

# 第二十章 深度生成模型

在本章中，我们介绍几种具体的生成模型，这些模型可以使用第十六章至第十九章中出现的技术构建和训练。所有这些模型在某种程度上都代表了多个变量的概率分布。有些模型允许显式地计算概率分布函数。其他模型则不允许直接评估概率分布函数，但支持隐式获取分布知识的操作，如从分布中采样。这些模型中的一部分使用第十六章中的图模型语言，从图和因子的角度描述为结构化概率模型。其他的不能简单地从因子角度描述，但仍然代表概率分布。

## 20.1 玻尔兹曼机

玻尔兹曼机最初作为一种广义的“联结主义”引入，用来学习二值向量上的任意概率分布 (Fahlman *et al.*, 1983; Ackley *et al.*, 1985; Hinton *et al.*, 1984b; Hinton and Sejnowski, 1986)。玻尔兹曼机的变体（包含其他类型的变量）早已超过了原始玻尔兹曼机的流行程度。在本节中，我们简要介绍二值玻尔兹曼机并讨论训练模型和进行推断时出现的问题。

我们在  $d$  维二值随机向量  $\mathbf{x} \in \{0, 1\}^d$  上定义玻尔兹曼机。玻尔兹曼机是一种基于能量的模型（第 16.2.4 节），意味着我们可以使用能量函数定义联合概率分布：

$$P(\mathbf{x}) = \frac{\exp(-E(\mathbf{x}))}{Z}, \quad (20.1)$$

其中  $E(\mathbf{x})$  是能量函数， $Z$  是确保  $\sum_x P(\mathbf{x}) = 1$  的配分函数。玻尔兹曼机的能量函数如下给出：

$$E(\mathbf{x}) = -\mathbf{x}^\top \mathbf{U}\mathbf{x} - \mathbf{b}^\top \mathbf{x}, \quad (20.2)$$

其中  $\mathbf{U}$  是模型参数的“权重”矩阵， $\mathbf{b}$  是偏置向量。

在一般设定下，给定一组训练样本，每个样本都是  $n$  维的。式 (20.1) 描述了观察到的变量的联合概率分布。虽然这种情况显然可行，但它限制了观察到的变量和权重矩阵描述的变量之间相互作用的类型。具体来说，这意味着一个单元的概率由其他单元值的线性模型（逻辑回归）给出。

当不是所有变量都能被观察到时，玻尔兹曼机变得更强大。在这种情况下，潜变量类似于多层感知机中的隐藏单元，并模拟可见单元之间的高阶交互。正如添加隐藏单元将逻辑回归转换为 MLP，导致 MLP 成为函数的万能近似器，具有隐藏单元的玻尔兹曼机不再局限于建模变量之间的线性关系。相反，玻尔兹曼机变成了离散变量上概率质量函数的万能近似器 (Le Roux and Bengio, 2008)。

正式地，我们将单元  $\mathbf{x}$  分解为两个子集：可见单元  $\mathbf{v}$  和潜在（或隐藏）单元  $\mathbf{h}$ 。能量函数变为

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^\top \mathbf{R}\mathbf{v} - \mathbf{v}^\top \mathbf{W}\mathbf{h} - \mathbf{h}^\top \mathbf{S}\mathbf{h} - \mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h}. \quad (20.3)$$

**玻尔兹曼机的学习** 玻尔兹曼机的学习算法通常基于最大似然。所有玻尔兹曼机都具有难以处理的配分函数，因此最大似然梯度必须使用第十八章中的技术来近似。

玻尔兹曼机有一个有趣的性质，当基于最大似然的学习规则训练时，连接两个单元的特定权重的更新仅取决于这两个单元在不同分布下收集的统计信息： $P_{\text{model}}(\mathbf{v})$  和  $\hat{P}_{\text{data}}(\mathbf{v})P_{\text{model}}(\mathbf{h} \mid \mathbf{v})$ 。网络的其余部分参与塑造这些统计信息，但权重可以在完全不知道网络其余部分或这些统计信息如何产生的情况下更新。这意味着学习规则是“局部”的，这使得玻尔兹曼机的学习似乎在某种程度上是生物学合理的。我们可以设想每个神经元都是玻尔兹曼机中随机变量的情况，那么连接两个随机变量的轴突和树突只能通过观察与它们物理上实际接触细胞的激发模式来学习。特别地，正相期间，经常同时激活的两个单元之间的连接会被加强。这是 Hebbian 学习规则 (Hebb, 1949) 的一个例子，经常总结为好记的短语——“fire together, wire together”。Hebbian 学习规则是生物系统学习中最古老的假设性解释之一，直至今天仍然有重大意义 (Giudice *et al.*, 2009)。

不仅仅使用局部统计信息的其他学习算法似乎需要假设更多的学习机制。例如，对于大脑在多层感知机中实现的反向传播，似乎需要维持一个辅助通信的网络，并借此向后传输梯度信息。已经有学者 (Hinton, 2007a; Bengio, 2015) 提出生物学上可行（和近似）的反向传播实现方案，但仍然有待验证，Bengio (2015) 还将梯度的反向传播关联到类似于玻尔兹曼机（但具有连续潜变量）的能量模型中的推断。

从生物学的角度看，玻尔兹曼机学习中的负相阶段有点难以解释。正如第 18.2 节所主张的，人类在睡眠时做梦可能是一种形式的负相采样。尽管这个想法更多的只是猜测。

## 20.2 受限玻尔兹曼机

受限玻尔兹曼机以簧风琴（harmonium）之名 (Smolensky, 1986) 面世之后，成为了深度概率模型中最常见的组件之一。我们之前在第 16.7.1 节简要介绍了 RBM。在这里我们回顾以前的内容并探讨更多的细节。RBM 是包含一层可观察变量和单层潜变量的无向概率图模型。RBM 可以堆叠起来（一个在另一个的顶部）形成更深的模型。图 20.1 展示了一些例子。特别地，图 20.1 a 显示 RBM 本身的图结构。它是一个二分图，观察层或潜层中的任何单元之间不允许存在连接。

我们从二值版本的受限玻尔兹曼机开始，但如我们之后所见，这还可以扩展为其他类型的可见和隐藏单元。

更正式地说，令观察层由一组  $n_v$  个二值随机变量组成，我们统称为向量  $\mathbf{v}$ 。我们将  $n_h$  个二值随机变量的潜在或隐藏层记为  $\mathbf{h}$ 。

就像普通的玻尔兹曼机，受限玻尔兹曼机也是基于能量的模型，其联合概率分布由能量函数指定：

$$P(\mathbf{v} = \mathbf{v}, \mathbf{h} = \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h})). \quad (20.4)$$

RBM 的能量函数由下给出

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h} - \mathbf{v}^\top \mathbf{W} \mathbf{h}, \quad (20.5)$$

其中  $Z$  是被称为配分函数的归一化常数：

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp\{-E(\mathbf{v}, \mathbf{h})\} \quad (20.6)$$

从配分函数  $Z$  的定义显而易见，计算  $Z$  的朴素方法（对所有状态进行穷举求和）计算上可能是难以处理的，除非有巧妙设计的算法可以利用概率分布中的规则来更快地计算  $Z$ 。在受限玻尔兹曼机的情况下，Long and Servedio (2010) 正式证明配分函数  $Z$  是难解的。难解的配分函数  $Z$  意味着归一化联合概率分布  $P(\mathbf{v})$  也难以评估。

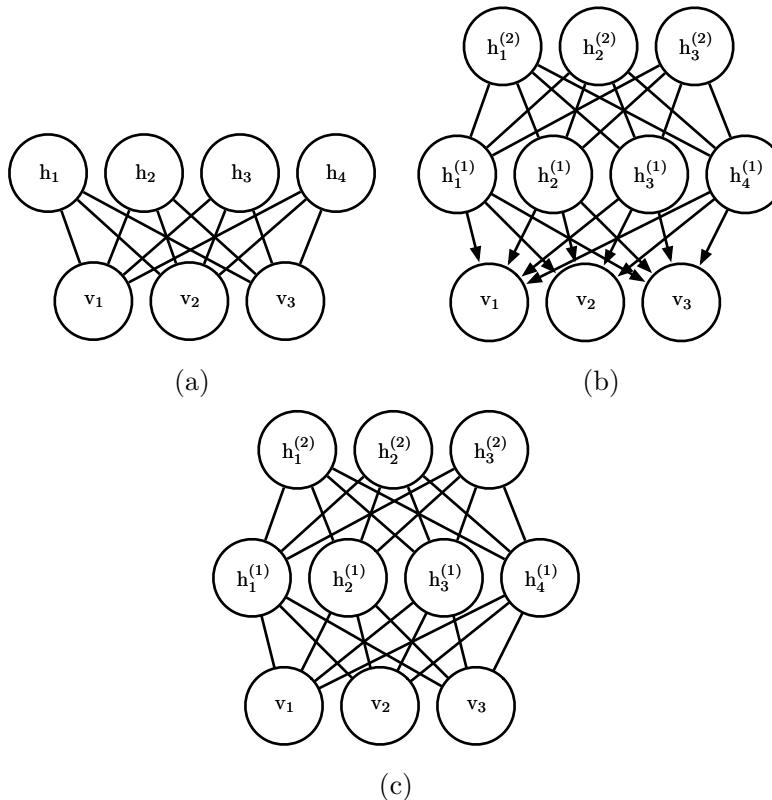


图 20.1: 可以用受限玻尔兹曼机构建的模型示例。(a)受限玻尔兹曼机本身是基于二分图的无向图模型，在图的一部分具有可见单元，另一部分具有隐藏单元。可见单元之间没有连接，隐藏单元之间也没有任何连接。通常每个可见单元连接到每个隐藏单元，但也可以构造稀疏连接的 RBM，如卷积 RBM。(b)深度信念网络是涉及有向和无向连接的混合图模型。与 RBM 一样，它也没有层内连接。然而，DBN 具有两个隐藏层，因此隐藏单元之间的连接在分开的层中。深度信念网络所需的所有局部条件概率分布都直接复制 RBM 的局部条件概率分布。或者，我们也可以用完全无向图表示深度信念网络，但是它需要层内连接来捕获父节点间的依赖关系。(c)深度玻尔兹曼机是具有几层潜变量的无向图模型。与 RBM 和 DBN 一样，DBM 也缺少层内连接。DBM 与 RBM 的联系不如 DBN 紧密。当从 RBM 堆栈初始化 DBM 时，有必要对 RBM 的参数稍作修改。某些种类的 DBM 可以直接训练，而不用先训练一组 RBM。

### 20.2.1 条件分布

虽然  $P(\mathbf{v})$  难解，但 RBM 的二分图结构具有非常特殊的性质，其条件分布  $P(\mathbf{h} | \mathbf{v})$  和  $P(\mathbf{v} | \mathbf{h})$  是因子的，并且计算和采样是相对简单的。

从联合分布中导出条件分布是直观的：

$$P(\mathbf{h} \mid \mathbf{v}) = \frac{P(\mathbf{h}, \mathbf{v})}{P(\mathbf{v})} \quad (20.7)$$

$$= \frac{1}{P(\mathbf{v})} \frac{1}{Z} \exp \left\{ \mathbf{b}^\top \mathbf{v} + \mathbf{c}^\top \mathbf{h} + \mathbf{v}^\top \mathbf{W} \mathbf{h} \right\} \quad (20.8)$$

$$= \frac{1}{Z'} \exp \left\{ \mathbf{c}^\top \mathbf{h} + \mathbf{v}^\top \mathbf{W} \mathbf{h} \right\} \quad (20.9)$$

$$= \frac{1}{Z'} \exp \left\{ \sum_{j=1}^{n_h} \mathbf{c}_j^\top \mathbf{h}_j + \sum_{n_h}^{j=1} \mathbf{v}^\top \mathbf{W}_{:,j} \mathbf{h}_j \right\} \quad (20.10)$$

$$= \frac{1}{Z'} \prod_{j=1}^{n_h} \exp \left\{ \mathbf{c}_j^\top \mathbf{h}_j + \mathbf{v}^\top \mathbf{W}_{:,j} \mathbf{h}_j \right\}. \quad (20.11)$$

由于我们相对可见单元  $\mathbf{v}$  计算条件概率，相对于分布  $P(\mathbf{h} \mid \mathbf{v})$  我们可以将它们视为常数。条件分布  $P(\mathbf{h} \mid \mathbf{v})$  因子相乘的本质，我们可以将向量  $\mathbf{h}$  上的联合概率写成单独元素  $h_j$  上（未归一化）分布的乘积。现在原问题变成了对单个二值  $h_j$  上的分布进行归一化的简单问题。

$$P(h_j = 1 \mid \mathbf{v}) = \frac{\tilde{P}(h_j = 1 \mid \mathbf{v})}{\tilde{P}(h_j = 0 \mid \mathbf{v}) + \tilde{P}(h_j = 1 \mid \mathbf{v})} \quad (20.12)$$

$$= \frac{\exp\{c_j + \mathbf{v}^\top \mathbf{W}_{:,j}\}}{\exp\{0\} + \exp\{c_j + \mathbf{v}^\top \mathbf{W}_{:,j}\}} \quad (20.13)$$

$$= \sigma(c_j + \mathbf{v}^\top \mathbf{W}_{:,j}). \quad (20.14)$$

现在我们可以将关于隐藏层的完全条件分布表达为因子形式：

$$P(\mathbf{h} \mid \mathbf{v}) = \prod_{j=1}^{n_h} \sigma((2\mathbf{h} - 1) \odot (\mathbf{c} + \mathbf{W}^\top \mathbf{v}))_j. \quad (20.15)$$

类似的推导将显示我们感兴趣的另一条件分布， $P(\mathbf{v} \mid \mathbf{h})$  也是因子形式的分布：

$$P(\mathbf{v} \mid \mathbf{h}) = \prod_{i=1}^{n_v} \sigma((2\mathbf{v} - 1) \odot (\mathbf{b} + \mathbf{W} \mathbf{h}))_i. \quad (20.16)$$

## 20.2.2 训练受限玻尔兹曼机

因为 RBM 允许高效计算  $\tilde{P}(\mathbf{v})$  的估计和微分，并且还允许高效地（以块吉布斯采样的形式）进行 MCMC 采样，所以我们很容易使用第十八章中训练具有难以计

算配分函数的技术来训练 RBM。这包括 CD、SML (PCD)、比率匹配等。与深度学习中使用的其他无向模型相比，RBM 可以相对直接地训练，因为我们可以以闭解形式计算  $P(\mathbf{h} | \mathbf{v})$ 。其他一些深度模型，如深度玻尔兹曼机，同时具备难处理的配分函数和难以推断的难题。

## 20.3 深度信念网络

深度信念网络 (deep belief network, DBN) 是第一批成功应用深度架构训练的非卷积模型之一 (Hinton *et al.*, 2006a; Hinton, 2007b)。2006 年深度信念网络的引入开始了当前深度学习的复兴。在引入深度信念网络之前，深度模型被认为太难以优化。具有凸目标函数的核机器引领了研究前沿。深度信念网络在 MNIST 数据集上表现超过内核化支持向量机，以此证明深度架构是能够成功的 (Hinton *et al.*, 2006a)。尽管现在与其他无监督或生成学习算法相比，深度信念网络大多已经失去了青睐并很少使用，但它们在深度学习历史中的重要作用仍应该得到承认。

深度信念网络是具有若干潜变量层的生成模型。潜变量通常是二值的，而可见单元可以是二值或实数。尽管构造连接比较稀疏的 DBN 是可能的，但在一般的模型中，每层的每个单元连接到每个相邻层中的每个单元（没有层内连接）。顶部两层之间的连接是无向的。而所有其他层之间的连接是有向的，箭头指向最接近数据的层。见图 20.1 b 的例子。

具有  $l$  个隐藏层的 DBN 包含  $l$  个权重矩阵： $\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(l)}$ 。同时也包含  $l+1$  个偏置向量： $\mathbf{b}^{(0)}, \dots, \mathbf{b}^{(l)}$ ，其中  $\mathbf{b}^{(0)}$  是可见层的偏置。DBN 表示的概率分布由下式给出：

$$P(\mathbf{h}^{(l)}, \mathbf{h}^{(l-1)}) \propto \exp(\mathbf{b}^{(l)\top} \mathbf{h}^{(l)} + \mathbf{b}^{(l-1)\top} \mathbf{h}^{(l-1)} + \mathbf{h}^{(l-1)\top} \mathbf{W}^{(l)} \mathbf{h}^{(l)}), \quad (20.17)$$

$$P(h_i^{(k)} = 1 | \mathbf{h}^{(k+1)}) = \sigma(b_i^{(k)} + \mathbf{W}_{:,i}^{(k+1)\top} \mathbf{h}^{(k+1)}) \quad \forall i, \forall k \in 1, \dots, l-2, \quad (20.18)$$

$$P(v_i = 1 | \mathbf{h}^{(1)}) = \sigma(b_i^{(0)} + \mathbf{W}_{:,i}^{(1)\top} \mathbf{h}^{(1)}) \quad \forall i. \quad (20.19)$$

在实值可见单元的情况下，替换

$$\mathbf{v} \sim \mathcal{N}(\mathbf{v}; \mathbf{b}^{(0)} + \mathbf{W}^{(1)\top} \mathbf{h}^{(1)}, \boldsymbol{\beta}^{-1}) \quad (20.20)$$

为便于处理， $\boldsymbol{\beta}$  为对角形式。至少在理论上，推广到其他指数族的可见单元是直观的。只有一个隐藏层的 DBN 只是一个 RBM。

为了从 DBN 中生成样本，我们先在顶部的两个隐藏层上运行几个 Gibbs 采样步骤。这个阶段主要从 RBM（由顶部两个隐藏层定义）中采一个样本。然后，我们可以对模型的其余部分使用单次原始采样，以从可见单元绘制样本。

深度信念网络引发许多与有向模型和无向模型同时相关的问题。

由于每个有向层内的相消解释效应，并且由于无向连接的两个隐藏层之间的相互作用，深度信念网络中的推断是难解的。评估或最大化对数似然的标准证据下界也是难以处理的，因为证据下界基于大小等于网络宽度的团的期望。

评估或最大化对数似然，不仅需要面对边缘化潜变量时难以处理的推断问题，而且还需要处理顶部两层无向模型内难处理的配分函数问题。

为训练深度信念网络，我们可以先使用对比散度或随机最大似然方法训练 RBM 以最大化  $\mathbb{E}_{\mathbf{v} \sim p_{\text{data}}} \log p(\mathbf{v})$ 。RBM 的参数定义了 DBN 第一层的参数。然后，第二个 RBM 训练为近似最大化

$$\mathbb{E}_{\mathbf{v} \sim p_{\text{data}}} \mathbb{E}_{\mathbf{h}^{(1)} \sim p^{(1)}(\mathbf{h}^{(1)} | \mathbf{v})} \log p^{(2)}(\mathbf{h}^{(1)}), \quad (20.21)$$

其中  $p^{(1)}$  是第一个 RBM 表示的概率分布， $p^{(2)}$  是第二个 RBM 表示的概率分布。换句话说，第二个 RBM 被训练为模拟由第一个 RBM 的隐藏单元采样定义的分布，而第一个 RBM 由数据驱动。这个过程能无限重复，从而向 DBN 添加任意多层，其中每个新的 RBM 对前一个 RBM 的样本建模。每个 RBM 定义 DBN 的另一层。这个过程可以被视为提高数据在 DBN 下似然概率的变分下界 (Hinton *et al.*, 2006a)。

在大多数应用中，对 DBN 进行贪心逐层训练后，不再花功夫对其进行联合训练。然而，使用醒眠算法对其进行生成精调是可能的。

训练好的 DBN 可以直接用作生成模型，但是 DBN 的大多数兴趣来自于它们改进分类模型的能力。我们可以从 DBN 获取权重，并使用它们定义 MLP：

$$\mathbf{h}^{(1)} = \sigma(b^{(1)} + \mathbf{v}^\top \mathbf{W}^{(1)}), \quad (20.22)$$

$$\mathbf{h}^{(l)} = \sigma(b_i^{(l)} + \mathbf{h}^{(l-1)\top} \mathbf{W}^{(l)}) \quad \forall l \in 2, \dots, m. \quad (20.23)$$

利用 DBN 的生成训练后获得的权重和偏置初始化该 MLP 之后，我们可以训练该 MLP 来执行分类任务。这种 MLP 的额外训练是判别性精调的示例。

与第十九章中从基本原理导出的许多推断方程相比，这种特定选择的 MLP 有些随意。这个 MLP 是一个启发式选择，似乎在实践中效果不错，并在文献中一贯使用。许多近似推断技术是由它们在一些约束下，并在对数似然上找到最大紧变分下

界的能力所驱动的。我们可以使用 DBN 中 MLP 定义的隐藏单元的期望，构造对数似然的变分下界，但这对于隐藏单元上的任何概率分布都是如此，并没有理由相信该 MLP 提供了一个特别的紧界。特别地，MLP 忽略了 DBN 图模型中许多重要的相互作用。MLP 将信息从可见单元向上传播到最深的隐藏单元，但不向下或侧向传播任何信息。DBN 图模型解释了同一层内所有隐藏单元之间的相互作用以及层之间的自顶向下的相互作用。

虽然 DBN 的对数似然是难处理的，但它可以使用 AIS 近似 (Salakhutdinov and Murray, 2008)。通过近似，可以评估其作为生成模型的质量。

术语“深度信念网络”通常不正确地用于指代任意种类的深度神经网络，甚至没有潜变量意义的网络。这个术语应特指最深层中具有无向连接，而在所有其他连续层之间存在向下有向连接的模型。

这个术语也可能导致一些混乱，因为术语“信念网络”有时指纯粹的有向模型，而深度信念网络包含一个无向层。深度信念网络也与动态贝叶斯网络 (dynamic Bayesian networks) (Dean and Kanazawa, 1989) 共享首字母缩写 DBN，动态贝叶斯网络表示马尔可夫链的贝叶斯网络。

## 20.4 深度玻尔兹曼机

**深度玻尔兹曼机** (Deep Boltzmann Machine, DBM) (Salakhutdinov and Hinton, 2009a) 是另一种深度生成模型。与深度信念网络 (DBN) 不同的是，它是一个完全无向的模型。与 RBM 不同的是，DBM 有几层潜变量 (RBM 只有一层)。但是像 RBM 一样，每一层内的每个变量是相互独立的，并条件于相邻层中的变量。见图 20.2 中的图结构。深度玻尔兹曼机已经被应用于各种任务，包括文档建模 (Srivastava *et al.*, 2013)。

与 RBM 和 DBN 一样，DBM 通常仅包含二值单元 (正如我们为简化模型的演示而假设的)，但很容易就能扩展到实值可见单元。

DBM 是基于能量的模型，这意味着模型变量的联合概率分布由能量函数  $E$  参数化。在一个深度玻尔兹曼机包含一个可见层  $\mathbf{v}$  和三个隐藏层  $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}$  和  $\mathbf{h}^{(3)}$  的情况下，联合概率由下式给出：

$$P(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}) = \frac{1}{Z(\boldsymbol{\theta})} \exp(-E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}; \boldsymbol{\theta})). \quad (20.24)$$

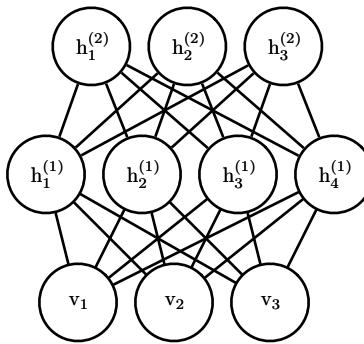


图 20.2: 具有一个可见层 (底部) 和两个隐藏层的深度玻尔兹曼机的图模型。仅在相邻层的单元之间存在连接。没有层内连接。

为简化表示, 下式省略了偏置参数。DBM 能量函数定义如下:

$$E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}; \boldsymbol{\theta}) = -\mathbf{v}^\top \mathbf{W}^{(1)} \mathbf{h}^{(1)} - \mathbf{h}^{(1)\top} \mathbf{W}^{(2)} \mathbf{h}^{(2)} - \mathbf{h}^{(2)\top} \mathbf{W}^{(3)} \mathbf{h}^{(3)}. \quad (20.25)$$

与 RBM 的能量函数 (式 (20.5)) 相比, DBM 能量函数以权重矩阵 ( $\mathbf{W}^{(2)}$  和  $\mathbf{W}^{(3)}$ ) 的形式表示隐藏单元 (潜变量) 之间的连接。正如我们将看到的, 这些连接对模型行为以及我们如何在模型中进行推断都有重要的影响。

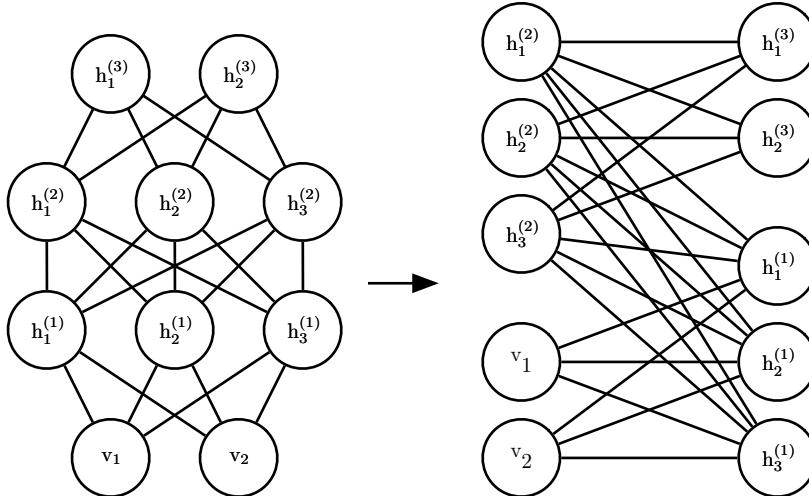


图 20.3: 深度玻尔兹曼机, 重新排列后显示为二分图结构。

与全连接的玻尔兹曼机 (每个单元连接到其他每个单元) 相比, DBM 提供了类

似于 RBM 的一些优点。

具体来说，如图 20.3 所示，DBM 的层可以组织成一个二分图，其中奇数层在一侧，偶数层在另一侧。容易发现，当我们条件于偶数层中的变量时，奇数层中的变量变得条件独立。当然，当我们条件于奇数层中的变量时，偶数层中的变量也会变得条件独立。

DBM 的二分图结构意味着我们可以应用之前用于 RBM 条件分布的相同式子来确定 DBM 中的条件分布。在给定相邻层值的情况下，层内的单元彼此条件独立，因此二值变量的分布可以由 Bernoulli 参数（描述每个单元的激活概率）完全描述。在具有两个隐藏层的示例中，激活概率由下式给出：

$$P(v_i = 1 \mid \mathbf{h}^{(1)}) = \sigma(\mathbf{W}_{i,:}^{(1)} \mathbf{h}^{(1)}), \quad (20.26)$$

$$P(h_i^{(1)} = 1 \mid \mathbf{v}, \mathbf{h}^{(2)}) = \sigma(\mathbf{v}^\top \mathbf{W}_{:,i}^{(1)} + \mathbf{W}_{i,:}^{(2)} \mathbf{h}^{(2)}), \quad (20.27)$$

和

$$P(h_k^{(2)} = 1 \mid \mathbf{h}^{(1)}) = \sigma(\mathbf{h}^{(1)\top} \mathbf{W}_{:,k}^{(2)}). \quad (20.28)$$

二分图结构使 Gibbs 采样能在深度玻尔兹曼机中高效采样。Gibbs 采样的方法是一次只更新一个变量。RBM 允许所有可见单元以一个块的方式更新，而所有隐藏单元在另一个块上更新。我们可以简单地假设具有  $l$  层的 DBM 需要  $l+1$  次更新，每次迭代更新由某层单元组成的块。然而，我们可以仅在两次迭代中更新所有单元。Gibbs 采样可以将更新分成两个块，一块包括所有偶数层（包括可见层），另一个包括所有奇数层。由于 DBM 二分连接模式，给定偶数层，关于奇数层的分布是因子的，因此可以作为块同时且独立地采样。类似地，给定奇数层，可以同时且独立地将偶数层作为块进行采样。高效采样对使用随机最大似然算法的训练尤其重要。

### 20.4.1 有趣的性质

深度玻尔兹曼机具有许多有趣的性质。

DBM 在 DBN 之后开发。与 DBN 相比，DBM 的后验分布  $P(\mathbf{h} \mid \mathbf{v})$  更简单。有点违反直觉的是，这种后验分布的简单性允许更加丰富的后验近似。在 DBN 的情况下，我们使用启发式的近似推断过程进行分类，其中我们可以通过 MLP（使用 sigmoid 激活函数并且权重与原始 DBN 相同）中的向上传播猜测隐藏单元合理的均匀场期望值。任何分布  $Q(\mathbf{h})$  可用于获得对数似然的变分下界。因此这种启发

式的过程让我们能够获得这样的下界。但是，该界没有以任何方式显式优化，所以该界可能是远远不紧的。特别地， $Q$  的启发式估计忽略了相同层内隐藏单元之间的相互作用以及更深层中隐藏单元对更接近输入的隐藏单元自顶向下的反馈影响。因为 DBN 中基于启发式 MLP 的推断过程不能考虑这些相互作用，所以得到的  $Q$  想必远不是最优的。DBM 中，在给定其他层的情况下，层内的所有隐藏单元都是条件独立的。这种层内相互作用的缺失使得通过不动点方程优化变分下界并找到真正最佳的均匀场期望（在一些数值容差内）变得可能的。

使用适当的均匀场允许 DBM 的近似推断过程捕获自顶向下反馈相互作用的影响。这从神经科学的角度来看是有趣的，因为根据已知，人脑使用许多自上而下的反馈连接。由于这个性质，DBM 已被用作真实神经科学现象的计算模型 (Series *et al.*, 2010; Reichert *et al.*, 2011)。

DBM 一个不理想的特性是从中采样是相对困难的。DBN 只需要在其顶部的一对层中使用 MCMC 采样。其他层仅在采样过程末尾涉及，并且只需在一个高效的原始采样过程。要从 DBM 生成样本，必须在所有层中使用 MCMC，并且模型的每一层都参与每个马尔可夫链转移。

## 20.4.2 DBM 均匀场推断

给定相邻层，一个 DBM 层上的条件分布是因子的。在有两个隐藏层的 DBM 的示例中，这些分布是  $P(\mathbf{v} | \mathbf{h}^{(1)})$ ,  $P(\mathbf{h}^{(1)} | \mathbf{v}, \mathbf{h}^{(2)})$  和  $P(\mathbf{h}^{(2)} | \mathbf{h}^{(1)})$ 。因为层之间的相互作用，所有隐藏层上的分布通常不是因子的。在有两个隐藏层的示例中，由于  $\mathbf{h}^{(1)}$  和  $\mathbf{h}^{(2)}$  之间的交互权重  $\mathbf{W}^{(2)}$  使得这些变量相互依赖， $P(\mathbf{h}^{(1)} | \mathbf{v}, \mathbf{h}^{(2)})$  不是因子的。

与 DBN 的情况一样，我们还是要找出近似 DBM 后验分布的方法。然而，与 DBN 不同，DBM 在其隐藏单元上的后验分布（复杂的）很容易用变分近似来近似（如第 19.4 节所讨论），具体是一个均匀场近似。均匀场近似是变分推断的简单形式，其中我们将近似分布限制为完全因子的分布。在 DBM 的情况下，均匀场方程捕获层之间的双向相互作用。在本节中，我们推导出由 Salakhutdinov and Hinton (2009a) 最初引入的迭代近似推断过程。

在推断的变分近似中，我们通过一些相当简单的分布族近似特定目标分布——在这里指给定可见单元时隐藏单元的后验分布。在均匀场近似的情况下，近似族是隐藏单元条件独立的分布集合。

我们现在为具有两个隐藏层的示例推导均匀场方法。令  $Q(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v})$  为  $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v})$  的近似。均匀场假设意味着

$$Q(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v}) = \prod_j Q(h_j^{(1)} | \mathbf{v}) \prod_k Q(h_k^{(2)} | \mathbf{v}). \quad (20.29)$$

均匀场近似试图找到这个分布族中最适合真实后验  $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v})$  的成员。重要的是，每次我们使用  $\mathbf{v}$  的新值时，必须再次运行推断过程以找到不同的分布  $Q$ 。

我们可以设想很多方法来衡量  $Q(\mathbf{h} | \mathbf{v})$  与  $P(\mathbf{h} | \mathbf{v})$  的拟合程度。均匀场方法是最小化

$$\text{KL}(Q || P) = \sum_h Q(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v}) \log \left( \frac{Q(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v})}{P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v})} \right). \quad (20.30)$$

一般来说，除了要保证独立性假设，我们不必提供参数形式的近似分布。变分近似过程通常能够恢复近似分布的函数形式。然而，在二值隐藏单元（我们在那里推导的情况）的均匀场假设的情况下，不会由于预先固定模型的参数而损失一般性。

我们将  $Q$  作为 Bernoulli 分布的乘积进行参数化，即我们将  $\mathbf{h}^{(1)}$  每个元素的概率与一个参数相关联。具体来说，对于每个  $j$ ,  $\hat{h}_j^{(1)} = Q(h_j^{(1)} = 1 | \mathbf{v})$ , 其中  $\hat{h}_j^{(1)} \in [0, 1]$ 。另外，对于每个  $k$ ,  $\hat{h}_k^{(2)} = Q(h_k^{(2)} = 1 | \mathbf{v})$ , 其中  $\hat{h}_k^{(2)} \in [0, 1]$ 。因此，我们有以下近似后验：

$$Q(\mathbf{h}^{(1)}, \mathbf{h}^{(2)} | \mathbf{v}) = \prod_j Q(h_j^{(1)} | \mathbf{v}) \prod_k Q(h_k^{(2)} | \mathbf{v}) \quad (20.31)$$

$$= \prod_j (\hat{h}_j^{(1)})^{h_j^{(1)}} (1 - \hat{h}_j^{(1)})^{(1-h_j^{(1)})} \times \prod_k (\hat{h}_k^{(2)})^{h_k^{(2)}} (1 - \hat{h}_k^{(2)})^{(1-h_k^{(2)})}. \quad (20.32)$$

当然，对于具有更多层的 DBM，近似后验的参数化可以通过明显的方式扩展，即利用图的二分结构，遵循 Gibbs 采样相同的调度，同时更新所有偶数层，然后同时更新所有奇数层。

现在我们已经指定了近似分布  $Q$  的函数族，但仍然需要指定用于选择该函数族中最适合  $P$  的成员的过程。最直接的方法是使用式 (19.56) 指定的均匀场方程。这些方程是通过求解变分下界导数为零的位置而导出。他们以抽象的方式描述如何优化任意模型的变分下界（只需对  $Q$  求期望）。

应用这些一般的方程，我们得到以下更新规则（再次忽略偏置项）：

$$h_j^{(1)} = \sigma \left( \sum_i v_i \mathbf{W}_{i,j}^{(1)} + \sum_{k'} \mathbf{W}_{j,k'}^{(2)} \hat{h}_{k'}^{(2)} \right), \forall j, \quad (20.33)$$

$$\hat{h}_k^{(2)} = \sigma \left( \sum_{j'} \mathbf{W}_{j',k}^{(2)} \hat{h}_{j'}^{(1)} \right), \forall k. \quad (20.34)$$

在该方程组的不动点处，我们具有变分下界  $\mathcal{L}(Q)$  的局部最大值。因此，这些不动点更新方程定义了迭代算法，其中我们交替更新  $h_j^{(1)}$ （使用式 (20.33)）和  $h_k^{(2)}$ （使用式 (20.34)）。对于诸如 MNIST 的小问题，少至 10 次迭代就足以找到用于学习的近似正相梯度，而 50 次通常足以获得要用于高精度分类的单个特定样本的高质量表示。将近似变分推断扩展到更深的 DBM 是直观的。

### 20.4.3 DBM 的参数学习

DBM 中的学习必须面对难解配分函数的挑战（使用第十八章中的技术），以及难解后验分布的挑战（使用第十九章中的技术）。

如第 20.4.2 节中所描述的，变分推断允许构建近似难处理的  $P(\mathbf{h} \mid \mathbf{v})$  的分布  $Q(\mathbf{h} \mid \mathbf{v})$ 。然后通过最大化  $\mathcal{L}(\mathbf{v}, Q, \boldsymbol{\theta})$ （难处理的对数似然的变分下界  $\log P(\mathbf{v}; \boldsymbol{\theta})$ ）学习。

对于具有两个隐藏层的深度玻尔兹曼机， $\mathcal{L}$  由下式给出

$$\mathcal{L}(Q, \boldsymbol{\theta}) = \sum_i \sum_{j'} v_i W_{i,j'}^{(1)} \hat{h}_{j'}^{(1)} + \sum_{j'} \sum_{k'} \hat{h}_{j'}^{(1)} W_{j',k'}^{(2)} \hat{h}_{k'}^{(2)} - \log Z(\boldsymbol{\theta}) + \mathcal{H}(Q). \quad (20.35)$$

该表达式仍然包含对数配分函数  $\log Z(\boldsymbol{\theta})$ 。由于深度玻尔兹曼机包含受限玻尔兹曼机作为组件，用于计算受限玻尔兹曼机的配分函数和采样的困难同样适用于深度玻尔兹曼机。这意味着评估玻尔兹曼机的概率质量函数需要近似方法，如退火重要采样。同样，训练模型需要近似对数配分函数的梯度。见第十八章对这些方法的一般性描述。DBM 通常使用随机最大似然训练。第十八章中描述的许多其他技术都不适用。诸如伪似然的技术需要评估非归一化概率的能力，而不是仅仅获得它们的变分下界。对于深度玻尔兹曼机，对比散度是缓慢的，因为它们不能在给定可见单元时对隐藏单元进行高效采样——反而，每当需要新的负相样本时，对比散度将需要磨合一条马尔可夫链。

非变分版本的随机最大似然算法已经在第 18.2 节讨论过。算法 20.1 给出了应用

于 DBM 的变分随机最大似然算法。回想一下，我们描述的是 DBM 的简化变体（缺少偏置参数）；很容易推广到包含偏置参数的情况。

#### 20.4.4 逐层预训练

不幸的是，随机初始化后使用随机最大似然训练（如上所述）的 DBM 通常导致失败。在一些情况下，模型不能学习如何充分地表示分布。在其他情况下，DBM 可以很好地表示分布，但是没有比仅使用 RBM 获得更高的似然。除第一层之外，所有层都具有非常小权重的 DBM 与 RBM 表示大致相同的分布。

如第 20.4.5 节所述，目前已经开发了允许联合训练的各种技术。然而，克服 DBM 的联合训练问题最初和最流行的方法是贪心逐层预训练。在该方法中，DBM 的每一层被单独视为 RBM，进行训练。第一层被训练为对输入数据进行建模。每个后续 RBM 被训练为对来自前一 RBM 后验分布的样本进行建模。在以这种方式训练了所有 RBM 之后，它们可以被组合成 DBM。然后可以用 PCD 训练 DBM。通常，PCD 训练将仅使模型的参数、由数据上的对数似然衡量的性能、或区分输入的能力发生微小的变化。见图 20.4 展示的训练过程。

这种贪心逐层训练过程不仅仅是坐标上升。因为我们在每个步骤优化参数的一个子集，它与坐标上升具有一些传递相似性。这两种方法是不同的，因为贪心逐层训练过程中，我们在每个步骤都使用了不同的目标函数。

DBM 的贪心逐层预训练与 DBN 的贪心逐层预训练不同。每个单独的 RBM 的参数可以直接复制到相应的 DBN。在 DBM 的情况下，RBM 的参数在包含到 DBM 中之前必须修改。RBM 栈的中间层仅使用自底向上的输入进行训练，但在栈组合形成 DBM 后，该层将同时具有自底向上和自顶向下的输入。为了解释这种效应，Salakhutdinov and Hinton (2009a) 提倡在将其插入 DBM 之前，将所有 RBM（顶部和底部 RBM 除外）的权重除 2。另外，必须使用每个可见单元的两个“副本”来训练底部 RBM，并且两个副本之间的权重约束为相等。这意味着在向上传播时，权重能有效地加倍。类似地，顶部 RBM 应当使用最顶层的两个副本训练。

为了使用深度玻尔兹曼机获得最好结果，我们需要修改标准的 SML 算法，即在联合 PCD 训练步骤的负相期间使用少量的均匀场 (Salakhutdinov and Hinton, 2009a)。具体来说，应当相对于其中所有单元彼此独立的均匀场分布来计算能量梯度的期望。这个均匀场分布的参数应该通过运行一次均匀场不动点方程获得。Goodfellow *et al.* (2013d) 比较了在负相中使用和不使用部分均匀场的中心化 DBM 的性能。

---

**算法 20.1** 用于训练具有两个隐藏层的DBM的变分随机最大似然算法
 

---

设步长  $\epsilon$  为一个小正数

设定吉布斯步数  $k$ , 大到足以让  $p(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}; \boldsymbol{\theta} + \epsilon\Delta_{\boldsymbol{\theta}})$  的马尔可夫链能磨合 (从来自  $p(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}; \boldsymbol{\theta})$  的样本开始)。

初始化三个矩阵,  $\tilde{\mathbf{V}}$ ,  $\tilde{\mathbf{H}}^{(1)}$  和  $\tilde{\mathbf{H}}^{(2)}$  每个都将  $m$  行设为随机值 (例如, 来自 Bernoulli 分布, 边缘分布大致与模型匹配)。

**while** 没有收敛 (学习循环) **do**

从训练数据采包含  $m$  个样本的小批量, 并将它们排列为设计矩阵  $\mathbf{V}$  的行。

初始化矩阵  $\hat{\mathbf{H}}^{(1)}$  和  $\hat{\mathbf{H}}^{(2)}$ , 使其大致符合模型的边缘分布。

**while** 没有收敛 (均匀场推断循环) **do**

$$\hat{\mathbf{H}}^{(1)} \leftarrow \text{sigmoid} \left( \mathbf{V}\mathbf{W}^{(1)} + \hat{\mathbf{H}}^{(2)}\mathbf{W}^{(2)\top} \right).$$

$$\hat{\mathbf{H}}^{(2)} \leftarrow \text{sigmoid} \left( \hat{\mathbf{H}}^{(1)}\mathbf{W}^{(2)} \right).$$

**end while**

$$\Delta_{\mathbf{W}^{(1)}} \leftarrow \frac{1}{m} \mathbf{V}^\top \hat{\mathbf{H}}^{(1)}$$

$$\Delta_{\mathbf{W}^{(2)}} \leftarrow \frac{1}{m} \hat{\mathbf{H}}^{(1)\top} \hat{\mathbf{H}}^{(2)}$$

**for**  $l = 1$  to  $k$  (Gibbs 采样) **do**

Gibbs block 1:

$$\forall i, j, \tilde{V}_{i,j} \text{ 采自 } P(\tilde{V}_{i,j} = 1) = \text{sigmoid} \left( \mathbf{W}_{j,:}^{(1)} \left( \hat{\mathbf{H}}_{i,:}^{(1)} \right)^\top \right).$$

$$\forall i, j, \tilde{H}_{i,j}^{(2)} \text{ 采自 } P(\tilde{H}_{i,j}^{(2)} = 1) = \text{sigmoid} \left( \tilde{\mathbf{H}}_{i,:}^{(1)} \mathbf{W}_{:,j}^{(2)} \right).$$

Gibbs block 2:

$$\forall i, j, \tilde{H}_{i,j}^{(1)} \text{ 采自 } P(\tilde{H}_{i,j}^{(1)} = 1) = \text{sigmoid} \left( \tilde{\mathbf{V}}_{i,:} \mathbf{W}_{:,j}^{(1)} + \tilde{\mathbf{H}}_{i,:}^{(2)} \mathbf{W}_{j,:}^{(2)\top} \right).$$

**end for**

$$\Delta_{\mathbf{W}^{(1)}} \leftarrow \Delta_{\mathbf{W}^{(1)}} - \frac{1}{m} \mathbf{V}^\top \tilde{\mathbf{H}}^{(1)}$$

$$\Delta_{\mathbf{W}^{(2)}} \leftarrow \Delta_{\mathbf{W}^{(2)}} - \frac{1}{m} \tilde{\mathbf{H}}^{(1)\top} \tilde{\mathbf{H}}^{(2)}$$

$\mathbf{W}^{(1)} \leftarrow \mathbf{W}^{(1)} + \epsilon \Delta_{\mathbf{W}^{(1)}}$  (这是大概的描述, 实践中使用的算法更高效, 如具有衰减学习率的动量)

$$\mathbf{W}^{(2)} \leftarrow \mathbf{W}^{(2)} + \epsilon \Delta_{\mathbf{W}^{(2)}}$$

**end while**

---

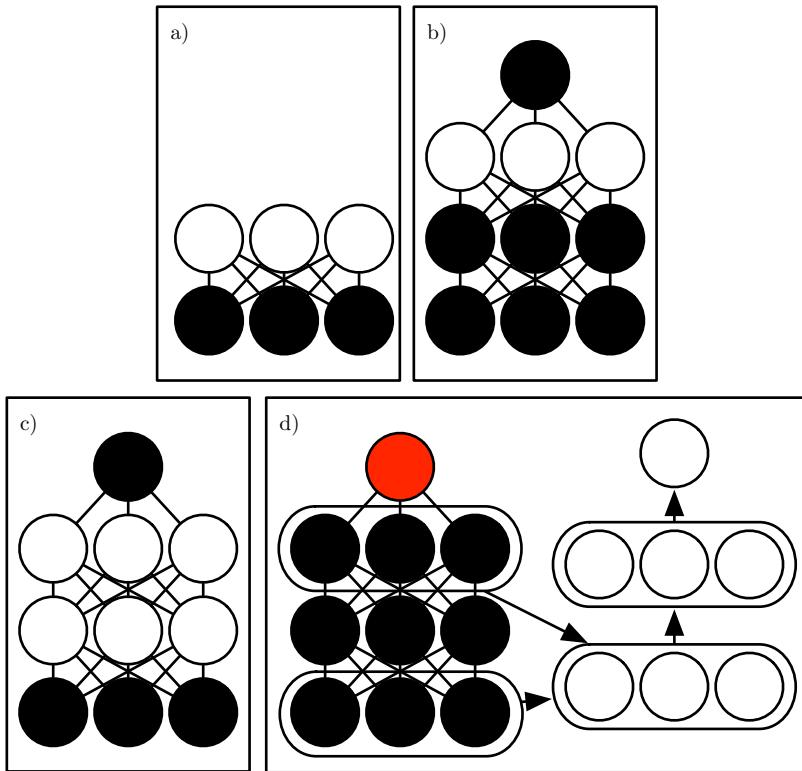


图 20.4: 用于分类 MNIST 数据集的深度玻尔兹曼机训练过程 (Salakhutdinov and Hinton, 2009a; Srivastava *et al.*, 2014)。(a) 使用CD近似最大化  $\log P(\mathbf{v})$  来训练RBM。(b) 训练第二个RBM, 使用CD- $k$  近似最大化  $\log P(\mathbf{h}^{(1)}, \mathbf{y})$  来建模  $\mathbf{h}^{(1)}$  和目标类  $\mathbf{y}$ , 其中  $\mathbf{h}^{(1)}$  采自第一个RBM条件于数据的后验。在学习期间将  $k$  从 1 增加到 20。(c) 将两个RBM组合为DBM。使用  $k = 5$  的随机最大似然训练, 近似最大化  $\log P(\mathbf{v}, \mathbf{y})$ 。(d) 将  $\mathbf{y}$  从模型中删除。定义新的一组特征  $\mathbf{h}^{(1)}$  和  $\mathbf{h}^{(2)}$ , 可在缺少  $\mathbf{y}$  的模型中运行均匀场推断后获得。使用这些特征作为MLP的输入, 其结构与均匀场的额外轮相同, 并且具有用于估计  $\mathbf{y}$  的额外输出层。初始化MLP的权重与DBM的权重相同。使用随机梯度下降和Dropout训练MLP近似最大化  $\log P(\mathbf{y} | \mathbf{v})$ 。图来自 Goodfellow *et al.* (2013d)。

## 20.4.5 联合训练深度玻尔兹曼机

经典 DBM 需要贪心无监督预训练, 并且为了更好的分类, 需要在它们提取的隐藏特征之上, 使用独立的基于 MLP 的分类器。这种方法有一些不理想的性质。因为我们不能在训练第一个 RBM 时评估完整 DBM 的属性, 所以在训练期间难以跟踪性能。因此, 直到相当晚的训练过程, 我们都很难知道我们的超参数表

现如何。DBM 的软件实现需要很多不同的模块，如用于单个 RBM 的 CD 训练、完整 DBM 的 PCD 训练以及基于反向传播的 MLP 训练。最后，玻尔兹曼机顶部的 MLP 失去了玻尔兹曼机概率模型的许多优点，例如当某些输入值丢失时仍能够进行推断的优点。

主要有两种方法可以处理深度玻尔兹曼机的联合训练问题。第一个是**中心化深度玻尔兹曼机**(centered deep Boltzmann machine) (Montavon and Muller, 2012)，通过重参数化模型使其在开始学习过程时代价函数的 Hessian 具有更好的条件数。这个模型不用经过贪心逐层预训练阶段就能训练。这个模型在测试集上获得出色的对数似然，并能产生高质量的样本。不幸的是，作为分类器，它仍然不能与适当正则化的 MLP 竞争。联合训练深度玻尔兹曼机的第二种方式是使用**多预测深度玻尔兹曼机** (multi-prediction deep Boltzmann machine, MP-DBM) (Goodfellow *et al.*, 2013d)。该模型的训练准则允许反向传播算法，以避免使用 MCMC 估计梯度的问题。不幸的是，新的准则不会导致良好的似然性或样本，但是相比 MCMC 方法，它确实会导致更好的分类性能和良好的推断缺失输入的能力。

如果我们回到玻尔兹曼机的一般观点，即包括一组权重矩阵  $\mathbf{U}$  和偏置  $\mathbf{b}$  的单元  $\mathbf{x}$ ，玻尔兹曼机中心化技巧是最容易描述的。回顾式(20.2)，能量函数由下式给出

$$E(\mathbf{x}) = -\mathbf{x}^\top \mathbf{U}\mathbf{x} - \mathbf{b}^\top \mathbf{x}. \quad (20.36)$$

在权重矩阵  $\mathbf{U}$  中使用不同的稀疏模式，我们可以实现不同架构的玻尔兹曼机，如 RBM 或具有不同层数的 DBM。将  $\mathbf{x}$  分割成可见和隐藏单元并将  $\mathbf{U}$  中不相互作用的单元的归零可以实现这些架构。中心化玻尔兹曼机引入了一个向量  $\boldsymbol{\mu}$ ，并从所有状态中减去：

$$E'(\mathbf{x}; \mathbf{U}, \mathbf{b}) = -(\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{U}(\mathbf{x} - \boldsymbol{\mu}) - (\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{b}. \quad (20.37)$$

通常  $\boldsymbol{\mu}$  在开始训练时固定为一个超参数。当模型初始化时，通常选择为  $\mathbf{x} - \boldsymbol{\mu} \approx 0$ 。这种重参数化不改变模型可表示的概率分布的集合，但它确实改变了应用于似然的随机梯度下降的动态。具体来说，在许多情况下，这种重参数化导致更好条件数的 Hessian 矩阵。Melchior *et al.* (2013) 通过实验证实了 Hessian 矩阵条件数的改善，并观察到中心化技巧等价于另一个玻尔兹曼机学习技术——增强梯度 (enhanced gradient) (Cho *et al.*, 2011)。即使在困难的情况下，例如训练多层的深度玻尔兹曼机，Hessian 矩阵条件数的改善也能使学习成功。

联合训练深度玻尔兹曼机的另一种方法是多预测深度玻尔兹曼机 (MP-DBM)，

它将均匀场方程视为定义一系列用于近似求解每个可能推断问题的循环网络 (Goodfellow *et al.*, 2013d)。模型被训练为使每个循环网络获得对相应推断问题的准确答案，而不是训练模型来最大化似然。训练过程如图 20.5 所示。它包括随机采一个训练样本、随机采样推断网络的输入子集，然后训练推断网络来预测剩余单元的值。

这种用于近似推断，通过计算图进行反向传播的一般原理已经应用于其他模型 (Stoyanov *et al.*, 2011; Brakel *et al.*, 2013)。在这些模型和 MP-DBM 中，最终损失不是似然的下界。相反，最终损失通常基于近似推断网络对缺失值施加的近似条件分布。这意味着这些模型的训练有些启发式。如果我们检查由 MP-DBM 学习出来的玻尔兹曼机表示  $p(\mathbf{v})$ ，在 Gibbs 采样产生较差样本的意义下，它倾向于有些缺陷。

通过推断图的反向传播有两个主要优点。首先，它以模型真正使用的方式训练模型——使用近似推断。这意味着在 MP-DBM 中，进行如填充缺失的输入或执行分类（尽管存在缺失的输入）的近似推断比在原始 DBM 中更准确。原始 DBM 不会自己做出准确的分类器；使用原始 DBM 的最佳分类结果是基于 DBM 提取的特征训练独立的分类器，而不是通过使用 DBM 中的推断来计算关于类标签的分布。MP-DBM 中的均匀场推断作为分类器，不需要进行特殊修改就获得良好的表现。通过近似推断反向传播的另一个优点是反向传播计算损失的精确梯度。对于优化而言，比 SML 训练中具有偏差和方差的近似梯度更好。这可能解释了为什么 MP-DBM 可以联合训练，而 DBM 需要贪心逐层预训练。近似推断图反向传播的缺点是它不提供一种优化对数似然的方法，而提供广义伪似然的启发式近似。

MP-DBM 启发了对 NADE 框架的扩展 NADE- $k$  (Raiko *et al.*, 2014)，我们将在第 20.10.10 节中描述。

MP-DBM 与 Dropout 有一定联系。Dropout 在许多不同的计算图之间共享相同的参数，每个图之间的差异是包括还是排除每个单元。MP-DBM 还在许多计算图之间共享参数。在 MP-DBM 的情况下，图之间的差异是每个输入单元是否被观察到。当没有观察到单元时，MP-DBM 不会像 Dropout 那样将其完全删除。相反，MP-DBM 将其视为要推断的潜变量。我们可以想象将 Dropout 应用到 MP-DBM，即额外去除一些单元而不是将它们变为潜变量。

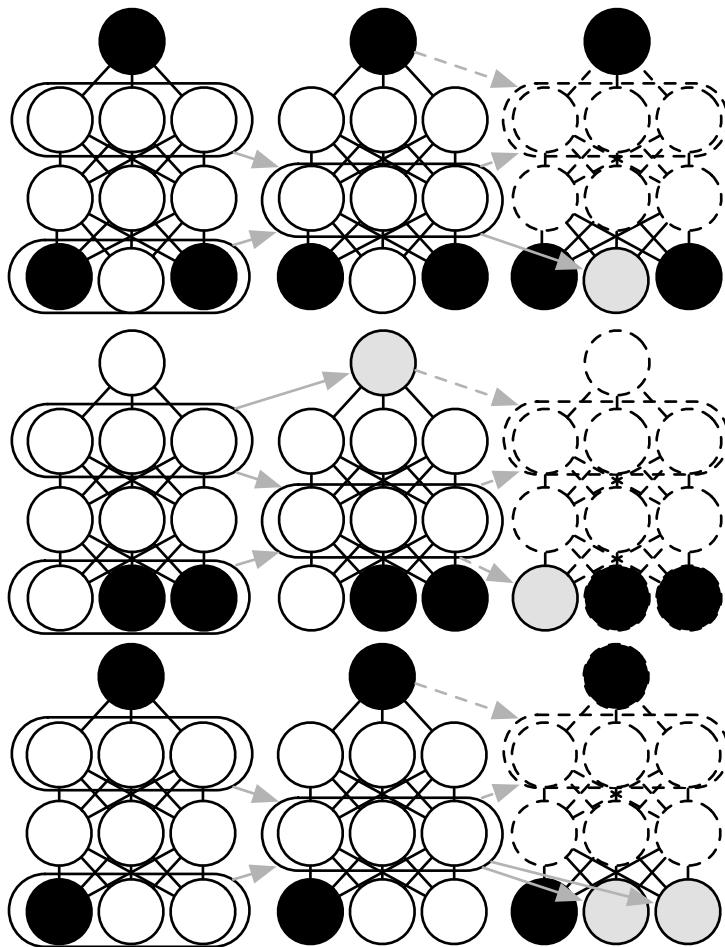


图 20.5: 深度玻尔兹曼机多预测训练过程的示意图。每一行指示相同训练步骤内小批量中的不同样本。每列表示均匀场推断过程中的时间步。对于每个样本，我们对数据变量的子集进行采样，作为推断过程的输入。这些变量以黑色阴影表示条件。然后我们运行均匀场推断过程，箭头指示过程中的哪些变量会影响其他变量。在实际应用中，我们将均匀场展开为几个步骤。在此示意图中，我们只展开为两个步骤。虚线箭头表示获得更多步骤需要如何展开该过程。未用作推断过程输入的数据变量成为目标，以灰色阴影表示。我们可以将每个样本的推断过程视为循环网络。为了使其在给定输入后能产生正确的目标，我们使用梯度下降和反向传播训练这些循环网络。这可以训练 MP-DBM 均匀场过程产生准确的估计。图改编自 Goodfellow *et al.* (2013d)。

## 20.5 实值数据上的玻尔兹曼机

虽然玻尔兹曼机最初是为二值数据而开发的，但是许多应用，例如图像和音频建模似乎需要表示实值上概率分布的能力。在一些情况下，我们可以将区间  $[0, 1]$  中的实值数据视为表示二值变量的期望。例如，Hinton (2000) 将训练集中灰度图像的像素值视为定义  $[0, 1]$  间的概率值。每个像素定义二值变量为 1 的概率，并且二值像素的采样都彼此独立。这是评估灰度图像数据集上二值模型的常见过程。然而，这种方法理论上并不特别令人满意，并且以这种方式独立采样的二值图像具有噪声表象。在本节中，我们介绍概率密度定义在实值数据上的玻尔兹曼机。

### 20.5.1 Gaussian-Bernoulli RBM

受限玻尔兹曼机可以用于许多指数族的条件分布 (Welling *et al.*, 2005)。其中，最常见的是具有二值隐藏单元和实值可见单元的 RBM，其中可见单元上的条件分布是高斯分布（均值为隐藏单元的函数）。

有很多方法可以参数化 Gaussian-Bernoulli RBM。首先，我们可以选择协方差矩阵或精度矩阵来参数化高斯分布。这里，我们介绍选择精度矩阵的情况。我们可以通过简单的修改获得协方差的形式。我们希望条件分布为

$$p(\mathbf{v} \mid \mathbf{h}) = \mathcal{N}(\mathbf{v}; \mathbf{Wh}, \boldsymbol{\beta}^{-1}). \quad (20.38)$$

通过扩展未归一化的对数条件分布可以找到需要添加到能量函数中的项：

$$\log \mathcal{N}(\mathbf{v}; \mathbf{Wh}, \boldsymbol{\beta}^{-1}) = -\frac{1}{2}(\mathbf{v} - \mathbf{Wh})^\top \boldsymbol{\beta} (\mathbf{v} - \mathbf{Wh}) + f(\boldsymbol{\beta}). \quad (20.39)$$

此处  $f$  封装所有的参数，但不包括模型中的随机变量。因为  $f$  的唯一作用是归一化分布，并且我们选择的任何可作为配分函数的能量函数都能起到这个作用，所以我们忽略  $f$ 。

如果我们在能量函数中包含式 (20.39) 中涉及  $\mathbf{v}$  的所有项（其符号被翻转），并且不添加任何其他涉及  $\mathbf{v}$  的项，那么我们的能量函数就能表示想要的条件分布  $p(\mathbf{v} \mid \mathbf{h})$ 。

其他条件分布比较自由，如  $p(\mathbf{h} \mid \mathbf{v})$ 。注意式 (20.39) 包含一项

$$\frac{1}{2} \mathbf{h}^\top \mathbf{W}^\top \boldsymbol{\beta} \mathbf{Wh}. \quad (20.40)$$

因为该项包含  $h_i h_j$  项，它不能被全部包括在内。这些对应于隐藏单元之间的边。如果我们包括这些项，我们将得到一个线性因子模型，而不是受限玻尔兹曼机。当设计我们的玻尔兹曼机时，我们简单地省略这些  $h_i h_j$  交叉项。省略这些项不改变条件分布  $p(\mathbf{v} | \mathbf{h})$ ，因此式 (20.39) 仍满足。然而，我们仍然可以选择是否包括仅涉及单个  $h_i$  的项。如果我们假设精度矩阵是对角的，就能发现对于每个隐藏单元  $h_i$ ，我们有一项

$$\frac{1}{2} h_i \sum_j \beta_j W_{j,i}^2. \quad (20.41)$$

在上面，我们使用了  $h_i^2 = h_i$  的事实（因为  $h_i \in \{0, 1\}$ ）。如果我们在能量函数中包含此项（符号被翻转），则当该单元的权重较大且以高精度连接到可见单元时，偏置  $h_i$  将自然被关闭。是否包括该偏置项不影响模型可以表示的分布族（假设我们包括隐藏单元的偏置参数），但是它确实会影响模型的学习动态。包括该项可以帮助隐藏单元（即使权重在幅度上快速增加时）保持合理激活。

因此，在 Gaussian-Bernoulli RBM 上定义能量函数的一种方式：

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2} \mathbf{v}^\top (\boldsymbol{\beta} \odot \mathbf{v}) - (\mathbf{v} \odot \boldsymbol{\beta})^\top \mathbf{W} \mathbf{h} - \mathbf{b}^\top \mathbf{h}, \quad (20.42)$$

但我们还可以添加额外的项或者通过方差而不是精度参数化能量。

在这个推导中，我们没有在可见单元上添加偏置项，但添加这样的偏置是容易的。Gaussian-Bernoulli RBM 参数化一个最终变化的来源是如何处理精度矩阵的选择。它可以被固定为常数（可能基于数据的边缘精度估计）或学习出来。它也可以是标量乘以单位矩阵，或者是一个对角矩阵。在此情况下，由于一些操作需要对矩阵求逆，我们通常不允许非对角的精度矩阵，因为高斯分布的一些操作需要对矩阵求逆，一个对角矩阵可以非常容易地被求逆。在接下来的章节中，我们将看到其他形式的玻尔兹曼机，它们允许对协方差结构建模，并使用各种技术避免对精度矩阵求逆。

## 20.5.2 条件协方差的无向模型

虽然高斯 RBM 已成为实值数据的标准能量模型，Ranzato *et al.* (2010a) 认为高斯 RBM 感应偏置不能很好地适合某些类型的实值数据中存在的统计变化，特别是自然图像。问题在于自然图像中的许多信息内容嵌入于像素之间的协方差而不是原始像素值中。换句话说，图像中的大多数有用信息在于像素之间的关系，而不是其绝对值。由于高斯 RBM 仅对给定隐藏单元的输入条件均值建模，所以它不能捕

获条件协方差信息。为了回应这些评论，已经有学者提出了替代模型，设法更好地考虑实值数据的协方差。这些模型包括 **均值和协方差 RBM** ( mean and covariance RBM, mcRBM )<sup>1</sup>、**学生 t 分布均值乘积** ( mean product of Student t-distribution, mPoT ) 模型和 **尖峰和平板 RBM** ( spike and slab RBM, ssRBM )。

**均值和协方差 RBM** mcRBM 使用隐藏单元独立地编码所有可观察单元的条件均值和协方差。mcRBM 的隐藏层分为两组单元：均值单元和协方差单元。建模条件均值的那组单元是简单的高斯 RBM。另一半是 **协方差 RBM** ( covariance RBM, cRBM ) (Ranzato *et al.*, 2010a)，对条件协方差的结构进行建模（如下所述）。

具体来说，在二值均值的单元  $\mathbf{h}^{(m)}$  和二值协方差单元  $\mathbf{h}^{(c)}$  的情况下，mcRBM 模型被定义为两个能量函数的组合：

$$E_{\text{mc}}(\mathbf{x}, \mathbf{h}^{(m)}, \mathbf{h}^{(c)}) = E_{\text{m}}(\mathbf{x}, \mathbf{h}^{(m)}) + E_{\text{c}}(\mathbf{x}, \mathbf{h}^{(c)}), \quad (20.43)$$

其中  $E_{\text{m}}$  为标准的 Gaussian-Bernoulli RBM 能量函数<sup>2</sup>，

$$E_{\text{m}}(\mathbf{x}, \mathbf{h}^{(m)}) = \frac{1}{2} \mathbf{x}^\top \mathbf{x} - \sum_j \mathbf{x}^\top \mathbf{W}_{:,j} h_j^{(m)} - \sum_j b_j^{(m)} h_j^{(m)}, \quad (20.44)$$

$E_{\text{c}}$  是 cRBM 建模条件协方差信息的能量函数：

$$E_{\text{c}}(\mathbf{x}, \mathbf{h}^{(c)}) = \frac{1}{2} \sum_j h_j^{(c)} (\mathbf{x}^\top \mathbf{r}^{(j)})^2 - \sum_j b_j^{(c)} h_j^{(c)}. \quad (20.45)$$

参数  $\mathbf{r}^{(j)}$  与  $h_j^{(c)}$  关联的协方差权重向量对应， $\mathbf{b}^{(c)}$  是一个协方差偏置向量。组合后的能量函数定义联合分布，

$$p_{\text{mc}}(\mathbf{x}, \mathbf{h}^{(m)}, \mathbf{h}^{(c)}) = \frac{1}{Z} \exp \left\{ -E_{\text{mc}}(\mathbf{x}, \mathbf{h}^{(m)}, \mathbf{h}^{(c)}) \right\}, \quad (20.46)$$

以及给定  $\mathbf{h}^{(m)}$  和  $\mathbf{h}^{(c)}$  后，关于观察数据相应的条件分布（为一个多元高斯分布）：

$$p_{\text{mc}}(\mathbf{x} | \mathbf{h}^{(m)}, \mathbf{h}^{(c)}) = \mathcal{N} \left( \mathbf{x}; \mathbf{C}_{\mathbf{x}|\mathbf{h}}^{\text{mc}} \left( \sum_j \mathbf{W}_{:,j} h_j^{(m)} \right), \mathbf{C}_{\mathbf{x}|\mathbf{h}}^{\text{mc}} \right). \quad (20.47)$$

注意协方差矩阵  $\mathbf{C}_{\mathbf{x}|\mathbf{h}}^{\text{mc}} = \left( \sum_j h_j^{(c)} \mathbf{r}^{(j)} \mathbf{r}^{(j)T} + \mathbf{I} \right)^{-1}$  是非对角的，且  $\mathbf{W}$  是与建模条件均值的高斯 RBM 相关联的权重矩阵。由于非对角的条件协方差结构，难以通过对

<sup>1</sup>术语“mcRBM”根据字母 M-C-R-B-M 发音；“mc”不是“McDonald’s”中的“Mc”的发音。

<sup>2</sup>这个版本的 Gaussian-Bernoulli RBM 能量函数假定图像数据的每个像素具有零均值。考虑非零像素均值时，可以简单地将像素偏移添加到模型中。

比散度或持续性对比散度来训练 mcRBM。CD 和 PCD 需要从  $\mathbf{x}, \mathbf{h}^{(m)}, \mathbf{h}^{(c)}$  的联合分布中采样，这在标准RBM 中可以通过 Gibbs 采样在条件分布上采样实现。但是，在 mcRBM 中，从  $p_{\text{mc}}(\mathbf{x} | \mathbf{h}^{(m)}, \mathbf{h}^{(c)})$  中抽样需要在学习的每个迭代计算  $(\mathbf{C}^{\text{mc}})^{-1}$ 。这对于更大的观察数据可能是不切实际的计算负担。Ranzato and Hinton (2010) 通过使用 mcRBM 自由能上的哈密尔顿（混合）蒙特卡罗 (Neal, 1993) 直接从边缘  $p(\mathbf{x})$  采样，避免了直接从条件  $p_{\text{mc}}(\mathbf{x} | \mathbf{h}^{(m)}, \mathbf{h}^{(c)})$  抽样。

**学生  $t$  分布均值乘积** 学生  $t$  分布均值乘积 ( mPoT ) 模型 (Ranzato *et al.*, 2010b) 以类似 mcRBM 扩展 cRBM 的方式扩展 PoT 模型 (Welling *et al.*, 2003a)。通过添加类似高斯 RBM 中隐藏单元的非零高斯均值来实现。与 mcRBM 一样，观察值上的 PoT 条件分布是多元高斯（具有非对角的协方差）分布；然而，不同于 mcRBM，隐藏变量的互补条件分布是由条件独立的 Gamma 分布给出。Gamma 分布  $\mathcal{G}(k, \theta)$  是关于正实数且均值为  $k\theta$  的概率分布。我们只需简单地了解 Gamma 分布就足以理解 mPoT 模型的基本思想。

mPoT 的能量函数为：

$$E_{\text{mPoT}}(\mathbf{x}, \mathbf{h}^{(m)}, \mathbf{h}^{(c)}) \quad (20.48)$$

$$= E_m(\mathbf{x}, \mathbf{h}^{(m)}) + \sum_j \left( h_j^{(c)} \left( 1 + \frac{1}{2} (\mathbf{r}^{(j)T} \mathbf{x})^2 \right) + (1 - \gamma_j) \log h_j^{(c)} \right), \quad (20.49)$$

其中  $\mathbf{r}^{(j)}$  是与单元  $h_j^{(c)}$  相关联的协方差权重向量， $E_m(\mathbf{x}, \mathbf{h}^{(m)})$  如式 (20.44) 所定义。

正如 mcRBM 一样，mPoT 模型能量函数指定一个多元高斯分布，其中关于  $\mathbf{x}$  的条件分布具有非对角的协方差。mPoT 模型中的学习（也像 mcRBM）由于无法从非对角高斯条件分布  $p_{\text{mPoT}}(\mathbf{x} | \mathbf{h}^{(m)}, \mathbf{h}^{(c)})$  采样而变得复杂。因此 Ranzato *et al.* (2010b) 也倡导通过哈密尔顿（混合）蒙特卡罗 (Neal, 1993) 直接采样  $p(\mathbf{x})$ 。

**尖峰和平板 RBM** 尖峰和平板 RBM (spike and slab RBM, ssRBM) (Courville *et al.*, 2011b) 提供对实值数据的协方差结构建模的另一种方法。与 mcRBM 相比，ssRBM 具有既不需要矩阵求逆也不需要哈密尔顿蒙特卡罗方法的优点。就像 mcRBM 和 mPoT 模型，ssRBM 的二值隐藏单元通过使用辅助实值变量来编码跨像素的条件协方差。

尖峰和平板 RBM 有两类隐藏单元：二值 尖峰 (spike) 单元  $\mathbf{h}$  和实值 平板 (slab) 单元  $\mathbf{s}$ 。条件于隐藏单元的可见单元均值由  $(\mathbf{h} \odot \mathbf{s}) \mathbf{W}^\top$  给出。换句话说，每一列  $\mathbf{W}_{:, i}$

定义当  $h_i = 1$  时可出现在输入中的分量。相应的尖峰变量  $h_i$  确定该分量是否存在。如果存在的话，相应的平板变量  $s_i$  确定该分量的强度。当尖峰变量激活时，相应的平板变量将沿着  $\mathbf{W}_{:,i}$  定义的轴的输入增加方差。这允许我们对输入的协方差建模。幸运的是，使用 Gibbs 采样的对比散度和持续性对比散度仍然适用。此处无需对任何矩阵求逆。

形式上，ssRBM 模型通过其能量函数定义：

$$E_{ss}(\mathbf{x}, \mathbf{s}, \mathbf{h}) = -\sum_i \mathbf{x}^\top \mathbf{W}_{:,i} s_i h_i + \frac{1}{2} \mathbf{x}^\top \left( \mathbf{\Lambda} + \sum_i \mathbf{\Phi}_i h_i \right) \mathbf{x} \quad (20.50)$$

$$+ \frac{1}{2} \sum_i \alpha_i s_i^2 - \sum_i \alpha_i \mu_i s_i h_i - \sum_i b_i h_i + \sum_i \alpha_i \mu_i^2 h_i, \quad (20.51)$$

其中  $b_i$  是尖峰  $h_i$  的偏置， $\mathbf{\Lambda}$  是观测值  $\mathbf{x}$  上的对角精度矩阵。参数  $\alpha_i > 0$  是实值平板变量  $s_i$  的标量精度参数。参数  $\mathbf{\Phi}_i$  是定义  $\mathbf{x}$  上的  $\mathbf{h}$  调制二次惩罚的非负对角矩阵。每个  $\mu_i$  是平板变量  $s_i$  的均值参数。

利用能量函数定义的联合分布，能相对容易地导出 ssRBM 条件分布。例如，通过边缘化平板变量  $\mathbf{s}$ ，给定二值尖峰变量  $\mathbf{h}$ ，关于观察量的条件分布由下式给出

$$p_{ss}(\mathbf{x} | \mathbf{h}) = \frac{1}{P(\mathbf{h})} \frac{1}{Z} \int \exp\{-E(\mathbf{x}, \mathbf{s}, \mathbf{h})\} d\mathbf{s} \quad (20.52)$$

$$= \mathcal{N}\left(\mathbf{x}; \mathbf{C}_{x|h}^{ss} \sum_i \mathbf{W}_{:,i} \mu_i h_i, \mathbf{C}_{x|h}^{ss}\right) \quad (20.53)$$

其中  $\mathbf{C}_{x|h}^{ss} = (\mathbf{\Lambda} + \sum_i \mathbf{\Phi}_i h_i - \sum_i \alpha_i^{-1} h_i \mathbf{W}_{:,i} \mathbf{W}_{:,i}^\top)^{-1}$ 。最后的等式只有在协方差矩阵  $\mathbf{C}_{x|h}^{ss}$  正定时成立。

由尖峰变量选通意味着  $\mathbf{h} \odot \mathbf{s}$  上的真实边缘分布是稀疏的。这不同于稀疏编码，其中来自模型的样本在编码中“几乎从不”（在测度理论意义上）包含零，并且需要MAP推断来强加稀疏性。

相比 mcRBM 和 mPoT 模型，ssRBM 以明显不同的方式参数化观察量的条件协方差。mcRBM 和 mPoT 都通过  $(\sum_j h_j^{(c)} \mathbf{r}^{(j)} \mathbf{r}^{(j)\top} + \mathbf{I})^{-1}$  建模观察量的协方差结构，使用  $h_j > 0$  的隐藏单元的激活来对方向  $\mathbf{r}^{(j)}$  的条件协方差施加约束。相反，ssRBM 使用隐藏尖峰激活  $h_i = 1$  来指定观察结果的条件协方差，以沿着由相应权重向量指定的方向捏合精度矩阵。ssRBM 条件协方差与一个不同模型给出的类似：概率主成分分析的乘积（PoPPCA）(Williams and Agakov, 2002)。在过完备的设定下，ssRBM 参数化的稀疏激活仅允许在稀疏激活  $h_i$  的所选方向上有显著方差（高

于由  $\Lambda^{-1}$  给出的近似方差)。在 mcRBM 或 mPoT 模型中, 过完备的表示意味着, 捕获观察空间中特定方向上的变化需要在该方向上的正交投影下去除潜在的所有约束。这表明这些模型不太适合于过完备设定。

尖峰和平板 RBM 的主要缺点是参数的一些设置会对应于非正定的协方差矩阵。这种协方差矩阵会在离均值更远的值上放置更大的未归一化概率, 导致所有可能结果上的积分发散。通常这个问题可以通过简单的启发式技巧来避免。理论上还没有任何令人满意的解决方法。使用约束优化来显式地避免概率未定义的区域(不过分保守是很难做到的), 并且这还会阻止模型到达参数空间的高性能区域。

定性地, ssRBM 的卷积变体能产生自然图像的优秀样本。图 16.1 中展示了一些样例。

ssRBM 允许几个扩展, 包括平板变量的高阶交互和平均池化 (Courville *et al.*, 2014) 使得模型能够在标注数据稀缺时为分类器学习到出色的特征。向能量函数添加一项能防止配分函数在稀疏编码模型下变得不确定, 如尖峰和平板稀疏编码 (Goodfellow *et al.*, 2013g), 也称为 S3C。

## 20.6 卷积玻尔兹曼机

如第九章所示, 超高维度输入(如图像)会对机器学习模型的计算、内存和统计要求造成很大的压力。通过使用小核的离散卷积来替换矩阵乘法是解决具有空间平移不变性或时间结构的输入问题的标准方式。Desjardins 和 Bengio (2008) 表明这种方法应用于 RBM 时效果很好。

深度卷积网络通常需要池化操作, 使得每个连续层的空间大小减小。前馈卷积网络通常使用池化函数, 例如池化元素的最大值。目前尚不清楚如何将其推广到基于能量的模型的设定中。我们可以在  $n$  个二值检测器单元  $\mathbf{d}$  上引入二值池化单元  $p$ , 强制  $p = \max_i d_i$ , 并且当违反约束时将能量函数设置为  $\infty$ 。因为它需要评估  $2^n$  个不同的能量设置来计算归一化常数, 这种方式不能很好地扩展。对于小的  $3 \times 3$  池化区域, 每个池化单元需要评估  $2^9 = 512$  个能量函数!

Lee *et al.* (2009) 针对这个问题, 开发了一个称为 **概率最大池化** (probabilistic max pooling) 的解决方案(不要与“随机池化”混淆, “随机池化”是用于隐含地构建卷积前馈网络集成的技术)。概率最大池化背后的策略是约束检测器单元, 使得一次最多只有一个可以处于活动状态。这意味着仅存在  $n + 1$  个总状态 ( $n$  个检测器

单元中某一个状态为开和一个对应于所有检测器单元关闭的附加状态)。当且仅当检测器单元中的一个开启时,池化单元打开。所有单元的状态关闭时,能量被分配为零。我们可以认为这是在用包含  $n + 1$  个状态的单个变量来描述模型,或者等价地具有  $n + 1$  个变量的模型,除了  $n + 1$  个联合分配的变量之外的能量赋为  $\infty$ 。

虽然高效的概率最大池化确实能强迫检测器单元互斥,这在某些情景下可能是有用的正则化约束而在其他情景下是对模型容量有害的限制。它也不支持重叠池化区域。从前馈卷积网络获得最佳性能通常需要重叠的池化区域,因此这种约束可能大大降低了卷积玻尔兹曼机的性能。

Lee *et al.* (2009) 证明概率最大池化可以用于构建卷积深度玻尔兹曼机<sup>3</sup>。该模型能够执行诸如填补输入缺失部分的操作。虽然这种模型在理论上具有吸引力,让它在实践中工作是具有挑战性的,作为分类器通常不如通过监督训练的传统卷积网络。

许多卷积模型对于许多不同空间大小的输入同样有效。对于玻尔兹曼机,由于各种原因很难改变输入尺寸。配分函数随着输入大小的改变而改变。此外,许多卷积网络按与输入大小成比例地缩放池化区域来实现尺寸不变性,但缩放玻尔兹曼机池化区域是不优雅的。传统的卷积神经网络可以使用固定数量的池化单元并且动态地增加它们池化区域的大小,以此获得可变大小输入的固定尺寸的表示。对于玻尔兹曼机,大型池化区域的计算成本比朴素方法高很多。Lee *et al.* (2009) 的方法使得每个检测器单元在相同的池化区域中互斥,解决了计算问题,但仍然不允许大小可变的池化区域。例如,假设我们在学习边缘检测器时,检测器单元上具有  $2 \times 2$  的概率最大池化。这强制约束在每个  $2 \times 2$  的区域中只能出现这些边中的一条。如果我们随后在每个方向上将输入图像的大小增加 50%,则期望边缘的数量会相应地增加。相反,如果我们在每个方向上将池化区域的大小增加 50% 到  $3 \times 3$ ,则互斥性约束现在指定这些边中的每一个在  $3 \times 3$  区域中仅可以出现一次。当我们以这种方式增长模型的输入图像时,模型会生成密度较小的边。当然,这些问题只有在模型必须使用可变数量的池化,以便产出固定大小的输出向量时才会出现。只要模型的输出是可以与输入图像成比例缩放的特征图,使用概率最大池化的模型仍然可以接受可变大小的输入图像。

图像边界处的像素也带来一些困难,由于玻尔兹曼机中的连接是对称的事实而加剧。如果我们不隐式地补零输入,则将会导致比可见单元更少的隐藏单元,并且图像边界处的可见单元将不能被良好地建模,因为它们位于较少隐藏单元的接受场

<sup>3</sup>该论文将模型描述为“深度信念网络”,但因为它可以被描述为纯无向模型(具有易处理逐层均匀场不动点更新),所以它最适合深度玻尔兹曼机的定义。

中。然而，如果我们隐式地补零输入，则边界处的隐藏单元将由较少的输入像素驱动，并且可能在需要时无法激活。

## 20.7 用于结构化或序列输出的玻尔兹曼机

在结构化输出场景中，我们希望训练可以从一些输入  $\mathbf{x}$  映射到一些输出  $\mathbf{y}$  的模型， $\mathbf{y}$  的不同条目彼此相关，并且必须遵守一些约束。例如，在语音合成任务中， $\mathbf{y}$  是波形，并且整个波形听起来必须像连贯的发音。

表示  $\mathbf{y}$  中的条目之间关系的自然方式是使用概率分布  $p(\mathbf{y} \mid \mathbf{x})$ 。扩展到建模条件分布的玻尔兹曼机可以支持这种概率模型。

使用玻尔兹曼机条件建模的相同工具不仅可以用于结构化输出任务，还可以用于序列建模。在后一种情况下，模型必须估计变量序列上的概率分布  $p(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau)})$ ，而不仅仅是将输入  $\mathbf{x}$  映射到输出  $\mathbf{y}$ 。为完成这个任务，条件玻尔兹曼机可以表示  $p(\mathbf{x}^{(\tau)} \mid \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau-1)})$  形式的因子。

视频游戏和电影工业中一个重要序列建模任务是建模用于渲染 3-D 人物骨架关节角度的序列。这些序列通常通过记录角色移动的运动捕获系统收集。人物运动的概率模型允许生成新的（之前没见过的）但真实的动画。为了解决这个序列建模任务，Taylor *et al.* (2007) 针对小的  $m$  引入了条件 RBM 建模  $p(\mathbf{x}^{(t)} \mid \mathbf{x}^{(t-1)}, \dots, \mathbf{x}^{(t-m)})$ 。该模型是  $p(\mathbf{x}^{(t)})$  上的 RBM，其偏置参数是  $\mathbf{x}$  前面  $m$  个值的线性函数。当我们条件于  $\mathbf{x}^{(t-1)}$  的不同值和更早的变量时，我们会得到一个关于  $\mathbf{x}$  的新 RBM。RBM 关于  $\mathbf{x}$  的权重不会改变，但是条件于不同的过去值，我们可以改变 RBM 中的不同隐藏单元处于活动状态的概率。通过激活和去激活隐藏单元的不同子集，我们可以对  $\mathbf{x}$  上诱导的概率分布进行大的改变。条件 RBM 的其他变体 (Mnih *et al.*, 2011) 和使用条件 RBM 进行序列建模的其他变体是可能的 (Taylor and Hinton, 2009; Sutskever *et al.*, 2009; Boulanger-Lewandowski *et al.*, 2012)。

另一个序列建模任务是对构成歌曲音符序列的分布进行建模。Boulanger-Lewandowski *et al.* (2012) 引入了 **RNN-RBM** 序列模型并应用于这个任务。RNN-RBM 由 RNN（产生用于每个时间步的 RBM 参数）组成，是帧序列  $\mathbf{x}^{(t)}$  的生成模型。与之前只有 RBM 的偏置参数会在一个时间步到下一个发生变化的方法不同，RNN-RBM 使用 RNN 来产生 RBM 的所有参数（包括权重）。为了训练模型，我们需要能够通过 RNN 反向传播损失函数的梯度。损失函数不直接应用于 RNN 输出。

相反，它应用于 RBM。这意味着我们必须使用对比散度或相关算法关于 RBM 参数进行近似的微分。然后才可以使用通常的通过时间反向传播算法通过 RNN 反向传播该近似梯度。

## 20.8 其他玻尔兹曼机

玻尔兹曼机的许多其他变种是可能的。

玻尔兹曼机可以用不同的训练准则扩展。我们专注于训练为大致最大化生成标准  $\log p(\mathbf{v})$  的玻尔兹曼机。相反，旨在最大化  $\log p(y \mid \mathbf{v})$  来训练判别的 RBM 也是有可能的 (Larochelle and Bengio, 2008a)。当使用生成性和判别性标准的线性组合时，该方法通常表现最好。不幸的是，至少使用现有的方法来看，RBM 似乎并不如 MLP 那样的监督学习器强大。

在实践中使用的大多数玻尔兹曼机在其能量函数中仅具有二阶相互作用，意味着它们的能量函数是许多项的和，并且每个单独项仅包括两个随机变量之间的乘积。这种项的一个例子是  $v_i W_{i,j} h_j$ 。我们还可以训练高阶玻尔兹曼机 (Sejnowski, 1987)，其中能量函数项涉及许多变量的乘积。隐藏单元和两个不同图像之间的三向交互可以建模从一个视频帧到下一个帧的空间变换 (Memisevic and Hinton, 2007, 2010)。通过one-hot类别变量的乘法可以根据存在哪个类来改变可见单元和隐藏单元之间的关系 (Nair and Hinton, 2009)。使用高阶交互的一个最近的示例是具有两组隐藏单元的玻尔兹曼机，一组同时与可见单元  $\mathbf{v}$  和类别标签  $y$  交互，另一组仅与输入值  $\mathbf{v}$  交互 (Luo *et al.*, 2011)。这可以被解释为鼓励一些隐藏单元学习使用与类相关的特征来建模输入，而且还学习额外的隐藏单元（不需要根据样本类别，学习逼真  $\mathbf{v}$  样本所需的繁琐细节）。高阶交互的另一个用途是选通一些特征。Sohn *et al.* (2013) 介绍了一个带有三阶交互的玻尔兹曼机，以及与每个可见单元相关的二进制掩码变量。当这些掩码变量设置为零时，它们消除可见单元对隐藏单元的影响。这允许将与分类问题不相关的可见单元从估计类别的推断路径中移除。

更一般地说，玻尔兹曼机框架是一个丰富的模型空间，允许比迄今为止已经探索的更多的模型结构。开发新形式的玻尔兹曼机相比于开发新的神经网络层需要更多细心和创造力，因为它通常很难找到一个能保持玻尔兹曼机所需的所有不同条件分布的可解性的能量函数。尽管这需要努力，该领域仍对创新开放。

## 20.9 通过随机操作的反向传播

传统的神经网络对一些输入变量  $\mathbf{x}$  施加确定性变换。当开发生成模型时，我们经常希望扩展神经网络以实现  $\mathbf{x}$  的随机变换。这样做的一个直接方法是使用额外输入  $\mathbf{z}$ （从一些简单的概率分布采样得到，如均匀或高斯分布）来增强神经网络。神经网络在内部仍可以继续执行确定性计算，但是函数  $f(\mathbf{x}, \mathbf{z})$  对于不能访问  $\mathbf{z}$  的观察者来说将是随机的。假设  $f$  是连续可微的，我们可以像往常一样使用反向传播计算训练所需的梯度。

作为示例，让我们考虑从均值  $\mu$  和方差  $\sigma^2$  的高斯分布中采样  $y$  的操作：

$$y \sim \mathcal{N}(\mu, \sigma^2). \quad (20.54)$$

因为  $y$  的单个样本不是由函数产生的，而是由一个采样过程产生，它的输出会随我们的每次查询变化，所以取  $y$  相对于其分布的参数  $\mu$  和  $\sigma^2$  的导数似乎是违反直觉的。然而，我们可以将采样过程重写，对基本随机变量  $\mathbf{z} \sim \mathcal{N}(0, 1)$  进行转换以从期望的分布获得样本：

$$y = \mu + \sigma z. \quad (20.55)$$

现在我们将其视为具有额外输入  $\mathbf{z}$  的确定性操作，可以通过采样操作来反向传播。至关重要的是，额外输入是一个随机变量，其分布不是任何我们想对其计算导数的变量的函数。如果我们可以用相同的  $\mathbf{z}$  值再次重复采样操作，结果会告诉我们  $\mu$  或  $\sigma$  的微小变化将会如何改变输出。

能够通过该采样操作反向传播允许我们将其并入更大的图中。我们可以在采样分布的输出之上构建图元素。例如，我们可以计算一些损失函数  $J(y)$  的导数。我们还可以构建这样的图元素，其输出是采样操作的输入或参数。例如，我们可以通过  $\mu = f(\mathbf{x}; \boldsymbol{\theta})$  和  $\sigma = g(\mathbf{x}; \boldsymbol{\theta})$  构建更大的图。在这个增强图中，我们可以通过这些函数的反向传播导出  $\nabla_{\boldsymbol{\theta}} J(y)$ 。

在该高斯采样示例中使用的原理能更广泛地应用。我们可以将任何形为  $p(\mathbf{y}; \boldsymbol{\theta})$  或  $p(\mathbf{y} | \mathbf{x}; \boldsymbol{\theta})$  的概率分布表示为  $p(\mathbf{y} | \boldsymbol{\omega})$ ，其中  $\boldsymbol{\omega}$  是同时包含参数  $\boldsymbol{\theta}$  和输入  $\mathbf{x}$  的变量（如果适用的话）。给定从分布  $p(\mathbf{y} | \boldsymbol{\omega})$  采样的值  $\mathbf{y}$ （其中  $\boldsymbol{\omega}$  可以是其他变量的函数），我们可以将

$$\mathbf{y} \sim p(\mathbf{y} | \boldsymbol{\omega}) \quad (20.56)$$

重写为

$$\mathbf{y} = f(\mathbf{z}; \boldsymbol{\omega}), \quad (20.57)$$

其中  $\mathbf{z}$  是随机性的来源。只要  $f$  是几乎处处连续可微的，我们就可以使用传统工具（例如应用于  $f$  的反向传播算法）计算  $\mathbf{y}$  相对于  $\boldsymbol{\omega}$  的导数。至关重要的是， $\boldsymbol{\omega}$  不能是  $\mathbf{z}$  的函数，且  $\mathbf{z}$  不能是  $\boldsymbol{\omega}$  的函数。这种技术通常被称为**重参数化技巧** (reparametrization trick)、**随机反向传播**(stochastic back-propagation) 或**扰动分析** (perturbation analysis)。

要求  $f$  是连续可微的，当然需要  $\mathbf{y}$  是连续的。如果我们希望通过产生离散值样本的采样过程进行反向传播，则可以使用强化学习算法（如 REINFORCE 算法 (Williams, 1992) 的变体）来估计  $\boldsymbol{\omega}$  上的梯度，这将在第 20.9.1 节中讨论。

在神经网络应用中，我们通常选择从一些简单的分布中采样  $\mathbf{z}$ ，如单位均匀分布或单位高斯分布，并通过网络的确定性部分重塑其输入来实现更复杂的分布。

通过随机操作扩展梯度或优化的想法可追溯到二十世纪中叶 (Price, 1958; Bonnet, 1964)，并且首先在强化学习 (Williams, 1992) 的情景下用于机器学习。最近，它已被应用于变分近似 (Opper and Archambeau, 2009) 和随机生成神经网络 (Bengio *et al.*, 2013b; Kingma, 2013; Kingma and Welling, 2014b,a; Rezende *et al.*, 2014; Goodfellow *et al.*, 2014c)。许多网络，如去噪自编码器或使用 Dropout 的正则化网络，也被自然地设计为将噪声作为输入，而不需要任何特殊的重参数化就能使噪声独立于模型。

### 20.9.1 通过离散随机操作的反向传播

当模型发射离散变量  $\mathbf{y}$  时，重参数化技巧不再适用。假设模型采用输入  $\mathbf{x}$  和参数  $\boldsymbol{\theta}$ ，两者都封装在向量  $\boldsymbol{\omega}$  中，并且将它们与随机噪声  $\mathbf{z}$  组合以产生  $\mathbf{y}$ :

$$\mathbf{y} = f(\mathbf{z}; \boldsymbol{\omega}). \quad (20.58)$$

因为  $\mathbf{y}$  是离散的， $f$  必须是一个阶跃函数。阶跃函数的导数在任何点都是没用的。在每个阶跃边界，导数是未定义的，但这是一个小问题。大问题是导数在阶跃边界之间的区域几乎处处为零。因此，任何代价函数  $J(\mathbf{y})$  的导数无法给出如何更新模型参数  $\boldsymbol{\theta}$  的任何信息。

REINFORCE 算法 (REward Increment = nonnegative Factor  $\times$  Offset Reinforcement  $\times$  Characteristic Eligibility) 提供了定义一系列简单而强大解决方案的框架 (Williams, 1992)。其核心思想是, 即使  $J(f(\mathbf{z}; \boldsymbol{\omega}))$  是具有无用导数的阶跃函数, 期望代价  $\mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} J(f(\mathbf{z}; \boldsymbol{\omega}))$  通常是服从梯度下降的光滑函数。虽然当  $\mathbf{y}$  是高维 (或者是许多离散随机决策组合的结果) 时, 该期望通常是难解的, 但我们可以使用蒙特卡罗平均进行无偏估计。梯度的随机估计可以与 SGD 或其他基于随机梯度的优化技术一起使用。

通过简单地微分期望成本, 我们可以推导出 REINFORCE 最简单的版本:

$$\mathbb{E}_{\mathbf{z}}[J(\mathbf{y})] = \sum_{\mathbf{y}} J(\mathbf{y})p(\mathbf{y}), \quad (20.59)$$

$$\frac{\partial \mathbb{E}[J(\mathbf{y})]}{\partial \boldsymbol{\omega}} = \sum_{\mathbf{y}} J(\mathbf{y}) \frac{\partial p(\mathbf{y})}{\partial \boldsymbol{\omega}} \quad (20.60)$$

$$= \sum_{\mathbf{y}} J(\mathbf{y})p(\mathbf{y}) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \quad (20.61)$$

$$\approx \frac{1}{m} \sum_{\mathbf{y}^{(i)} \sim p(\mathbf{y}), i=1}^m J(\mathbf{y}^{(i)}) \frac{\partial \log p(\mathbf{y}^{(i)})}{\partial \boldsymbol{\omega}}. \quad (20.62)$$

式 (20.60) 依赖于  $J$  不直接引用  $\boldsymbol{\omega}$  的假设。放松这个假设来扩展该方法是简单的。式 (20.61) 利用对数的导数规则,  $\frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} = \frac{1}{p(\mathbf{y})} \frac{\partial p(\mathbf{y})}{\partial \boldsymbol{\omega}}$ 。式 (20.62) 给出了该梯度的无偏蒙特卡罗估计。

在本节中我们写的  $p(\mathbf{y})$ , 可以等价地写成  $p(\mathbf{y} | \mathbf{x})$ 。这是因为  $p(\mathbf{y})$  由  $\boldsymbol{\omega}$  参数化, 并且如果  $\mathbf{x}$  存在,  $\boldsymbol{\omega}$  包含  $\boldsymbol{\theta}$  和  $\mathbf{x}$  两者。

简单 REINFORCE 估计的一个问题是其具有非常高的方差, 需要采  $\mathbf{y}$  的许多样本才能获得对梯度的良好估计, 或者等价地, 如果仅绘制一个样本, SGD 将收敛得非常缓慢并将需要较小的学习率。通过使用 **方差减小** (variance reduction) 方法 (Wilson, 1984; L'Ecuyer, 1994), 可以减少该估计的方差。想法是修改估计量, 使其预期值保持不变, 但方差减小。在 REINFORCE 的情况下提出的方差减小方法, 涉及计算用于偏移  $J(\mathbf{y})$  的基线 (baseline)。注意, 不依赖于  $\mathbf{y}$  的任何偏移  $b(\mathbf{w})$

都不会改变估计梯度的期望，因为

$$E_{p(\mathbf{y})} \left[ \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \right] = \sum_{\mathbf{y}} p(\mathbf{y}) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \quad (20.63)$$

$$= \sum_{\mathbf{y}} \frac{\partial p(\mathbf{y})}{\partial \boldsymbol{\omega}} \quad (20.64)$$

$$= \frac{\partial}{\partial \boldsymbol{\omega}} \sum_{\mathbf{y}} p(\mathbf{y}) = \frac{\partial}{\partial \boldsymbol{\omega}} 1 = 0, \quad (20.65)$$

这意味着

$$E_{p(\mathbf{y})} \left[ (J(\mathbf{y}) - b(\boldsymbol{\omega})) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \right] = E_{p(\mathbf{y})} \left[ J(\mathbf{y}) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \right] - b(\boldsymbol{\omega}) E_{p(\mathbf{y})} \left[ \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \right] \quad (20.66)$$

$$= E_{p(\mathbf{y})} \left[ J(\mathbf{y}) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}} \right]. \quad (20.67)$$

此外，我们可以通过计算  $(J(\mathbf{y}) - b(\boldsymbol{\omega})) \frac{\partial \log p(\mathbf{y})}{\partial \boldsymbol{\omega}}$  关于  $p(\mathbf{y})$  的方差，并关于  $b(\boldsymbol{\omega})$  最小化获得最优  $b(\boldsymbol{\omega})$ 。我们发现这个最佳基线  $b^*(\boldsymbol{\omega})_i$  对于向量  $\boldsymbol{\omega}$  的每个元素  $\omega_i$  是不同的：

$$b^*(\boldsymbol{\omega})_i = \frac{E_{p(\mathbf{y})} \left[ J(\mathbf{y}) \frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i} \right]}{E_{p(\mathbf{y})} \left[ \frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i} \right]}. \quad (20.68)$$

相对于  $\omega_i$  的梯度估计则变为

$$(J(\mathbf{y}) - b(\boldsymbol{\omega})_i) \frac{\partial \log p(\mathbf{y})}{\partial \omega_i}, \quad (20.69)$$

其中  $b(\boldsymbol{\omega})_i$  估计上述  $b^*(\boldsymbol{\omega})_i$ 。获得估计  $b$  通常需要将额外输出添加到神经网络，并训练新输出对  $\boldsymbol{\omega}$  的每个元素估计  $E_{p(\mathbf{y})}[J(\mathbf{y}) \frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i}]$  和  $E_{p(\mathbf{y})}[\frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i}]$ 。这些额外的输出可以用均方误差目标训练，对于给定的  $\boldsymbol{\omega}$ ，从  $p(\mathbf{y})$  采样  $\mathbf{y}$  时，分别用  $J(\mathbf{y}) \frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i}$  和  $\frac{\partial \log p(\mathbf{y})^2}{\partial \omega_i}$  作目标。然后可以将这些估计代入式 (20.68) 就能恢复估计  $b$ 。Mnih and Gregor (2014) 倾向于使用通过目标  $J(\mathbf{y})$  训练的单个共享输出（跨越  $\boldsymbol{\omega}$  的所有元素  $i$ ），并使用  $b(\boldsymbol{\omega}) \approx E_{p(\mathbf{y})}[J(\mathbf{y})]$  作为基线。

在强化学习背景下引入的方差减小方法 (Sutton *et al.*, 2000; Weaver and Tao, 2001), Dayan (1990) 推广了二值奖励的前期工作。可以参考 Bengio *et al.* (2013b)、

Mnih and Gregor (2014)、Ba *et al.* (2014)、Mnih *et al.* (2014) 或 Xu *et al.* (2015) 中在深度学习的背景下使用减少方差的 REINFORCE 算法的现代例子。除了使用与输入相关的基线  $b(\omega)$ , Mnih and Gregor (2014) 发现可以在训练期间调整 ( $J(\mathbf{y}) - b(\omega)$ ) 的尺度 (即除以训练期间的移动平均估计的标准差), 即作为一种适应性学习率, 可以抵消训练过程中该量大小发生的重要变化的影响。Mnih and Gregor (2014) 称之为启发式方差归一化 (variance normalization)。

基于 REINFORCE 的估计器可以被理解为将  $\mathbf{y}$  的选择与  $J(\mathbf{y})$  的对应值相关联来估计梯度。如果在当前参数化下不太可能出现  $\mathbf{y}$  的良好值, 则可能需要很长时间来偶然获得它, 并且获得所需信号的配置应当被加强。

## 20.10 有向生成网络

如第十六章所讨论的, 有向图模型构成了一类突出的图模型。虽然有向图模型在更大的机器学习社群中非常流行, 但在较小的深度学习社群中, 大约直到 2013 年它们都掩盖在无向模型 (如 RBM) 的光彩之下。

在本节中, 我们回顾一些传统上与深度学习社群相关的标准有向图模型。

我们已经描述过部分有向的模型——深度信念网络。我们还描述过可以被认为是浅度有向生成模型的稀疏编码模型。尽管在样本生成和密度估计方面表现不佳, 在深度学习的背景下它们通常被用作特征学习器。我们接下来描述多种深度完全有向的模型。

### 20.10.1 sigmoid 信念网络

sigmoid 信念网络 (Neal, 1990) 是一种具有特定条件概率分布的有向图模型的简单形式。一般来说, 我们可以将 sigmoid 信念网络视为具有二值向量的状态  $\mathbf{s}$ , 其中状态的每个元素都受其祖先影响:

$$p(s_i) = \sigma\left(\sum_{j < i} W_{j,i}s_j + b_i\right). \quad (20.70)$$

sigmoid 信念网络最常见的结构是被分为许多层的结构, 其中原始采样通过一系列多个隐藏层进行, 然后最终生成可见层。这种结构与深度信念网络非常相似, 但它们在采样过程开始时的单元彼此独立, 而不是从受限玻尔兹曼机采样。这种结构

由于各种原因而令人感兴趣。一个原因是该结构是可见单元上概率分布的通用近似，即在足够深的情况下，可以任意良好地近似二值变量的任何概率分布（即使各个层的宽度受限于可见层的维度）(Sutskever and Hinton, 2008)。

虽然生成可见单元的样本在 sigmoid 信念网络中是非常高效的，但是其他大多数操作不是很高效。给定可见单元，对隐藏单元的推断是难解的。因为变分下界涉及对包含整个层的团求期望，均匀场推断也是难以处理的。这个问题一直困难到足以限制有向离散网络的普及。

在 sigmoid 信念网络中执行推断的一种方法是构造专用于 sigmoid 信念网络的不同下界 (Saul *et al.*, 1996)。这种方法只适用于非常小的网络。另一种方法是使用学成推断机制，如第 19.5 节中描述的。Helmholtz 机 (Dayan *et al.*, 1995; Dayan and Hinton, 1996) 结合了一个 sigmoid 信念网络与一个预测隐藏单元上均匀场分布参数的推断网络。sigmoid 信念网络的现代方法 (Gregor *et al.*, 2014; Mnih and Gregor, 2014) 仍然使用这种推断网络的方法。因为潜变量的离散本质，这些技术仍然是困难的。人们不能简单地通过推断网络的输出反向传播，而必须使用相对不可靠的机制即通过离散采样过程进行反向传播（如第 20.9.1 节所述）。最近基于重要采样、重加权的睡眠(Bornschein and Bengio, 2015) 或双向 Helmholtz 机 (Bornstein *et al.*, 2015) 的方法使得我们可以快速训练 sigmoid 信念网络，并在基准任务上达到最好的表现。

sigmoid 信念网络的一种特殊情况是没有潜变量的情况。在这种情况下学习是高效的，因为没有必要将潜变量边缘化到似然之外。一系列称为自回归网络的模型将这个完全可见的信念网络泛化到其他类型的变量（除二值变量）和其他结构（除对数线性关系）的条件分布。自回归网络将在第 20.10.7 节中描述。

### 20.10.2 可微生成器网络

许多生成模型基于使用可微生成器网络 (generator network) 的想法。这种模型使用可微函数  $g(\mathbf{z}; \theta^{(g)})$  将潜变量  $\mathbf{z}$  的样本变换为样本  $\mathbf{x}$  或样本  $\mathbf{x}$  上的分布，可微函数通常可以由神经网络表示。这类模型包括将生成器网络与推断网络配对的变分自编码器、将生成器网络与判别器网络配对的生成式对抗网络，以及孤立地训练生成器网络的技术。

生成器网络本质上仅是用于生成样本的参数化计算过程，其中的体系结构提供了从中采样的可能分布族以及选择这些族内分布的参数。

作为示例，从具有均值  $\mu$  和协方差  $\Sigma$  的正态分布绘制样本的标准过程是将来自零均值和单位协方差的正态分布的样本  $z$  馈送到非常简单的生成器网络中。这个生成器网络只包含一个仿射层：

$$\mathbf{x} = g(\mathbf{z}) = \boldsymbol{\mu} + \mathbf{L}\mathbf{z}, \quad (20.71)$$

其中  $\mathbf{L}$  由  $\Sigma$  的 Cholesky 分解给出。

伪随机数发生器也可以使用简单分布的非线性变换。例如，逆变换采样 (inverse transform sampling) (Devroye, 2013) 从  $U(0, 1)$  中采一个标量  $z$ ，并且对标量  $x$  应用非线性变换。在这种情况下， $g(z)$  由累积分布函数  $F(x) = \int_{-\infty}^x p(v)dv$  的反函数给出。如果我们能够指定  $p(x)$ ，在  $x$  上积分，并取所得函数的反函数，我们不用通过机器学习就能从  $p(x)$  进行采样。

为了从更复杂的分布（难以直接指定、难以积分或难以求所得积分的反函数）中生成样本，我们使用前馈网络来表示非线性函数  $g$  的参数族，并使用训练数据来推断参数以选择所期望的函数。

我们可以认为  $g$  提供了变量的非线性变化，将  $\mathbf{z}$  上的分布转换成  $\mathbf{x}$  上想要的分布。

回顾式 (3.47)，对于可求反函数的、可微的、连续的  $g$ ，

$$p_z(\mathbf{z}) = p_x(g(\mathbf{z})) \left| \det\left(\frac{\partial g}{\partial \mathbf{z}}\right) \right|. \quad (20.72)$$

这隐含地对  $\mathbf{x}$  施加概率分布：

$$p_x(\mathbf{x}) = \frac{p_z(g^{-1}(\mathbf{x}))}{\left| \det\left(\frac{\partial g}{\partial \mathbf{z}}\right) \right|}. \quad (20.73)$$

当然，取决于  $g$  的选择，这个公式可能难以评估，因此我们经常需要使用间接学习  $g$  的方法，而不是直接尝试最大化  $\log p(\mathbf{x})$ 。

在某些情况下，我们使用  $g$  来定义  $\mathbf{x}$  上的条件分布，而不是使用  $g$  直接提供  $\mathbf{x}$  的样本。例如，我们可以使用一个生成器网络，其最后一层由 sigmoid 输出组成，可以提供 Bernoulli 分布的平均参数：

$$p(\mathbf{x}_i = 1 | \mathbf{z}) = g(\mathbf{z})_i. \quad (20.74)$$

在这种情况下，我们使用  $g$  来定义  $p(\mathbf{x} | \mathbf{z})$  时，我们通过边缘化  $\mathbf{z}$  来对  $\mathbf{x}$  施加分布：

$$p(\mathbf{x}) = \mathbb{E}_{\mathbf{z}} p(\mathbf{x} | \mathbf{z}). \quad (20.75)$$

两种方法都定义了一个分布  $p_g(\mathbf{x})$ ，并允许我们使用第 20.9 节中的重参数化技巧来训练  $p_g$  的各种评估准则。

表示生成器网络的两种不同方法（发出条件分布的参数相对直接发射样品）具有互补的优缺点。当生成器网络在  $\mathbf{x}$  上定义条件分布时，它不但能生成连续数据，也能生成离散数据。当生成器网络直接提供采样时，它只能产生连续的数据（我们可以在前向传播中引入离散化，但这样做意味着模型不再能够使用反向传播进行训练）。直接采样的优点是，我们不再被迫使用条件分布（可以容易地写出来并由人类设计者进行代数操作的形式）。

基于可微生成器网络的方法是由分类可微前馈网络中梯度下降的成功应用而推动的。在监督学习的背景下，基于梯度训练学习的深度前馈网络在给定足够的隐藏单元和足够的训练数据的情况下，在实践中似乎能保证成功。这个同样的方案能成功转移到生成式建模上吗？

生成式建模似乎比分类或回归更困难，因为学习过程需要优化难以处理的准则。在可微生成器网络的情况下，准则是难以处理的，因为数据不指定生成器网络的输入  $\mathbf{z}$  和输出  $\mathbf{x}$ 。在监督学习的情况下，输入  $\mathbf{x}$  和输出  $\mathbf{y}$  同时给出，并且优化过程只需学习如何产生指定的映射。在生成建模的情况下，学习过程需要确定如何以有用的方式排布  $\mathbf{z}$  空间，以及额外的如何从  $\mathbf{z}$  映射到  $\mathbf{x}$ 。

Dosovitskiy *et al.* (2015) 研究了一个简化问题，其中  $\mathbf{z}$  和  $\mathbf{x}$  之间的对应关系已经给出。具体来说，训练数据是计算机渲染的椅子图。潜变量  $\mathbf{z}$  是渲染引擎的参数，描述了椅子模型的选择、椅子的位置以及影响图像渲染的其他配置细节。使用这种合成的生成数据，卷积网络能够学习将图像内容的描述  $\mathbf{z}$  映射到渲染图像的近似  $\mathbf{x}$ 。这表明当现代可微生成器网络具有足够的模型容量时，足以成为良好的生成模型，并且现代优化算法具有拟合它们的能力。困难在于当每个  $\mathbf{x}$  的  $\mathbf{z}$  的值不是固定的且在每次训练前是未知时，如何训练生成器网络。

在接下来的章节中，我们讨论仅给出  $\mathbf{x}$  的训练样本，训练可微生成器网络的几种方法。

### 20.10.3 变分自编码器

变分自编码器 ( variational auto-encoder, VAE ) (Kingma, 2013; Rezende *et al.*, 2014) 是一个使用学好的近似推断的有向模型，可以纯粹地使用基于梯度的方法进行

训练。

为了从模型生成样本，VAE 首先从编码分布  $p_{\text{model}}(\mathbf{z})$  中采样  $\mathbf{z}$ 。然后使样本通过可微生成器网络  $g(\mathbf{z})$ 。最后，从分布  $p_{\text{model}}(\mathbf{x}; g(\mathbf{z})) = p_{\text{model}}(\mathbf{x} | \mathbf{z})$  中采样  $\mathbf{x}$ 。然而在训练期间，近似推断网络（或编码器） $q(\mathbf{z} | \mathbf{x})$  用于获得  $\mathbf{z}$ ，而  $p_{\text{model}}(\mathbf{x} | \mathbf{z})$  则被视为解码器网络。

变分自编码器背后的关键思想是，它们可以通过最大化与数据点  $\mathbf{x}$  相关联的变分下界  $\mathcal{L}(q)$  来训练：

$$\mathcal{L}(q) = \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} | \mathbf{x})} \log p_{\text{model}}(\mathbf{z}, \mathbf{x}) + \mathcal{H}(q(\mathbf{z} | \mathbf{x})) \quad (20.76)$$

$$= \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} | \mathbf{x})} \log p_{\text{model}}(\mathbf{x} | \mathbf{z}) - D_{\text{KL}}(q(\mathbf{z} | \mathbf{x}) || p_{\text{model}}(\mathbf{z})) \quad (20.77)$$

$$\leq \log p_{\text{model}}(\mathbf{x}). \quad (20.78)$$

在式 (20.76) 中，我们将第一项视为潜变量的近似后验下可见和隐藏变量的联合对数似然性（正如 EM 一样，不同的是我们使用近似而不是精确后验）。第二项则可视为近似后验的熵。当  $q$  被选择为高斯分布，其中噪声被添加到预测平均值时，最大化该熵项促使该噪声标准偏差的增加。更一般地，这个熵项鼓励变分后验将高概率质量置于可能已经产生  $\mathbf{x}$  的许多  $\mathbf{z}$  值上，而不是坍缩到单个估计最可能值的点。在式 (20.77) 中，我们将第一项视为在其他自编码器中出现的重构对数似然。第二项试图使近似后验分布  $q(\mathbf{z} | \mathbf{x})$  和模型先验  $p_{\text{model}}(\mathbf{z})$  彼此接近。

变分推断和学习的传统方法是通过优化算法推断  $q$ ，通常是迭代不动点方程（第 19.4 节）。这些方法是缓慢的，并且通常需要以闭解形式计算  $\mathbb{E}_{\mathbf{z} \sim q} \log p_{\text{model}}(\mathbf{z}, \mathbf{x})$ 。变分自编码器背后的主要思想是训练产生  $q$  参数的参数编码器（有时也称为推断网络或识别模型）。只要  $\mathbf{z}$  是连续变量，我们就可以通过从  $q(\mathbf{z} | \mathbf{x}) = q(\mathbf{z}; f(\mathbf{x}; \boldsymbol{\theta}))$  中采样  $\mathbf{z}$  的样本反向传播，以获得相对于  $\boldsymbol{\theta}$  的梯度。学习则仅包括相对于编码器和解码器的参数最大化  $\mathcal{L}$ 。 $\mathcal{L}$  中的所有期望都可以通过蒙特卡罗采样来近似。

变分自编码器方法是优雅的，理论上令人愉快的，并且易于实现。它也获得了出色的结果，是生成式建模中的最先进方法之一。它的主要缺点是从在图像上训练的变分自编码器中采样的样本往往有些模糊。这种现象的原因尚不清楚。一种可能性是模糊性是最大似然的固有效应，因为我们需要最小化  $D_{\text{KL}}(p_{\text{data}} || p_{\text{model}})$ 。如图 3.6 所示，这意味着模型将为训练集中出现的点分配高的概率，但也可能为其他点分配高的概率。还有其他原因可以导致模糊图像。模型选择将概率质量置于模糊图像而不是空间的其他部分的部分原因是实际使用的变分自编码器通常在  $p_{\text{model}}(\mathbf{x}; g(\mathbf{z}))$  使用高斯分布。最大化这种分布似然性的下界与训练具有均方误差的传统自编码器类似，

这意味着它倾向于忽略由少量像素表示的特征或其中亮度变化微小的像素。如 Theis *et al.* (2015) 和 Huszar (2015) 指出的，该问题不是 VAE 特有的，而是与优化对数似然或  $D_{\text{KL}}(p_{\text{data}} \parallel p_{\text{model}})$  的生成模型共享的。现代 VAE 模型另一个麻烦的问题是，它们倾向于仅使用  $z$  维度中的小子集，就像编码器不能够将具有足够局部方向的输入空间变换到边缘分布与分解前匹配的空间。

VAE 框架可以直接扩展到大范围的模型架构。相比玻尔兹曼机，这是关键的优势，因为玻尔兹曼机需要非常仔细地设计模型来保持易解性。VAE 可以与广泛的可微算子族一起良好工作。一个特别复杂的 VAE 是深度循环注意写者 (DRAW) 模型 (Gregor *et al.*, 2015)。DRAW 使用一个循环编码器和循环解码器并结合注意力机制。DRAW 模型的生成过程包括顺序访问不同的小图像块并绘制这些点处的像素值。我们还可以通过在 VAE 框架内使用循环编码器和解码器来定义变分 RNN (Chung *et al.*, 2015b) 来扩展 VAE 以生成序列。从传统 RNN 生成样本仅在输出空间涉及非确定性操作。而变分 RNN 还具有由 VAE 潜变量捕获的潜在更抽象层的随机变化性。

VAE 框架已不仅仅扩展到传统的变分下界，还有 **重要加权自编码器**(importance-weighted autoencoder)(Burda *et al.*, 2015) 的目标：

$$\mathcal{L}_k(\mathbf{x}, q) = \mathbb{E}_{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(k)} \sim q(\mathbf{z}|\mathbf{x})} \left[ \log \frac{1}{k} \sum_{i=1}^k \frac{p_{\text{model}}(\mathbf{x}, \mathbf{z}^{(i)})}{q(\mathbf{z}^{(i)} | \mathbf{x})} \right]. \quad (20.79)$$

这个新的目标在  $k = 1$  时等同于传统的下界  $\mathcal{L}$ 。然而，它也可以被解释为基于提议分布  $q(\mathbf{z} | \mathbf{x})$  中  $z$  的重要采样而形成的真实  $\log p_{\text{model}}(\mathbf{x})$  估计。重要加权自编码器目标也是  $\log p_{\text{model}}(\mathbf{x})$  的下界，并且随着  $k$  增加而变得更紧。

变分自编码器与 MP-DBM 和其他涉及通过近似推断图的反向传播方法有一些有趣的联系 (Goodfellow *et al.*, 2013d; Stoyanov *et al.*, 2011; Brakel *et al.*, 2013)。这些以前的方法需要诸如均匀场不动点方程的推断过程来提供计算图。变分自编码器被定义为任意计算图，这使得它能适用于更广泛的概率模型族，因为它不需要将模型的选择限制到具有易处理的均匀场不动点方程的那些模型。变分自编码器还具有增加模型对数似然边界的优点，而 MP-DBM 和相关模型的准则更具启发性，并且除了使近似推断的结果准确外很少有概率的解释。变分自编码器的一个缺点是它仅针对一个问题学习推断网络，即给定  $\mathbf{x}$  推断  $\mathbf{z}$ 。较老的方法能够在给定任何其他变量子集的情况下对任何变量子集执行近似推断，因为均匀场不动点方程指定如何在所有这些不同问题的计算图之间共享参数。

变分自编码器的一个非常好的特性是，同时训练参数编码器与生成器网络的组合迫使模型学习一个编码器可以捕获的可预测的坐标系。这使得它成为一个优秀的流形学习算法。图 20.6 展示了由变分自编码器学到的低维流形的例子。图中所示的情况之一，算法发现了存在于面部图像中两个独立的变化因素：旋转角和情绪表达。



图 20.6: 由变分自编码器学习的高维流形在 2 维坐标系中的示例 (Kingma and Welling, 2014a)。我们可以在纸上直接绘制两个可视化的维度，因此可以使用 2 维潜在编码训练模型来了解模型的工作原理（即使我们认为数据流形的固有维度要高得多）。图中所示的图像不是来自训练集的样本，而是仅仅通过改变 2 维“编码” $z$ ，由模型  $p(x|z)$  实际生成的图像  $x$ （每个图像对应于“编码” $z$  位于 2 维均匀网格的不同选择）。（左）Frey 人脸流形的 2 维映射。其中一个维度（水平）已发现大致对应于面部的旋转，而另一个（垂直）对应于情绪表达。（右）MNIST 流形的 2 维映射。

#### 20.10.4 生成式对抗网络

**生成式对抗网络** (generative adversarial network, GAN) (Goodfellow *et al.*, 2014c) 是基于可微生成器网络的另一种生成式建模方法。

生成式对抗网络基于博奕论场景，其中生成器网络必须与对手竞争。生成器网络直接产生样本  $x = g(z; \theta^{(g)})$ 。其对手，**判别器网络** (discriminator network)，试图区分从训练数据抽取的样本和从生成器抽取的样本。判别器发出由  $d(x; \theta^{(d)})$  给出的概率值，指示  $x$  是真实训练样本而不是从模型抽取的伪造样本的概率。

形式化表示生成式对抗网络中学习的最简单方式是零和游戏，其中函数  $v(\theta^{(g)}, \theta^{(d)})$  确定判别器的收益。生成器接收  $-v(\theta^{(g)}, \theta^{(d)})$  作为它自己的收益。在学习期间，每个玩家尝试最大化自己的收益，因此收敛在

$$g^* = \arg \min_g \max_d v(g, d). \quad (20.80)$$

$v$  的默认选择是

$$v(\theta^{(g)}, \theta^{(d)}) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \log d(\mathbf{x}) + \mathbb{E}_{\mathbf{x} \sim p_{\text{model}}} \log(1 - d(\mathbf{x})). \quad (20.81)$$

这驱使判别器试图学习将样品正确地分类为真的或伪造的。同时，生成器试图欺骗分类器以让其相信样本是真实的。在收敛时，生成器的样本与实际数据不可区分，并且判别器处处都输出  $\frac{1}{2}$ 。然后就可以丢弃判别器。

设计 GAN 的主要动机是学习过程既不需要近似推断也不需要配分函数梯度的近似。当  $\max_d v(g, d)$  在  $\theta^{(g)}$  中是凸的（例如，在概率密度函数的空间中直接执行优化的情况）时，该过程保证收敛并且是渐近一致的。

不幸的是，在实践中由神经网络表示的  $g$  和  $d$  以及  $\max_d v(g, d)$  不凸时，GAN 中的学习可能是困难的。Goodfellow (2014) 认为不收敛可能会引起 GAN 的欠拟合问题。一般来说，同时对两个玩家的成本梯度下降不能保证达到平衡。例如，考虑价值函数  $v(a, b) = ab$ ，其中一个玩家控制  $a$  并产生成本  $ab$ ，而另一玩家控制  $b$  并接收成本  $-ab$ 。如果我们将每个玩家建模为无穷小的梯度步骤，每个玩家以另一个玩家为代价降低自己的成本，则  $a$  和  $b$  进入稳定的圆形轨迹，而不是到达原点处的平衡点。注意，极小极大化游戏的平衡不是  $v$  的局部最小值。相反，它们是同时最小化的两个玩家成本的点。这意味着它们是  $v$  的鞍点，相对于第一个玩家的参数是局部最小值，而相对于第二个玩家的参数是局部最大值。两个玩家可以永远轮流增加然后减少  $v$ ，而不是正好停在玩家没有能力降低其成本的鞍点。目前不知道这种不收敛的问题会在多大程度上影响 GAN。

Goodfellow (2014) 确定了另一种替代的形式化收益公式，其中博弈不再是零和，每当判别器最优时，具有与最大似然学习相同的预期梯度。因为最大似然训练收敛，这种 GAN 博弈的重述在给定足够的样本时也应该收敛。不幸的是，这种替代的形式化似乎并没有提高实践中的收敛，可能是由于判别器的次优性或围绕期望梯度的高方差。

在真实实验中，GAN 博弈的最佳表现形式既不是零和也不等价于最大似然，而是 Goodfellow et al. (2014c) 引入的带有启发式动机的不同形式化。在这种最佳性能

的形式中，生成器旨在增加判别器发生错误的对数概率，而不是旨在降低判别器进行正确预测的对数概率。这种重述仅仅是观察的结果，即使在判别器确信地拒绝所有生成器样本的情况下，它也能导致生成器代价函数的导数相对于判别器的对数保持很大。

稳定 GAN 学习仍然是一个开放的问题。幸运的是，当仔细选择模型架构和超参数时，GAN 学习效果很好。Radford *et al.* (2015) 设计了一个深度卷积 GAN (DCGAN)，在图像合成的任务上表现非常好，并表明其潜在的表示空间能捕获到变化的重要因素，如图 20.7 所示。图 20.7 展示了 DCGAN 生成器生成的图像示例。



图 20.7：在 LSUN 数据集上训练后，由 GAN 生成的图像。(左) 由 DCGAN 模型生成的卧室图像，经 Radford *et al.* (2015) 许可转载。(右) 由 LAPGAN 模型生成的教堂图像，经 Denton *et al.* (2015) 许可转载。

GAN 学习问题也可以通过将生成过程分成许多级别的细节来简化。我们可以训练有条件的 GAN (Mirza and Osindero, 2014)，并学习从分布  $p(\mathbf{x} | \mathbf{y})$  中采样，而不是简单地从边缘分布  $p(\mathbf{x})$  中采样。Denton *et al.* (2015) 表明一系列的条件 GAN 可以被训练为首先生成非常低分辨率的图像，然后增量地向图像添加细节。由于使用拉普拉斯金字塔来生成包含不同细节水平的图像，这种技术被称为 LAPGAN 模型。LAPGAN 生成器不仅能够欺骗判别器网络，而且能够欺骗人类观察者，实验主体将高达 40% 的网络输出识别为真实数据。请看图 20.7 中 LAPGAN 生成器生成的图像示例。

GAN 训练过程中一个不寻常的能力是它可以拟合向训练点分配零概率的概率分布。生成器网络学习跟踪其点在某种程度上类似于训练点的流形，而不是最大化特定点的对数概率。有点矛盾的是，这意味着模型可以将负无穷大的对数似然分配

给测试集，同时仍然表示人类观察者判断为能捕获生成任务本质的流形。这不是明显的优点或缺点，并且只要向生成器网络最后一层所有生成的值添加高斯噪声，就可以保证生成器网络向所有点分配非零概率。以这种方式添加高斯噪声的生成器网络从相同分布的采样，即使用生成器网络参数化条件高斯分布的均值所获得的分布。

Dropout 似乎在判别器网络中很重要。特别地，在计算生成器网络的梯度时，单元应当被随机地丢弃。使用权重除以二的确定性版本的判别器的梯度似乎不是那么有效。同样，从不使用 Dropout 似乎会产生不良的结果。

虽然 GAN 框架被设计为用于可微生成器网络，但是类似的原理可以用于训练其他类型的模型。例如，**自监督提升** (self-supervised boosting) 可以用于训练 RBM 生成器以欺骗逻辑回归判别器 (Welling *et al.*, 2002)。

### 20.10.5 生成矩匹配网络

**生成矩匹配网络** (generative moment matching network) (Li *et al.*, 2015; Dziugaite *et al.*, 2015) 是另一种基于可微生成器网络的生成模型。与 VAE 和 GAN 不同，它们不需要将生成器网络与任何其他网络配对，如不需要与用于 VAE 的推断网络配对，也不需要与 GAN 的判别器网络。

生成矩匹配网络使用称为 **矩匹配** (moment matching) 的技术训练。矩匹配背后的基本思想是以如下的方式训练生成器——令模型生成的样本的许多统计量尽可能与训练集中的样本相似。在此情景下，**矩** (moment) 是对随机变量不同幂的期望。例如，第一矩是均值，第二矩是平方值的均值，以此类推。多维情况下，随机向量的每个元素可以被升高到不同的幂，因此使得矩可以是任意数量的形式

$$\mathbb{E}_x \prod_i x_i^{n_i}, \quad (20.82)$$

其中  $\mathbf{n} = [n_1, n_2, \dots, n_d]^\top$  是一个非负整数的向量。

在第一次检查时，这种方法似乎在计算上是不可行的。例如，如果我们想匹配形式为  $x_i x_j$  的所有矩，那么我们需要最小化在  $\mathbf{x}$  的维度上是二次的多个值之间的差。此外，甚至匹配所有第一和第二矩将仅足以拟合多变量高斯分布，其仅捕获值之间的线性关系。我们使用神经网络的野心是捕获复杂的非线性关系，这将需要更多的矩。GAN 通过使用动态更新的判别器避免了穷举所有矩的问题，该判别器自动将其注意力集中在生成器网络最不匹配的统计量上。

相反，我们可以通过最小化一个被称为最大平均偏差（maximum mean discrepancy, MMD）(Schölkopf and Smola, 2002; Gretton *et al.*, 2012) 的代价函数来训练生成矩匹配网络。该代价函数通过向核函数定义的特征空间隐式映射，在无限维空间中测量第一矩的误差，使得对无限维向量的计算变得可行。当且仅当所比较的两个分布相等时，MMD 代价为零。

从可视化方面看，来自生成矩匹配网络的样本有点令人失望。幸运的是，它们可以通过将生成器网络与自编码器组合来改进。首先，训练自编码器以重构训练集。接下来，自编码器的编码器用于将整个训练集转换到编码空间。然后训练生成器网络以生成编码样本，这些编码样本可以经解码器映射到视觉上令人满意的样本。

与 GAN 不同，代价函数仅关于一批同时来自训练集和生成器网络的实例定义。我们不可能将训练更新作为一个训练样本或仅来自生成器网络的一个样本的函数。这是因为必须将矩计算为许多样本的经验平均值。当批量大小太小时，MMD 可能低估采样分布的真实变化量。有限的批量大小都不足以大到完全消除这个问题，但是更大的批量大小减少了低估的量。当批量大小太大时，训练过程就会慢得不可行，因为计算单个小梯度步长必须一下子处理许多样本。

与 GAN 一样，即使生成器网络为训练点分配零概率，仍可以使用 MMD 训练生成器网络。

### 20.10.6 卷积生成网络

当生成图像时，将卷积结构的引入生成器网络通常是有用的（见 Goodfellow *et al.* (2014c) 或 Dosovitskiy *et al.* (2015) 的例子）。为此，我们使用卷积算子的“转置”，如第 9.5 节所述。这种方法通常能产生更逼真的图像，并且比不使用参数共享的全连接层使用更少的参数。

用于识别任务的卷积网络具有从图像到网络顶部的某些概括层（通常是类标签）的信息流。当该图像通过网络向上流动时，随着图像的表示变得对于有害变换保持不变，信息也被丢弃。在生成器网络中，情况恰恰相反。要生成图像的表示通过网络传播时必须添加丰富的详细信息，最后产生图像的最终表示，这个最终表示当然是带有所有细节的精细图像本身（具有对象位置、姿势、纹理以及明暗）。在卷积识别网络中丢弃信息的主要机制是池化层。而生成器网络似乎需要添加信息。由于大多数池化函数不可逆，我们不能将池化层求逆后放入生成器网络。更简单的操作是仅仅增加表示的空间大小。似乎可接受的方法是使用 Dosovitskiy *et al.* (2015) 引入的

“去池化”。该层对应于某些简化条件下最大池化的逆操作。首先，最大池化操作的步幅被约束为等于池化区域的宽度。其次，每个池化区域内的最大输入被假定为左上角的输入。最后，假设每个池化区域内所有非最大的输入为零。这些是非常强和不现实的假设，但它们允许我们对最大池化算子求逆。逆去池化的操作分配一个零张量，然后将每个值从输入的空间坐标  $i$  复制到输出的空间坐标  $i \times k$ 。整数值  $k$  定义池化区域的大小。即使驱动去池化算子定义的假设是不现实的，后续层也能够学习补偿其不寻常的输出，所以由整体模型生成的样本在视觉上令人满意。

### 20.10.7 自回归网络

自回归网络是没有潜在随机变量的有向概率模型。这些模型中的条件概率分布由神经网络表示（有时是极简单的神经网络，例如逻辑回归）。这些模型的图结构是完全图。它们可以通过概率的链式法则分解观察变量上的联合概率，从而获得形如  $P(x_d | x_{d-1}, \dots, x_1)$  条件概率的乘积。这样的模型被称为 **完全可见的贝叶斯网络**（fully-visible Bayes networks, FVBN），并成功地以许多形式使用，首先是对每个条件分布逻辑回归 (Frey, 1998)，然后是带有隐藏单元的神经网络 (Bengio and Bengio, 2000b; Larochelle and Murray, 2011)。在某些形式的自回归网络中，例如在第 20.10.10 节中描述的 NADE (Larochelle and Murray, 2011)，我们可以引入参数共享的一种形式，它能带来统计优点（较少的唯一参数）和计算优势（较少计算量）。这是深度学习中反复出现的主题——特征重用的另一个实例。

### 20.10.8 线性自回归网络

自回归网络的最简单形式是没有隐藏单元、没有参数或特征共享的形式。每个  $P(x_i | x_{i-1}, \dots, x_1)$  被参数化为线性模型（对于实值数据的线性回归，对于二值数据的逻辑回归，对于离散数据的softmax回归）。这个模型由 Frey (1998) 引入，当有  $d$  个变量要建模时，该模型有  $\mathcal{O}(d^2)$  个参数。如图 20.8 所示。

如果变量是连续的，线性自回归网络只是表示多元高斯分布的另一种方式，只能捕获观察变量之间线性的成对相互作用。

线性自回归网络本质上是线性分类方法在生成式建模上的推广。因此，它们具有与线性分类器相同的优缺点。像线性分类器一样，它们可以用凸损失函数训练，并且有时允许闭解形式（如在高斯情况下）。像线性分类器一样，模型本身不提供增加

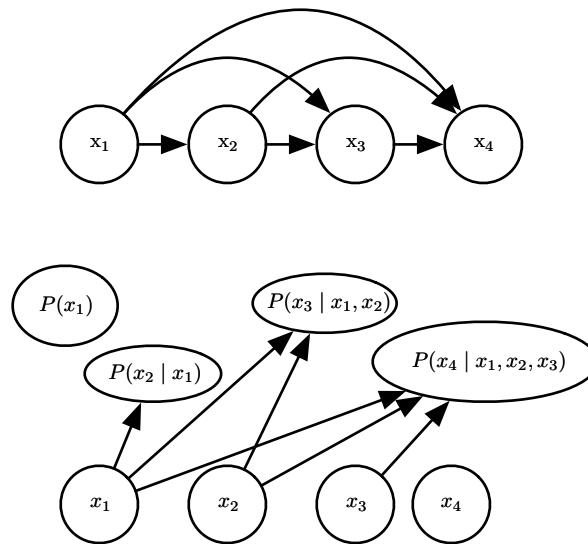


图 20.8: 完全可见的信念网络从前  $i - 1$  个变量预测第  $i$  个变量。(上) FVBN 的有向图模型。(下) 对数 FVBN 相应的计算图, 其中每个预测由线性预测器作出。

其容量的方法, 因此必须使用其他技术 (如输入的基扩展或核技巧) 来提高容量。

### 20.10.9 神经自回归网络

神经自回归网络 (Bengio and Bengio, 2000a,b) 具有与逻辑自回归网络相同的从左到右的图模型 (图 20.8), 但在该图模型结构内采用不同的条件分布参数。新的参数化更强大, 它可以根据需要随意增加容量, 并允许近似任意联合分布。新的参数化还可以引入深度学习中常见的参数共享和特征共享原理来改进泛化能力。设计这些模型的动机是避免传统表格图模型引起的维数灾难, 并与图 20.8 共享相同的结构。在表格离散概率模型中, 每个条件分布由概率表表示, 其中所涉及的变量的每个可能配置都具有一个条目和一个参数。通过使用神经网络, 可以获得两个优点:

1. 通过具有  $(i - 1) \times k$  个输入和  $k$  个输出的神经网络 (如果变量是离散的并有  $k$  个值, 使用one-hot编码) 参数化每个  $P(x_i | x_{i-1}, \dots, x_1)$ , 让我们不需要指数量级参数 (和样本) 的情况下就能估计条件概率, 然而仍然能够捕获随机变量之间的高阶依赖性。
2. 不需要对预测每个  $x_i$  使用不同的神经网络, 如图 20.9 所示的从左到右连接, 允

许将所有神经网络合并成一个。等价地，它意味着为预测  $x_i$  所计算的隐藏层特征可以重新用于预测  $x_{i+k}$  ( $k > 0$ )。因此隐藏单元被组织成第  $i$  组中的所有单元仅依赖于输入值  $x_1, \dots, x_i$  的特定的组。用于计算这些隐藏单元的参数被联合优化以改进对序列中所有变量的预测。这是重用原理的一个实例，这是从循环和卷积网络架构到多任务和迁移学习的场景中反复出现的深度学习原理。

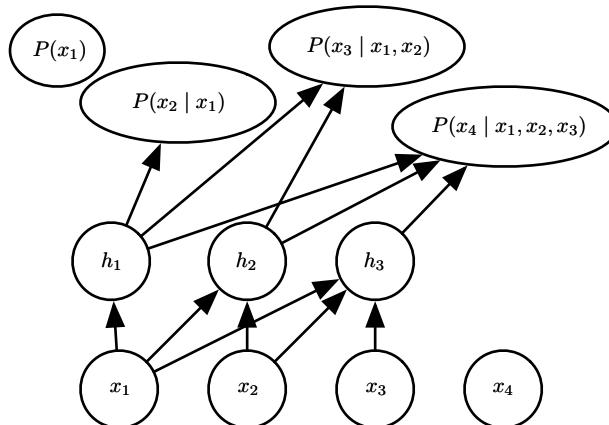


图 20.9: 神经自回归网络从前  $i - 1$  个变量预测第  $i$  个变量  $x_i$ ，但经参数化后，作为  $x_1, \dots, x_i$  函数的特征（表示为  $h_i$  的隐藏单元的组）可以在预测所有后续变量  $x_{i+1}, x_{i+2}, \dots, x_d$  时重用。

如在第 6.2.2.1 节中讨论的，使神经网络的输出预测  $x_i$  条件分布的参数，每个  $P(x_i | x_{i-1}, \dots, x_1)$  就可以表示一个条件分布。虽然原始神经自回归网络最初是在纯粹离散多变量数据（带有 sigmoid 输出的 Bernoulli 变量或 softmax 输出的 Multinoulli 变量）的背景下评估，但我们可以自然地将这样的模型扩展到连续变量或同时涉及离散和连续变量的联合分布。

## 20.10.10 NADE

**神经自回归密度估计器** (neural auto-regressive density estimator, NADE) 是最近非常成功的神经自回归网络的一种形式 (Larochelle and Murray, 2011)。与 Bengio and Bengio (2000b) 的原始神经自回归网络中的连接相同，但 NADE 引入了附加的参数共享方案，如图 20.10 所示。不同组  $j$  的隐藏单元的参数是共享的。

从第  $i$  个输入  $x_i$  到第  $j$  组隐藏单元的第  $k$  个元素  $h_k^{(j)}$  ( $j \geq i$ ) 的权重  $W'_{j,k,i}$  是

组内共享的：

$$W'_{j,k,i} = W_{k,i}. \quad (20.83)$$

其余  $j < i$  的权重为零。

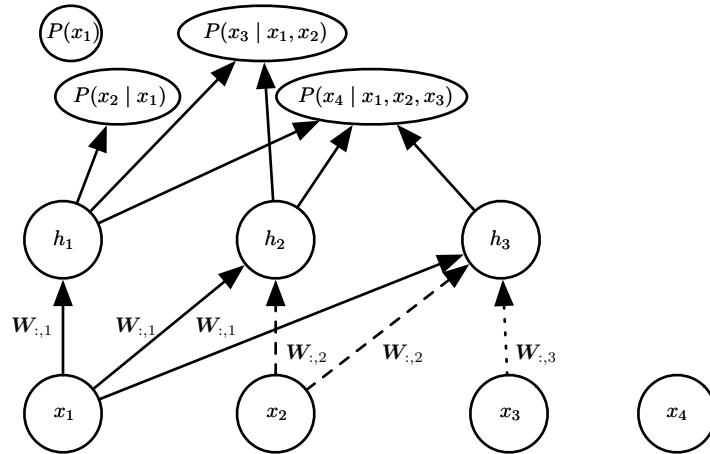


图 20.10：神经自回归密度估计器（NADE）的示意图。隐藏单元被组织在组  $\mathbf{h}^{(j)}$  中，使得只有输入  $x_1, \dots, x_i$  参与计算  $\mathbf{h}^{(i)}$  和预测  $P(x_j | x_{j-1}, \dots, x_1)$ （对于  $j > i$ ）。NADE 使用特定的权重共享模式区别于早期的神经自回归网络： $W'_{j,k,i} = W_{k,i}$  被共享于所有从  $x_i$  到任何  $j \geq i$  组中第  $k$  个单元的权重（在图中使用相同的线型表示复制权重的每个实例）。注意向量  $(W_{1,i}, W_{2,i}, \dots, W_{n,i})$  记为  $\mathbf{W}_{:,i}$ 。

Larochelle and Murray (2011) 选择了这种共享方案，使得 NADE 模型中的正向传播与在均匀场推断中执行的计算大致相似，以填充 RBM 中缺失的输入。这个均匀场推断对应于运行具有共享权重的循环网络，并且该推断的第一步与 NADE 中的相同。使用 NADE 的唯一区别是，连接隐藏单元到输出的输出权重独立于连接输入单元和隐藏单元的权重进行参数化。在 RBM 中，隐藏到输出的权重是输入到隐藏权重的转置。NADE 架构可以扩展为不仅仅模拟均匀场循环推断的一个时间步，而是  $k$  步。这种方法称为 NADE- $k$  (Raiko *et al.*, 2014)。

如前所述，自回归网络可以被扩展成处理连续数据。用于参数化连续密度的特别强大和通用的方法是混合权重为  $\alpha_i$ （组  $i$  的系数或先验概率），每组条件均值为  $\mu_i$  和每组条件方差为  $\sigma_i^2$  的高斯混合体。一个称为 RNADE 的模型 (Uria *et al.*, 2013) 使用这种参数化将 NADE 扩展到实值。与其他混合密度网络一样，该分布的参数是网络的输出，由 softmax 单元产生混合的权量概率以及参数化的方差，因此可使它

们为正的。由于条件均值  $\mu_i$  和条件方差  $\sigma_i^2$  之间的相互作用，随机梯度下降在数值上可能会表现不好。为了减少这种困难，Uria *et al.* (2013) 在后向传播阶段使用伪梯度代替平均值上的梯度。

另一个非常有趣的神经自回归架构的扩展摆脱了为观察到的变量选择任意顺序的需要 (Murray and Larochelle, 2014)。在自回归网络中，该想法是训练网络以能够通过随机采样顺序来处理任何顺序，并将信息提供给指定哪些输入被观察的隐藏单元（在条件条的右侧），以及哪些是被预测并因此被认为是缺失的（在条件条的左侧）。这是不错的性质，因为它允许人们非常高效地使用训练好的自回归网络来执行任何推断问题（即从给定任何变量的子集，从任何子集上的概率分布预测或采样）。最后，由于变量的许多顺序是可能的（对于  $n$  个变量是  $n!$ ），并且变量的每个顺序  $o$  产生不同的  $p(\mathbf{x} | o)$ ，我们可以组成许多  $o$  值模型的集成：

$$p_{\text{ensemble}}(\mathbf{x}) = \frac{1}{k} \sum_{i=1}^k p(\mathbf{x} | o^{(i)}). \quad (20.84)$$

这个集成模型通常能更好地泛化，并且为测试集分配比单个排序定义的单个模型更高的概率。

在同一篇文章中，作者提出了深度版本的架构，但不幸的是，这立即使计算成本像原始神经自回归网络一样高 (Bengio and Bengio, 2000b)。第一层和输出层仍然可以在  $\mathcal{O}(nh)$  的乘法-加法操作中计算，如在常规 NADE 中，其中  $h$  是隐藏单元的数量（图 20.10 和图 20.9 中的组  $h_i$  的大小），而它在 Bengio and Bengio (2000b) 中是  $\mathcal{O}(n^2h)$ 。然而，对于其他隐藏层的计算量是  $\mathcal{O}(n^2h^2)$ （假设在每个层存在  $n$  组  $h$  个隐藏单元，且在  $l$  层的每个“先前”组参与预测  $l+1$  层处的“下一个”组）。如在 Murray and Larochelle (2014) 中，使  $l+1$  层上的第  $i$  个组仅取决于第  $i$  个组， $l$  层处的计算量将减少到  $\mathcal{O}(nh^2)$ ，但仍然比常规 NADE 差  $h$  倍。

## 20.11 从自编码器采样

在第十四章中，我们看到许多种学习数据分布的自编码器。得分匹配、去噪自编码器和收缩自编码器之间有着密切的联系。这些联系表明某些类型的自编码器以某些方式学习数据分布。我们还没有讨论如何从这样的模型中采样。

某些类型的自编码器，例如变分自编码器，明确地表示概率分布并且允许直接的原始采样。而大多数其他类型的自编码器则需要 MCMC 采样。

收缩自编码器被设计为恢复数据流形切面的估计。这意味着使用注入噪声的重复编码和解码将引起沿着流形表面的随机游走 (Rifai *et al.*, 2012; Mesnil *et al.*, 2012)。这种流形扩散技术是马尔可夫链的一种。

更一般的马尔可夫链还可以从任何去噪自编码器中采样。

### 20.11.1 与任意去噪自编码器相关的马尔可夫链

上述讨论留下了一个开放问题——注入什么噪声和从哪获得马尔可夫链（可以根据自编码器估计的分布生成样本）。Bengio *et al.* (2013c) 展示了如何构建这种用于广义去噪自编码器(generalized denoising autoencoder) 的马尔可夫链。广义去噪自编码器由去噪分布指定，给定损坏输入后，对干净输入的估计进行采样。

根据估计分布生成的马尔可夫链的每个步骤由以下子步骤组成，如图 20.11 所示：

1. 从先前状态  $\mathbf{x}$  开始，注入损坏噪声，从  $C(\tilde{\mathbf{x}} | \mathbf{x})$  中采样  $\tilde{\mathbf{x}}$ 。
2. 将  $\tilde{\mathbf{x}}$  编码为  $\mathbf{h} = f(\tilde{\mathbf{x}})$ 。
3. 解码  $\mathbf{h}$  以获得  $p(\mathbf{x} | \omega = g(\mathbf{h})) = p(\mathbf{x} | \tilde{\mathbf{x}})$  的参数  $\omega = g(\mathbf{h})$ 。
4. 从  $p(\mathbf{x} | \omega = g(\mathbf{h})) = p(\mathbf{x} | \tilde{\mathbf{x}})$  采样下一状态  $\mathbf{x}$ 。

Bengio *et al.* (2014) 表明，如果自编码器  $p(\mathbf{x} | \tilde{\mathbf{x}})$  形成对应真实条件分布的一致估计量，则上述马尔可夫链的平稳分布形成数据生成分布  $\mathbf{x}$  的一致估计量（虽然是隐式的）。

### 20.11.2 夹合与条件采样

与玻尔兹曼机类似，去噪自编码器及其推广（例如下面描述的 GSN）可用于从条件分布  $p(\mathbf{x}_f | \mathbf{x}_o)$  中采样，只需夹合观察单元  $\mathbf{x}_f$  并在给定  $\mathbf{x}_f$  和采好的潜变量（如果有的话）下仅重采样自由单元  $\mathbf{x}_o$ 。例如，MP-DBM 可以被解释为去噪自编码器的一种形式，并且能够采样丢失的输入。GSN 随后将 MP-DBM 中的一些想法推广以执行相同的操作 (Bengio *et al.*, 2014)。Alain *et al.* (2015) 从 Bengio *et al.* (2014) 的命题 1 中发现了一个缺失条件，即转移算子（由从链的一个状态到下一个

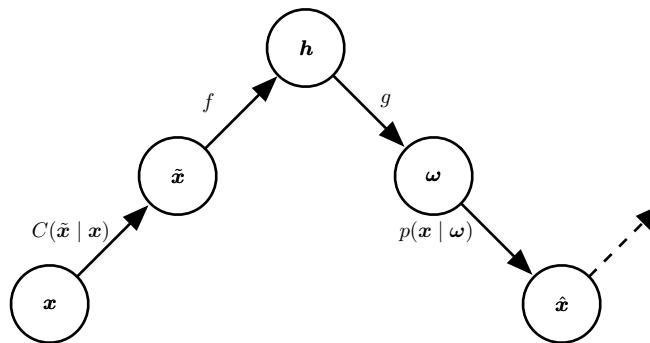


图 20.11：马尔可夫链的每个步骤与训练好的去噪自编码器相关联，根据由去噪对数似然准则隐式训练的概率模型生成样本。每个步骤包括：(a) 通过损坏过程  $C$  向状态  $x$  注入噪声产生  $\tilde{x}$ ，(b) 用函数  $f$  对其编码，产生  $h = f(\tilde{x})$ ，(c) 用函数  $g$  解码结果，产生用于重构分布的参数  $\omega$ ，(d) 给定  $\omega$ ，从重构分布  $p(x | \omega) = g(f(\tilde{x}))$  采样新状态。在典型的平方重构误差情况下， $g(h) = \hat{x}$ ，并估计  $E[x | \tilde{x}]$ ，损坏包括添加高斯噪声，并且从  $p(x | \omega)$  的采样包括第二次向重构  $\hat{x}$  添加高斯噪声。后者的噪声水平应对应于重构的均方误差，而注入的噪声是控制混合速度以及估计器平滑经验分布程度的超参数 (Vincent, 2011)。在这所示的例子中，只有  $C$  和  $p$  条件是随机步骤 ( $f$  和  $g$  是确定性计算)，我们也可以在自编码器内部注入噪声，如生成随机网络 (Bengio et al., 2014)。

状态的随机映射定义) 应该满足 **细致平衡** (detailed balance) 的属性，表明无论转移算子正向或反向运行，马尔可夫链都将保持平衡。

在图 20.12 中展示了夹合一半像素 (图像的右部分) 并在另一半上运行马尔可夫链的实验。

### 20.11.3 回退训练过程

回退训练过程由 Bengio et al. (2013c) 等人提出，作为一种加速去噪自编码器生成训练收敛的方法。不像执行一步编码-解码重建，该过程由交替的多个随机编码-解码步骤组成 (如在生成马尔可夫链中)，以训练样本初始化 (正如在第 18.2 节中描述的对比散度算法)，并惩罚最后的概率重建 (或沿途的所有重建)。

训练  $k$  个步骤与训练一个步骤是等价的 (在实现相同稳态分布的意义上)，但是实际上可以更有效地去除来自数据的伪模式。



图 20.12: 在每步仅重采样左半部分, 夹合图像的右半部分并运行马尔可夫链的示意图。这些样本来自重构 MNIST 数字的 GSN (每个时间步使用回退过程)。

## 20.12 生成随机网络

生成随机网络 (generative stochastic network, GSN) (Bengio *et al.*, 2014) 是去噪自编码器的推广, 除可见变量 (通常表示为  $\mathbf{x}$ ) 之外, 在生成马尔可夫链中还包括潜变量  $\mathbf{h}$ 。

GSN 由两个条件概率分布参数化, 指定马尔可夫链的一步:

1.  $p(\mathbf{x}^{(k)} \mid \mathbf{h}^{(k)})$  指示在给定当前潜在状态下如何产生下一个可见变量。这种“重建分布”也可以在去噪自编码器、RBM、DBN 和 DBM 中找到。
2.  $p(\mathbf{h}^{(k)} \mid \mathbf{h}^{(k-1)}, \mathbf{x}^{(k-1)})$  指示在给定先前的潜在状态和可见变量下如何更新潜在状态变量。

去噪自编码器和 GSN 不同于经典的概率模型 (有向或无向), 它们自己参数化生成过程而不是通过可见和潜变量的联合分布的数学形式。相反, 后者如果存在则隐式地定义为生成马尔可夫链的稳态分布。存在稳态分布的条件是温和的, 并且需要与标准 MCMC 方法相同的条件 (见第 17.3 节)。这些条件是保证链混合的必要条

件，但它们可能被某些过渡分布的选择（例如，如果它们是确定性的）所违反。

我们可以想象 GSN 不同的训练准则。由 Bengio *et al.* (2014) 提出和评估的只对可见单元上对数概率的重建，如应用于去噪自编码器。通过将  $\mathbf{x}^{(0)} = \mathbf{x}$  夹合到观察到的样本并且在一些后续时间步处使生成  $\mathbf{x}$  的概率最大化，即最大化  $\log p(\mathbf{x}^{(k)} = \mathbf{x} | \mathbf{h}^{(k)})$ ，其中给定  $\mathbf{x}^{(0)} = \mathbf{x}$  后， $\mathbf{h}^{(k)}$  从链中采样。为了估计相对于模型其他部分的  $\log p(\mathbf{x}^{(k)} = \mathbf{x} | \mathbf{h}^{(k)})$  的梯度，Bengio *et al.* (2014) 使用了在第 20.9 节中介绍的重参数化技巧。

回退训练过程（在第 20.11.3 节中描述）可以用来改善训练 GSN 的收敛性 (Bengio *et al.*, 2014)。

### 20.12.1 判别性 GSN

GSN 的原始公式 (Bengio *et al.*, 2014) 用于无监督学习和对观察数据  $\mathbf{x}$  的  $p(\mathbf{x})$  的隐式建模，但是我们可以修改框架来优化  $p(\mathbf{y} | \mathbf{x})$ 。

例如，Zhou and Troyanskaya (2014) 以如下方式推广 GSN，只反向传播输出变量上的重建对数概率，并保持输入变量固定。他们将这种方式成功应用于建模序列（蛋白质二级结构），并在马尔可夫链的转换算子中引入（一维）卷积结构。重要的是要记住，对于马尔可夫链的每一步，我们需要为每个层生成新序列，并且该序列用于在下一时间步计算其他层的值（例如下面一个和上面一个）的输入。

因此，马尔可夫链确实不只是输出变量（与更高层的隐藏层相关联），并且输入序列仅用于条件化该链，其中反向传播使得它能够学习输入序列如何条件化由马尔可夫链隐含表示的输出分布。因此这是在结构化输出中使用 GSN 的一个例子。

Zöhrer and Pernkopf (2014) 引入了一个混合模型，通过简单地添加（使用不同的权重）监督和非监督成本即  $\mathbf{y}$  和  $\mathbf{x}$  的重建对数概率，组合了监督目标（如上面的工作）和无监督目标（如原始的 GSN）。Larochelle and Bengio (2008a) 以前在 RBM 中就提出了这样的混合标准。他们展示了在这种方案下分类性能的提升。

## 20.13 其他生成方案

目前为止我们已经描述的方法，使用 MCMC 采样、原始采样或两者的一些混合来生成样本。虽然这些是生成式建模中最流行的方法，但它们绝不是唯一的方法。

Sohl-Dickstein *et al.* (2015) 开发了一种基于非平衡热力学学习生成模型的扩散反演 (diffusion inversion) 训练方案。该方法基于我们希望从中采样的概率分布具有结构的想法。这种结构会被递增地使概率分布具有更多熵的扩散过程逐渐破坏。为了形成生成模型，我们可以反过来运行该过程，通过训练模型逐渐将结构恢复到非结构化分布。通过迭代地应用使分布更接近目标分布的过程，我们可以逐渐接近该目标分布。在涉及许多迭代以产生样本的意义上，这种方法类似于 MCMC 方法。然而，模型被定义为由链的最后一步产生的概率分布。在这个意义上，没有由迭代过程诱导的近似。Sohl-Dickstein *et al.* (2015) 介绍的方法也非常接近于去噪自编码器的生成解释（第 20.11.1 节）。与去噪自编码器一样，扩散反演训练一个尝试概率地撤消添加的噪声效果的转移算子。不同之处在于，扩散反演只需要消除扩散过程的一个步骤，而不是一直返回到一个干净的数据点。这解决了去噪自编码器的普通重建对数似然目标中存在的以下两难问题：小噪声的情况下学习者只能看到数据点附近的配置，而在大噪声的情况下，去噪自编码器被要求做几乎不可能的工作（因为去噪分布是高度复杂和多峰值的）。利用扩散反演目标，学习者可以更精确地学习数据点周围的密度形状，以及去除可能在远离数据点处出现的假性模式。

样本生成的另一种方法是近似贝叶斯计算 (approximate Bayesian computation, ABC) 框架 (Rubin *et al.*, 1984)。在这种方法中，样本被拒绝或修改以使样本选定函数的矩匹配期望分布的那些矩。虽然这个想法与矩匹配一样使用样本的矩，但它不同于矩匹配，因为它修改样本本身，而不是训练模型来自动发出具有正确矩的样本。Bachman and Precup (2015) 展示了如何在深度学习的背景下使用 ABC 中的想法，即使用 ABC 来塑造 GSN 的 MCMC 轨迹。

我们期待更多其他等待发现的生成式建模方法。

## 20.14 评估生成模型

研究生成模型的研究者通常需要将一个生成模型与另一个生成模型比较，通常是为了证明新发明的生成模型比之前存在的模型更能捕获一些分布。

这可能是一个困难且微妙的任务。通常，我们不能实际评估模型下数据的对数概率，但仅可以评估一个近似。在这些情况下，重要的是思考和沟通清楚正在测量什么。例如，假设我们可以评估模型 A 对数似然的随机估计和模型 B 对数似然的确定性下界。如果模型 A 得分高于模型 B，哪个更好？如果我们关心确定哪个模型

具有分布更好的内部表示，我们实际上不能说哪个更好，除非我们有一些方法来确定模型 B 的边界有多松。然而，如果我们关心在实践中该模型能用得多好，例如执行异常检测，则基于特定于感兴趣的 actual task 的准则，可以公平地说模型是更好的，例如基于排名测试样例和排名标准，如精度和召回率。

评估生成模型的另一个微妙之处是，评估指标往往是自身困难的研究问题。可能很难确定模型是否被公平比较。例如，假设我们使用 AIS 来估计  $\log Z$  以便为我们刚刚发明的新模型计算  $\log \tilde{p}(\mathbf{x}) - \log Z$ 。AIS 计算经济的实现可能无法找到模型分布的几种模式并低估  $Z$ ，这将导致我们高估  $\log p(\mathbf{x})$ 。因此可能难以判断高似然估计是否是良好模型或不好的 AIS 实现导致的结果。

机器学习的其他领域通常允许在数据预处理中有一些变化。例如，当比较对象识别算法的准确性时，通常可接受的是对每种算法略微不同地预处理输入图像（基于每种算法具有何种输入要求）。而因为预处理的变化，会导致生成式建模的不同，甚至非常小和微妙的变化也是完全不可接受的。对输入数据的任何更改都会改变要捕获的分布，并从根本上改变任务。例如，将输入乘以 0.1 将人为地将概率增加 10 倍。

预处理的问题通常在基于 MNIST 数据集上的生成模型产生，MNIST 数据集是非常受欢迎的生成式建模基准之一。MNIST 由灰度图像组成。一些模型将 MNIST 图像视为实向量空间中的点，而其他模型将其视为二值。还有一些将灰度值视为二值样本的概率。我们必须将实值模型仅与其他实值模型比较，二值模型仅与其他二值模型进行比较。否则，测量的似然性不在相同的空间。对于二值模型，对数似然可以最多为零，而对于实值模型，它可以是任意高的，因为它是关于密度的测度。在二值模型中，比较使用完全相同的二值化模型是重要的。例如，我们可以将 0.5 设为阈值后，将灰度像素二值化为 0 或 1，或者通过由灰度像素强度给出样本为 1 的概率来采一个随机样本。如果我们使用随机二值化，我们可能将整个数据集二值化一次，或者我们可能为每个训练步骤采不同的随机样例，然后采多个样本进行评估。这三个方案中的每一个都会产生极不相同的似然数，并且当比较不同的模型时，两个模型使用相同的二值化方案来训练和评估是重要的。事实上，应用单个随机二值化步骤的研究者共享包含随机二值化结果的文件，使得基于二值化步骤的不同输出的结果没有差别。

因为从数据分布生成真实样本是生成模型的目标之一，所以实践者通常通过视觉检查样本来评估生成模型。在最好的情况下，这不是由研究人员本身，而是由不知道样品来源的实验受试者完成 (Denton *et al.*, 2015)。不幸的是，非常差的概率

模型可能会产生非常好的样本。验证模型是否仅复制一些训练示例的常见做法如图 16.1 所示。该想法是根据在  $\mathbf{x}$  空间中的欧几里得距离，为一些生成的样本显示它们在训练集中的最近邻。此测试旨在检测模型过拟合训练集并仅再现训练实例的情况。甚至可能同时欠拟合和过拟合，但仍然能产生单独看起来好的样本。想象一下，生成模型用狗和猫的图像训练时，但只是简单地学习来重现狗的训练图像。这样的模型明显过拟合，因为它不能产生不在训练集中的图像，但是它也欠拟合，因为它不给猫的训练图像分配概率。然而，人类观察者将判断狗的每个个体图像都是高质量的。在这个简单的例子中，对于能够检查许多样本的人类观察者来说，确定猫的不存在是容易的。在更实际的设定中，在具有数万个模式的数据上训练后的生成模型可以忽略少数模式，并且人类观察者不能容易地检查或记住足够的图像以检测丢失的变化。

由于样本的视觉质量不是可靠的标准，所以当计算可行时，我们通常还评估模型分配给测试数据的对数似然。不幸的是，在某些情况下，似然性似乎不可能测量我们真正关心的模型的任何属性。例如，MNIST 的实值模型可以将任意低的方差分配给从不改变的背景像素，获得任意高的似然。即使这不是一个非常有用的事情，检测这些常量特征的模型和算法可以获得无限的奖励。实现接近负无穷代价的可能性存在于任何实值的最大似然问题中，但是对于 MNIST 的生成模型问题尤为严重，因为许多输出值是不需要预测的。这强烈地表明需要开发评估生成模型的其他方法。

Theis *et al.* (2015) 回顾了评估生成模型所涉及的许多问题，包括上述的许多想法。他们强调了生成模型有许多不同的用途，并且指标的选择必须与模型的预期用途相匹配。例如，一些生成模型更好地为大多数真实的点分配高概率，而其他生成模型擅长于不将高概率分配给不真实的点。这些差异可能源于生成模型是设计为最小化  $D_{\text{KL}}(p_{\text{data}} \parallel p_{\text{model}})$  还是  $D_{\text{KL}}(p_{\text{model}} \parallel p_{\text{data}})$ ，如图 3.6 所示。不幸的是，即使我们将每个指标的使用限制在最适合的任务上，目前使用的所有指标仍存在严重的缺陷。因此，生成式建模中最重要的研究课题之一不仅仅是如何提升生成模型，事实上还包括了设计新的技术来衡量我们的进步。

## 20.15 结论

为了让模型理解表示在给定训练数据中的大千世界，训练具有隐藏单元的生成模型是一种有力方法。通过学习模型  $p_{\text{model}}(\mathbf{x})$  和表示  $p_{\text{model}}(\mathbf{h} \mid \mathbf{x})$ ，生成模型可以解答  $\mathbf{x}$  输入变量之间关系的许多推断问题，并且可以在层次的不同层对  $\mathbf{h}$  求期望来