生成数据。

在机器学习中，生成模型是非常重要的模型，既可以用在无监督学习中，也可以用在有监督学习中。在无监督学习中，生成模型可以直接建模一些观测数据的概率密度函数，比如高斯混合模型。在有监督学习中，生成模型可以计算条件概率密度函数，作为计算类别后验概率的中间步骤，然后通过贝叶斯规则可以从生成模型中得到条件分布，比如朴素贝叶斯分类器。

生成模型还有一个很重要的功能就是生成数据，比如生成图像、文本、声音等。自然图像就是可以看作是在连续高维空间中的样本，每一个像素点就是一个变量的一维。不同像素之间存在某种未知的依赖关系，比如一些相邻的像素点的颜色一般是相似的。通过建立一个图像生成模型，可以来生成一些逼真的自然图像。同样，自然语言可以看作是在离散高维空间中的样本。一个句子中每一个词可以看出一维离散变量，不同词之间也存在一个的语法或语义约束关系。通过建立一个语言生成模型，可以来生成一些人类语言。

但在很多实际的应用上，根据可观测的样本来建立这样的生成模型是非常困难的。因为我们观测到的样本往往只是真实样本的一部分变量，叫做可**观测变量**。除了可观测变量外，还有一些变量是不可观测的，叫做**隐藏变量**（Latent

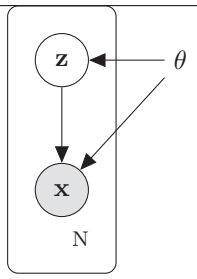


图 11.1: 带隐变量的生成模型

Variables），或**隐变量**。假设隐变量 Z 是另外一个相对低维的空间 $\mathcal{Z}$ 中的变量，完整的生成式模式应该是建模 $P_{model}(X, Z)$。根据链式法则 $P_{model}(X, Z) = P_{model}(X|Z)P_{model}(Z)$，生成式模式可以转换为对两个分布的建模：一个是观测变量 X 的条件分布 $P_{model}(X|Z)$，另一个是隐变量的先验分布 $P_{model}(Z)$。假设 $P_{model}(Z)$ 和 $P_{model}(X|Z)$ 分别为可以某种参数化的分布族 $p(\mathbf{z};\theta)$ 和 $p(\mathbf{x}|\mathbf{z};\theta)$。已知这些分布的形式（比如正态分布），只是参数 θ 未知。参数可以通过最大似然估计来进行求解。图11.1给出了带隐变量的生成模型的图模型结构。

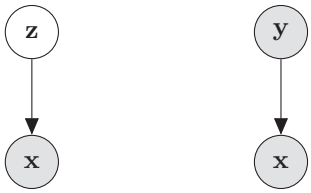
这样，生成数据 $\mathbf{x}$ 的过程可以分为两步进行：

1. 根据隐变量的先验分布 $p(\mathbf{z};\theta)$ 进行采样，得到样本 $\mathbf{z}$；

2. 根据条件分布 $p(\mathbf{x}|\mathbf{z};\theta)$ 进行采样，得到 $\mathbf{x}$。

一般为了简化模型，假设隐变量先验分布 $p(\mathbf{z};\theta)$ 为标准高斯分布 $\mathcal{N}(\mathbf{z};\mathbf{0},\mathbf{I})$。隐变量 $\mathbf{z}$ 的每一维之间都是独立的。在这个假设下，先验分布 $p(\mathbf{z};\theta)$ 中没有参数。

因此在生成模型中，重点是估计条件分布 $p(\mathbf{x}|\mathbf{z};\theta)$。所谓的深度生成模型，就是利用神经网络来建模条件分布 $p(\mathbf{x}|\mathbf{z})$。在本章，我们介绍两种深度生成模型：对抗生成式网络（Generative Adversarial Network，GAN）[Goodfellow et al., 2014] 和变分自动编码器（Variational Autoencoder，VAE）[Kingma and Welling, 2013, Rezende et al., 2014]。



(a) 带隐变量的生成模型 (b) 带类别的生成模型

图 11.2: 生成模型

11.1 变分自编码器

变分自编码器（Variational Autoencoder，VAE）是一种深度生成模型，其思想是利用神经网络来分别建模（1）生成模型 $p(\mathbf{x}|\mathbf{z};\theta)$ 和（2）后验概率分布 $p_\theta(\mathbf{z}|\mathbf{x})$ 的变分近似分布 $q_\phi(\mathbf{z}|\mathbf{x})$。其中，生成模型 $p(\mathbf{x}|\mathbf{z};\theta)$ 可以看作是解码器，将隐变量映射为可观测变量；变分近似分布 $q_\phi(\mathbf{z}|\mathbf{x})$ 可以看作是编码器，即将可观测变量映射为隐变量。这种编码-解码的结构和自编码器比较类似，因此叫做变分自编码器。不同点在于，变分自编码器中的编码器和解码器的输出为分布，而不是确定的值。

11.1.1 解码器：生成网络

首先来看生成模型，即解码器。生成模型可以分解为两部分：隐变量 $\mathbf{z}$ 的先验分布 $p(\mathbf{z};\theta)$ 和条件概率分布 $p(\mathbf{x}|\mathbf{z};\theta)$。

先验分布 $p(\mathbf{z};\theta)$ 隐变量先验分布 $p(\mathbf{z};\theta)$ 一般假设为各向同性的标准高斯分布 $\mathcal{N}(\mathbf{z};\mathbf{0},\mathbf{I})$。隐变量 $\mathbf{z}$ 的每一维之间都是独立的。

条件概率分布 $p(\mathbf{x}|\mathbf{z};\theta)$ 根据变量 $\mathbf{x}$ 的类型不同，我们可以不同的分布族来表示条件概率分布 $p(\mathbf{x}|\mathbf{z};\theta)$，这些分布族的参数可以用神经网络来计算得到。

如果 $\mathbf{x} \in \{0,1\}^d$ 是 d 维的二值的向量，并假设 $\log p(\mathbf{x}|\mathbf{z};\theta)$ 服从多变量的贝努力分布，则

$$\log p(\mathbf{x}|\mathbf{z};\theta) = \sum_{i=1}^{d} \log p(x_i|\mathbf{z};\theta) \tag{11.1}$$

$$= \sum_{i=1}^{d} x_i \log \gamma_i + (1 - x_i) \log(1 - \gamma_i), \tag{11.2}$$

其中，$\gamma_i = p(x_i = 1|\mathbf{z}; \theta)$ 为第 i 维分布的参数，可以用神经网络来计算。比如一个两层的神经网络

$$\boldsymbol{\gamma} = \sigma(W^{(2)}\sigma(W^{(l)}\mathbf{z} + \mathbf{b}^{(l)}) + \mathbf{b}^{(2)}), \tag{11.3}$$

$\theta = \{W^{(2)}, W^{(l)}, \mathbf{b}^{(2)}, \mathbf{b}^{(l)}\}$ 为网络参数。

如果 $\mathbf{x} \in \mathbb{R}^d$ 是 d 维的连续向量，并假设 $\log p(\mathbf{x}|\mathbf{z}; \theta)$ 服从对角化协方差的正态分布，则

$$\log p(\mathbf{x}|\mathbf{z}; \theta) = \log \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 I) \tag{11.4}$$

其中，$\boldsymbol{\mu}$ 和 $\boldsymbol{\sigma}^2$ 可以用神经网络来计算。

这里，生成模型采用神经网络模型，因此也叫作**生成网络**。

11.1.2 参数估计

假设有 N 从未知数据分布中抽取的独立同分布样本 $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(N)}$。根据生成模型 $p(\mathbf{z}; \theta)p(\mathbf{x}|\mathbf{z}; \theta)$，每个样本 $\mathbf{x}$ 的边际对数似然函数为

$$l(\theta; \mathbf{x}) = \log p(\mathbf{x}; \theta) \tag{11.5}$$

$$= \log \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}; \theta) d\mathbf{z}. \tag{11.6}$$

通过最大化所有样本的边际对数似然，我们可以估计最优的参数 θ^*.

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} l(\theta; \mathbf{x}) \tag{11.7}$$

$$= \arg\max_{\theta} \sum_{i=1}^{N} \log \int_{\mathbf{z}} p(\mathbf{x}^{(i)}, \mathbf{z}; \theta) d\mathbf{z}. \tag{11.8}$$

然而，在上面的边际似然函数中需要在对数函数内部进行积分。除非 $p(\mathbf{x}, \mathbf{z}; \theta)$ 的形式非常简单，否则这个积分是很难计算的。

为了可以计算 $\log p(\mathbf{x}; \theta)$，我们引入一个额外的变分函数 $q(\mathbf{z}|\mathbf{x})$，样本 $\mathbf{x}$ 的边际对数似然函数为

$$l(\theta; \mathbf{x}) = \log \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \frac{p(\mathbf{x}, \mathbf{z}; \theta)}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} \tag{11.9}$$

$$\geq \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x},\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} \tag{11.10}$$

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} \tag{11.11}$$

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}|\mathbf{z};\theta) d\mathbf{z} - \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} \tag{11.12}$$

$$= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}|\mathbf{z};\theta) d\mathbf{z} - \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z};\theta)} d\mathbf{z} \tag{11.13}$$

$$= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z};\theta)) \tag{11.14}$$

$$\triangleq L(q,\theta;\mathbf{x}) \tag{11.15}$$

其中，$L(q,\theta;\mathbf{x})$ 为对数似然函数 $l(\theta;\mathbf{x})$ 的下界；公式（11.10）是利用 Jensen 不等式的性质。仅当 $q(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\mathbf{x};\theta)$ 时，

$$l(\theta;\mathbf{x}) = L(q,\theta;\mathbf{x}).$$

> **定义 11.1** – Jensen **不等式：** 如果 X 是随机变量，g 是凸函数，则
>
> $$g\left(\mathbb{E}[X]\right) \leq \mathbb{E}\left[g(X)\right].$$
>
> 等式当且仅当 X 是一个常数或 g 是线性时成立。

理论上，我们可以通过**期望最大化**（Expectation-Maximum，EM）算法来最大化 $l(\theta;\mathbf{x})$。EM 算法可以分解为两步：

- E-step: 找到一个 $q(\mathbf{z}|\mathbf{x})$ 使得 $L(q,\theta;\mathbf{x})$ 最大。
- M-step: 然后保持 $q(\mathbf{z}|\mathbf{x})$ 固定，找到一个 θ，使得来最大化 $L(q,\theta;\mathbf{x})$。

这样个步骤不断重复，直到收敛。

在 E-step 中，最优的 $q(\mathbf{z}|\mathbf{x})$ 为隐变量的后验分布 $p(\mathbf{z}|\mathbf{x};\theta)$。而计算后验分布 $p(\mathbf{z}|\mathbf{x};\theta)$ 是一个**统计推理**问题。

$$p(\mathbf{z}|\mathbf{x};\theta) = \frac{p(\mathbf{x},\mathbf{z};\theta)}{p(\mathbf{x};\theta)} \tag{11.16}$$

$$= \frac{p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)}{\int_z p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)d\mathbf{z}} \tag{11.17}$$

$p(\mathbf{z}|\mathbf{x};\theta)$ 同样涉及到积分计算。如果 $\mathbf{z}$ 是有限的一维离散变量，则计算起来比较容易。一般情况下，$p(\mathbf{z}|\mathbf{x};\theta)$ 是很难计算的。

11.1.3 编码器：推理网络

计算后验概率 $p(\mathbf{z}|\mathbf{x};\theta)$ 一般情况下比较难以计算。为了解决这个问题，在变分自编码器中，同样用一个神经网络来近似计算，叫做**推理网络**。因为其过程是有样本空间到隐变量空间，其形式和自编码器中的编码器比较类似，因此也叫做**基于概率的编码器**（Probabilistic Encoder）。

在统计推理中，传统方法是利用采样或者变分方法来近似计算后验概率，叫做**近似推理**。基于采样的方法效率很低且估计也不是很准确，所以一般使用的是**变分推理**方法，即用简单的分布 q 去近似复杂的分布 $p(\mathbf{z}|\mathbf{x};\theta)$。但是在深度生成模型中，$p(\mathbf{z}|\mathbf{x};\theta)$ 是通过神经网络来计算的，很难用简单的分布去近似。

在变分自编码器中，利用一个神经网络来估计 $p(\mathbf{z}|\mathbf{x};\theta)$ 的近似分布 $q(\mathbf{z}|\mathbf{x};\phi)$，这里 ϕ 是网络参数。这里一般假设隐变量 $\mathbf{z} \in \mathbb{R}^m$ 是 m 维的连续向量，并假设 $q(\mathbf{z}|\mathbf{x};\phi)$ 服从对角化协方差的正态分布，则

$$\log q(\mathbf{z}|\mathbf{x};\phi) = \log \mathcal{N}(\mathbf{x};\boldsymbol{\mu}', \boldsymbol{\sigma}'^2 I) \tag{11.18}$$

其中，$\boldsymbol{\mu}'$ 和 $\boldsymbol{\sigma}'^2$ 可以用神经网络来计算。

我们需要找到一个 $\log q(\mathbf{z}|\mathbf{x};\phi)$ 来尽可能接近真实的后验 $p(\mathbf{z}|\mathbf{x};\theta)$。如果用 KL 散度来衡量两个分布之间的距离，则需要找到变分参数 ϕ^* 来最小化两个分布的 KL 散度。

$$\phi^* = \arg\min_{\phi} D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)||p(\mathbf{z}|\mathbf{x};\theta)). \tag{11.19}$$

然而，直接计算这个散度是不可能的，因为我们无法计算 $p(\mathbf{z}|\mathbf{x};\theta)$。因此，我们需要找到一种间接的计算方法。

将 $\log q(\mathbf{z}|\mathbf{x};\phi)$ 代入公式（11.15）得到，样本 $\mathbf{x}$ 的边际对数似然函数为

$$\mathcal{L}(\phi,\theta;\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)||p(\mathbf{z};\theta)) \tag{11.20}$$

其中，D_{KL} 是 KL 散度，后一项是负的平均重构误差。隐变量 $\mathbf{z}$ 的先验分布设为标准正态分布 $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{0}, I)$。

$l(\theta;\mathbf{x})$ 和其下界 $\mathcal{L}(\phi,\theta;\mathbf{x})$ 的差异为

$$l(\theta;\mathbf{x}) - \mathcal{L}(\phi,\theta;\mathbf{x})$$

$$= \log p(\mathbf{x};\theta) - \mathcal{L}(\phi,\theta;\mathbf{x}) \tag{11.21}$$

$$= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log p(\mathbf{x};\theta)] - \mathcal{L}(\phi,\theta;\mathbf{x}) \tag{11.22}$$

$$= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log p(\mathbf{x};\theta)] - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log \frac{p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x};\phi)}] \tag{11.23}$$

$$= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log \frac{p(\mathbf{z}|\mathbf{x};\theta)}{q(\mathbf{z}|\mathbf{x};\phi)}] \tag{11.24}$$

$$= D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)||p(\mathbf{z}|\mathbf{x};\theta)). \tag{11.25}$$

即近似分布 $q(\mathbf{z}|\mathbf{x};\phi)$ 与真实后验 $p(\mathbf{z}|\mathbf{x};\theta))$ 的KL散度等于对数边际似然与其下界的差。这样，

$$\phi^* = \arg\min_{\phi} D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)||p(\mathbf{z}|\mathbf{x};\theta)) \tag{11.26}$$

$$= \arg\min_{\phi} l(\theta;\mathbf{x}) - \mathcal{L}(\phi,\theta;\mathbf{x}) \tag{11.27}$$

$$= \arg\max_{\phi} \mathcal{L}(\phi,\theta;\mathbf{x}). \tag{11.28}$$

11.1.4 模型汇总

变分自编码器的模型结构可以分为两个部分：

1. 寻找后验分布 $p(\mathbf{z}|\mathbf{x};\theta)$ 的变分近似 $q(\mathbf{z}|\mathbf{x};\phi^*)$；
2. 在已知 $q(\mathbf{z}|\mathbf{x};\phi^*)$ 的情况下，估计更好的生成模型 $p(\mathbf{x}|\mathbf{z};\theta)$。

变分自编码器的图模型表示见图11.3所示。实线表示生成模型，虚线表示变分近似。

在得到之后，我们将其代入变分下界 $L(\phi,\theta;\mathbf{x})$，然后找到一组 θ^* 最大化其下界。

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\phi^*,\theta;\mathbf{x}). \tag{11.29}$$

结合公式（11.28）和（11.29），我们可以看作变分下界 $L(\phi,\theta;\mathbf{x})$。因此，变分自编码器的目标函数为

$$\mathcal{L}(\phi,\theta;\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x};\phi)}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)||p(\mathbf{z};\theta)). \tag{11.30}$$

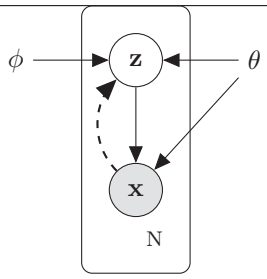


图 11.3: 变分自编码器

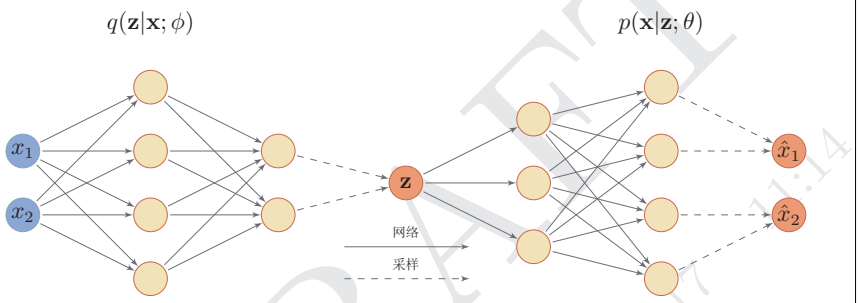


图 11.4: 变分自编码器的神经网络结构

变分自编码器可以看作神经网络和贝叶斯网络的混合体。贝叶斯网络中的节点可以看成是一个随机变量。在变分自编码器中，我们仅仅将隐藏编码对应的节点看成是随机变量，其它节点还是作为普通神经元。这样，编码器变成一个变分推理网络，而解码器可以看作是将隐变量映射到观测变量的生成网络。

图11.4给出了变分自编码器的神经网络结构示例。

11.1.5 训练

给定一个数据集 D，包含 N 个从未知数据分布中抽取的独立同分布样本 $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(N)}$。变分自编码器的目标函数为

$$\mathcal{L}(\phi, \theta; \mathbf{x}) = \sum_{i=1}^{N} \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x}^{(i)};\phi)}[\log p(\mathbf{x}^{(i)}|\mathbf{z};\theta)] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}^{(i)};\phi)\|p(\mathbf{z};\theta)), \tag{11.31}$$

其中，$\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x}^{(i)};\phi)}[\log p(\mathbf{x}^{(i)}|\mathbf{z};\theta)]$ 需要通过采样的方式进行计算。对于每个样本 $\mathbf{x}^{(i)}$，通过采集 M 个 $\mathbf{z}^{(i,m)}$ 来估计期望。

$$\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x}^{(i)};\phi)}[\log p(\mathbf{x}^{(i)}|\mathbf{z};\theta)] = \frac{1}{M}\sum_{l=1}^{M}\log p(\mathbf{x}^{(i)}|\mathbf{z}^{(i,j)};\theta). \tag{11.32}$$

这样目标函数变为了

$$\mathcal{L}(\phi,\theta;D) = \sum_{i=1}^{N}\left(\frac{1}{M}\sum_{l=1}^{M}\log p(\mathbf{x}^{(i)}|\mathbf{z}^{(i,j)};\theta) - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}^{(i)};\phi)\|p(\mathbf{z};\theta))\right), \tag{11.33}$$

如果采用随机梯度方法，每次从数据集中采一个样本 $\mathbf{x}$，然后根据 $q(\mathbf{z}|\mathbf{x};\phi)$ 采一个隐变量 $\mathbf{z}$，并计算下面函数的梯度。

$$\mathcal{L}(\phi,\theta;\mathbf{x}) = \log p(\mathbf{x}|\mathbf{z};\theta) - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)\|p(\mathbf{z};\theta)). \tag{11.34}$$

假设 $q(\mathbf{z}|\mathbf{x}^{(i)};\phi)$ 和 $p(\mathbf{z};\theta)$ 都是正态分布，目标函数中的第二项 $D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}^{(i)};\phi)\|p(\mathbf{z};\theta))$ 可以用解析方法进行计算。

对于两个正态分布 $\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1)$ 和 $\mathcal{N}(\boldsymbol{\mu}_2,\Sigma_2)$，其 KL 散度有精确解。

$$\begin{aligned} &D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1)\|\mathcal{N}(\boldsymbol{\mu}_2,\Sigma_2)) \\ &= \frac{1}{2}\left(tr(\Sigma_2^{-1}\Sigma_1) + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^{\top}\Sigma_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) - k + \log(\frac{det\Sigma_2}{det\Sigma_1})\right) \end{aligned} \tag{11.35}$$

当 $q(\mathbf{z}|\mathbf{x}^{(i)};\phi)$ 为 $\mathcal{N}(\boldsymbol{\mu}(\mathbf{x}),\boldsymbol{\sigma}^2(\mathbf{x})I)$，$p(\mathbf{z};\theta)$ 取标准正态分布 $\mathcal{N}(0,I)$ 时，

$$D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x};\phi)\|p(\mathbf{z};\theta)) = \frac{1}{2}\left(tr(\boldsymbol{\sigma}^2(\mathbf{x})I) + \boldsymbol{\mu}(\mathbf{x})^{\top}\boldsymbol{\mu}(\mathbf{x}) - k - \log(det\boldsymbol{\sigma}^2(\mathbf{x})I)\right). \tag{11.36}$$

重新参数化 在变分自编码器中，一个问题是如何求随机变量的参数的导数。如果随机变量 $\mathbf{z}$ 采样自后验分布 $q(\mathbf{z}|\mathbf{x};\phi)$，我们希望求函数 $f(\mathbf{z})$ 关于 ϕ 的导数，这个导数应该是非零的，但是难以直接计算。

如果 $q(\mathbf{z}|\mathbf{x};\phi)$ 的随机性独立于参数，我们可以通过**重新参数化**来计算导数。假设 $q(\mathbf{z}|\mathbf{x};\phi)$ 为正态分布 $N(\boldsymbol{\mu},\boldsymbol{\sigma}^2I)$，我们可以通过下面方式来采样 $\mathbf{z}$

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma}\odot\boldsymbol{\epsilon}, \tag{11.37}$$

其中，$\epsilon \sim \mathcal{N}(0, I)$。这样，我们就将$\mathbf{z}$和$\boldsymbol{\mu}, \boldsymbol{\sigma}$的关系从采样关系变为函数关系。这样，我们就可以求$f(\mathbf{z})$关于$\boldsymbol{\mu}$和$\boldsymbol{\sigma}$的导数。

重新参数化（Reparameterization）是实现通过随机变量实现反向传播的一种重要手段，并用随机梯度下降训练整个网络，可以提高变分自编码器的训练效率。

11.2 生成式对抗网络

深度生成模型，就是利用神经网络来隐式地建模条件分布$p_{model}(\mathbf{x}|\mathbf{z})$所谓的隐式建模，是指并不对条件分布$P(x|z)$本身进行建模，而是建模生成过程，即学习一个映射函数$g : \mathcal{Z} \to \mathcal{X}$。神经网络的输入为隐变量$\mathbf{z}$，输出为观测变量$\mathbf{x}$。

11.3 总结和深入阅读

变分自动编码器（Variational Autoencoder，VAE）Kingma and Welling [2013], Rezende et al. [2014]

Kingma and Welling [2013]Rezende et al. [2014]

Bowman et al. [2015]

Springenberg [2015] CatGAN Goodfellow et al. [2014]

Chen et al. [2016]

Radford et al. [2015]

Denton et al. [2015] laggan

Salimans et al. [2016]

Mirza and Osindero [2014]

参考文献

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

James A Anderson and Edward Rosenfeld. *Talking nets: An oral history of neural networks*. MiT Press, 2000.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *ArXiv e-prints*, September 2014.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Yoshua Bengio and Jean-Sébastien Senécal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4):713–722, 2008.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994.

Yoshua Bengio, Rejean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003a.

Yoshua Bengio, Jean-Sébastien Senécal, et al. Quick training of probabilistic neural nets by importance sampling. In *AISTATS Conference*, 2003b.

Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8624–8628. IEEE, 2013.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010.

C.M. Bishop. *Pattern recognition and machine learning*. Springer New York., 2006.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, 2002.

Hal Daumé III. A course in machine learning. http://ciml.info/. [Online].

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, New York, 2nd edition, 2001. ISBN 0471056693.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in Neural Information Processing Systems*, pages 472–478, 2001.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

Ian Goodfellow, Aaron Courville, and Yoshua Bengio. Deep learning. Book in preparation for MIT Press, 2015. URL http://goodfeli.github.io/dlbook/.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron C Courville, and Yoshua Bengio. Maxout networks. In *ICML*, volume 28, pages 1319–1327, 2013.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, volume 1, page 6, 2010.

Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,

In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79 (8):2554–2558, 1982.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2012.

David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

M.I. Jordan. *Learning in Graphical Models*. Kluwer Academic Publishers, 1998.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *NIPS*, 2014.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, 2013.

Ryan McDonald, Keith Hall, and Gideon Mann. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464. Association for Computational Linguistics, 2010.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, page 3, 2010.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013b.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Marvin Minsky and Papert Seymour. Perceptrons. 1969.

Marvin L Minsky and Seymour A Papert. *Perceptrons - Expanded Edition: An Introduction to Computational Geometry*. MIT press Boston, MA:, 1987.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

T.M. Mitchell. *Machine learning*. Burr Ridge, IL: McGraw Hill, 1997.

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*, pages 2265–2273, 2013.

Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014.

Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252, 2005.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

Albert BJ Novikoff. On convergence proofs for perceptrons. Technical report, DTIC Document, 1963.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, pages 2226–2234, 2016.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from

overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, pages 2431–2439, 2015.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

DE Rumelhart GE Hinton RJ Williams and GE Hinton. Learning representations by back-propagating errors. *Nature*, pages 323–533, 1986.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5, 2015.

Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

索引

dropout, 100

k 近邻算法, 56

Logistic 回归, 47

maxout 网络, 80

one-hot 向量, 8

丑小鸭定理, 60
二项分布, 20
伯努利分布, 20
信息论, 26
修正线性单元, 77

全 1 向量, 8
内部协变量偏移, 97
决策树, 54
准确率, 58
分布式表示, 153
分类器, 35
前馈神经网络, 81
协变量偏移, 96
卷积, 104
卷积神经网络, 103
双向循环神经网络, 123
反向传播算法, 87
变分自编码器, 144
召回率, 59
向量空间模型, 154
噪声对比估计, 174
困惑度, 161
均值, 25
均匀分布, 21
堆叠循环神经网络, 123
多元正态分布, 23
多层感知器, 82
多项分布, 23
宏平均, 59
局部表示, 153
平均感知器, 70
循环神经网络, 114
微平均, 59

感知器, 62
批量归一化, 97
投票感知器, 69
提前停止, 43
支持向量机, 57
数值微分, 88
数学期望, 25
数据, 39
数据集, 40
方差, 25, 26
时序反向传播, 115

最优化问题, 12
最近邻算法, 56
期望最大化算法, 146
朴素贝叶斯分类器, 55
条件分布, 24
标准归一化, 95
样本, 40
梯度下降法, 16
梯度消失问题, 92
概率, 18
概率图模型, 134
正态分布, 21
正确率, 58
没有免费午餐定理, 60
泛化错误, 38
注意力, 127
深度信念网络, 140

特征映射, 107
盘子表示法, 135
神经网络语言模型, 162
符号微分, 88
符号计算, 89
简单循环网络, 116

自动微分, 89
词嵌入, 155
词袋模型, 154
语料库, 40
贝叶斯公式, 25
贝叶斯分类器, 54
边际分布, 24
过拟合, 38
连接表, 108
配分函数, 166
重要性采样, 171
错误率, 58
长短时记忆神经网络, 119
门机制, 119
门限循环单元, 121
随机过程, 26
马尔可夫链, 26