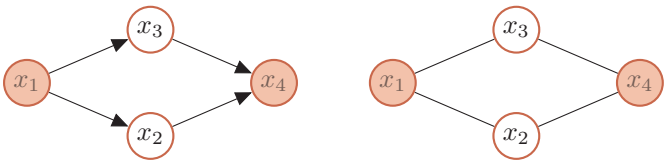
第十一章　概率图模型

在概率论和统计学中，**概率图模型**（Probabilistic Graphical Model，PGM），简称**图模型**（Graphical Model，GM），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。图结构可以将概率模型的结构可视化，以一种直观、简单的方式描述了随机变量之间的条件独立性的性质，并可以将一个复杂的概率模型分解为一些简单模型的组合。在机器学习中，图模型越来越多地用来设计和分析各种学习算法。很多机器学习模型都可以很作是概率模型（Probabilistic Model），将学习任务归结于计算输入和输出之间的条件概率分布。因此，图模型提供了一种新的角度来解释机器学习模型，并且这种角度有很多优点，比如了解不同机器学习模型之间的联系，方便设计新模型等。

一个图由节点和节点之间的边组成。在概率图模型中，每一个节点都表示一个随机变量（或一组随机变量），边表示这些随机变量之间的概率依赖关系。常见的概率图模型可以分为两类：有向图模型和无向图模型。有向图模型也叫做*贝叶斯网络*，为有向非循环图，边的方向代表了因果关系。无向图模型，也叫做*马尔可夫随机场*，每条边代表来两个变量之间有概率依赖关系，但是并不表明任何因果关系。对于一个非全连接的图模型，都可以根据条件独立性将联合概率分布进行因子分解，表示为一组局部的条件概率分布的乘积。图11.1给出了两个代表性图模型的例子：有向图和无向图，分别表示了四个变量$\{x_1, x_2, x_3, x_4\}$之间的依赖关系。

11.1　贝叶斯网络

贝叶斯网络（Bayesian Network），又称**信念网络**（Belief Network），或**有向图模型**（Directed Graphical Model），是指用有向图来表示概率分布的图模



(a) 有向图：贝叶斯网络 (b) 无向图：马尔可夫随机场

图 11.1: 有向图和无向图示例

带阴影的节点表示可观测变量，不带阴影的节点表示不可观测的变量，连边表示两变量间的条件依赖关系。

型。假设一个有向图$G(\mathcal{V},\mathcal{E})$，节点集合$\mathcal{V}=\{X_1,X_2,\cdots,X_K\}$表示$K$个随机变量，每个节点对应一个随机变量$X_k$，边集合$\mathcal{E}$中的每个连接表示两个变量之间的因果关系。

在本章后文中，“节点”与“随机变量”、“变量”的概念会经常混用。每个节点对应一个随机变量。

我们用x_i表示变量X_i的一个取值，K个变量的联合概率分布可以分解为K个条件概率的乘积。

$$p(x_1,x_2,\cdots,x_k) \triangleq P(X_1=x_1,X_2=x_2,\cdots,X_K=x_K) \tag{11.1}$$

$$=p(x_1)p(x_2|x_1)\cdots p(x_K|x_1,\cdots,x_{K-1}), \tag{11.2}$$

$$=\prod_{k=1}^{K} p(x_k|x1,\cdots,x_{k-1}), \tag{11.3}$$

其中，条件概率分布$p(x_k|x_1,\cdots,x_{k-1})$可以用图结构来表示，将在$v_1,\cdots,v_{k-1}$中的每个节点和节点$v_k$用一个有向边来连接。那么，整个联合概率分布对应的图为一个全连接的有向图。

更一般地，对任何结构（非全连接）的有向非循环图，比如图11.1a，我们也需要将其对应的概率模型进行分解。

定义 11.1 – 贝叶斯网络： 对于K随机变量$\{X_1,X_2,\cdots,X_K\}$和一个有向非循环图G，G中的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；G中的每个连接e_{ij}表示两个随机变量X_i和X_j之间具有非独立的因果关系。我们定义X_{π_k}表示变量X_k的所有父节点变量集合，每个随机变量的*局部条件概率*（local conditional probability distribution）为$P(X_k|X_{\pi_k})$。

如果$X=X_1,X_2,\cdots,X_K$的联合概率分布可以分解为每个

随机变量 X_k 的局部条件概率的连乘形式，即

$$p(x_1, x_2, \cdots, x_k) = \prod_{k=1}^{K} p(x_k|x_{\pi_k}), \tag{11.4}$$

那么 (G, X) 构成了一个**贝叶斯网络**。

从公式(11.3)和(11.4)可得到。参见习题（11-1），第161页。

局部马尔可夫性质 贝叶斯网络具有局部马尔可夫性质：每个随机变量在给定父节点的情况下，条件独立于它的非后代节点。

$$X_k \perp\!\!\!\perp Z|X_{\pi_k}, \tag{11.5}$$

其中，Z 为 X_k 的非后代节点的变量。

利用贝叶斯网络的局部马尔可夫性，我们可以对多元变量的联合概率进行简化，从而降低建模的复杂度。以图11.1a中的4个变量（假设每个 X_i 为二值变量）为例，如果我们不进行分解，直接用*联合概率表*来保存每一种的 X 的取值，则独立参数为 $2^4 - 1 = 15$ 个。

如果按图11.1a的结构进行分解，其联合概率为

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2, x_3), \tag{11.6}$$

是4个局部条件概率的乘积，这样只需要 $1 + 2 + 2 + 4 = 9$ 个独立参数。

11.1.1 条件独立性

在贝叶斯网络中，如果两个节点是直接连接的，它们肯定是非条件独立的，是*直接因果关系*。父节点是“因”，子节点是“果”。

如果两个节点不是直接连接的，但是它们之间有一条经过其他节点的路径连接互连接，它们之间的条件独立性就比较复杂。我们先来看一个三个节点的简单例子。给定三个节点 x_1, x_2, x_3，x_1 和 x_3 是不直接连接的，可以通过节点 x_2 连接。这三个节点之间可以有四种连接关系，如图11.2所示。

间接因果关系（图11.2a） 在已知 x_2 时，x_1 和 x_3 为条件独立；
间接果因关系（图11.2b） 在已知 x_2 时，x_1 和 x_3 为条件独立；
共因关系（图11.2c） x_1 和 x_3 是不独立的，在已知 x_2 时，x_1 和 x_3 条件独立；
共果关系（图11.2d） x_1 和 x_3 是独立的，在已知 x_2 时，x_1 和 x_3 不独立

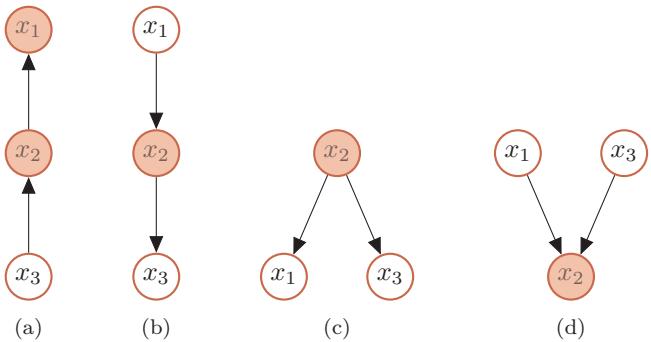


图 11.2: 由 x_1 到 x_3 并经过 x_2 的四种路径类型。在 (a)(b) 中，在已知 x_2 时，x_1 和 x_3 为条件独立；在 (c) 中，x_1 和 x_3 是不独立的，在已知 x_2 时，x_1 和 x_3 条件独立；在 (d) 中，x_1 和 x_3 是独立的，在已知 x_2 时，x_1 和 x_3 不独立。

11.1.2 常用的贝叶斯网络模型

很多经典的机器学习模型都可以看做是贝叶斯网络，比如模型朴素贝叶斯、隐马尔可夫模型、神经网络等。

朴素贝叶斯分类器

朴素贝叶斯分类器（Naive Bayes Classifiers）是一类简单的概率分类器，在强（朴素）独立性假设的条件下运用贝叶斯公式来计算每个类别的后验概率。

给定一个有 d 维特征的样本 $\mathbf{x}$ 和类别 y，类别的后验概率为

$$p(y|\mathbf{x}) = p(y|x_1, \cdots, x_d) \tag{11.7}$$

$$= \frac{p(x_1, \cdots, x_d|y)p(y)}{p(x_1, \cdots, x_d)} \tag{11.8}$$

$$\propto p(x_1, \cdots, x_d|y)p(y). \tag{11.9}$$

$$\tag{11.10}$$

我们假设在给定 y 的情况下，x_i 之间是条件独立的，即 $x_i \perp\!\!\!\perp x_j|y, \forall i \neq j$。那么，$p(y|\mathbf{x})$ 可以简化为

$$p(y|\mathbf{x}) \propto p(y) \prod_{i=1}^{d} p(x_i|y). \tag{11.11}$$

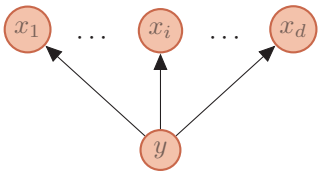


图 11.3: 朴素贝叶斯模型的图模型表示。

图11.3给出了朴素贝叶斯模型的图形表示。条件概率分布 $p(x_i|y)$ 可以是带有可学习参数的密度函数。如果 x_i 为连续值，$p(x_i|y)$ 可以为高斯分布。如果 x_i 为离散值，$p(x_i|y)$ 可以为多项分布。

显然，这个条件独立性假设太严格了。实际任务中的样本特征应该不满足这样的理想假设。但是，朴素贝叶斯分类器在很多任务上也能得到很好的结果。

Sigmoid 信念网络

Sigmoid 信念网络（Sigmoid Belief Network，SBN）[Neal, 1992] 是用 logistic sigmoid 函数来建模有向图中的条件概率分布。

Sigmoid 信念网络网络中的变量为二值变量，取值为 $\{0, 1\}$。对于变量 x_k 和它的父节点集合 π_k，其条件概率分布为

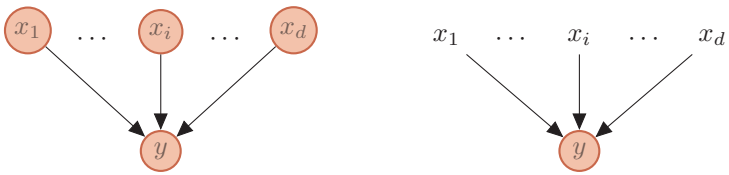
$$p(x_k = 1|x_{\pi_k}) = \sigma(w_0 + \sum_{x_i \in \pi_k} w_i x_i), \tag{11.12}$$

其中 $\sigma(\cdot)$ 是 logistic sigmoid 函数，w_i 是可学习的参数。

图11.4a给出了一个只有一层的简单 Sigmoid 信念网络。

Sigmoid 信念网络与 Logistic 回归模型 Sigmoid 信念网络与 Logistic 回归模型都采用 logistic sigmoid 函数来计算条件概率。如果将 Sigmoid 信念网络简化为只有一个叶子节点，其所有的父节点之间没有连接，且取值为实数，那么 Sigmoid 信念网络的网络结构和 logistic 回归模型类似，如图11.4所示。但是，这两个模型区别在于 logistic 回归模型中的 $\mathbf{x}$ 作为一种确定性的参数，而非变量。因此，logistic 回归模型只建模条件概率 $p(y|\mathbf{x})$，是一种判别模型；而 sigmoid 信念网络建模 $p(\mathbf{x}, y)$，是一种生成模型。

logistic 回归模型也可以看做是一种条件无向图模型。



(a) Sigmoid 信念网络 (b) Logistic 回归

图 11.4: Sigmoid 信念网络和 Logistic 回归模型的比较。

11.1.3 参数估计

贝叶斯网络的学习可以分为两部分：一是网络结构估计，估计每个条件概率分布的参数；二是寻找最优的网络结构。由于后者的优化一般比较困难，因此贝叶斯网络的结构一般是由领域专家来构建。在给定网络结构的条件下，网络的参数一般通过最大似然来进行估计。

在贝叶斯网络中，所有变量 $\mathbf{x}$ 的联合概率分布可以分解为每个随机变量 x_k 的局部条件概率的连乘形式。假设每个局部条件概率 $p(x_k|x_{\pi_k})$ 的参数为 θ_k，则 $\mathbf{x}$ 的对数似然函数为

$$\log p(\mathbf{x}, \Theta) = \sum_{k=1}^{K} \log p(x_k|x_{\pi_k}, \theta_k), \tag{11.13}$$

其中，Θ 为模型中的所有参数。

最大化 $\mathbf{x}$ 的对数似然，只需要分别地最大化每个条件似然 $\log p(x_k|x_{\pi_k}, \theta_k)$。如果 $\mathbf{x}$ 中所有变量都是可观测的并且是离散的，只需要在训练集上统计每个变量的条件概率表即可。但是条件概率表需要的参数比较多。假设条件概率 $p(x_k|x_{\pi_k})$ 的父节点数量为 M，所有变量为二值变量，其条件概率表需要 2^M 个参数。有时为了减少参数数量，我们可以使用参数化的模型，比如 logistic sigmoid 函数。如果所有变量是连续的，我们可以使用高斯函数来表示条件概率分布。前者就是 sigmoid 信念网络，后者就是高斯信念网络。在此基础上，我们可以所有的条件概率分布共享使用同一组参数来进一步减少参数数量。

如果 $\mathbf{x}$ 中所有变量都是可观测的，对于每个对数条件似然函数 $\log p(x_k|x_{\pi_k}, \theta_k)$，我们可以将 x_{π_k} 作为输入变量，x_k 作为输出变量，并使用监督学习的方法来需要每个参数 θ_k。如果变量中有一部分变量为隐变量，就需要使用 EM 算法来进行参数估计。

11.1.4 EM 算法

边际似然（marginal likelihood）也称为**证据**（evidence）。

假设有一组变量，有部分变量是是不可观测的，如何进行参数估计呢？对于一个样本$\mathbf{x}$，令其“缺失”的变量为$\mathbf{z}$，样本$\mathbf{x}$的**对数边际似然函数**（Log Marginal Likelihood）为

$$l(\theta;\mathbf{x}) = \log p(\mathbf{x};\theta) \tag{11.14}$$

$$= \log \int_{\mathbf{z}} p(\mathbf{x},\mathbf{z};\theta)d\mathbf{z}, \tag{11.15}$$

其中，θ为模型参数。

通过最大化整个训练集上所有样本的对数边际似然，可以估计出最优的参数θ^*。然而，在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，计算这个积分十分麻烦。

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

$$l(\theta;\mathbf{x}) = \log \int_{\mathbf{z}} q(\mathbf{z})\frac{p(\mathbf{x},\mathbf{z};\theta)}{q(\mathbf{z})}d\mathbf{z} \tag{11.16}$$

$$\geq \int_{\mathbf{z}} q(\mathbf{z})\log\frac{p(\mathbf{x},\mathbf{z};\theta)}{q(\mathbf{z})}d\mathbf{z} \tag{11.17}$$

利用Jensen不等式。

$$= \int_{\mathbf{z}} q(\mathbf{z})\log\frac{p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)}{q(\mathbf{z})}d\mathbf{z} \tag{11.18}$$

$$= \int_{\mathbf{z}} q(\mathbf{z})\log p(\mathbf{x}|\mathbf{z};\theta)d\mathbf{z} + \int_{\mathbf{z}} q(\mathbf{z})\log\frac{p(\mathbf{z};\theta)}{q(\mathbf{z})}d\mathbf{z} \tag{11.19}$$

$$= \int_{\mathbf{z}} q(\mathbf{z})\log p(\mathbf{x}|\mathbf{z};\theta)d\mathbf{z} - \int_{\mathbf{z}} q(\mathbf{z})\log\frac{q(\mathbf{z})}{p(\mathbf{z};\theta)}d\mathbf{z} \tag{11.20}$$

$$= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{\mathrm{KL}}(q(\mathbf{z})||p(\mathbf{z};\theta)) \tag{11.21}$$

$$\triangleq L(q,\theta;\mathbf{x}) \tag{11.22}$$

其中，$L(q,\theta;\mathbf{x})$为对数边际似然函数$l(\theta;\mathbf{x})$的下界，称为**证据下界**（Evidence Lower Bound，ELBO）。

由Jensen不等式的性质可知，仅当$q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x};\theta)$时，对数边际似然函数$l(\theta;\mathbf{x})$和其下界$L(q,\theta;\mathbf{x})$相等，

练习11-2。

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

数学小知识 | 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。

这样，最大化对数边际似然函数 $l(\theta;\mathbf{x})$ 的过程可以分解为两个步骤：先找到近似分布 $q(\mathbf{z})$ 使得 $l(\theta;\mathbf{x}) = L(q,\theta;\mathbf{x})$，再寻找参数 θ 最大化 $L(q,\theta;\mathbf{x})$。这就是**期望最大化**（Expectation-Maximum，EM）算法。

EM 算法是常用的含隐变量的参数估计方法，通过迭代的方法来最大化边际似然。其具体的过程可以分为两步：

- E 步（Expectation）：固定参数 θ，找到一个分布 $q(\mathbf{z})$ 使得 $L(q,\theta;\mathbf{x})$ 最大。
- M 步（Maximization）：固定 $q(\mathbf{z})$，找到一组参数 θ，使得 $L(q,\theta;\mathbf{x})$ 最大。

这两个步骤不断重复，直到收敛到某个局部最优解。

在 E 步中，最理想的变分分布 $q(\mathbf{z})$ 是等于后验分布 $p(\mathbf{z}|\mathbf{x};\theta)$。而后验分布 $p(\mathbf{z}|\mathbf{x};\theta)$ 是一个推断问题。如果 $\mathbf{z}$ 是有限的一维离散变量（比如混合高斯模型），计算起来还比较容易。否则，$p(\mathbf{z}|\mathbf{x};\theta)$ 一般情况下很难计算的。因此需要通过近似推断的方法来进行估计。

参见第12.1节中的变分自编码器。

11.2 马尔可夫随机场

马尔可夫随机场（Markov Random Fields），也叫无向图模型，或马尔可夫网络（Markov Network），是一类用无向图来表示一组具有马尔可夫性质的随机变量 X 的联合概率分布模型。

和贝叶斯网络类似，马尔可夫随机场也图结构来随机变量之间的依赖关系。但是，贝叶斯网络是有向非循环图，而马尔可夫随机场是一个无向图，并且可以存在循环。这样，马尔可夫随机场可以表示贝叶斯网络无法表示的一些依赖关系，如循环依赖；但它不能表示贝叶斯网络能够表示的某些关系，如推导关系。

给定个有 K 个节点的无向图 $G(\mathcal{V}, \mathcal{E})$，其中 $\mathcal{V} = \{v_1, v_2, \cdots, v_K\}$ 表示节点集合。每个节点 v_k 表示一个随机变量 X_k。如果 (G, X) 满足局部马尔可夫性质，即一个变量 X_k 在给定它的邻居的情况下独立于其它所有变量，那么 (G, X) 就构成了一个马尔可夫随机场。

局部马尔可夫性表示为 $X_v \perp\!\!\!\perp X_{\mathcal{V}\backslash \mathrm{N}[v]} \mid X_{\mathrm{N}(v)}$，即

$$P(X_v = x_v | X_u, u \neq v) = P(X_v = x_v | X_u, v \in N_u), \tag{11.23}$$

吉布斯（Gibbs）分布 如果无向图模型能够表示成一系列在 G 的最大团（们）上的非负函数乘积的形式，这个无向图模型的概率分布 P(X) 就称为 Gibbs 分布。

一个函数集合 f_k（也称为因子或者团因子有时也称为特征），每一个 f_k 的定义域是图 G 的团或子团 k。

马尔可夫网络的联合分布可以表示为：

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c), \tag{11.24}$$

其中，$\phi_c(\mathbf{x}_c)$ 是定义在团 c 上的*势能函数*（Potential Function），Z 是*配分函数*（Partition Function），

$$Z = \sum_{\mathbf{x} \in \mathcal{X}} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c). \tag{11.25}$$

乘积包括了图中的所有团。注意马尔可夫性质在团内的节点存在，在团之间是不存在依赖关系的。这里，实际上，马尔可夫网联络经常表示为对数线性模型。

由于势能函数必须为正的，因此我们一般定义为

$$\phi_c(\mathbf{x}_c) = \exp(-E(X_c)), \tag{11.26}$$

其中，$E(X_c)$ 为*能量函数*（Energy Function），这个表示叫做*玻尔兹曼分布*（Boltzmann Distribution）。

马尔可夫网络的联合分布可以表示为：

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \exp(-E(X_c)) \tag{11.27}$$

$$= \frac{1}{Z} \exp(\sum_{c \in \mathcal{C}} -E(X_c)) \tag{11.28}$$

数学小知识 | 玻尔兹曼分布

在统计力学中，**玻尔兹曼分布**（Boltzmann Distribution）是描述粒子处于特定状态下的概率，是关于状态能量与系统温度的函数。

$$p_i = \frac{1}{Z}\exp\left(\frac{-E_\alpha}{kT}\right), \tag{11.29}$$

其中，p_α 粒子处于为状态 α 的概率，E_α 为状态 α 的能量，k 为玻尔兹曼常量，T 为系统温度，$\exp(\frac{-E_\alpha}{kT})$ 称为*玻尔兹曼因子*（Boltzmann Factor），是没有归一化的概率；Z 为归一化因子，是对系统所有状态进行总和。

在统计力学中，Z 一般称为**配分函数**（Partition Function），其定义为

$$Z = \sum_\alpha \exp\left(\frac{-E_\alpha}{kT}\right). \tag{11.30}$$

玻尔兹曼分布取自奥地利物理学家路德维希·玻尔兹曼（Ludwig Boltzmann），他在1868年研究热平衡气体的统计力学时首次提出了这一分布。

玻尔兹曼分布的一个性质是两个状态概率的比率仅仅依赖于两个状态能量的差值。

$$\frac{p_\alpha}{p_\beta} = \exp\left(\frac{E_\beta - E_\alpha}{kT}\right). \tag{11.31}$$

德维希·玻尔兹曼（Ludwig Boltzmann，1844 - 1906），奥地利物理学家、哲学家。主要贡献为分子动力学。

Hammersley Clifford理论认为，马尔可夫随机场和Gibbs分布是一致的。即吉布斯分布一定满足由node separation导致的条件独立性，并且马尔可夫随机场的概率分布一定可以表示成最大团们上的非负函数乘积形式。

11.3 推断

给定一组变量，推断（Inference）是指在观测到部分变量 $\mathbf{e} = \{e_1, e_2, \cdots, e_m\}$ 时，计算其它变量的某个子集 $\mathbf{q} = \{q_1, q_2, \cdots, q_n\}$ 的后验概率分布 $p(\mathbf{q}|\mathbf{e})$。

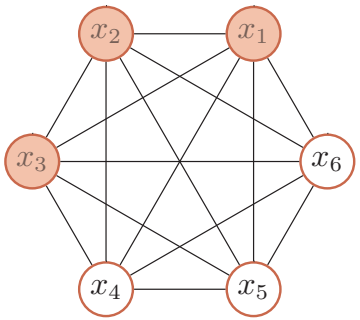


图 11.5: 一个有六个变量的玻尔兹曼机。

在图模型中，我们可以利用图结构来

11.3.1 近似推断

11.4 玻尔兹曼机

玻尔兹曼机（Boltzmann Machine）是一个特殊的概率无向图模型。玻尔兹曼机有以下三点性质。

1. 每个随机变量（节点）是二值的，我们用一个二值的随机向量$\mathbf{X}$来表示所有的变量。在应用时，所有变量一般可以分为两组：可见变量$\mathbf{V}$和隐变量$\mathbf{H}$。
2. 所有变量之间是全连接的。每个变量X_i的取值依赖于所有其它变量$\mathbf{X}_{\setminus i}$。为了简单起见，假设两个变量之间的相互影响（$X_i \to X_j$和$X_j \to X_i$）是对称的，权重相等。这样，玻尔兹曼机可以看做是一个无向图。
3. 整个能量函数定义为

$$E(\mathbf{X}=\mathbf{x}) = -\left(\sum_{i<j} w_{ij}\, x_i\, x_j + \sum_i b_i\, x_i\right), \tag{11.32}$$

其中，w_{ij}是两个变量x_i和x_j之间的连接权重，$x_i \in \{0,1\}$表示状态，b_i是节点i的偏置。

图11.5给出了一个包含3个可见变量和3个隐变量的玻尔兹曼机。

在玻尔兹曼机中，每个变量 X_i 可以解释为是否接受一个基本假设 [Ackley et al., 1985]，其取值为1或0分别表示系统接受或拒绝该假设。变量之间连接的权重为可正可负的实数，代表了两个假设之间的弱约束关系。一个正的权重表示两个假设可以相互支持。也就是说，如果一个假设被接受，另一个也很可能被接受。相反，一个负的权重表示两个假设不能同时被接受。

这也是玻尔兹曼机名称的由来。

变量 $\mathbf{X}$ 的联合概率由玻尔兹曼分布得到，即

$$P(\mathbf{X}=\mathbf{x})=\frac{1}{Z}\exp\left(\frac{-E(\mathbf{x})}{T}\right), \tag{11.33}$$

为了简单起见，这里我们把玻尔兹曼常数 k 吸收到温度 T 中。

玻尔兹曼机可以用来解决两类问题。一类是搜索问题。当给定变量之间的连接权重，需要找到一组二值向量，使得整个网络的能量最低。另一类是学习问题。当给一组定部分变量的观测值时，计算一组最优的权重。

11.4.1 能量最小化

动态系统是数学上的一个概念，用一个函数来描述一个空间中所有点随时间的变化情况，比如钟摆晃动、水的流动等。

玻尔兹曼机可以看做是一个*随机动力系统*（Stochastic Dynamical System），每个变量的状态都以一定的概率受到其它变量的影响。

要寻找一个变量使得整个网络的能量最小，一个简单（但是低效）的做法是选择一个变量，在其它变量保持不变的情况下，将这个变量设为会导致整个网络能量更低的状态。

因为整个网络的连接是对称的，一个变量 X_i 的两个状态0（关闭）和1（打开）之间的能量差异（Energy Gap）为

$$\begin{aligned}\Delta E_i(\mathbf{x}_{\backslash i}) &= E(x_i=0,\mathbf{x}_{\backslash i})-E(x_i=1,\mathbf{x}_{\backslash i}) &(11.34)\\ &=\sum_j w_{ij}\,x_j+b_i. &(11.35)\end{aligned}$$

如果能量差异 $\Delta E_i(\mathbf{x}_{\backslash i})$ 大于一定的阈值（比如0），我们就设 $X_i=1$，否则就设 $X_i=0$。这种简单、确定性的方法在运行一定时间之后总是可以收敛到一个解。但是这个解是局部最优的，不是全局最优。为了跳出局部最优，就必须允许“偶尔”可以将一个变量设置为使得能量变高的状态。这样我们就需要引入一定的随机性，我们以 $p_i=\sigma\left(\frac{\Delta E_i(\mathbf{x}_{\backslash i})}{T}\right)$ 的概率将变量 X_i 设为1，否则设为0。

p_i 和 X_i 的自身的历史状态无关。

玻尔兹曼机动态运行过程中，随机的选择一个变量 X_i，然后根据上面的概率以一定的随机性设置其状态。在固定温度 T 的情况下，玻尔兹曼机动态运行足够时间之后，系统会达到热平衡。此时，任何全局状态的概率服从玻尔兹曼分布。热平衡时的状态分布也只与系统能量有关，与初始状态无关。

要使得玻尔兹曼机达到热平衡，温度 T 的选择十分关键。当系统温度非常高 $T \to \infty$ 时，$p_i \to 0.5$，即每个变量状态的改变十分容易，每一种网络状态都是一样的，而从很快可以达到热平衡。当系统温度非常低 $T \to 0$ 时，如果 $\Delta E_i(\mathbf{x}_{\backslash i}) > 0$ 则 $p_i \to 1$，如果 $\Delta E_i(\mathbf{x}_{\backslash i}) < 0$ 则 $p_i \to 0$。因此，当 $T \to 0$ 时，随机性方法又变成了确定性的方法。

一个比较好的折中方法是让系统刚开始在一个比较高的温度下运行，然后逐渐降低，直到系统在一个比较低的温度下达到热平衡。这样我们就能够得到一个能量全局最小的分布。这个过程被称为模拟退火（Simulated Annealing）[Kirkpatrick et al., 1983]。

模拟退火是一种寻找全局最优的近似方法，其名字来自冶金学的专有名词“退火”，即将材料加热后再以一定的速度退火冷却，可以减少晶格中的缺陷。固体中的内部粒子会停留在使内能有局部最小值的位置，加热时能量变大，粒子会变得无序并随机移动。退火冷却时速度较慢，使得粒子在每个温度都达到平衡态。最后在常温时，粒子以很大的概率达到内能比原先更低的位置。可以证明，模拟退火算法所得解依概率收敛到全局最优解。

在上面的模拟退火过程中，我们以 $p_i = \sigma\left(\frac{\Delta E_i(\mathbf{x}_{\backslash i})}{T}\right)$ 的概率将变量 X_i 设为 1。这个概率是变量 X_i 的条件概率。

定理 11.1 – 玻尔兹曼机中变量的条件概率： 在玻尔兹曼机中，当给定其它变量 $\mathbf{x}_{\backslash i}$ 时，$p(x_i = 1|\mathbf{x}_{\backslash i})$ 的条件概率为

$$p(x_i = 1|\mathbf{x}_{\backslash i}) = \sigma\left(\frac{\Delta E_i(\mathbf{x}_{\backslash i})}{T}\right). \tag{11.36}$$

其中，σ 为 logistic sigmoid 函数。

证明. 根据玻尔兹曼分布的性质，见公式 (11.31)，可得

$$\Delta E_i(\mathbf{x}_{\backslash i}) = -T \ln P(X_i = 0, \mathbf{x}_{\backslash i}) - (-T \ln p(x_i = 1, \mathbf{x}_{\backslash i})) \tag{11.37}$$

$$= T \ln \frac{p(x_i = 1, \mathbf{x}_{\backslash i})}{P(X_i = 0, \mathbf{x}_{\backslash i})} \tag{11.38}$$

$$= T \ln \frac{p(x_i = 1|\mathbf{x}_{\setminus i})}{P(X_i = 0|\mathbf{x}_{\setminus i})} \tag{11.39}$$

$$= T \ln \frac{p(x_i = 1, |\mathbf{x}_{\setminus i})}{1 - p(x_i = 1|\mathbf{x}_{\setminus i})}, \tag{11.40}$$

进而得到，

$$p(x_i = 1|\mathbf{x}_{\setminus i}) = \frac{1}{1 + \exp\left(-\frac{\Delta E_i(\mathbf{x}_{\setminus i})}{T}\right)} \tag{11.41}$$

$$= \sigma\left(\frac{\Delta E_i(\mathbf{x}_{\setminus i})}{T}\right) \tag{11.42}$$

$$= \sigma\left(\frac{1}{T}\sum_j w_{ij}\, x_j + b_i\right). \tag{11.43}$$

□

11.4.2 参数学习

玻尔兹曼机中的变量分为可见变量 $\mathbf{v} \in \{0,1\}^m$ 和不可见变量 $\mathbf{h} \in \{0,1\}^n$。

给定一组可见向量 $\hat{\mathbf{v}}^{(1)}, \hat{\mathbf{v}}^{(2)}, \cdots, \hat{\mathbf{v}}^{(N)}$ 作为训练集，我们要调整玻尔兹曼机的分布使得训练集中所有样本的（对数）似然函数最大。而这个分布由权重参数 W 决定，所以我们需要更新权重 W。训练集的对数似然函数定义为

$$\mathcal{LL}(\mathcal{W}) = \frac{1}{N}\sum_{i=1}^{N} \log p(\hat{\mathbf{v}}^{(n)}) \tag{11.44}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \log \sum_{\mathbf{h}} p(\hat{\mathbf{v}}^{(n)}, \mathbf{h}) \tag{11.45}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \log \frac{\sum_{\mathbf{h}} \exp(E(\hat{\mathbf{v}}^{(n)}, \mathbf{h}))}{\sum_{\mathbf{v},\mathbf{h}} \exp(E(\mathbf{v}, \mathbf{h}))}, \tag{11.46}$$

其中，每个训练向量 $p(\hat{\mathbf{v}}^{(n)})$ 的对数似然对参数 w_{ij} 的导数为

$$\frac{\partial \log p(\hat{\mathbf{v}}^{(n)})}{\partial w_{ij}} = \frac{\partial \log \sum_{\mathbf{h}} p(\hat{\mathbf{v}}^{(n)}, \mathbf{h})}{\partial w_i j} \tag{11.47}$$

$$= \frac{\partial \log \sum_{\mathbf{h}} \exp(E(\hat{\mathbf{v}}^{(n)}, \mathbf{h})) - \log \sum_{\mathbf{v},\mathbf{h}} \exp(E(\mathbf{v}, \mathbf{h}))}{\partial w_{ij}} \tag{11.48}$$

$$= \sum_{\mathbf{h}} \frac{\exp(E(\hat{\mathbf{v}}^{(n)}, \mathbf{h}))}{\sum_{\mathbf{h}} \exp(E(\hat{\mathbf{v}}^{(n)}, \mathbf{h}))} x_i x_j - \sum_{\mathbf{v},\mathbf{h}} \frac{\exp(E(\mathbf{v}, \mathbf{h}))}{\sum_{\mathbf{v},\mathbf{h}} \exp(E(\mathbf{v}, \mathbf{h}))} x_i x_j \tag{11.49}$$

$$= \sum_{\mathbf{h}} p(\mathbf{h}|\hat{\mathbf{v}}^{(n)}) x_i x_j - \sum_{\mathbf{v},\mathbf{h}} p(\mathbf{v}, \mathbf{h}) x_i x_j \tag{11.50}$$

$$= \mathbb{E}_{p(\mathbf{h}|\hat{\mathbf{v}}^{(n)})}[x_i x_j] - \mathbb{E}_{p(\mathbf{x})}[x_i x_j], \tag{11.51}$$

其中，$p(\mathbf{h}|\hat{\mathbf{v}}^{(n)})$ 和 $p(\mathbf{v}, \mathbf{h})$ 在当前参数 W 下计算的条件概率和联合概率。

采用梯度上升法，参数 w_{ij} 的更新公式为：

$$w_{ij} \leftarrow w_{ij} + \alpha \left(\frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{p(\mathbf{h}|\hat{\mathbf{v}}^{(n)})}[x_i x_j] - \mathbb{E}_{p(\mathbf{x})}[x_i x_j] \right), \tag{11.52}$$

其中，$\alpha > 0$ 为学习率。

但是，我们很难精确计算这个梯度。公式 (11.51) 中，第一项为 $x_i x_j$ 在给定 $\mathbf{V} = \hat{\mathbf{v}}^{(n)}$ 时的期望，第二项为没有任何限制时的期望。因为涉及到计算配分函数，该两个分布很难计算。对于一个 D 维的随机向量 $\mathbf{X}$，其取值空间大小为 2^D。当 D 比较大时，这两项的计算会十分耗时。因此，只能通过一些采样方法（如 Gibbs 采样）来进行近似求解。

对于第一项，将可见变量固定，通过模拟退火的方法使得网络达到热平衡状态（温度 $T = 1$）。对于每一个连接，采样 $x_i x_j$ 的值。在训练集上所有的训练向量上重复此过程，得到 $x_i x_j$ 的近似期望 $\langle x_i x_j \rangle_{\texttt{data}}$。

对于第二项，不设任何限制，通过模拟退火的方法使得网络达到热平衡状态（温度 $T = 1$）。对于每一个连接，采样 $x_i x_j$ 的值。在训练集上所有的训练向量上重复此过程，得到 $x_i x_j$ 的近似期望 $\langle x_i x_j \rangle_{\texttt{model}}$。

这样，权重 w_{ij} 可以用下面公式近似地更新

$$w_{ij} \leftarrow w_{ij} + \alpha \left(\langle x_i x_j \rangle_{\texttt{data}} - \langle x_i x_j \rangle_{\texttt{model}} \right). \tag{11.53}$$

这个更新方法的一个特点是仅仅使用了局部信息。也就是说，虽然我们优化目前是整个网络的能量最低，但是一个权重的更新只依赖于它连接的相关变量的状态。

玻尔兹曼机可以用在监督学习和无监督学习中。在监督学习中，可见变量又可以分为输入和输出变量，隐变量则隐式地描述了可见变量之间复杂的约束关系。在无监督学习中，隐变量可以看做是可见变量的内部特征表示。玻尔兹

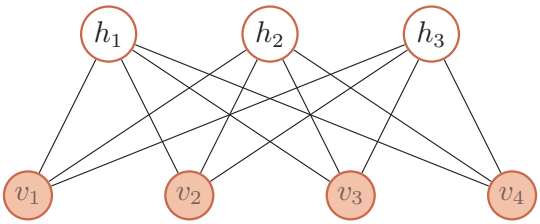


图 11.6: 一个有7个变量的受限玻尔兹曼机。

曼机也可以看做是一种随机型的神经网络，是Hopfield神经网络的扩展，并且可以生成的相应的Hopfield神经网络。在没有时间限制时，玻尔兹曼机还可以用来解决复杂的组合优化问题。

全连接的玻尔兹曼机虽然只在理论上十分有趣，但是由于其复杂性，目前为止并没有被广泛使用。虽然基于采样的方法在很大程度提高了学习效率，但是每更新一次权重，就需要网络重新达到热平衡状态，这个过程依然比较低效，需要很长时间。在实际应用中，使用比较广泛的一种带限制的版本，也就是受限玻尔兹曼机。

受限玻尔兹曼机因其结构最初黉风琴，2000年后受限玻兹曼机的名称才变得流行。

11.5 受限玻尔兹曼机

和两层的全连接神经网络的结构相同。

受限玻尔兹曼机（Restricted Boltzmann Machines，RBM）是一个二分图结构的无向图模型，如图11.6所示。在受限玻尔兹曼机中，变量可以为两组，分别为隐藏层和可见层（或输入层）。同一层中的节点之间没有连接，一个层中的节点与另一层中的所有节点连接。节点变量的取值为0或1。

假设一个受限玻尔兹曼机由 m 个可见层节点和 n 个隐层节点组成，其定义如下：

- 可见层节点：$\mathbf{v} = [v_1, \cdots, v_m]^{\mathrm{T}}$
- 隐藏层节点：$\mathbf{h} = [h_1, \cdots, h_n]^{\mathrm{T}}$
- 权重矩阵：$W = \{w_{i,j}\} \in \mathbb{R}^{n \times m}$，其中每个元素为隐层单元 h_i 和可见层单元 v_j 之间边的权重
- 偏置：每个可见层单元 v_i 有偏置 a_i，对每个隐层单元 h_j 有偏置 b_j

受限玻尔兹曼机的能量函数定义为

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j h_j w_{i,j} v_i \tag{11.54}$$

$$= -\mathbf{a}^{\mathrm{T}}\mathbf{v} - \mathbf{b}^{\mathrm{T}}\mathbf{h} - \mathbf{h}^{\mathrm{T}} W \mathbf{v}, \tag{11.55}$$

随机向量 $(\mathbf{v},\mathbf{h})$ 的联合概率为

$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z}\exp(-E(\mathbf{v},\mathbf{h})) \tag{11.56}$$

$$= \frac{1}{Z}\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})\exp(\mathbf{b}^{\mathrm{T}}\mathbf{h})\exp(\mathbf{h}^{\mathrm{T}} W \mathbf{v}), \tag{11.57}$$

其中，$Z = \sum_{\mathbf{v},\mathbf{h}} \exp(-E(\mathbf{v},\mathbf{h}))$ 为配分函数。

定理 11.2 – 受限玻尔兹曼机中变量的条件概率： 在受限玻尔兹曼机中，每个可见变量和隐变量的条件概率为

$$p(v_i = 1|\mathbf{h}) = \sigma\left(a_i + \sum_j w_{i,j} h_j\right), \tag{11.58}$$

$$p(h_j = 1|\mathbf{v}) = \sigma\left(b_j + \sum_i w_{i,j} v_i)\right), \tag{11.59}$$

其中，σ 为logistic sigmoid函数。

证明.（1）我们先证明 $p(v_i = 1|\mathbf{h})$。

可见层变量$\mathbf{v}$的边际概率为

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{11.60}$$

$$= \frac{1}{Z} \sum_{\mathbf{h}} \exp\left(\mathbf{a}^{\mathrm{T}}\mathbf{v} + \sum_j b_j h_j + \sum_i \sum_j h_j w_{i,j} v_i \right) \tag{11.61}$$

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \sum_{\mathbf{h}} \exp\left(\sum_j h_j (b_j + \sum_i w_{i,j} v_i) \right) \tag{11.62}$$

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \sum_{\mathbf{h}} \prod_j \exp\left(h_j (b_j + \sum_i w_{i,j} v_i) \right) \tag{11.63}$$

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \sum_{h_1} \sum_{h_2} \cdots \sum_{h_n} \prod_j \exp\left(h_j (b_j + \sum_i w_{i,j} v_i) \right) \tag{11.64}$$

将h_j为0或1的取值代入计算。

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \prod_j \sum_{h_j} \exp\left(h_j (b_j + \sum_i w_{i,j} v_i) \right) \tag{11.65}$$

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \prod_j \left(1 + \exp(b_j + \sum_i w_{i,j} v_i) \right). \tag{11.66}$$

固定$h_j = 1$时，$p(h_j = 1, \mathbf{v})$的边际概率为

$$p(h_j = 1, \mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}, h_j = 1} \exp\left(-E(\mathbf{v}, \mathbf{h})\right) \tag{11.67}$$

$$= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{v})}{Z} \prod_{k, k \neq j} \left(1 + \exp(b_k + \sum_i w_{i,k} v_i) \right) \exp(b_j + \sum_i w_{i,j} v_i). \tag{11.68}$$

由公式11.66和 11.68，可以计算隐藏单元h_j的条件概率为：

$$p(h_j = 1 | \mathbf{v}) = \frac{p(h_i = 1, \mathbf{v})}{p(\mathbf{v})} \tag{11.69}$$

$$= \frac{\exp(b_j + \sum_i w_{i,j} v_i)}{1 + \exp(b_j + \sum_i w_{i,j} v_i)} \tag{11.70}$$

$$= \sigma\left(b_j + \sum_i w_{i,j} v_i \right), \tag{11.71}$$

其中，$\sigma(\cdot)$为logitic sigmoid函数。

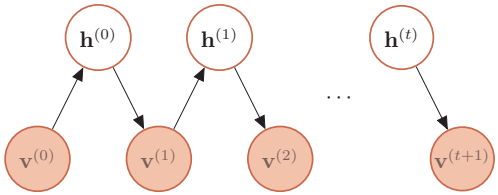


图 11.7: 受限玻尔兹曼机的采样过程

（2）同理，条件概率 $p(v_i = 1|\mathbf{h})$ 为

$$p(v_i = 1|\mathbf{h}) = \sigma\left(a_i + \sum_j w_{i,j} h_j\right). \tag{11.72}$$

□

公式11.71和11.72也可以写为向量形式。

$$p(\mathbf{h} = \underline{1}|\mathbf{v}) = \sigma\left(\mathbf{b} + W^{\mathrm{T}} v_i\right) \tag{11.73}$$

$$p(\mathbf{v} = \underline{1}|\mathbf{h}) = \sigma\left(\mathbf{a} + W h_j\right). \tag{11.74}$$

从上面定理可知，在给定可见变量时，隐变量之间相互条件独立的。同样，在给定隐变量时，可见变量之间也相互条件独立。因此，受限玻尔兹曼机可以并行地对所有的可见变量（或所有的隐变量）同时进行采样，而从可以更快地达到热平衡状态。

采样 受限玻尔兹曼机的采样过程如下：

- 随机初始化一个可见向量 $\hat{\mathbf{v}}_0$, 计算隐层节点的概率，并从中采样一个隐向量 $\mathbf{h}_0$；
- 基于 $\mathbf{h}_0$，计算可见变量概率，并从中采样一个可见向量 $\mathbf{v}_1$；
- 重复 t 次后，我们获得了 $(\mathbf{h}_t, \mathbf{v}_t)$
- 当 $t \to \infty$ 时, $(\mathbf{h}_t, \mathbf{v}_t)$ 的采样服从 $P(V, H)$ 分布。

图11.7也给出了上述过程的示例。

11.5.1 参数学习

和玻尔兹曼机一样，受限玻尔兹曼机通过最大化似然函数来找到最优的参数W。给定一组训练样本$\mathcal{V}=\hat{\mathbf{v}}^{(1)},\hat{\mathbf{v}}^{(2)},\cdots,\hat{\mathbf{v}}^{(N)}$，优化目标为

$$\mathcal{LL}(W)=\frac{1}{N}\sum_{n=1}^{N}\log p(\hat{\mathbf{v}}^{(n)}). \tag{11.75}$$

采用随机梯度上升法，对于样本$\hat{\mathbf{v}}^{(n)}$，目标函数关于参数w_{ij}的梯度为：

$$\frac{\partial\mathcal{LL}(W)}{\partial w_{ij}}=\sum_{h_j}p(h_j|\hat{\mathbf{v}}^{(n)})v_ih_j-\sum_{\mathbf{h},\mathbf{v}}p(\mathbf{v},\mathbf{h})v_ih_j, \tag{11.76}$$

将$h_j=0$或1的值代入上式，得到

$$\frac{\partial\mathcal{LL}(W)}{\partial w_{ij}}=p(h_j=1|\hat{\mathbf{v}}^{(n)})\hat{v}_i-\sum_{\mathbf{v}}p(\mathbf{v})p(h_j=1|\mathbf{v})v_i, \tag{11.77}$$

同理，

$$\frac{\partial\mathcal{LL}(W)}{\partial a_i}=\hat{v}_i-\sum_{\mathbf{v}}p(\mathbf{v})v_i, \tag{11.78}$$

$$\frac{\partial\mathcal{LL}(W)}{\partial b_j}=p(h_j=1|\hat{\mathbf{v}}^{(n)})-\sum_{\mathbf{v}}p(h_j=1|\mathbf{v}), \tag{11.79}$$

上述三个参数梯度的公式中，都需要计算$p(\mathbf{v})$，这就涉及到配分函数的计算，因此也需要通过Gibbs采样的方法来近似计算。根据受限玻尔兹曼机的条件独立性，可以按可见变量和隐变量两组轮流进行分批采样，如图11.7中所示。虽然比一般的玻尔兹曼机速度有很大提高，但一般还是需要通过很多步采样才可以采集到符合真实分布的样本。这就使得受限玻尔兹曼机的训练效率仍然不高。

对比散度学习算法 由于受限玻尔兹曼机的特殊结构，因此可以使用一种比Gibbs采样更有效的学习算法，即**对比散度**（Contrastive Divergence）[Hinton, 2002]。

因为目标是$p(v)\approx p_{train}(v)$

与Gibbs采样不同，对比散度算法仅需k步Gibbs采样。为了提高效率，对比散度算法用一个训练样本作为可见变量的初始值。然后，轮流进行Gibbs采样，不需要等到收敛，只需要k步就足够了。这就是CD-k算法。通常，$k=1$就可以学得很好。

对比散度的流程如算法11.1所示。

算法 11.1: 单步对比散度算法

输入: 训练集: $\hat{\mathbf{v}}^{(n)}, n = 1, \cdots, N$;

学习率: α

初始化: $W \leftarrow 0$, $\mathbf{a} \leftarrow 0$, $\mathbf{b} \leftarrow 0$;

for $t = 1 \cdots T$ **do**

  for $n = 1 \cdots N$ **do**

    选取一个样本 $\hat{\mathbf{v}}^{(n)}$，用公式 (11.71) 计算 $p(\mathbf{h} = \underline{1}|\hat{\mathbf{v}}^{(n)})$，并根据这个分布采集一个隐向量 $\mathbf{h}$;

    计算**正向梯度** $\hat{\mathbf{v}}^{(n)}\mathbf{h}^{\mathrm{T}}$;

    根据 $\mathbf{h}$，用公式 (11.72) 计算 $p(\mathbf{v} = \underline{1}|\mathbf{h})$，并根据这个分布采集重构的可见变量 $\mathbf{v}'$;

    根据 $\mathbf{v}'$，重新计算 $p(\mathbf{h} = \underline{1}|\mathbf{v}')$ 并采样一个 $\mathbf{h}'$;

    计算**反向梯度** $\mathbf{v}'\mathbf{h}'^{\mathrm{T}}$;

    更新参数:

    $W \leftarrow W + \alpha(\hat{\mathbf{v}}^{(n)}\mathbf{h}^{\mathrm{T}} - \mathbf{v}'\mathbf{h}'^{\mathrm{T}})$;

    $\mathbf{a} \leftarrow \mathbf{a} + \alpha(\hat{\mathbf{v}}^{(n)} - \mathbf{v}')$;

    $\mathbf{b} \leftarrow \mathbf{b} + \alpha(\mathbf{h} - \mathbf{h}')$;

  end

end

输出: $W, \mathbf{a}, \mathbf{b}$

受限玻尔兹曼机是一个生成模型，使用隐变量来描述输入数据的分布。同时受限玻尔兹曼机也是一个无监督模型，不需要数据的标签信息。

11.5.2 受限玻尔兹曼机的类型

在具体的不同任务中，需要处理的数据类型不一定都是二值的，也可能是连续值。为了能够处理这些数据，就需要根据输入或输出的数据类型来设计新的能量函数。

一般来说，常见的受限玻尔兹曼机有以下三种：

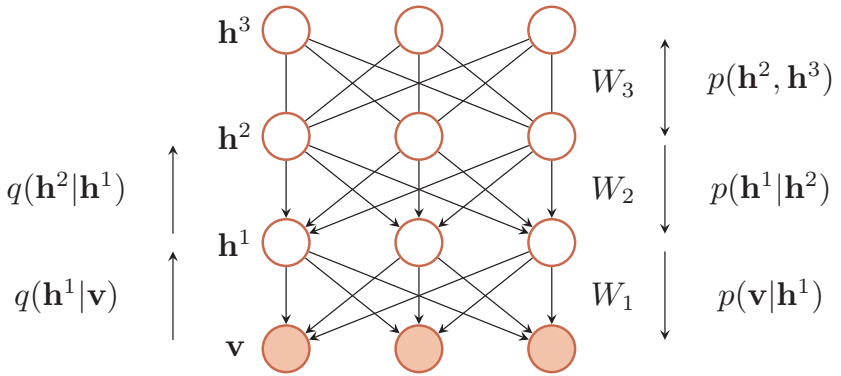


图 11.8: 一个有 4 层结构的深度信念网络。

“贝努力-贝努力”受限玻尔兹曼机 “贝努力-贝努力”受限玻尔兹曼机（Bernoulli-Bernoulli RBM, BB-RBM）就是上面介绍的可见变量和隐变量都为二值类型的受限玻尔兹曼机。

“高斯-贝努力”受限玻尔兹曼机 “高斯-贝努力”受限玻尔兹曼机（Gaussian-Bernoulli RBM, GB-RBM），其能量函数定义为

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i \frac{(v_i - \mu_i)^2}{2\sigma_i^2} - \sum_j b_j h_j - \sum_i \sum_j \frac{v_i}{\sigma_i} w_{i,j} h_j, \tag{11.80}$$

其中，每个可见变量 v_i 服从 (μ_i, σ_i) 的高斯分布。

“贝努力-高斯”受限玻尔兹曼机 “贝努力-高斯”受限玻尔兹曼机（Bernoulli-Gaussian RBM, BG-RBM），其能量函数定义为

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i a_i v_i - \sum_j \frac{(h_j - \mu_j)^2}{2\sigma_j^2} - \sum_i \sum_j v_i w_{i,j} \frac{h_j}{\sigma_j}, \tag{11.81}$$

其中，每个可见变量 v_i 服从 (μ_i, σ_i) 的高斯分布。

11.6 深度信念网络

和全连接的神经网络结构相同。

深度信念网络（Deep Belief Network，DBN）是深度的有向的概率图模型，其图结构由多层的节点构成。每层节点的内部没有连接，相邻两层的节点之间

为全连接。网络的最底层为可见变量，其它层节点都为隐变量。最顶部的两层间的连接是无向的，其他层之间有连接上下的有向连接。图11.8给出了一个深度信念网络的示例。对一个有L层隐变量的深度信念网络，最底层（第0层）为可见变量$\mathbf{v} = \mathbf{h}^{(0)}$，其余每层变量为$\mathbf{h}^{(1)}, \cdots, \mathbf{h}^{(L)}$。除了最顶上两层外，每一层变量$\mathbf{h}^{(l)}$依赖于其上面一层$\mathbf{h}^{(l+1)}$，即

$$p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}, \cdots, \mathbf{h}^{(L)}) = p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}), \tag{11.82}$$

其中，$l = \{0, \cdots, L-2\}$。

顶部的两层是一个无向图，可以看做是一个受限玻尔兹曼机，用来产生$p(\mathbf{h}^{(L-1)})$的先验分布。

深度信念网络中所有变量的联合概率可以分解为

$$p(\mathbf{v}, \mathbf{h}^{(1)}, \cdots, \mathbf{h}^{(L)}) = p(\mathbf{v}|\mathbf{h}^{(1)}) \left(\prod_{l=1}^{L-2} p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}) \right) p(\mathbf{h}^{(L-1)}, \mathbf{h}^{(L)}) \tag{11.83}$$

$$= \left(\prod_{l=0}^{L-2} p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}) \right) p(\mathbf{h}^{(L-1)}, \mathbf{h}^{(L)}), \tag{11.84}$$

其中，$p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)})$为sigmoid型条件概率分布为

$$p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}) = \sigma\left(\mathbf{b}^{(l+1)} + W^{(l+1)}\mathbf{h}^{(l+1)}\right), \tag{11.85}$$

其中，$\sigma(\cdot)$为按位计算的logistic sigmoid函数，$\mathbf{b}^{(l+1)}$为偏置参数，$W^{(l+1)}$为权重参数。

生成模型 深度信念网络是一个生成模型，可以用来生成符合特定分布的样本。隐变量用来描述在可见变量之间的高阶相关性。假如训练数据服从分布$p(\mathbf{v})$，通过训练得到一个深度信念网络。在生成样本时，首先在最顶两层进行足够多次的吉布斯采样，生成$\mathbf{h}^{(L-1)}$，然后依次计算下一层隐变量的分布。因为在给定上一层变量取值时，下一层的变量是条件独立的，因为可以独立采样。这样，我们可以从第$L-1$层开始，自顶向下进行逐层采样，最终得到可见层的样本。

11.6.1 深度信念网络的训练

深度信念网络最直接的训练方式可以通过最大似然方法使得可见层变量$\mathbf{v}$的分布$p(\mathbf{v})$在训练集合上的似然达到最大。但在深度信念网络中，隐变量$\mathbf{h}$之

间的关系十分复杂，由于“贡献度分配问题”，很难直接学习。即使对于简单的单层信念网络 $p(v=1|\mathbf{h})=\sigma\left(b+\mathbf{w}^{\mathrm{T}}\mathbf{h}\right)$，在已知可见变量时，其隐变量的联合后验概率 $p(\mathbf{h}|v)$ 不再相互独立，因此很难精确估计所有隐变量的后验概率。早期深度信念网络的后验概率一般通过蒙特卡罗方法或变分方法来近似估计，但是效率比较低，而导致其参数学习比较困难。

为了有效地训练深度信念网络，我们将每一层的 sigmoid 信念网络转换为受限玻尔兹曼机。这样做的好处是隐变量的后验概率是相互独立的，从而可以很容易地进行采样。这样，深度信念网络可以看作是由多个受限玻尔兹曼机从下到上进行堆叠，每一层受限玻尔兹曼机的隐层作为上一层受限玻尔兹曼机的可见层。进一步地，深度信念网络可以采用逐层训练的方式来快速训练，即从最底层开始，每次只训练一层，直到最后一层 [Hinton et al., 2006]。

“逐层训练”是能够有效训练深度模型的最早的方法。

深度信念网络的训练过程可以分为**预训练**和**精调**两个阶段。先通过逐层预训练将模型的参数初始化为较优的值，再通过传统学习方法对参数进行精调。

预训练 在预训练阶段，我们采用逐层训练的方式，将深度信念网络的训练简化为对多个受限玻尔兹曼机的训练。

算法11.2给出一种深度信念网络的逐层预训练方法。大量的实践表明，预训练可以产生非常好的参数初始值，从而极大地降低了模型的学习难度。

精调 经过预训练之后，再结合具体的任务（监督学习或重构），通过传统的全局学习算法对网络进行精调（Fine-Tuning），使模型收敛到更好的局部最优点。

除了顶层的受限玻尔兹曼机，其它层之间的权重被分成向上的**认知权重**（“Recognition”Weights）R 和向下的**生成权重**（“Generative”Weights）W。认知权重用来进行后验概率计算，而生成权重用来进行定义模型。认知权重的初始值 $R^{(l)}=W^{(l)\mathrm{T}}$。

深度信念网络一般采样 Contrastive Wake-Sleep 算法进行精调，其算法过程是：

- 随机初始化权重；
- Wake阶段：认知过程，通过外界输入（可见变量）和向上认知权重，计算每一层隐变量的后验概率并采样。然后，修改下行的生成权重使得下一

算法 11.2: 深度信念网络的逐层训练方法

输入: 训练集: $\hat{\mathbf{v}}^{(n)}, n = 1, \cdots, N$;
学习率：α;
深度信念网络层数:L;
第 l 层权重：$W^{(l)}$;
第 l 层偏置 $\mathbf{a}^{(l)}$;
第 l 层偏置 $\mathbf{b}^{(l)}$;

```
for l = 1 ··· L do
    初始化：W^(l) ← 0, a^(l) ← 0, b^(l) ← 0;
    从训练集中采样 h^(0) = v̂;
    for i = 1 ··· l − 1 do
        根据分布 q(h^(i)|h^(i−1)) 采样 h^(i);
    end
    将 h^(l−1) 作为训练样本，充分训练第 l 个受限玻尔兹曼机
     W^(l), a^(l), b^(l);
end
```

输出: $W_1, \cdots, W_L$

层的变量的后验概率最大。也就是“如果现实跟我想象的不一样，改变我的权重使得我想象的东西就是这样的”;

- Sleep 阶段：生成过程，通过顶层的采样和向下的生成权重，逐层计算每一层的后验概率并采样。然后，修改向上的认知权重使得上一层变量的后验概率最大。也就是“如果梦中的景象不是我脑中的相应概念，改变我的认知权重使得这种景象在我看来就是这个概念”;
- 交替进行 Wake 和 Sleep 过程，直到收敛。

11.6.2 作为深度神经网络的预训练

只需要向上的认知权重。

深度信念网络的一个应用是作为深度神经网络的预训练部分，提供神经网络的初始权重。在深度信念网络的最顶层再增加一层输出层，然后再使用反向传播算法对这些权重进行调优。

特别是在训练数据比较少时，预训练的作用非常大。因为不恰当的初始化

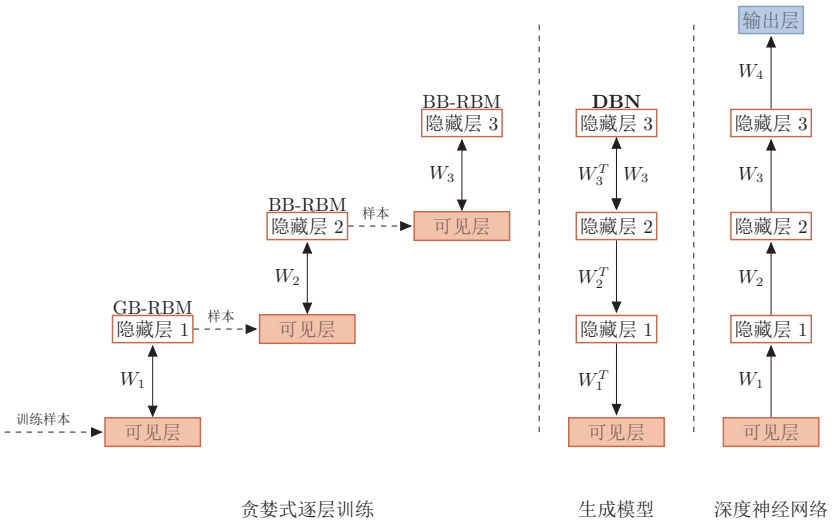


图 11.9: 一个有 4 层结构的深度信念网络。

权重会显著影响最终模型的性能，而预训练获得的权重在权值空间中比随机权重更接近最优的权重，避免了反向传播算法因随机初始化权值参数而容易陷入局部最优和训练时间长的缺点。这不仅提升了模型的性能，也加快了调优阶段的收敛速度 [Larochelle et al., 2007]。

除了深度信念网络之外，自编码器 [Bengio et al., 2007] 以及它的变体，比如稀疏自编码器 [Ranzato et al., 2006] 和去噪自编码器 [Vincent et al., 2008]，也可以用来作为深度神经网络的初始化。

11.6.3 卷积深度置信网络

11.7 图模型与神经网络的关系

图模型和神经网络有着类似的网络结构，但两者也有很大的不同。图模型的节点是随机变量，其图结构的主要功能是用来描述变量之间的依赖关系，一般是稀疏连接。使用图模型的好处是可以有效进行统计推断。而神经网络中的节点是神经元，是一个计算节点。如果将神经网络中每个神经元看做是一个二值随机变量，那神经网络就变成一个 sigmoid 信念网络。

图模型中的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。而神经网络中的神经元则没有直观的解释。

图模型一般是生成模型，可以用生成样本，也可以通过贝叶斯公式用来做分类。而神经网络是判别模型，直接用来分类。

判别模型也可以用图模型来表示。

图模型的参数学习的目标函数为似然函数或条件似然函数，若包含隐变量则通常通过EM算法来求解。而神经网络参数学习的目标为交叉熵或平方误差等损失函数。

11.8 总结和深入阅读

习题 11-1 证明公式(11.5)。

习题 11-2 证明仅当$q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x};\theta)$时，对数边际似然函数$l(\theta;\mathbf{x})$和其下界$L(q,\theta;\mathbf{x})$相等。

参考文献

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598): 671–680, 1983.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480. ACM, 2007.

Radford M Neal. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992.

Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, pages 1137–1144. MIT Press, 2006.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.