Learning Deep Architectures for AI

Yoshua Bengio

摘要

理论结果表明，为了学习用于表示高层次的抽象（例如视觉、语言以及其他 AI 级别的任务）的复杂函数，我们需要深度结构。深度结构的组成包括了多层次的非线性操作，比如具有许多隐含层的神经网络，或者重用了许多子公式的复杂命题公式。搜索深度结构的参数空间是一件很困难的任务，但是最近提出的诸如用于深度信念网络等的学习算法，对于探索这类问题取得了显著的成功，在某些领域达到了最新的水平。本书讨论深度学习算法的方法和原理，尤其是那些被充分用作基石的单层模型的非监督学习算法例如受限玻尔兹曼机(RBM)，它用于构建深度信念网络等深度模型。

第一章　介绍

允许让计算机来模拟我们的世界，足以表明我们所说的智力已经超过半个世纪一直是的研究重点。为了实现这一点，很显然，有大量的关于我们的世界的信息应该以某种方式被明确地或含蓄地存储在计算机中。因为手工地将那些信息转换，并使计算机可以用它们来回答问题并推广到新的环境中是一件令人怯步的事，许多研究者纷纷转向学习算法来获取大部分信息。了解和改进的学习算法已经取得了很大的进展，但人工智能仍然面临挑战。我们有算法可以去理解场景并用自然语言来描述它们？不完全是，除非在非常有限的情况下。我们有算法可以推断出足够的语义概念以便能够使用这些概念进行大多数人类互动？不，如果我们考虑图像理解，人工智能任务里最棒的一个分支，我们认识到，我们还没有能发现许多视觉和语义概念的学习算法，而它们(指概念)似乎有必要出现在网络上解释大部分图像。这种情况就类似于其他的 AI 任务。

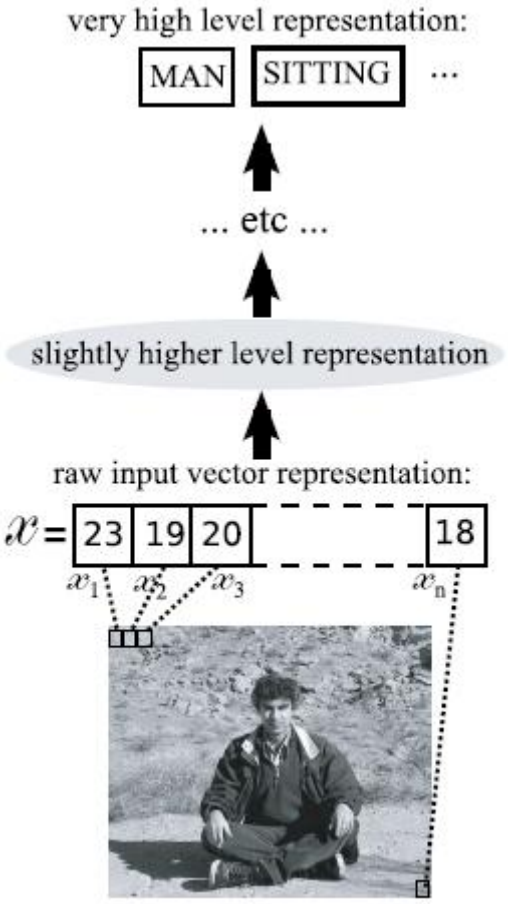


图 1.1 我们希望原始输入图像能被转换为更高水平的表述，并表现原始输入表达式越来越多的抽象功能，如：边、局域形状、对象部分等等。在实践中，我们不能提前知道“正确”的表达式应该是什么，在所有这些抽象层中，尽管语言概念可以帮助我们猜测到更高层次表述可能隐含的信息

考虑一个例子，要求解释一个如图 1.1 中输入的图像。当人们尝试着去解决特定的 AI 任务（例如机器视觉或自然语言处理），他们往往会直接利用他们的有关如何分解问题转化子问题和多级表示的经验，例如，在对象的部分和一系列模型中[138,179,197],可以重新用在不同的对象实例模型的部件。例如，最先进机器视觉领域的当前语句涉及从像素开始并以线性或内核分类结束[134,145]模块的序列，以及中间模块混合工程的转换和学习。

例如，先提取低级别的功能，那些不变的小几何变化（如 Gabor 滤波器的边缘检测），并逐渐转变它们（例如，使它们在参照物改变或反转时保持不变，有时通过集中和子采样），然后检测最频繁的模式。一种貌似有理且常见的方式是从自然图像中提取有用的信息，包括将所述原始像素表示成逐渐更抽象的表示，例如，从边缘表达式，以及更复杂的，局部的形状的检测，到抽象类别的识别与子对象，哪些是图像的部分对象相关联，并把这些放到一起来捕获足够关于这些场景的理解来回答关于它们的问题。

在这里，我们假设必要的计算机来表达复杂的行为（其中一个可能被标记为“智能”）需要

高度变化的数学函数，即，数学函数是高度非线性依据原始感知输入，并横跨感兴趣的领域显示一个非常大的的变化（增长和减小）。我们观察原始输入给学习系统作为一个高维实体，提出了许多观测变量的，这些都和未知的错综复杂的统计关系有关。例如，使用固态物体和照明的三维几何的知识，就可以在与底层物理和几何因素小的变化（如位置，方向，对象的照明）用的图像的所有像素的变化像素强度。我们称之为变量，是因为它们是数据的不同方面，并可以独立地经常变化。在这种情况下，所涉及的物理因素的明确知识允许人们获得该组图像的形状，这些依赖关系的数学形式的图片，和（如在像素强度的高维空间中的点）与相关联的同样的 3D 对象。如果一台机器捕获变量在数据中的统计变化，以及它们是如何相互作用以产生我们观察到的数据种类，我们将能够说，机器理解是指这些变化的因素覆盖的的世界。不幸的是，在一般情况和变化最底层的因素的自然图像，我们没有对分析这些变化因素的理解。我们没有足够形式化的关于世界的先验知识来解释所观察到的各种图像，即便是这样一个看似简单的抽象：MAN，像图 1.1 中那样。一个高层次的抽象，例如，MAN 有一个性质，他相当于一个巨大的可能图像的集合，这可能非常不同于那些简单欧几里得距离中的像素点的属性。该组图像能够识别此标签可能是适当的形式在像素空间中的高度旋绕区域即甚至没有必要是一个连接区域。MAN 类可以被看作是相对于图象的空间的高层次抽象。我们所说的抽象这里可以是一个类（如 MAN 类别）或特征，感觉数据的函数，它可以是离散的（例如，输入的句子是在过去时），或连续的（例如，输入视频显示了运动物体的移动是 2 米/秒）。对于构建 MAN-探测器，许多较低级和中间级的概念（我们这里也称之为抽象）将是有用的。低层次的抽象更直接关系到特定的知觉，而更高级别的那些就是我们所说的“更抽象的”，因为他们与实际知觉的连接比较微弱，是通过了其他中间层次的抽象。

除了想出合适的中间抽象的难度，我们希望用一种“智能”的机器来捕获的视觉和语义类别的数目（如 MAN）相当大了。深层结构学习的重点是能够自动发现这样的抽象，从最低级别的特性到最高层次的概念。理想情况下，我们希望使这一发现成真的学习算法只需很少人的努力，换言之，无需手动定义所有必要的抽象或不必提供一组相关的手工标记的巨大样本。如果这些算法可以挖掘到网络上的文本和图像的巨大资源，这肯定会有助于传递许多人类的知识转换为机器可解释的形式。

1.1 我们如何学习深度架构？

深度学习方法的目的是学习层次与更高水平由低级别的功能组合物形成深层次结构的特点。自动多层次的抽象学习功能允许系统学习复杂的功能映射，直接从数据的输入到输出，而不完全依赖于人类制作的功能。这对于更高层次的抽象特别重要，人们经常常常不知道怎么根据原始感受数据来明确给出指令。能够自动学习强大特征的能力将随着机器学习方法的数据量和应用范围继续增长而变得越来越重要。

架构的深度是指在功能学习中非线性运算组合的数目。而目前大多数学习算法对应于浅层结构（1，2 或 3 层），哺乳动物脑是一个组织组织在深度架构[173]与多个级别的抽象，每一级

对应于不同的区域的，代表一个给定的输入知觉。人类往往用分层的方式，借助多层次的抽象来描述这种概念。大脑似乎也处理通过转换和表达的多个阶段的信息。这在灵长类视觉系统[173]特别清楚，按顺序处理：检测边缘，原始形状，并且向上移动到逐渐更复杂的视觉形状。

灵感来自于大脑的深度架构，神经网络研究人员努力了几十年来训练深度多层神经网络[19，191]，但在 2006 年之前没有尝试成功[除具有特殊的结构的神经网络，称为卷积网络，讨论于 4.5 节]：研究人员报告了有两个或三个层次（即一个或两个隐藏层）的阳性试验结果，但更深层次的培训网络，一直都取得的是较差的结果。突破性的事情发生在 2006 年：Hinton 等人在多伦多大学引进深信念网络（DBNs）[73]，它具有学习算法，每次贪婪地学习一层，对每个层利用无监督学习算法，受限波尔兹曼机（RBM）[51]。不久后，相关的基于自动编码器的算法[17，153]也被提出，显然利用了同样的原理：采用无监督学习，这可以在本地在每个级别进行指导代表性的中等水平的培训。其他算法的深架构提出了近期的利用既没有 RBMS 也不是自动编码器，并且利用同样的原理[131，202]（见第 4 章）。

自 2006 年以来，深度网络已经不仅在分类任务[2，17，99，111，150，153，195]应用成功，而且在回归[160]，降维[74，158]，造型纹理[141]，建模运动[182，183]，对象分割[114]，信息检索[154，159，190]，机器人[60]，自然语言处理[37，130，202]，和协同过滤[162]。虽然自动编码器，如 RBM 和 DBN，可以用未标记的数据进行训练，在上述的许多应用中，它们已被成功地用于初始化一些深度有监督的正向反馈特定神经网络任务。

1.2 中间表示：在任务间共享特性和抽象

由于深度结构可以看作由一系列加工阶段组成，提高深度结构当前面临的问题是：每个阶段的输出（即：另一个阶段的输入）到底采用哪种数据表示？阶段之间采用哪种接口？最近深度结构研究的一个特点就是聚焦中间表示：深度结构的成功属于借鉴 RBMs [73]，普通自动编码器[17]，稀疏自动编码器[150，153]，或者降噪自动编码器[195]等学习无监督的方式表示。这些算法（详见 7.2 节）可以被看作是学习将一个表示（前一阶段的输出）变换为另一个表示，每次变换也许能解开潜在变化数据中的更好因素。正如在 4 节中我们讨论的长度，它被一次又一次观察到，一旦一个良好表示在各级中被发现，它可以用于初始化，并成功地通过监督基于梯度的优化培养深神经网络。

在大脑中发现的每一级抽象包括了大量特征的小的子集的“激活”（神经激发），在一般情况下，它们不是相互排斥的。因为这些特征不是相互排斥的，它们形成所谓的分布式表示[68，156]：该信息不被局限在一个特定的神经元，但分布在许多中。除了被分发，它似乎显示大脑使用的表示较稀疏：在给定的时间里，神经元中仅一个围绕 1-4％的区域是同时激活 [5，113]。3.2 节介绍稀疏分布表示的概念，第 7.1 节详细介绍了机器学习的方法。部分灵感来自于大脑中的稀疏表示的观察，已被用于采用稀疏表示构建深层结构。

而稠密分布表示是一个频谱的一个极端，和稀疏表示是在该光谱的中间，纯粹本地表示是

另一个极端。代表性的地方是密切与当地泛化的概念连接。许多现有的机器学习方法是局部的输入空间：获得一个有学问的功能表现不同的数据空间的不同区域，它们在每一个地区（详见 3.1 节）需要不同的可调参数。即使在可调参数的数目很大时，统计效率不一定差，良好的泛化可以加入某种形式的先验（例如，参数的较小值是首选）。当先验不是特定任务时，它往往迫使解决方案是很平滑的，如在第 3.1 节的末尾讨论那样。对比基于局部泛化的学习方法，模式总数，可以区分使用一个分布式表示比例可能呈指数级表示维度（即:学习特征的数量）。

在许多机器视觉系统，学习算法都被限制在这样一个处理链的特定部分。其余的设计仍然是劳动密集型，这可能限制了这种系统的规模。另一方面，我们认为的智能机包括了一个足够大的概念指令表。认识到 MAN 是不够的。我们需要的算法应可以解决一个非常大的任务和概念。在这种情况下，手动定义许多任务和必要的学习看起来令人气馁。此外，在这些任务之间和需要的概念之间不利用基本的共性是看起来很愚蠢的。这已是多任务学习[7，8，32，88,186]研究的重点。具有多层次自然结构提供了共享和重用组件：低级别的视觉特征（如边缘检测）和中等水平的视觉特征（如对象的部分），用于检测人，也可用于一大组的其他视觉任务。深度学习算法都是基于学习可共享的任务的中间表示。因此，他们可以利用无监督的数据和来自类似任务[148]的数据，以来提高大型的、具有挑战性的问题性能，经常遭受贫困的标记的数据，已经显示出[37]，在国家的最先进的多种自然语言处理任务。深层结构相似的多任务的方法应用在视觉任务[2]。考虑有不同任务不同输出的一个多任务环境，他们都来自于一个高层次的功能共享池。在多任务中，事实上许多这些学习特性是共享的，并为提供统计强度共享比例。现在认为，这些学习的高级功能，可以自表示在一个普通池低级别的中间表示。再次统计强度可以通过类似的方式获得，这种策略可以用于深层结构的各个层次。

此外，学习大量的相关概念，可能提供人类出现能做的那种广泛概括的一个关键，我们不期望于独立的培训对象检测器，而每个视觉范畴都有个探测器。如果每一个高层次的范畴本身是代表通过一个特定的分布式结构的抽象特征从一个普通的池，推广到看不见的类别可以遵循自然，来自这些特征的新配置。尽管这些功能只有一些配置会出现于训练样本中，但如果他们代表不同方面的数据，新的例子可以有效地被这些特征的新配置代表。

1.3 人工智能学习中的迫切需求

总结上文所提到的各个问题，并从人工智能的更为广泛的角度来观察，我们提出了我们认为在人工智能的学习算法中至关重要的若干要求，它们中促进了科学研究的列举如下：

- 学习复杂且高度灵活的函数的能力。在这种情况下，变量的数量往往远大于训练集实例的数量。

- 在人工干预极少的情况下，学习那些用于表示复杂函数的各个层次（低级，普通，高级；译者按：这里是指各个层次的语义）的抽象概念的能力；而表示这些复杂函数是人工智能任务所需要的。
- 从海量实例中学习的能力：训练所需的计算时间应与所使用的实例数量呈良好的可度量关系，即接近线性。
- 在大部分数据未进行标记的情况下进行学习的能力，即半监督情况下的学习。这种情况下，并不是所有的数据都被进行了完整、正确的语义标记。
- 探寻大量任务间出现的增效作用的能力，即多任务学习的能力。之所以会存在这样的增效作用，是因为所有的人工智能任务都对相同的潜在现实现象提供了不同的视角。
- 完善的非监督学习的能力（即捕捉样本数据中的大部分统计结构），这在面对海量任务量以及未来学习任务不明确的情况下是非常关键的。

其他一些要素也同样重要，但与本文将要着重叙述的内容并无直接关系。这些要素包括：（1）学习表达长度及结构不定的语境的能力[146], 这样也就赋予了我们的机器在依赖语境的环境下工作的能力; （2） 在所采取的行为会影响未来观测值以及未来激励值的情况下，做出决策的能力[181]；（3）以收集更多相关信息为目的去影响未来观测值的能力，即某种形式的主动学习[34]。

1.4 论文概述

第二章回顾了理论上的成果（此处跳过不影响理解后文），表明一个深度不足的结构，与一个深度与任务相匹配的结构相比，可能需要更多的计算单元，甚至可能是指数级的（相对于输入大小）。我们认为，深度不足会不利于学习。实际上，如果一个任务的解决方案被表达成一个非常大的浅层的结构（有很多计算单元），可能需要大量的训练样本，来调整每个计算单元并学习一个高度变化的函数。3.1 节同样是为了激发读者的兴趣，这次是突出了局部泛化和局部估计的局限性，我们希望能够避免在分布式表达下使用深度结构（3.2 节）。

在后面的章节中，专门描述和分析了一些已经被用来训练深度结构的算法。第四章介绍了神经网络文献中与训练深度结构相关的一些概念。我们首先考虑了以前训练多层神经网络的难点问题，随后引进了无监督算法，它可以被用来初始化深度神经网络。这其中的许多算法（包括用于受限玻尔兹曼机的）都与自动编码器有关。自动编码器是一个简单的无监督算法，可以用来学习一层模型来计算其输入的一个分布式表达。 为了全面理解受限玻尔兹曼机和许多相关的无监督学习算法，第五章介绍了基于能量的模型，包括那些被用来构建含有隐变量生成模型的模型，比如玻尔兹曼机。 第六章的重点是深度信念网络和多层自编码器的贪婪式逐层训练算法。 第七章讨论了对受限玻尔兹曼机和自编码器进行扩展和改进后的一些变化形式，包括稀疏性的使用，以及对时间依赖性的建模等。第八章讨论了采用变分边界对深度信念网络的所有层

进行联合训练的算法。 最后在第九章，我们考虑了一些前瞻性问题，比如训练深度结构中涉及到的假设的困难优化问题。特别是地，我们跟进这样一个假设：现有深度结构学习策略的成功一部分是与低层的优化相连的。我们讨论了延拓法的原理，它逐步地降低了对代价估价函数平滑性的要求，在深度结构优化方面取得了初步的进展。

第二章 深层架构的理论优势

在本节中，我们提出了一个激励参数学习深层结构的算法，通过理论计算分析深度不足所潜在的局限性。这部分专著（本节和下一节）中的一些算法会在后面被描述，这部分即使跳过，剩下的部分也不会难以理解。

这一节的主要观点是一些函数不能被太浅显的架构有效的描述出来（在可调元素的数量）。这些结果表明探索深层架构学习算法很有意义，它也许可以代表这些函数，而不是有效的表现。在简单和浅层的结构不能有效地描述（而且很难学习）一个感兴趣的任务的地方，我们可以寄希望于设置一个深层结构参数的学习算法去描述这个任务。

当一个函数表达式只有几个计算元素时，我们通常说它是简洁的，就是有几个自由度需要通过学习来调节。所以对一个固定数量的训练样本，在学习算法中缺少其他学习源的注入，我们期望目标函数的简洁表示可以更好的泛化。

更精确的说，可以由一个深度为 K 架构被紧凑的表示的函数可能需要计算元素的指数由一个深度为 k-1 架构来表示。由于计算的元素可以负担得起的数量取决于可用的调整或选择他们训练实例的数量，其后果不仅是计算也是统计：当使用不够深的结构去表现一些函数时，泛化不好是可以被预期的。

我们考虑固定维数输入的情况，由机器执行的计算部分可通过有向无环图来表示，其中每个节点对输入执行的运算是这个应用的功能，它们的输出是另一个节点的输入。整个图看以看作是一个可以计算一个函数被应用到外部输入的电路。当所设置的允许计算节点的函数仅限于逻辑门，例如{与，或，非}，这是一个布尔电路或逻辑电路。

为了使深层架构的概念更加正式，必须引入一组计算元素的概念。这样的一组的一个实例是一组可以执行逻辑门的计算， 另一个是一组可以由人工神经元来进行的计算（取决于它的突触权重的值）。一个函数可以通过给定的计算元素组成的集合来表示。它可以用一个图来定义，这个图的每个节点都是计算单元。架构的深度与那张图的深度有关，也就是指从一个输入节点到输出节点的最长路径。当所设置的计算元素是人工神经可以执行的计算，深度就对应于神经网路的层次数。让我们用不同深度架构的例子来探索这个深度的概念。思考一下这个函数 f(x) = x * sin(a * x + b). 它可以由一些简单的操作如，加法，减法，乘法和正弦描述，如插图 2.1. 在这个例子中有一个不同的节点用于乘法 a*x 和用于最后一个乘法。图中的每个节点

与通过其它节点作为输入值应用于函数而输出的值相关联。

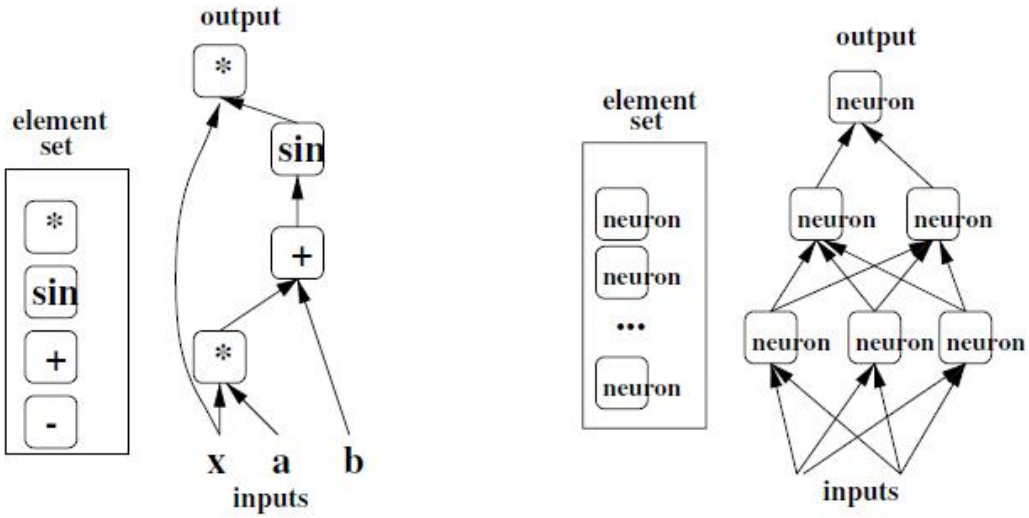


图 2.1 一个可以计算的图表示的函数例子，这里的每个节点取自于可以被允许计算的集合。左边图中的元素是{*,+,−,sin} ∪ R。 这个结构计算 x * sin(a * x + b)， 而且深度是 4. 右图是元素的人工神经元计算 f(x) = tanh (b + w ' x);这个集合里的每个节点都有个不同的（w,b）参数。 这个架构是一个深度为 3 的多层次的神经网络。

例如，在一个逻辑电路中，每个节点都可以从一些小的布尔函数来计算另一个布尔函数。这张图作为一个整体有输入节点，有输出节点，有一个计算从输入到输出的函数。一个架构的深度是这张图中任意一个输入节点到任意一个输出节点最长的路径,就像图 2.1 中的 x * sin(a * x + b)的情况下，深度就是 4。

- 如果我们包含一个映射操作以及它们与在该组的计算原件，线性回归和逻辑回归的 S 型可能组合有深度 1，也就是说只有一个单一的层次。
- 当我们把一个固定的内核计算 K(u,v) 放在一系列允许的操作中，与映射操作一起，具有一个固定内核的内核机器[166]可以被认为有两个层次。第一层关于每一个原型 x i(一个被选中的训练样本代表)都有一个元素计算 K(x, xi) ， 而且匹配每个输入向量 x 和原型 xi。第二层执行一个映射操作 $b + \sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i)$ 去用预期的结果联合匹配的原型。
- 当我们把神经网络（非线性映射转化）放在我们的元素集合中，我们获取到普通多层次的神经网络[156].有一个隐藏层次的最常见的选择，它们都有两层（隐藏层和输出层）。
- 决策树也可以被看成有两层，我们会在章节 3.1 讨论它。
- Boosting[52]对于基础学习通常增加一层：这一层可以计算一个表决或者是基于基础学习

者的输出的线性成分。

- Stacking[205]是一另外一个元学习算法，它可以添加一层。
- 基于现有的关于大脑解剖的知识[173]， 可以将大脑皮层看成一个深层的体系结构，仅仅一个视觉系统就用了 5-10 层。

尽管深度取决于对每一个元素的可允许的计算的集合的选择，与一组相关联的图形通常可以转换为与另一个通过图转换中相乘深度的方式相关联的图形。理论结果表明，它是没有电平的绝对数量的事项，但是层次的相对于多少需要有效地代表目标函数（与计算元素中的一些选择）的数目。

2.1 计算的复杂度

关于深度结构的强大的最正式的参数来自于对电路的计算复杂性的研究。基本结论是这些研究结果表明当一个函数可以被深层结构简洁的描述时，如果用深度不足的结构去描述，就需要复杂的结构。

一个双层逻辑门电路可以表示任何布尔函数。任意一个布尔函数[127]可以写作一个乘积和（析取范式：第一层为与门，输入可以包括非，第二层为或门）或者和项积（和取范式：第一层为或门，输入可以包括非作为，第二层为与门）。为了理解浅层架构的局限性，首先我们考虑对于两层逻辑电路，大部分布尔函数需要指数级（与输入大小相关）的逻辑门[198]来表达。

更有趣的是，当把深度限制为 k-1 时[62]，对于一个 k 层多项式门电路的可计算函数需要指数级大小。这个理论的证明依赖于先前的结果[208]，之前证明深度为 2 的 d 位奇偶电路有指数级的大小。d 位奇偶函数通常定义为：

$$\text{奇偶}:(b_1,...,b_d)\in\{0,1\}^d \mapsto \left\{\begin{array}{c} 1, \text{如果} \sum_{i=1}^{d} b_i \text{是偶数} \\ 0, \text{其他} \end{array}\right\}$$

有人可能会奇怪这些布尔电路的计算复杂度和机器学习有什么关系。参考[140]早期的一个计算复杂度与学习算法相关的理论研究的综述。很多布尔电路的结果可以推广为那些计算元素是线性阈值单元（也被称作人工神经元[125]）的架构。用于计算

$$f(x)=1_{w'x+b\geq 0}$$

其中 w 和 b 为参数。电路的扇入是那些输入元素的最大值。电路通常以层来组织，如同多层神经网络，每一层的输入只从之前层获取，第一层为神经网络的输入。电路的大小是计算的元素（不包括那些不进行计算的元素）。

当想用一个 k 层电路表达一个函数紧致可表达时，下面这个使用单调加权阈值电路（就是线性阈值单元和正的权值的多层神经网络）的理论更为有趣：

定理 2.1 存在常数$C>0$和$N>N_0$，使 k-1 层单调加权阈值电路计算函数$f_k \in F_{k,N}$的大小至少为2^{cN} [63].

函数的集合$F_{k,N}$定义如下。包含有N^{2k-2}个输入的函数簇，用 k 层电路组成的树来定义。树的叶子节点是非负输入变量，函数值为根。当 i 为偶数时底部的第 i 层为一系列与门，当奇数时为或门。最顶端和低端的扇入为 N 其他层均为N^2.

上面的结果并不能证明其他函数簇（例如那些执行 AI 学习任务的）需要深度架构，也不能证明那些看到的局限性也适用于其他类型的电路。然而，这些理论结果回避了一个问题：那些 1、2 或 3 层架构是不是太浅以至于不能有效的表示 AI 任务需要的那些更复杂的函数。上述理论的结果表明其实没有一个普适的深度：每一个函数（每一个任务）都可能需要某个特定的最小深度（对于给定计算元素的集合）。所以我们要尽力发现一种算法，可以使用数据来决定最终架构的深度。同时要注意，递归计算所定义的计算图，图的深度随着迭代的数量线性增长。

2.2 非正式的探讨

网络结构的深度是与高变性函数这个概念相关联的。通常，我们认为，高变性函数可以用深层结构来有效的表达，但是如果用其他不恰当的结构来表示这些函数，可能会要用巨量的节点才行。我们说一个函数具有高变性指的是当用分段函数来逼近这个函数的时候，需要大量的分段。一个深层结构是由许多操作构成的，并且在任何情况下，它都可能由巨大的深度为 2 的结构来表达。可以将一个庞大的浅层网络进行重新组合，变成节点少，但是层数多的深层网络。这相当于进行了一项因式分解，深层网络的每一个计算单元可以看成是因式分解中的一个有效的因子，而那个庞大的浅层网络就相当于被分解的公式了。如果计算单元的构成方式组织合理，将会给消减网络的尺寸带来非常大的效果。比如，设想一个网络，其深度是 2k，代表了一个多项式。这个网络的奇数层进行乘法运算，偶数层进行加法运算。这个网络可以看成是一个特别有效的因式分解，如果将这个网络扩展成一个求和的 2 层结构，一层用于乘法运算，一层用于加法运算，那么在加法层上可能就会需要大量的因式。设想一下第一层的产出就知道了（就像图 2.2 中的 x2x3）。这种情况在深度为 2 的结构中经常发生。通过这个例子可以看到，在一些计算可以被共享的情况下，深层结构是其优越性的。在这种情况下，需要被表示的整个表达式能够被有效的表达。

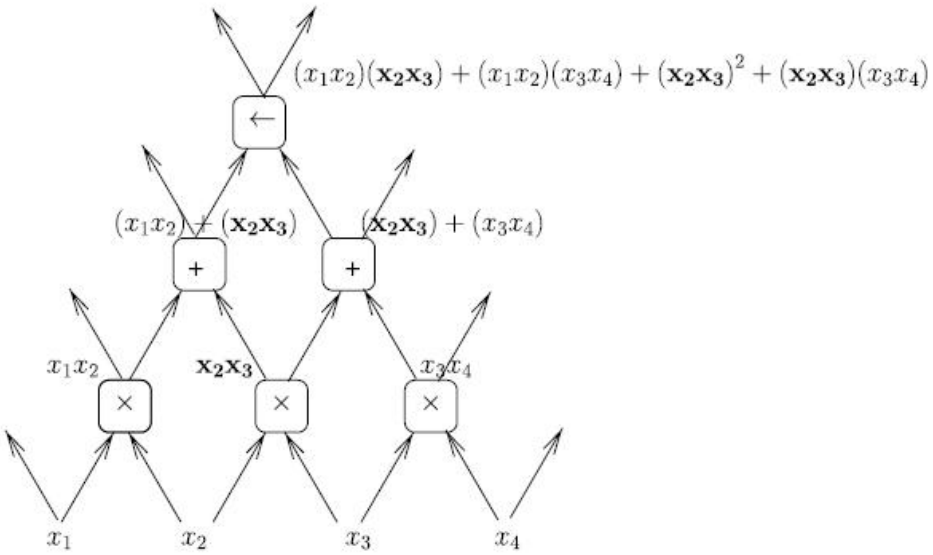


图 2.2 多项式运算（奇数层进行乘积运算，偶数层进行加法运算），它显示了使用深层网络给因式分解带来的优雅性。比如，如果使用一个 2 层的网络来表达上述的多项式，那么第一层的乘积 X_2X_3 将会出现许多次（层数的指数次）。

在文献【19】中，可以找到更多的例子来说明深层网络的表现力和它在人工智能与机器学习中的潜力。在更早一些的文献【191】中，更倾向于从认知角度讨论了深层结构的期望优点。需要注意的是，连接认知心理学家们已经研究这个问题很长时间了，他们设想了多层神经计算方式，每一层都代表了不同程度的抽象，同时每种程度都对应着一个分布式的表达，这些研究可以参考文献[67, 68, 123, 122, 124, 157]。我们这里讨论的现代深层网络方法有很大一部分要归功于他们的研究。为了解释早期认知模型无法解释的现象，这些概念引入到了认知心理学中（然后是计算机科学/人工智能），然后将认知理论和神经的本质联系了起来。

总结一下，大量的计算复杂性成果可以明显的看出来，能够用深度为 k 的深层网络表达的函数，如果要用浅层结构来表达，将会需要大量的节点。由于网络中的每个节点都需要使用样本进行训练，所以从效率的角度来看，采用深层网络是非常重要的。我们将会在下一章更深入的讨论这个概念，我们将会探讨到使用非参数学习算法的浅层结构网络的缺点。

第三章 Local vs Non-Local Generalization

3.1 局域模板匹配的限制

如何让一个学习算法用一个“复杂”的函数紧凑的描述输入，例如，拥有的变体数量多于可利用的训练例子?这个问题同时与问题深度和区域估计器问题相关联。我们认为局域估计对于

学习高度不同的函数是不合适的，尽管他们可能可以高效地用深度架构描述。一个局域输入空间估计器需要大多数情况下利用相邻 X 的训练例子给新输入 X 进行很好的概括。比如，在训练例子中，测试点 x 的 k 最近邻，用来预测 X。局域估计器含蓄的或者明确的对输入空间按区域分区（可能用软方式代替硬方式），并要求用不同的参数或自由程度来说明每个区域的目标函数的可能形状。当由于函数高度不同而需要大量区域时，需要的参数数量也将是巨大的，同时例子的数量需要达到很好的概括。

局域概括问题直接关系到维度灾难，但是我们引用的结果表明对概括产生影响的不是维度，而是我们希望在学习后获得的函数“变体”数量。比如，如果模型描述的函数是 （如决策树），那么有影响的问题就是接近正确目标函数所需要的片段数量。在变体数量与输入维度之间存在着联系：容易为变体数量设计目标函数族在输入维度将呈几何级数的增长，比如以 d 输入的平等函数。

可以认为基于局域模板匹配的架构有两个层次。第一个层次由匹配输入的一组模板组成。一个模板单元将输出一个代表匹配程度的数值。第二层会联合这些数值，为了估计出预期的输出，典型的是以简单的线性联合（一个 OR-like 操作）。我们可以认为这种线性联合是一种插值操作，为了在模板之间的输入空间产生一个答案。

基于局域模型匹配的架构，这个典型的例子是核机器[166]

$$f(\mathbf{x}) = b + \sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i), \qquad (3.1)$$

其中的 b 和 ai 来自第二层，同时在第一层，核函数 K（x,xi）将输入 x 与训练例子 xi 匹配（总和运行在训练集中部分或全部的输入模式）。在上面的等式中，f(x)可以是一个分类器的判别函数，或一个回归预测器的输出。

当 K(x,xi)>p 只在 xi 附近的一些联合区域的 x 是真的时，核函数是局域的（对于一些阈值 p）。这种联合区域的尺寸通常可以有核函数的超参数控制。局域核的一个例子是高斯核 K(x,xi)= e-||x-xi||2/σ2，其中的σ控制 xi 附近区域的大小。我们可以把高斯核看作计算一个软结合，因为它可以被重写成一维状态的乘积：K(u,v)= ∏je-(uj-vj)2/σ2。如果|uj -vj|/σ对所有维数 j 都是小的，那么模式匹配成功并且 K(u,v)是大的。如果|uj -vj|/σ对于一个单独的 j 是大的，那么没有匹配结果并且 K(u,v)是小的。

为人所熟知的核机器不仅包括支持向量机（SVMs）[24,39]和分类器与回归的高斯处理器[203]1，还有分类器，回归和密度估计的经典非参数学习算法，例如 k 最近邻算法，Nadaraya-Watson 或 Parzen 窗密度，回归估计等。接下来，我们讨论流行学习算法，比如像 Isomap 和 LLE 同样可以被看作局域核机器，以及相关的半监督学习算法也基于邻域图的建立（每个例子一个节点和弧线连接相邻例子）。

局域核的核机器利用平滑优先(smoothness prior)生成泛化：假设目标函数是平滑的或者可以很好的近似于一个平滑函数。例如，在监督学习中，如果我们从训练样本(xi,yi)，可以建立一个预测器 f(x)，其在 x 接近 xi 时将输出接近 yi 的值。这种优先要求在输入空间定义一个

近似(proximity)的概念。这种优先是有用的，但是当目标函数在输入空间中高度不同时，这种优化对泛化是不充足的。

泛化核如高斯核的限制，促进了基于知识优先的设计核(designing kernels)的大量研究。然而，如果我们缺少充足的知识用来设计合适的核，我们能否通过学习得到呢？这个问题也促进了很多研究[40，96，196]，深度架构可以看作是这个研究方向可以预见的发展。事实表明高斯进程核机器可以通过深度可信网络改进后，用来学习一个特征空间[160]：经过深度可信网络训练后,核机器的参数初始化一个计算特征矩阵(一个新的数据特征空间)的确定非线性变换(一个多层神经网络)，并且这个变换可以基于梯度优化调整后，最小化高斯进程产生的预测错误。特征空间可以看作数据的学习表示。好的表现会接近共用抽象特性的数据分布变体的相关因素。深度架构的学习算法可以看成是为核机器学习一个好的特征空间。

考虑一个方向 V，其中一个目标函数（学习者应该理想的捕获的）上升和下降（例如，随着 a 的增加，f(x+av)-b 经过 0，变为正，然后又变为负，又变为正），处于不断的“颠簸”。如下[165]，[13，19]显示了对于高斯核这样的核机器，当目标函数不断振荡时，需要学习的必要例子的数量将呈线性增长。它们还表明，对于变化巨大的函数，例如就函数，要实现高斯核机器一定错误率，必要的例子在输入维度中是指数级的。对于一个只依靠优先的目标函数是局域平滑的（比如，高斯核机器），学习在一个方向上有许多符号变化的函数是非常困难的（需要一个大 VC 维与相应大数量的例子）。然而，学习可以通过在其他模式变体中紧凑捕捉的函数进行（一个大致符合的简单的例子是当变体是周期性的，并且函数类包含周期性函数）。

对于高维的复杂任务，决策表面上的复杂性会使用局域刻机器学习变得不切实际。也可以认为如果曲线有很多变体，并且这些变体在底层规律上彼此不相关，那么没有学习算法能够比输入空间的局域估计器完成得好多少。然而，寻找变体更加紧凑的表示还是值得的，因为如果能够发现一种，它可能可以导致更好的泛化，尤其当变体在训练集中为被发现时。当然前提时我们能够在目标函数中捕捉到深层的规律；我们期待这样的属性能在 AI 任务中出现。

估计器在输入空间中是居域性的，这不仅在监督学习算法中发现，如上面讨论过的，同时还在无监督学习和半监督学习算法中发现，例如线性嵌入[155]，Isomap[185]，核主成分分析[168],(或称作 kernel PCA)拉普拉斯特征映射[10],Manifold Charting[26],谱聚类算法[199],和基于内核的非参数半监督算法[9，44，209，210]。大多数无监督和半监督算法依赖邻接图：图中每个例子一个结点，邻接结点之间用圆弧连接。用这些算法，我们可以得到一个几何的直观表现，知道这些算法的处理过程，同时知道估计器如何妨碍他们。图 3.1 展示了流行学习中的一个例子。在这里再一次声明，我们发现为了覆盖学习过程中许多可能的变体，需要与覆盖变体数量成比例的例子[21]。

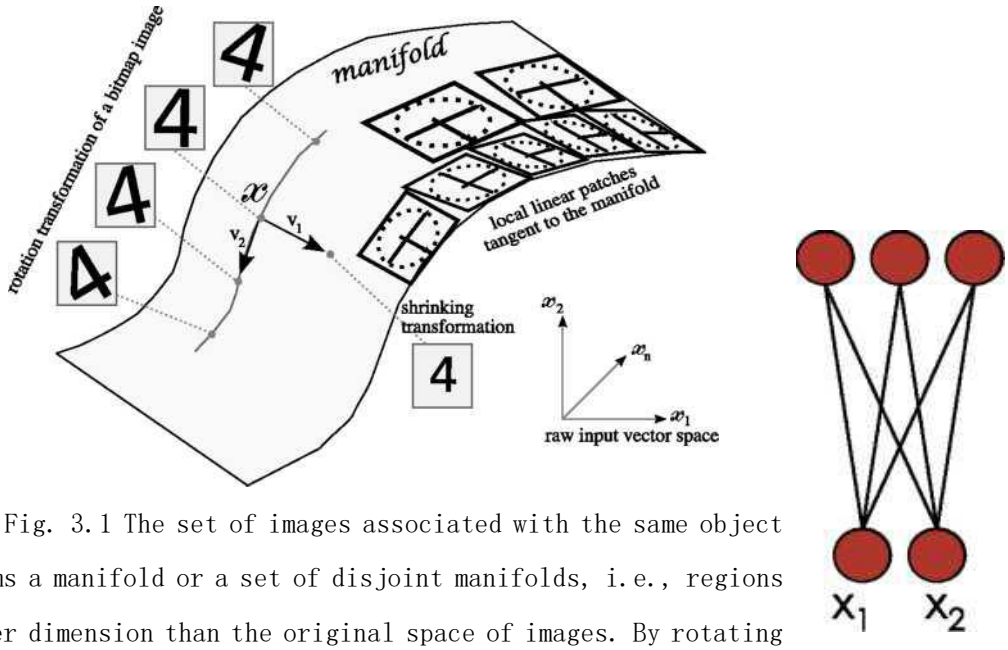


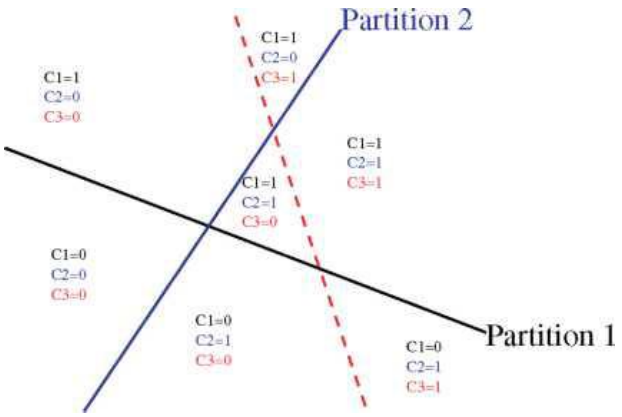
Fig. 3.1 The set of images associated with the same object class forms a manifold or a set of disjoint manifolds, i.e., regions of lower dimension than the original space of images. By rotating or shrinking, e.g., a digit 4, we get other images of the same class, i.e., on the same manifold. Since the manifold is locally smooth, it can in principle be approximated locally by linear patches, each being tangent to the manifold. Unfortunately, if the manifold is highly curved, the patches are required to be small, and exponentially many might be needed with respect to manifold dimension. Graph graciously provided by Pascal Vincent.（图表介绍译者未译，请审校老师补充一下）

最后让我们考虑一下基于邻接图的半监督学习算法这个例子[9,44,209,210]。这些算法以上下文标签对邻接图进行分区。可以观察到邻接图带上下文标签的分区数量不会比已标记例子的数量[13]。因此我们需要至少与分类变体相当数量的标记例子。如果感兴趣的决策面有非常大的变体数量，这将是令人望而却步的。

决策树[28]是最好的学习算法之一。因为它们能够只关注输入变量中的具体子集，在初步印象中，它们看起来是非局域的。然而，它们从依赖于输入空间分区，并且为每个分区使用单独参数的意义上来说也是局域估计器[14]，每个分区与决策树的一个叶子关联。这意味着它们也会遭遇上述中讨论过的关于其他非监督学习算法的限制：它们需要至少与目标函数中感兴趣的变体相同数量的训练例子，而且它们无法泛化到训练集未覆盖到的新变体。理论分析[14]显示函数具体类要达到一个给定的错误率，必要的训练例子数量在输入维中呈现指数级的。这种

分析建立在先前在计算复杂性中利用过的想法[41]。这些结果与之前的实证结果同时也说明决策书的泛化性能表现在目标函数中的变体数量增加时会降低。

一系列树（像 boosted trees[52]，和 forests[80，27]）比单棵树处理能力更强。它们向



架构添加了第三级，以此来允许模型在指数级的参数中区分一定数量的分区[14]。如图 3.2 所示，它们用树林中所有树的输出生成了一个分布表现（一种将在后续章节 3.2 中进行讨论）。一个系列中的每棵树能与树的输入例子中分离标志标记的叶子\分区。输入模式与梅克树关联的叶子节点的标记生成一个数组，这个数组对输入模式是一个非常丰富的描述：它能够呈现大量的可能模式因为 n 棵树相关的叶子区域的交叉点数量是 n 的指数级。

3.2 学习分布的模型表示

在 1.2 节中，我们讨论过了深度结构需要在系统层级的界面上选择各种模型表征。

同时我们也介绍了局部表征（在前一节中进行了深入的阐释），分布式表征和稀疏分布式表征这样的基本概念。分布式表征在机器学习和神经网络中是一个很老的概念，它可能对解决维度灾难和局部泛化限制有一定的帮助。一个整数集构成的 cartoon 局部表征是一个拥有单独 1 和 N-1 的 0 的 N 比特的向量 r(i)，也就是说第 j 个元素拥有 rj(i) = 1i=j，叫做 单热点表征 [ref:http://www.zhihu.com/question/21714667]。一个词向量（参考同上）对于相同的整数可以表示为一个 log2N 比特的向量，是一种更紧凑的表示方法。对于这两种有相同数量的立体基阵，词向量能以指数量级的方式相比于局部表征(local representation)的方式更紧凑。引入系数矩阵这个概念（也就是让很多单元为 0）可以允许展示在 local（比如大量的稀疏值）和非稀疏值之间的词向量。神经元在大脑皮质的链接方式被认为拥有分散式的和稀疏的表现方式[139]，在任意时间段内大概有 1-4%的神经元是激活状态的[5, 113]。

在实践中，我们常常充分利用连续值的模型表示，这样可以增加他们的表现力。对于连续值的局部表征的一个例子就是第 i 个元素依据输入到原型或者区域中心的距离的多重复合，就如同 3.1 节中展示的高斯核分布一样。在分布式表达中，输入模式由一系列不互斥甚至有些统计独立的特种构成。举例来说，聚类算法不能用来建立词向量，因为各个簇本质上来说是相互

独立的然而，独立成分分析(ICA)[11, 142]和主成分分析(PCA)[82]可以用来建立一个词向量。

考虑一个独立的词向量 r(x)对于一个输入模式 x，这里 ri(x) ∈ {1,...M}, i ∈ {1,...,N}.

每个 ri(x)可以被认为是一个 x 类别分类到 M 类中。正如在图 3.2 中（M=2）所展示的，每个 ri(x) 在 M 个区间中分割成 x-空间，但是不通的分割可能会由于不通的立体矩阵 r(x)而潜在的在 x-空间中造成潜在可能的指数量级的交集。注意当展示一个特殊的输入分布时，一些立体基阵是不可行的，原因是他们不能同时成立。距离来说，在语言模型中，一个单词的局部表征可能被在字典里的索引编码它的身份特征，或者相当于一个拥有字典大小数量的条目的单热点编码（one-hot code）。

另一方面，一个词向量可以通过连接一个语法特征(比如：它在说话中的各部分的分布)的向量指标来表示一个词。形态学特征（所拥有的后缀前缀），语义特征（是否是某种动物的名字等）。就像在聚类中，我们构造独立的类，但是潜在的合并类的数目非常大：我们得到了一个我们称之为的多重簇，同时它又和有交叠的簇和局部的聚类元素非常类似，也就是说就整个聚类数据[65, 66]而言，簇之间不是相互独立的。然而聚类形成单个类，通常而言就输入信息而言都会伴随着严重的信息损失，一个多重聚类提供了一套输入空间的独立分区。多重聚类鉴别每一个分区所用到的例子的输入的样例取决于输入模式的描述形式，而这种输入模式可以非常丰富，有可能并没有损失信息。元组的符号规格化（输入所属的每个分类的区间）可以可以看做输入转换到一个新的空间，这样数据的统计结构和要素的多元化问题就可以得到解决。就如前面章节所讨论的，这和聚类树可以作为一个整体展示的多种 x-空间的分类相对应。

这也是我们希望深度结构可以捕捉到的，但是，分布具有多重水平，高层级就会更加抽象，输入空间的表示也会更加复杂。

在监督学习领域内，多层神经网络[157, 156]和非监督学习，玻尔兹曼机[1]已经引入了在隐藏层内以内部分布为模型表示的学习目标。和上面语言学的例子不同，它的目的是让学习算法发现构成分布式表达的特征。在多层神经网络中，有对于一个的隐藏层，它在每个层都有一个模型表示。

学习多重水平的模型表示分布会有一个模型训练问题的挑战，我们在下一节中讨论。

第四章 深层神经网络

4.1 多层神经网络

以下是一系列具有代表性的多层神经网络[156]的公式。如(图 4.1)所示，对于网络的每一层，利用前一层h^{k-1}层的输出作为输入，在第 k 层经过计算后得到输出向量h^k，起始输入为 $x = h^0$。

$$\mathrm{h^k = \tanh\left(b^k + W^k h^{k-1}\right)} \qquad (4.1)$$

其中变量$\mathrm{b^k}$是由偏置项组成的向量，$\mathrm{W^k}$是权值矩阵。公式中 $tanh$ 是逐元素进行的，它也可以替换为 $sigm(u) = 1/(1+e^{-u}) = \frac{1}{2}(tanh(u)+1)$或者其他饱和非线性环节替代。最后一层的输出$h^\ell$被用来进行结果预测，作为监督学习，我们将它和训练目标 y 一起放入一个损失函数 $L(h^\ell, y)$，通常情况下令$b^\ell + W^\ell h^{\ell-1}$为凸函数。输出层可以使用另一种非线性环节，比如 Softmax

$$h_i^\ell = \frac{e^{b_i^\ell + W_i^\ell h^{\ell-1}}}{\sum_j e^{b_j^\ell + W_j^\ell h^{\ell-1}}} \qquad (4.2)$$

其中，W_i^ℓ表示W^ℓ的第 i 行，h_i^ℓ中元素为正并且$\sum_i h_i^\ell = 1$。Softmax 的输出项h_i^ℓ可以用来预测 $P(Y=i|x)$，其中 Y 是输入模式 x 对应的输出类别。在此种情况下，我们经常使用负条件概率对数似然函数 $L(h^\ell, y) = -\log P(Y=y|x) = -\log h_y^\ell$作为损失函数，并且使$(x,y)$对应的期望值最小。

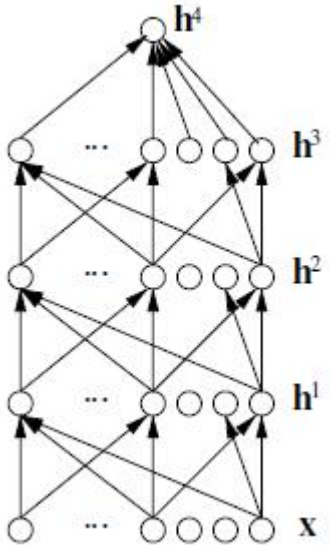


图 4.1. 多层神经网络，通常在监督学习中使用，用来预测或者分类。对于一系列由神经元组成的层中的每一层，都是由一个仿射操作和一个非线性操作组成。通过前馈传播，可以计算出确定性的转移。从输入 x，到隐层h^k，再到输出层h^ℓ，之后通过与标记好的 y 比较，从而去达到试损失函数 $L(h^\ell, y)$最小。

4.2 训练深度神经网络的挑战

在意识到深度结构必要性且深度结构为非局部估计量之后，我们开始分析训练深度结构这个难题。实验结果显示训练深层结构会比训练浅层结构更加困难[17,50]。

由于通常使用标准的参数随机初始化产生较大的训练误差和推广误差[17]，直到 2006 年深度结构才开始在机器学习领域开始被广泛讨论。4.5 节曾讨论过深度卷积神经网络[104, 101, 175, 153]被发现更容易训练，但是其原因尚未被解释清楚。

[注 1]：我们叫它们明显的局部最小值在某种意义上表示梯度下降的学习轨迹停止在那，这并没有排除更强大的优化器可以找到比这好的多的解。

许多未被报道的负面发现和[17, 50]中的实验结果显示基于梯度训练的深度有监督的多层神经网络（从随机初始化开始）会遇到“明显的局部最小值和平稳值”[注 1]，并且随着结构变深要获得更好的推广能力变得更加困难。从随机初始化开始，结构更深的神经网络通常会比只有一或两层隐层的网络对应更差的解[17,98]。这个现象是因为虽然 k+1 层网络能很容易表示一个 k 层网络所能表示内容（没有增加明显的容量），但是反过来是不对的。然而，[73] 当通过无监督的学习算法从第一层（直接从观测值 x 作为输入）开始逐层训练预训练每一次层发现结果会变得好的多。初步实验对每层使用受限玻尔兹曼机生成式模型[73]，并且用自编码器来训练每层也能产生类似的结果[17,153,195]。这些文章的大多是利用贪婪的逐层无监督学习的思路（在下一节有详细的描述）：首先利用无监督的学习算法训练较低的层（例如受限玻尔兹曼机中的一层或者某个自编码器），对神经网络的第一层的初始化参数集给予改进。然后使用第一层的输出作为另一层的输入（一种原始输入的新的表示），并且相似的利用无监督的学习算法初始化该层。在这样初始化数层之后，整个神经网络可以像往常一样根据某种监督训练准则进行精调。无监督的预训练相对随机初始化在多个统计比较里[17,50,98,99]被清晰的证明。什么原理能解释文献中提到的使用无监督预训练改进分类错误率的原因？有一个线索可能会帮助解释深度结构训练算法背后成功的原因，它既不是利用受限玻尔兹曼机也不使自编码器[131,202]。这些算法和基于受限玻尔兹曼机和自编码器的训练算法的共性是都是基于逐层无监督准则，比如：在每一层注入一个无监督的训练信号可能会帮助指导该层参数在参数空间朝更优的区域移动。在[202]，神经网络用成对的样本（x, x˜）来训练，它们要么是“邻居”（同类的）要么不是。设想 $h^k(x)$为模型中 x 的一个 k 层表示。每层一个局部的训练准则会被定义，它使中间表示 $h^k(x)$和 $h^k(x˜)$根据 x 和 x˜是否为邻居变得彼此更加靠近或彼此更加远离（例如输入空间的 k 近邻）。类似的准则已经被成功用在使用无监督流型学习算法[59]学习低维嵌入，但是在这里[202]被用在神经网络的一层或者更多的中间层。照着慢特征分析的思路，[23,131,204]利用高层抽象的时间一致性来提供给中间层的无监督的指导：连续帧可能包含同样的物体。

很明显，至少对所研究的任务类型测试误差能利用这些技术被极大的改进，但是为什么呢？要问的一个基本问题是是否这个改进本质是因为有更好的优化或者更好的正则化技术。正如下述讨论，答案可能不会适用优化和正则的常规定义。

在一些实验中[17,98]，很明显即使使用没有无监督预训练的深度神经网络也能使训练分类误差降低到 0，指明了更多在正则化效果方面而不是优化效果。[50]中的实验也在相同的方向中给出实例：对于相同训练误差（训练中不同的点），带有无监督预训练的测试误差会系统性的更低。在[50]中讨论过，无监督的预训练可以被看成一种正则化形式（和先验）：无监督的预训练

相当于一种参数空间某个区域可行解的一个约束。这个约束迫使解靠近无监督训练（的结果）[注2]，即希望其对应输入空间显著统计结构的解。另一方面，其他实验[17,98]显示对较低层欠佳的调节可以对没有预训练得到更差结果进行解释：当顶层隐层被约束（强制设为较小值）带有随机初始化深度网路（没有无监督预训练）在训练集和测试集都表现的很差，比预训练的网络差的多。在之前提到的训练误差可以到零实验中，隐层单元的个数（超参数）被允许必要时足够大（在验证集里使误差最小化）。在[17,98]中建议的解释性假设为当最高层隐层是无约束的，使用较低层计算表示作为输入，最高的两层（对应一个常规的一个隐层的神经网络）足以拟合训练集，即使这个表示是欠佳的。另一方面，使用无监督的预训练，较低层被更好的优化，且较小的顶层足以获得一个低的训练误差，还产生更好的泛化。在[50]中的描述的其他实验如随机参数初始化，较低层（靠近输入层）训练欠佳也和这个解释一致。这些实验显示无监督预训练的作用更多的是表现在深度结构的较低层。

[注 2]梯度下降过程中同一个吸引盆。

从经验我们知道一个两层的网络（一个隐层）大体上能被很好的训练，并且从深度网络的最高两层这个观点来看，它们组成了一个浅层网络结构，它的输入是较低层的输出。优化深度神经网络的最后一层是一个训练准则中广泛使用的凸优化问题。优化最后两层，虽然是非凸的，但是已知比优化深度网络会容易很多（事实上，当隐层节点的个数变成无穷，一个两层网络的训练准则可以被映射成凸的[18]）。

如果在最高层隐层有足够的隐节点（也就是有足够的容量），即使当较低层没有被恰当的训练，训练误差也能被降到很低（只要它们保存了原始输入的大部分信息），然而这可能会比浅层神经网络带来更糟糕的泛化能力。当训练误差是低的同时测试误差是高的，我们通常称这个现象为过拟合。由于无监督的预训练会带来测试误差的下降，那会把它当成一种数据依赖的正则项。其它有力证据被显示无监督的预训练表现的就像一个正则项[50]：尤其是当没有足够容量的时候，无监督的预训练会倾向于对对泛化造成负面影响，并且当训练集容量小时（例如 MNIST，不到十万个样本），即使无监督的预训练会改进测试误差，它也会倾向于产生更大的训练误差。

在另一方面，对于大得多的训练集，有更好初始化的较低层的隐层，当使用无监督的预训练时训练误差和泛化误差都会被显著降低（如图 4.2 和下面的讨论所示）。我们推测在一个训练良好的深度神经网络里，隐层形成了数据的一个好的表示，这有助于做好的预测。当较低层被不良的初始化，这些确定的和连续的表示大致保存了输入的大部分信息，但是这些表示可能会扰乱输入并且产生伤害而不是有助于完成良好泛化的分类。

根据这个推测，虽然用如高斯过程或支持向量机之类的凸机替换深度神经网络的最高两层

能产生一些改进[19]，尤其是在训练误差，但是较低层如果没有被充分优化，例如没有发现原始输入的一个好的表示，在泛化方面不会有太多的帮助。

因此，一个推断是通过允许对深度结构的较低层进行更好的调优，无监督的预训练可以帮助泛化。虽然可以只通过拟合训练样本开发顶层能力来降低训练误差，但是当所有层都被恰当的调节之后会获得更好的推广能力。另一种更优泛化来源于一个正则化的形式：带有无监督的预训练，较低层被约束来捕获输入分布的正则化。细想随机输入队(X,Y)。这样的正则化类似半监督学习[100]中无标注样本的假设效果或者 P(X,Y)（生成模型）相对 P(Y|X)（判别模型）[118,137]的极大似然带来的正则化效果。如果真实的 P(X)和 P(Y|X)是 X 的不相关函数（例如是独立挑选的，以致关于其中一个的学习不会通知我们另一个），那么 P(X)的无监督学习也不会帮助 P(Y|X)的学习。但是如果它们是相关的[注 3]，并且如果在估计 P(X)和 P(Y|X)用到的参数是相同的[注 4]，那么每对(X,Y)在 P(Y|X)上带来的信息不仅是以一种通常的方式还会通过 P(X)。例如，在深度信念网，两者的分布在本质上共享着同样的参数，因此在估计 P(Y|X)中涉及的参数会从一种数据依赖的正则化受益：在某种程度上它们必须跟 P(X)和 P(Y|X)达成一致。

[注 3]例如，MNIST 数字图片组成了分离的非常开的簇群，尤其在学习好的表示的时候，甚至是无监督的[192]，以致于在看到任何标注之前就可以合理的猜测决策面了。

[注 4]例如，多层神经网络所有的较低层估计的 P(Y|X)都能从一个深度信念网络估计的 P(X)来进行参数初始化。

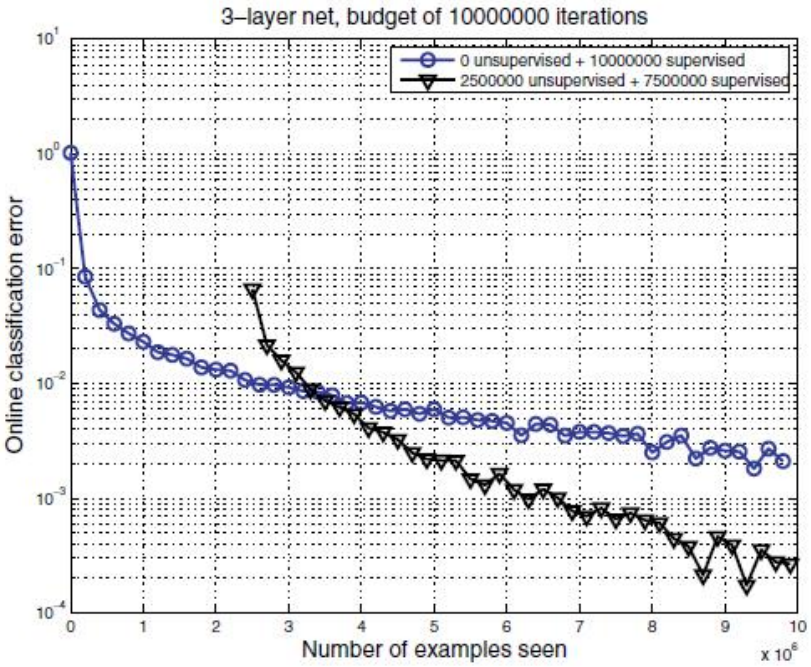


图 4.2 深度结构使用一千万个数字图像样本在线训练，使用预训练（三角形）或不使用预训练（圆圈）。显示的分类误差（纵轴，对数尺度）是在下一个一千个样本上在线计算的，从开头看

起相对样本的数量进行绘制。前两百五十万个样本用于无监督的预训练（一堆去噪自编码器）。在尾部的震荡是因为错误率太接近零，使采样方差在对数尺度显得大。然而，有一个非常巨大的训练集的话，正则化作用会消失，有人会发现没有预训练，训练会收敛到一个更差的明显的局部最小值：无监督的预训练帮助找到在线误差的一个更优的最小值。实验是 Dumitru Erhan 做的。

让我们回到无监督预训练带来更好结果的最优化解释和正则化解释（这个话题）。要注意在这里当用到最优化这个词，应该如何变得严谨。从一般意义来讲，在优化方面我们没有难点。事实上，从整个网络的观点来看（也）没有难点，因为主要依靠头两层就能使训练误差变得很低。但是，如果有人关心调节较低层（参数）的问题（在保持倒数第二层（即隐层最高层）隐节点个数或者最高两层权重幅值较小时），那么他可能是在讨论优化难点。一种调和最优化和正则化观点的方法就是考虑真实的网络设置（在这里样本来源于一个无限的流，并且通过训练集不会循环回去）。那样的话，在线的梯度下降就是在实现泛化误差的随机优化。如果无监督预训练的效果纯粹是正则化中的一种，那么有人会期望在一个虚拟的无限训练集，带预训练和不带预训练的在线误差会收敛到同样的等级。另一方面，如果这个呈现在这里的解释性假设是正确的，我们会期望无监督的预训练在网络设置方面会带来清晰的优势。要探索那个问题，我们使用的是“无限的 MNIST”数据集[120]，即一个虚拟的类 MNIST 数字图像无限流（通过随机平移，旋转，尺度变换等，在[176]定义）。如图 4.2 所示，一个带三个隐层的神经网络，当它被预训练时在线训练会收敛到一个明显更低的误差（见 7.2 节，堆砌的去噪自编码器）。这幅图显示随着在线误差的优化（在下一千个样本上），一个无偏的泛化误差的蒙特卡罗估计。头两百五十万个更新用于无监督的预训练。这幅图强烈显示无监督的预训练收敛到一个较低的误差，即它表现的不仅仅是一个正则子还是找到优化准则的更好的最小值。事实上，这并不和正则化假设冲突：由于局部最小值，即使样本数趋向无穷，正则化效果（仍然）持续。这个解释的另一面是一旦动力被困在一些明显的局部最小值处，更多的标注样本不会提供更多的新信息。

要解释更低层会更难优化，上述线索暗示反向传播到较低层的梯度可能不足以把参数移动到好的解对应的区域。根据那个假设，关于较低层参数的优化被卡在一个欠佳的明显的局部最小值或者稳定水平（即小的梯度）。因为基于梯度的顶层训练方法表现的相当好，因此它将意味随着我们向较低层移回，梯度在所需的参数变化方面变得含更少信息量，或者误差函数对梯度下降来逃离这些局部最小值方面变得很病态。在 4.5 节争论过，这可能一个观察报告相关，深度卷积神经网络是较容易训练的，也许是因为它们在每一层有一个非常特别的稀疏链接。深度网络利用梯度的难点和通过长序列训练再现神经网络也可能会有个关联，在[22,81,119]中有分析。一个再现神经网络可能是适时展开的，通过考虑不同时间每个神经元的输出作为不同变量，使展开的网络通过一段长的输入序列变成一个非常深的结构。在再现神经网络，训练的困难可以归结到通过许多非线性（变换）传播的消失的（或有时为爆炸的）梯度。如果是再现神经网络，由于梯度中短时（即计算展开图中更短的路径）和长时（和那副图中更长路径相关）之间的一

个误匹配，还会有一个额外的难点。

4.3 Unsupervised Learning for Deep Architectures

我们已经在上文中看到，到目前为止，分层的无监督学习是所有成功的深度学习算法的关键部分。 如果输出层的评价函数的梯度随着反向传播作用效果变得越来越小，那么可以认为单层的无监督评估函数可以被用来往更优的方向调整它的参数。我们也有理由期待利用单层的学习算法去形成一种形式可以挖掘出每层输入里统计上的规律。PCA（主成分分析法）和因素数与信号量相同的 ICA（独立成分分析法）变形看起来并不合适，因为它们并不能处理一种叫做“过完备”的情形，即一层的输出数大于它的输入数。这里建议去看一下 ICA 的扩展去处理过完备的情形 [78, 87, 115, 184]，还有与 PCA、ICA 相关的算法，如 auto-encoders 和 RBMs，它们都能应用于过完备的情形。事实上，在多层系统的环境下进行单层无监督学习算法的实验也证实了这一想法[17, 73, 153]。值得一提的是，叠加线性投影（如两层的 PCA）依然是线性变换，也就是说无法形成深度结构。

无监督学习可以减少对有监督学习评估函数梯度给出的不可靠更新方向的依赖，除了这一动机，我们还提出另一个在深度结构里每层都使用无监督学习的动机——这可能是一种自然分解一个问题到不同抽象层的子问题的一种方式。我们知道无监督学习算法可以提取出有关输入分布的显著信息。这些信息可以以分布的形式获取到，也就是一系列能引起输入变化的显著因素的特征。一层的无监督学习算法可以提出这样的显著特征，但是由于一层的容量限制，第一层提取出的特征只能被看作是低级特征。很容易想到第二层学习基于同样的原则，将第一层学习到的特征做为输入会得到更高级一点的特征。这样，能够想像，用于描绘输入的更高级的抽象最终会出现。注意在此过程中，如何使所有的学习在每层都能保证局部性，因此当我们尝试去求解一个全局评估函数最优时，同时要处理梯度传播可能会损害深度神经网络基于梯度学习的问题。在接下来的第三章，我们将讨论深度生成结构并且正式介绍深度信念网络。

4.4 深层次生成结构

除了对职前培训一个准确的预测者有用外，在没有监管下的深层次学习对学会分配和从中产生样品也是有意义的。生成的模型常常被看作是图解模型[91]：这些模型像图片那样是可视化的，模型里面的节点代表随即变量，弧线表明了随机变量之间存在的从属类型。所有随机变量的联合分布可以以图片里只含有一个节点和它邻近的节点的乘积来表示。那些直接的弧线(定义为子根)，考虑到父根一个子节点通常是不随它变化的。在图示的模型中，可以观察到一些随机变量，还有一些观察不到的量(定义为阴变量)。乙状结肠信念网络生成多层神经网络的提出和研

究在 2006 年之前并且用变分近似法进行了训练[42，72，164，189]。在乙状结肠信念网络里，每层里面的基元(通常是二进制随机变量)和上层给出的基元的值都是独立的，如图 4.3 所示。这些边界条件分布的典型参数化(在常见的神经网络里自上而下分布而不是往上)和神经元激活方程非常类似(4.1):

$$P(\mathbf{h}_i^k = 1|\mathbf{h}^{k+1}) = \mathrm{sigm}(\mathbf{b}_i^k + \sum W_{i,j}^{k+1}\mathbf{h}_j^{k+1}) \qquad (4.3)$$

其中，hik 是在 k 层中隐藏的节点 i 的二次激活，hk 是矢量(h1k,h2k,...)。我们把输入的矢量用 x=h0 表示。注：符号 P(...)一直代表和我们模型相关的概率分布,然而 P^是受训练的分布(受训练的集合的经验分布，或是我们训练样本的产生分布)。在输入空间的最底层产生了一个矢量 x，我们希望这个模型能给出训练数据高的概率。考虑到多个层次的网络，因此生成的模型可以按下式分解：

$$P(\mathbf{x}, \mathbf{h}^1, \ldots, \mathbf{h}^\ell) = P(\mathbf{h}^\ell)\left(\prod_{k=1}^{\ell-1} P(\mathbf{h}^k|\mathbf{h}^{k+1})\right) P(\mathbf{x}|\mathbf{h}^1) \qquad (4.4)$$

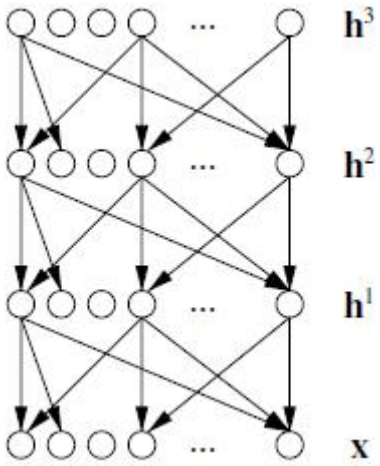


图 4.3 一个生成的多层神经网络的例子，这里一个乙状的信念网络，被描绘成一个直接的图解模型(每个随机变量用一个节点表示，有方向的线表示直接相关)。观察的数据是 x，在 k 层隐藏因子是矢量 hk 的分量。顶层 h3 优先分解。

得到 P(x)，但是实际中除了得到这些微小模型外处理起来是非常困难的。在乙状结肠信念网络里，顶层之前的 P(hl)通常选择因式分解，也就是说，这样处理会非常简单：P(hl)= ∏iP(hli)，并且对于每个 P(hl=1)的二次激活的基元单一的伯努利参数也是必要的。

深度信息网络类似于乙状结肠信念网络，但是对于前两层的参数化有少许不同，如图 4.4 所示：

$$P(\mathbf{x}, \mathbf{h}^1, \ldots, \mathbf{h}^\ell) = P(\mathbf{h}^{\ell-1}, \mathbf{h}^\ell)\left(\prod_{k=1}^{\ell-2} P(\mathbf{h}^k|\mathbf{h}^{k+1})\right) P(\mathbf{x}|\mathbf{h}^1). \qquad (4.5)$$

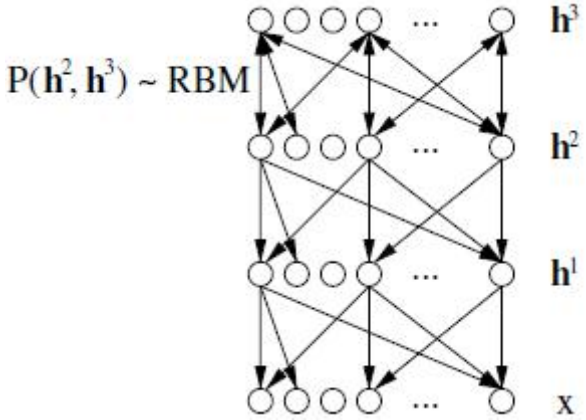


图 4.4 一个深度信念网络的观察矢量 x 和隐藏层 h1，h2 和 h3。符号的含义和图 4.3 中所示的一致。深度信念网络的结构除了最上面两层外和单层信念网络的结构近似。它的最上面两层 P(h2，h3)的节点受波尔兹曼机理(RBM)的限制，而不是对 P(h3)进行因式分解。这个模型是混合的，在最上面两层之间用双箭头连接，因为 RBM 是间接的而不是直接的模型。

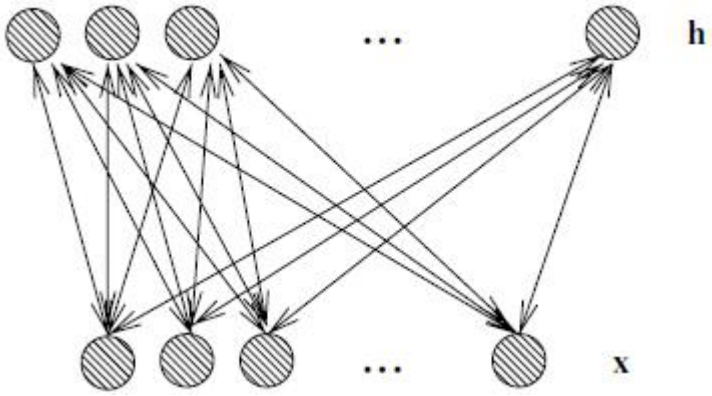


图 4.5 一个受波尔兹曼原理限制的间接的图解模型。在同层之间没有联系，只有输入基元(或输入变量)xj 和隐藏基元 hi 有联系，使得条件 P(h｜x)和 P(x｜h)方便因式分解。

最上面两层的节点分布是受波尔兹曼限制的分布:

$$P(\mathbf{h}^{\ell-1},\mathbf{h}^{\ell}) \propto e^{\mathbf{b}'\mathbf{h}^{\ell-1}+\mathbf{c}'\mathbf{h}^{\ell}+\mathbf{h}^{\ell\prime}W\mathbf{h}^{\ell-1}} \qquad (4.6)$$

正如如图 4.5 所示，而且它的推理和训练算法分别在 5.3 和 5.4 章节中有更详细的描述。这种来自于乙状信念网络和来自于利用粗略地一次训练一层这个概念的算法的微小差别正逐步对原始输入数据的后续反应 P(hk｜x)建立更加抽象的表示。对深度结构的受波尔兹曼限制模型和粗略的 layer-wise 训练算法的详细描述在接下来的 5 和 6 部分。

4.5 卷积神经网络

尽管深度监督式神经网络在未采用无监督式的预训练的时候训练起来很困难，但卷积神经网络是一个显著的例外。卷积网络是由视觉系统的结构激发而来，尤其是【83】里面提出的模

型。基于神经元的局部连接性质和图像的分级组织提出的最早的计算模型是 Fukushima 的神经认知机【54】。正如他所提到的，在相同的参数被用于前一层不同位置的神经元时可以得到一种翻译不变性。而后，LeCum 和他的同事根据这种思想，通过误差梯度的方法设计并训练了他们的卷积网络，得到了当时模式识别领域最好的性能【101，104】。当前对视觉系统的生理功能的认识与在卷积网络中所发现的处理方式一致【173】，至少在快速识别对象上是一样的，比如，并不需要自顶向下的反馈连接。当下，基于卷积神经网络的模式识别系统是性能最优的系统之一。这个在手写字体识别中可以明显体会到【101】，而手写字体识别则在很多年内被当作机器学习基准 [5]。

（5. 可能是很多年？当然这个领域向更多更伟大的基准发展必然是好的，例如那些在【108， 99】中所介绍的。）

考虑到我们这里讨论的深度结构的训练，卷积神经网络的例子【101，104，153，175】是非常有趣的，因为他们一般有 5，6 或者 7 层，层数的增多在随机初始化的时候使得完全连接的神经网络几乎没办法训练。那么到底是哪种结构特性使得卷积神经网络在视觉领域具有很好的泛化性能呢？

LeCun 的卷积神经网络具有两种类型的层结构：卷积层和降抽样层。每一层都有一种拓扑图结构，例如，每个神经元都对应着输入图像上一个固定的坐标，伴随一个感受野（输入图像上影响神经元活动的区域）。在每层的每个位置处，都有很多不同的神经元，每个神经元都有它自己的输入权值，连接着上层中一个立方体区域的神经元。不同位置的神经元都具有相同的一组权值，但对应着不同位置的立方体区域。

一个未被证明的假设是这些小的扇入神经元（每个神经元具有较少的输入神经元）帮助梯度能够传播多层而未被扩散，从而避免训练失败。请注意单单这个原因并不足以解释卷积神经网络的成功，因为随机的稀疏连接并不能得到很好的深度神经网络。然而，这种扇入的一个后果是让梯度沿着多个路径传播，导致其过于扩散，例如，对输出误差的奖励或惩罚会分布的非常宽或窄。另一个假设（未必排斥了第一个）是这种分级的局部连接结构是一种很好的先验知识，对视觉任务非常适合，并且整个网络的参数集在一个很有利的区域（未连接位置的权值为零）使得基于梯度的优化过程非常实用。事实上，即使用随机权值初始化第一层，对应的卷积神经网络仍然表现很好【151】，优于一个训练好的完全连接的神经网络但差于一个完全优化的卷积神经网络。

近来，这种卷积结构被引入到 RBMs【45】和 DBNs【111】中。【111】中一个重要的革新在于设计了一个生成版本的降抽样单元，在其报道的实验中具有漂亮的表现，不仅对 MNIST 数字而且对 Caltech-101 对象识别基准都获得了目前最优的结果。另外，每个层次得到的可视化特征（最可能被隐藏层表示的模式）清楚地明确了多层次比较的观念，这使得深度结构成为了首要的选择，即以一种自然的方式从边到物体的部件再到物体的认知过程。

4.6 自编码器

以下将要介绍的一些深度体系结构（深度信念网络和栈式自编码器）被作为一些特殊类型神经网络的组成部分或者验证装置：自编码器，也被称作为自联合器，或者“空竹网络”[25，79，90，156，172]。在章节 5.4.3，我们也讨论过自编码器和受限玻尔兹曼机(RBM)之间的联系，即，自编码器通过利用对比散度(CD)训练的方法来近似受限玻尔兹曼机的训练。犹豫对自编码器的训练比对受限玻尔兹曼机的训练要容易，所以它们经常被用做训练深度网络的模块，即每一层都附带一个自编码器，并且每一层可以分别训练[17，99，153，195]。

自编码器的目的是训练出一种编码方法使得输入 x 可以被编码得到某种表述 $c(x)$，从而输入可以通过这种表述被重构出来。因此，自编码器的输出就是输入本身。假设用单线性隐层，并且使用均方差作为准则，那么 k 个隐藏节点经过学习后将输入映射至数据的前 k 个主成分上[25]。如果隐层是非线性的，那么自编码器以不同于主成分分析(PCA)的方式工作，可以获取输入数据在多种形式下的分布信息[90]。我们通常将均方差泛化为重构信息的最小负对数似然函数，对于编码过的信息 $c(x)$:

$$RE = -\log P(x|c(x)) \qquad (4.7)$$

如果 $x|c(x)$服从高斯分布，这个问题又回到了我们熟悉的方差问题。如果输入x_i是二值的，或者服从二项分布，则损失函数演变为

$$-\log P(x|c(x)) = -\sum_i x_i \log f_i(c(x)) + (1 - x_i) \log (1 - f_i(c(x))) \qquad (4.8)$$

其中 $f(\cdot)$被称作解码器，$f(c(x))$是这个网络产生的重构信息，比如在此时应是一个经过 sigmoid 函数生成，由$(0, 1)$之间数组成的数组。我们期望编码后得到的 $c(x)$是一种对于数据中主要变化因子的分散式的表述：原因在于，我们可以将编码 $c(x)$看做一种对于输入 x 的有损压缩过程，它无法做到对于所有 x 都进行很好的压缩方法(存在一些小的损耗)，所以我们不指望它胜任于任意样本，而是通过学习过程驱使它成为一个只针对训练样本的很好的压缩方法，并且希望对于其他输入也可以做到同样好（这也正是自编码器泛化的意义）。

关于这种方法，有一个严重的问题在于，如果不加任何约束，一个 n 维输入，大于 n 维编码的自编码器，有可能只会学到恒等函数，也就是说很多编码过程是无用的（只是单纯复制输入数据）。令人惊奇的是，在[17]中的实验显示，如果利用随机梯度下降法训练一个隐层节点数多于输入维度的非线性自编码器（也称做过完备），能够获得有价值的表述（相对于用这种表述作为输入，测量分类误差而言）。对于此现象一种简单地解释是基于观察发现，提前停止训练的随机梯度下降与对参数使用 l2 正则化作用相似[211, 36]。为了得到对于连续输入的完美重构，一个单隐层非线性自编码器需要非常小的第一层权值（为了给隐层节点的线性体系引入非线性）和非常大的第二层权值。对于二值化输入，我们也需要非常大的权值用来尽量缩小重构误差。因为存在隐式和显式的正则化，使得我们很难得到大权值解决方案，优化算法发现编码只有在样

本和训练集相似的情况下表现良好，这也正是我们想要的。也可以说，这种表述运用了出现在训练集中的统计学的规律，而并非通过学习复制恒等函数。

有很多种方法可以让隐层数量大于输入维度的自编码器免于学到恒等函数，从而在它的隐层表述中获取到输入数据的一些有用信息。其中一种方法就是，替代对于权值隐式或者显式的规则化，或者在规则化基础上，在编码过程中增加噪音。这本质上就是受限玻尔兹曼机的原理，我们会在后续部分讨论到。另一种十分有效的方法[46，121，139，150，152，153]是基于对编码的稀疏性限制。有趣的是，通过这些方法得到的权值向量与在 V1 和 V2 神经[110]上观察到的感受域非常吻合，这些神经是哺乳动物视觉系统的主要区域。关于稀疏性的问题我们将会在 7.1 节讨论到。

鉴于稀疏性和正则化可以缩小描述能力从而避免学习到恒等函数，受限玻尔兹曼机可以有很大的处理容量同时不会学习到恒等函数，这是因为它不(止)尝试对输入数据进行编码，还通过近似最大化生成概率模型的似然函数去获取输入数据的统计学结构。有一种自编码器的变种模型也拥有受限玻尔兹曼机的这种特性，它被称作降噪自编码器[195]。降噪自编码器试图重建一个随机缺损版本的输入，并使重构误差最小化。我们可以发现，它可以将生成概率的对数似然函数的下限最大化。更多细节将会在 7.2 章节讲到。

第五章 能量模型和玻兹曼机

因为 Deep Belief Networks (DBNs)是以受限的玻兹曼机为基础，受限的玻兹曼机是典型的能量模型。这章我们介绍一些有用的数学概念来理解它们，其中包括偏差(Contrastive Divergence, CD)。

5.1 能量模型和期望(?experts)乘积

能量模型把标量能量和配置中的变量结合起来[107, 106, 149]。学习和修改能量函数对应起来，因此函数就可以得到期望的性质。例如，我们期望的配置消耗较少能量。基于能量的概率模型通过能量函数可以定义概率分布，定义如下：

$$\mathrm{P}(\boldsymbol{x}) = \frac{e^{-Energy(x)}}{Z}, \tag{5.1}$$

即，能量在对数概率域上起作用。上面一般化的指数簇模型[29]，其中能量函数 Energy(x)的形式为$\eta(x)\cdot\emptyset(x)$。下面我们将会看到，在 RBM 中，一层给出另一层的条件分布可以以指数的形式表现[200]。任何概率分布都可以转换为能量模型，一些特殊的概率分布，例如指数，可以得益于特殊的推论和学习过程。在能量模型中，学习非特殊分布是为了开辟学习方法而不是一般目的[84, 106, 149]。

与物理系统类似，正规因子 Z 叫做*配分函数*

$$Z = \sum_x e^{-Energy(x)} \tag{5.2}$$

如果 x 离散在输入域求和，如果 x 是连续的则做积分。甚至当求和或积分不存在 Z 时，这时候还可以定义能量模型(参考 5.1.2 节)。

在期望积定义中[69, 70]，能量函数是一些项的和，每一项出现的“期望”是f_i:

$$\text{Energy}(x) = \sum_i f_i(x), \tag{5.3}$$

即

$$P(x) \propto \prod_i P_i(x) \propto \prod_i e^{-f_i(x)}. \tag{5.4}$$

每个期望$P_i(x)$可以被看做配置中 **x** 的侦测器，或者看做强加在 **x** 上的限制。如果考虑特殊例子，就会容易理解，例如$f_i(x)$可以去两个值，一种情况（小）是满足限制，另一种情况（大）是不满足。[69]解释了期望乘积相对于 mixture of experts 的优点，mixture of expert 中概率的乘积被概率的加权和替换。为了简化，假设每个期望对应的限制或者可以满足，或者不可以。在一个混合模型中，期望的限制是一个区域排除其他区域的指示。期望乘积的一个优点是集合$f_i(\mathbf{x})$形成了分布的表示：不像在混合模型中用区域的期望来划分空间，他们根据配置的可能性来划分空间（其中每个期望或者有它的违反约束，或者没有）。[69]提出了一种算法，用跟期望有关的参数来估计式(5.4)的中 logP(x)的梯度，其中用到了[70]第一个示例中的偏差算法（参考 5.4 节）。

5.1.1 隐含变量的引入

在大部分情况下，$\mathbf{x}$ 由很多的变量 $\mathbf{x}_i$ 组成，我们无法同时观测这些组成部分。我们的想法是引入一些不可观测的变量，从而提高模型的表达能力。

因此，考虑一个观测部分(observed part)（仍然用 $\mathbf{x}$ 表示）和一个隐含部分(hidden part)$\mathbf{h}$的联合概率：

$$P(\mathbf{x}, \mathbf{h}) = \frac{e^{-\text{Energy}(\mathbf{x}, \mathbf{h})}}{Z} \tag{5.5}$$

由于只有$\mathbf{x}$可观测，所以我们考虑其边缘概率(marginal)：

$$P(\mathbf{x}) = \sum_{\mathbf{h}} \frac{e^{-\text{Energy}(\mathbf{x}, \mathbf{h})}}{Z}. \tag{5.6}$$

在这种情况下，对比类似式(5.1)中的定义形式，（受物理学启发）我们引入自由能(free energy)符号，定义如下：

$$P(\mathbf{x}) = \frac{e^{-\text{FreeEnergy}(\mathbf{x})}}{Z}, \tag{5.7}$$

其中，$Z = \sum_{\mathbf{x}} e^{-\text{FreeEnergy}(\mathbf{x})}$，即有：

$$\text{FreeEnergy}(\mathbf{x}) = -\log \sum_{\mathbf{h}} e^{-\text{Energy}(\mathbf{x},\mathbf{h})}. \tag{5.8}$$

因此，自由能可以理解为一个在对数域(log-domain)中进行边缘化的能量(marginalization of energies)。

数据的对数似然梯度(log-likelihood gradient)有一个非常有趣的形式。如果用 θ 来表示模型的参数，从式(5.7)出发，我们得到：

$$\begin{aligned}\frac{\partial \log P(\mathbf{x})}{\partial \theta} &= -\frac{\partial \text{FreeEnergy}(\mathbf{x})}{\partial \theta} + \frac{1}{Z}\sum_{\tilde{\mathbf{x}}} e^{-\text{FreeEnergy}(\tilde{\mathbf{x}})}\frac{\partial \text{FreeEnergy}(\tilde{\mathbf{x}})}{\partial \theta} \\ &= -\frac{\partial \text{FreeEnergy}(\mathbf{x})}{\partial \theta} + \sum_{\tilde{\mathbf{x}}} P(\tilde{\mathbf{x}})\,\frac{\partial \text{FreeEnergy}(\tilde{\mathbf{x}})}{\partial \theta}. \end{aligned} \tag{5.9}$$

所以，在训练集(training set)上对数似然梯度的期望(the average log-likelihood gradient)为：

$$E_{\hat{P}}\left[\frac{\partial \log P(\mathbf{x})}{\partial \theta}\right] = -E_{\hat{P}}\left[\frac{\partial \text{FreeEnergy}(\mathbf{x})}{\partial \theta}\right] + E_P\left[\frac{\partial \text{FreeEnergy}(\mathbf{x})}{\partial \theta}\right] \tag{5.10}$$

其中，该期望在 $\mathbf{x}$ 上，训练集的经验分布为 $\hat{P}$，并且期望 E_P 在模型分布 P 下。

那么，如果我们能够对 P 进行采样(sample)，并且简便的计算自由能，我们就可以使用蒙特卡洛方法(Monte-Carlo Method)求出对数似然梯度的随机估计量(stochastic estimator)。

如果将能量写成最多与一个隐单元(hidden unit)相联系的项之和

$$\text{Energy}(\mathbf{x},\mathbf{h}) = -\beta(\mathbf{x}) + \sum_i \gamma_i(\mathbf{x},\mathbf{h}_i), \tag{5.11}$$

这满足 RBM 的情况下的条件，那么自由能和似然值的分子项都能够被简便的计算出来（尽管它涉及一些指数项的和）：

$$\begin{aligned} P(\mathbf{x}) &= \frac{1}{Z}e^{-\text{FreeEnergy}(\mathbf{x})} = \frac{1}{Z}\sum_{\mathbf{h}} e^{-\text{Energy}(\mathbf{x},\mathbf{h})} \\ &= \frac{1}{Z}\sum_{\mathbf{h}_1}\sum_{\mathbf{h}_2}\cdots\sum_{\mathbf{h}_k} e^{\beta(\mathbf{x})-\sum_i \gamma_i(\mathbf{x},\mathbf{h}_i)} \\ &= \frac{1}{Z}\sum_{\mathbf{h}_1}\sum_{\mathbf{h}_2}\cdots\sum_{\mathbf{h}_k} e^{\beta(\mathbf{x})}\prod_i e^{-\gamma_i(\mathbf{x},\mathbf{h}_i)} \\ &= \frac{e^{\beta(\mathbf{x})}}{Z}\sum_{\mathbf{h}_1} e^{-\gamma_1(\mathbf{x},\mathbf{h}_1)}\sum_{\mathbf{h}_2} e^{-\gamma_2(\mathbf{x},\mathbf{h}_2)}\cdots\sum_{\mathbf{h}_k} e^{-\gamma_k(\mathbf{x},\mathbf{h}_k)} \\ &= \frac{e^{\beta(\mathbf{x})}}{Z}\prod_i\sum_{\mathbf{h}_i} e^{-\gamma_i(\mathbf{x},\mathbf{h}_i)} \end{aligned} \tag{5.12}$$

在上式中，$\sum_{\mathbf{h}_i}$ 是对 $\mathbf{h}_i$ 可取的所有值求和（比如在通常两个单元的情况下的两个值）；注意，这个和式比所有 $\mathbf{h}$ 值之和 $\sum_{\mathbf{h}}$ 容易计算得多。如果 h 是连续的，那么相同的原则适用于通过积分替换所有的和。

大多数情况下，求和或者积分（在所有的单层隐单元(sigle hidden unit)值中）的计算都很简单，而且似然值的分子项（即自由能）能在上述情况下被精确计算出来，在

$\text{Energy}(\mathbf{x},\mathbf{h}) = -\beta(\mathbf{x}) + \sum_i \gamma_i(\mathbf{x},\mathbf{h}_i)$, 情况下，我们有：

$$\text{FreeEnergy}(\mathbf{x}) = -\log P(\mathbf{x}) - \log Z = -\beta(\mathbf{x}) - \sum_i \log \sum_{\mathbf{h}_i} e^{-\gamma_i(\mathbf{x},\mathbf{h}_i)}. \tag{5.13}$$

5.1.2 有条件的基于能量的模型

尽管计算这个配分(分区)函数一般比较复杂，但如果我们最终只关心根据给出的一个变量 x 而得出的变量 y，而不是考虑所有配置(x,y),这样我们考虑每个给定的 x 所得出的 y 的配置就可以了。一个常见的例子就是 y 只能取值于一个小的离散集合中，即：

$$P(y|\mathbf{x}) = \frac{e^{-\text{Energy}(\mathbf{x},y)}}{\sum_y e^{-\text{Energy}(\mathbf{x},y)}}. \tag{5.14}$$

在这种情况下，该条件对数似然的梯度到能量函数的参数可以有效地计算出来。

这一提法适用于 RBM 的判别变种，我们称之为判别 RBM[97]。这种有条件的基于能量的模型也被应用于一系列基于神经网络的概率语言模型中 [15, 16, 130, 169, 170, 171, 207]。

该公式（或当其在一组分区函数方面很容易求和或最大化）的价值已经在探索中[37, 106, 107, 149, 153]。在后续工作中一个重要且有趣的内容是，它表明这种基于能量的模型不仅相对于对数似然可被优化，而相对于更一般的标准，其梯度具有使“正确”响应能量减小，同时增加竞争反应能量的属性。这些能量函数不一定会产生一个概率模型（因为这种负能量函数的指数不需要可被积分），但是它们仍然可以产生一个给出 x 所得出的 y 的函数，这往往是应用程序的最终目标。实际上，当 y 取值于一个有限集合中时，P（Y| X）总是可以被计算，因为能量函数只可在 y 的可能值中被归一化。

5.2 波兹曼机

波兹曼机是含有隐变量的基于能量模型的一种特殊形式，而 RBMs 则是波兹曼机的一种特殊形式，其 P(h|x)和 P(x|h)都是易于处理的，因为它们都可以进行因式分解。在波兹曼机中[1,76,77]，能量函数是一个一般的二次多项式：

$$\text{Energy}(\mathbf{x},\mathbf{h}) = -\mathbf{b}'\mathbf{x} - \mathbf{c}'\mathbf{h} - \mathbf{h}'W\mathbf{x} - \mathbf{x}'U\mathbf{x} - \mathbf{h}'V\mathbf{h}. \tag{5.15}$$

有两种类型的参数,我们统一用 θ 表示：

偏置 bi 和 ci(每一个都关联到向量 x 或者向量 h 中的一个元素),而权重 Wij,Uij 和 Vij (每一个都关联到一对单元).假设矩阵 U 和 V 是对称的[1]且对角线元素大多数都为 0.对角线上的非零元素可以被用来计算得到其他变量,例如,使用高斯而不是二项单元[200]。

因为 h 中有二次幂数量的元素,所以解析计算自由能量函数 (Equation(5.12))的技巧不能再这里使用。

然而,可以使用 MCMC(蒙特卡洛马尔科夫链[4])采样方法得到对梯度的随机估计。似然函数的梯

度可以写成如下形式,

根据(5.6)式:

$$\begin{aligned}\frac{\partial \log P(\mathbf{x})}{\partial \theta} &= \frac{\partial \log \sum_{\mathbf{h}} e^{-\text{Energy}(\mathbf{x},\mathbf{h})}}{\partial \theta} - \frac{\partial \log \sum_{\tilde{\mathbf{x}},\mathbf{h}} e^{-\text{Energy}(\tilde{\mathbf{x}},\mathbf{h})}}{\partial \theta} \\ &= -\frac{1}{\sum_{\mathbf{h}} e^{-\text{Energy}(\mathbf{x},\mathbf{h})}} \sum_{\mathbf{h}} e^{-\text{Energy}(\mathbf{x},\mathbf{h})} \frac{\partial \text{Energy}(\mathbf{x},\mathbf{h})}{\partial \theta} \\ &\quad + \frac{1}{\sum_{\tilde{\mathbf{x}},\mathbf{h}} e^{-\text{Energy}(\tilde{\mathbf{x}},\mathbf{h})}} \sum_{\tilde{\mathbf{x}},\mathbf{h}} e^{-\text{Energy}(\tilde{\mathbf{x}},\mathbf{h})} \frac{\partial \text{Energy}(\tilde{\mathbf{x}},\mathbf{h})}{\partial \theta} \\ &= -\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{x}) \frac{\partial \text{Energy}(\mathbf{x},\mathbf{h})}{\partial \theta} + \sum_{\tilde{\mathbf{x}},\mathbf{h}} P(\tilde{\mathbf{x}},\mathbf{h}) \frac{\partial \text{Energy}(\tilde{\mathbf{x}},\mathbf{h})}{\partial \theta}. \end{aligned} \tag{5.16}$$

(注 1：例如,如果 U 不是对称的,会产生一些额外的自由度.因为 xiUijxj +xjUjixi 可以被重写为 xi(Uij +Uji)x = 12xi(Uij +Uji)xj + 1xj(Uij + Uji)xi,也就是在对称矩阵形式中。)注意(∂Energy(x,h)/∂θ)是容易计算的. 因此如果我们可以对 P(h|x 金和 P(x,h)进行采样的话，那么我们就可以获得对数似然函数梯度的无偏估计。[1,76,77]介绍如下的术语：在正向阶段， x 绑定到被观察到的输入向量，我们在已知 x 的前提下采样 h；在负向阶段，理想情况下通过模型本身对 x 和 h 一起进行采样。一般，只有近似采样才便于处理，也就是说，使用一个迭代过程去构造 MCMC。 MCMC 采样方法是基于 Gibbs 采样[4, 57]，在 [1, 76, 77]中进行了介绍。通过一系列如下形式的 N 步子采样完成对于 N 个随机变量 S=(S1,...,SN)的联合分布的 Gibbs 采样

$S_i \sim P(S_i|S_{-i} = \mathbf{s}_{-i})$ (5.17)

S_{-i} 包含了S中除了S_i 以外的N-1个随机变量

在这 N 个采样都得到以后，链的一步已经完成,生成了 S 的一个采样，在某些条件下，随着步数逐渐增加到无穷大，S 的分布收敛于 P(S)。

马尔科夫链是非周期[2]的和不可约[3]的是其收敛的充分条件。

我们怎样把Gibbs采样用于波兹曼机呢?

用s=(x,h)表示波兹曼机里面的所有单元，$\mathbf{s}_{-i}$ 表示除了和第i个单元以外所有单元相连的值的集合.通过把所有参数置于向量d和一个对称矩阵A，波兹曼机的能量函数可以重写为如下格式，

$Energy(s) = -ds - sAs.$ (5.18)

用$\mathbf{d}_{-i}$表示表示向量d里除了 d_i 的所有元素，A_{-i}表示矩阵A里除了第i行的所有行，
$\mathbf{a}_{-i}$ 表示矩阵A里第i行中除了第i个元素的所有元素。

使用这种表示方式，我们可以得到(s_i/s_{-i})在波兹曼机中可以很容易通过计算和采样得到。例如,如果$s_i \in \{0,1\}$ 且矩阵A中的对角线元素都为空：

(注2 非周期性: 没有一个状态的周期为$k>1$;如果一个状态可以在$t+k$,$t+2k$的时刻转移到自己，那么该状态的周期为k。
注3 不可约：任何两个状态之间在经过有限次转移都能以非零的概率相互转移。）

$$
\begin{aligned}
P(\mathbf{s}_i = 1|\mathbf{s}_{-i}) &= \frac{\exp(\mathbf{d}_i + \mathbf{d}'_{-i}\mathbf{s}_{-i} + 2\mathbf{a}'_{-i}\mathbf{s}_{-i} + \mathbf{s}'_{-i}A_{-i}\mathbf{s}_{-i})}{\exp(\mathbf{d}_i + \mathbf{d}'_{-i}\mathbf{s}_{-i} + 2\mathbf{a}'_{-i}\mathbf{s}_{-i} + \mathbf{s}'_{-i}A_{-i}\mathbf{s}_{-i}) + \exp(\mathbf{d}'_{-i}\mathbf{s}_{-i} + \mathbf{s}'_{-i}A_{-i}\mathbf{s}_{-i})} \\
&= \frac{\exp(\mathbf{d}_i + 2\mathbf{a}'_{-i}\mathbf{s}_{-i})}{\exp(\mathbf{d}_i + 2\mathbf{a}'_{-i}\mathbf{s}_{-i}) + 1} = \frac{1}{1 + \exp(-\mathbf{d}_i - 2\mathbf{a}'_{-i}\mathbf{s}_{-i})} \\
&= \mathrm{sigm}(\mathbf{d}_i + 2\mathbf{a}'_{-i}\mathbf{s}_{-i}) \qquad (5.19)
\end{aligned}
$$

在人工神经网络中，以上公式基本上就是根据其他神经元 s_{-i} 计算 si 的常用公式。
因为对于每个样本 x 都需要两个 MCMC 链(一个是正向链，一个是负向链)，这样对于梯度的计算将会消耗很大的计算量，训练将会非常耗时。这也是为什么对于多层神经网络，在 1980 末的时候，反向传播能替代波兹曼机作为最主要的训练方法。然而，最近的工作表明短链在某些时候也可以被成功的运用，这也就是对数散度(5.4 节讨论)被用来训练 RBM 的原理。注意负向链阶段对于每个新的样本 x 都不需要被重新计算(因为它并没有依赖于训练数据)，这种观察被用于 5.4.2 小节讨论的一致 MCMC 估计。

5.3 限制玻尔兹曼机

限制玻尔兹曼机是深信度网络（DBN）的组成模块。限制玻尔兹曼机与 DBN 每层共享参数，通过训练玻尔兹曼机可以得到有效的学习算法。RBM 的无向图模型参见图 4.5。图中，在已知 x 的情况下，h_i 之间是相互独立的；在已知 h 的情况下，x_j 之间是相互独立的。在 RBM 模型中，等式 5.15 中的 U=0 切 V=0，换句话说同层之间各单元相互独立，隐藏单元和可视单元之间相互影响。该模型是由 Harmonium[178]首先提出的，其算法论述可参见[51]。已经证实的有效学习算法和衍生模型可参见[31,80,200]。能量方程是双线性的，其中输入单元和输入单元之间以及隐藏单元和隐藏单元之间没有关联：

$$\mathrm{Energy}(\mathbf{x},\mathbf{h}) = -\mathbf{b}'\mathbf{x} - \mathbf{c}'\mathbf{h} - \mathbf{h}'W\mathbf{x} \qquad (5.20)$$

输入自由能量参数可以通过等式 5.11,5.13 以及 $\beta(\mathbf{x}) = \mathbf{b}'\mathbf{x}$ $\gamma_i(\mathbf{x},\mathbf{h}_i) = -\mathbf{h}_i(\mathbf{c}_i + W_i\mathbf{x})$ 求出。其中 W_i 是 W 的第 i 行行向量。输入(非归一的对数概率)的自由能量可以通过下面的等式来计算：

$$\text{FreeEnergy}(\mathbf{x}) = -\mathbf{b}'\mathbf{x} - \sum_i \log \sum_{\mathbf{h}_i} e^{\mathbf{h}_i(\mathbf{c}_i + W_i\mathbf{x})}. \qquad (5.21)$$

参照等式 5.12，利用以 h 为参数的 Energy（x，h）的仿射模式，我们可以得到条件概率 P（h|x）的另一种表达方式：

$$\begin{aligned} P(\mathbf{h}|\mathbf{x}) &= \frac{\exp(\mathbf{b}'\mathbf{x} + \mathbf{c}'\mathbf{h} + \mathbf{h}'W\mathbf{x})}{\sum_{\tilde{\mathbf{h}}} \exp(\mathbf{b}'\mathbf{x} + \mathbf{c}'\tilde{\mathbf{h}} + \tilde{\mathbf{h}}'W\mathbf{x})} \\ &= \frac{\prod_i \exp(\mathbf{c}_i\mathbf{h}_i + \mathbf{h}_i W_i\mathbf{x})}{\prod_i \sum_{\tilde{\mathbf{h}}_i} \exp(\mathbf{c}_i\tilde{\mathbf{h}}_i + \tilde{\mathbf{h}}_i W_i\mathbf{x})} \\ &= \prod_i \frac{\exp(\mathbf{h}_i(\mathbf{c}_i + W_i\mathbf{x}))}{\sum_{\tilde{\mathbf{h}}_i} \exp(\tilde{\mathbf{h}}_i(\mathbf{c}_i + W_i\mathbf{x}))} \\ &= \prod_i P(\mathbf{h}_i|\mathbf{x}). \end{aligned}$$

通常情况下，$h_i \in \{0,1\}$，我们可以得到通用的神经等式:

$$P(\mathbf{h}_i = 1|\mathbf{x}) = \frac{e^{\mathbf{c}_i + W_i\mathbf{x}}}{1 + e^{\mathbf{c}_i + W_i\mathbf{x}}} = \text{sigm}(\mathbf{c}_i + W_i\mathbf{x}). \qquad (5.22)$$

在能量方程中，x 和 h 对称，P(x|h)可以用下面的公司来计算：

$$P(\mathbf{x}|\mathbf{h}) = \prod_i P(\mathbf{x}_i|\mathbf{h}) \qquad (5.23)$$

在二元制的情况下：

$$P(\mathbf{x}_j = 1|\mathbf{h}) = \text{sigm}(\mathbf{b}_j + W'_{.j}\mathbf{h}) \qquad (5.24)$$

Wj 是 W 的第 j 列。

在[73]中，用两项输入单元来编码输入图像的像素灰度级别。该方法对于手绘人物图像有效，在其他情况下，并不理想。

实验表明当输入为连续值时，使用高斯输入单元来模拟效果要优于二元单元模拟法，详情参见[17]。文[200]中给出了一个通用的公式，x 和 h（在一个已值的前提下），可以表示为任一种指数分布（离散和连续）。

虽然，在某些情况下利用非限制玻尔兹曼机要比 RBM 模型更有效，但是在有足够多隐藏单元的前提下，RBM 可以用于任何一种离散分布[51,102]。此外，在 RBM 尚未完美建模训练分布之前，增加一个隐藏单元（并选择合适的权重和偏移）可以提升对数似然概率[102]。

如图 3.2 所示，我们也可以把 RBM 看作一个多聚类（参见 3.2）。每个隐藏单元把输入空间划分成一个二元分区（借助线性划分方法）。通过线性分区方法，三个隐藏单元利用三组二元分

区，可以延伸出八种组合。每个分组对应于一个输入区域。一个区域内有相同的隐藏配置（即编码）。隐藏单元的一个二元设置可以识别出输入空间的一个区域。对所有该区域内的 X，P（H|x）在对应的 h 配置下取最大值。需要注意的是，并非所有的隐藏单元都有对应的输入空间。如图 3.2 所示，这种情况与二叉树的情况类似。

RBM 模型中，利用所有可能的隐藏层配置的概率以及在该配置下的输入，我们可以得到：

$$P(\mathbf{x}) = \sum_{\mathbf{h}} P(\mathbf{x}|\mathbf{h})P(\mathbf{h}) \qquad (5.25)$$

其中 P（x|h）是配置为 H 的模型。举例来说，如果 P（x|H）是高斯分布（参见[200，17]），则 P（x|h）共有 2^n 种可能（h 有 n 个比特）。这 2^n 种可能依赖于共同的参数（RBM 参数），因而他们之间并非相互独立的。利用该特性，RBM 模型经过归一化后可用于没有训练案例的场景。在高斯分布的场景下，搞死均值可以表示为 b+w’h，即每个隐藏单元 hi 对均值的贡献为 Wi。

5.3.1 RBM 的吉布斯采样

RBM 采样非常有用。首先利用采样可以得到对数似然梯度的估计。其次，通过分析该模型采样可以得到该模型能扑捉和不能扑捉的数据分布。DBN 模型中的最上面两层是 RBM，借助 RBM 采样可以为 DBN 提供样本，详情参见 6.1.

在全连接的玻尔兹曼机模型中，吉布斯采样很慢。网络单元决定了吉布斯链的子程序数量。换句话说，对 RBM 进行分解有两个好处：一是不需要正态采样，自由能量（以及其梯度）可以借助解析的方法计算出；二是，变量对（x，h）可以借助吉布斯链的两个阶段分别得到。首先我们在给定 x 的情况下计算 h，然后利用给定的 h 计算 x。在通用的 PoE 模型中（product of experts models），可以用混合蒙特卡洛采样[48,136]来替代吉布斯采样。MCMC 方法中每一步马尔科夫链包含数个自由能量梯度计算。因此，RBM 结构是 PoE 模型的特例：等式 5.21 中第 i 项 $\log\sum_{\mathbf{h}_i} e^{(\mathbf{c}_i + W_i\bar{\mathbf{x}})\mathbf{h}_i}$ 对应域一个专家单元，即每个隐藏神经和输入偏移都有一个专家单元。这样的结构使得吉布斯采样非常高效。从训练案例开始（即 $\hat{P}$ 采样），对于吉布斯采样的第 k 步：

$$
\begin{aligned}
\mathbf{x}_1 &\sim \hat{P}(\mathbf{x}) \\
\mathbf{h}_1 &\sim P(\mathbf{h}|\mathbf{x}_1) \\
\mathbf{x}_2 &\sim P(\mathbf{x}|\mathbf{h}_1) \\
\mathbf{h}_2 &\sim P(\mathbf{h}|\mathbf{x}_2) \\
&\vdots \\
\mathbf{x}_{k+1} &\sim P(\mathbf{x}|\mathbf{h}_k).
\end{aligned} \tag{5.26}
$$

通过捕捉训练数据中的结构可以优化 RBM 模型，模型分布 P 和训练分布 $\hat{P}$ 更简单了（两者具有相似的统计特性）。需要注意的是，如果我们从 P 开始，该模型会收敛成一步，所以从 $\hat{P}$ 开始，能保证必要的收敛步骤。

5.4 Contrastive Divergence（对比区分）

对比区分（Contrastive Divergence）是一种用以逼近似然函数梯度的方法，目前该方法，作为一种更新法则，成功地应用在了受限波尔茨曼机（RBMs）中[31]。算法 1 所示的就是关于这个方法的伪代码，其中有特定的针对 0-1 输入层和隐藏层来计算条件分布的公式。

5.4.1 Justifying Contrastive Divergence（CD 的正确性验证）

为了构造具有这种效果的算法，首先我们要用单次采样的结果来代替所有可能输入情况的平均值（在式子（5.10）的第二部分）。因为我们会经常更新参数（比方说采用随即梯度下降更新或者使用小部分训练数据的小样本（min-batch）梯度更新），所以在多次参数更新的过程中就隐含有一个取平均的过程（这个隐含的取平均值的方法在[105]中有较好的效果）；另外，由一次或者几次 MCMC 采样（而不是做完全加和）而带来的额外的参数，在参数的在线梯度更新过程中，也许是可以部分去除的。我们会因梯度的逼近而引入额外的变量，但是如果额外的变量要比由于在线梯度下降而带来的变量要小或者相差不多，那么这个额外的引入则不会有太大的副作用。

————————————————————————————

Algorithm 1

————————————————————————————

$RBMupdate(x_1,\varepsilon,W,b,c)$

这个 RBM 更新的算法是针对 0-1 输出单元的。该算法也可以比较容易改成其他类型的单元。

x_1，是根据训练数据的分布而产生的一个样本数据，以此来训练 RBM

ε，是 CD 中随即梯度下降的学习速率

W，是 RBM 权重矩阵，矩阵的维度是（隐含层的单元数目，输入层的单元数目）

b，是 RBM 中对输入单元的基准向量

c，是 RBM 中对隐含单元的基准向量

注意：$Q(h_2=1|x_2)$ 表示的是由 $Q(h_{2i}=1|x_2)$ 这些元素组成的向量

for 所有的隐含单元 i do

• 计算所有的 $Q(h_{1i}=1|x_1)$ (对于 0-1 的二元化单元，用 $sigm\left(c_i+\sum_j W_{ij}x_{1j}\right)$ 计算)

• 依据 $Q(h_{1i}=1|x_1)$ 来进行隐含层单元的采样，来得到 $h_{1i}\in\{0,1\}$

end for

for 所有的可见单元 j do

• 计算所有的 $P(x_{2j}=1|h_1)$ （对于 0-1 的二元化单元，用 $sigm\left(b_j+\sum_i W_{ij}h_{1i}\right)$ 计算）

• 依据 $P(x_{2j}=1|h_1)$ 来进行采样，以此得到 $x_{2j}\in\{0,1\}$

end for

for 所有的隐含单元 i do

• 计算所有的 $Q(h_{2i}=1|x_2)$ （对于 0-1 的二元化单元，用 $sigm\left(c_i+\sum_j W_{ij}x_{2j}\right)$ 计算）

end for

- $W\leftarrow W+\varepsilon\left(h_1x_1^{'}-Q\left(h_2=1|x_2\right)x_2^{'}\right)$
- $b\leftarrow b+\varepsilon\left(x_1-x_2\right)$
- $c\leftarrow c+\varepsilon\left(h_1-Q\left(h_2=1|x_2\right)\right)$

————————————————————————————————————

运行一个步长较长的 MCMC 采样链的代价非常大，为了改善这个大的代价，我们引入了 k 步 CD 的近似方法。k 步 CD（CD-k）[69，70]的想法较为简单，而且涉及另一个近似过程：在梯度中引入一些基值：从最开始可观测到的数据 $x_1=x$，运行 MCMC 采样链 k 次直到 x_{k+1}。CD-k 会在看见数据样本 x 后进行参数更新，因此，

$$\Delta\theta\propto\frac{\partial FreeEnergy(x)}{\partial\theta}-\frac{\partial FreeEnergy(\tilde{x})}{\partial\theta}\qquad(5.27)$$

上式中$\tilde{x} = x_{k+1}$是经过 k 步迭代之后的从马尔科夫链模型中得到的采样结果。经分析，我们可得知：在开始阶段加入的基值，当 k 趋近于无穷时，会最终不起任何作用。此外，我们还发现：当模型产生的分布十分接近经验分布时（比如$P \approx \hat{P}$），这时我们从 MCMC 采样链里面抽取数据样本已经收敛了，所以这个时候我们只需要一步就可以从模型的分布 P 中获得一个无偏采样结果了（虽然这个时候，结果还是与 x 相关的）。

一个令人惊奇的经验性的结果是：就算 k=1 的迭代，都总是可以得到较好的结果。一个扩展性的关于 CD-k 和准确的似然函数梯度的比较在[31]中可以找到。在这些实验中，取 k 大于 1 的时候可以得到更精准的结果，不过当 k 取 1 的时候也是可以得到一些很精准的结果的。在 5.4.3 章节给出的一些理论上的结果[12]可以帮助我们理解为什么小数目的 k 可以起作用：CD-k 对应着将开始的 k 个数值与似然函数梯度对应起来。

一种解释 CD 的方法是：CD 是一种在训练样本x_1周围局部逼近似然函数梯度的方法。随机重建$\tilde{x} = x_{k+1}$（对于 CD-k）是一种在某种程度上以x_1为中心的分布（给定x_1的情况下），而且随着 k 的变大，这种分布会更加扩散，直到收敛成为模型的分布。CD-k 的更新会降低训练样本x_1的自由能量（如果其他的自由能量保持不变，那么增加自由能量会提高似然函数数值），同时会增加$\tilde{x}$（x_1的邻点）的自由能量。注意到$\tilde{x}$是x_1的邻点，但可能同时在模型的高概率区域（特别是当 k 比较大的时候）。正如文章[106]中所说的那样，对于一个基于能量的模型，从训练数据那里获得的最多的是：模型会削弱观测到的输入数据的能量（在这里，也就是自由能量，来边缘化隐含层的变量），将能量转移到其他地方去。CD 算法受到来自两种情况下的数据对比：一种是当输入是真实的训练样本数据，而另一种是当输入是从采样链中得到的结果。更多的讨论会在接下来的章节展开，我们可以无监督学习问题想象成一个发现决策曲面的问题，这些决策曲面将高概率区域（这个区域里有很多给定的观测到的训练数据）与其他小概率区域分开。因此，当模型在错误的区域产生数据样本时，我们会给与一定的惩罚；也就是说，一个好的解决“分割面是否应该移动”问题的好方法是：比较训练样本的数据和由模型产生的数据。

5.4.2 对数散度的替代选择

关于RBM的学习算法，最近有一个激动人心的发现，使用一个叫做一致MCMC的方法对负向阶段进行处理[161,187],下面的方法在 [135]中已有介绍.

这个想法很简单:依然使用MCMC链... $\mathbf{x}t \rightarrow \mathbf{h}t \rightarrow \mathbf{x}t+1 \rightarrow \mathbf{h}t+1$ . . .去获得负向阶段的采样(应该源自模型本身). 但我们不会去执行一个像CD-k这样的短链, 我们会忽略在MCMC链生成的过程中参数是会不断改变的这一事实,也就是说,我们不会对参数的每个值单独执行一个MCMC链 (就像在传统的限制波兹曼机中的那样)。也许因为参数更新的比较慢, 所以逼近的效果非常好,一般比CD-k能得到更大的对数似然度(用k=1和k=10都进行了实验)。CD-1权衡的是方差

更大，但是偏差更小。 一些有趣的事情发生了[188]：模型系统地远离从负向阶段中获得的采样，模型和链不断的进行交互，防止了模型在一个区域中滞留过长时间，这极大的增加了链的混合率。这会产生期望得到但是无法观察到的影响，可以帮助更快的去遍历RBM配置参数的空间.另一种替代对比散度的是得分匹配[84, 85, 86]，一种对于能量容易计算，但是标准化常数Z不容易计算的时候，去训练基于能量模型的一般方法。 p(x)=q(x)/Z的评分函数为ψ=(∂logp(x))/∂x, 很明显评分函数不依赖于它的归一化常数Z,也就是,ψ=(∂logq(x))/∂x.一般的想法是匹配模型和经验密度的评分函数。这两个评分函数之间差值2范数的均值(基于经验密度)可以写成模型评分函数的平方项和二阶导数(∂2logq(x))/∂x2的形式。

评分匹配[84]已经被证明是局部一致性的,也就是,如果模型簇匹配了数据生成过程，那么评分匹配就会收敛,一般它也被用于音频数据和图像数据的非监督模型 [94]。

5.4.3 吉布斯链模型中的对数似然梯度的截断

这里，我们从另外一个角度来研究对比分歧，推广它，并建立他与重建误差的联系。后者常常被用来训练自动编码器以及监视性能（等式（4.7））。这受到[73]了的启发：第一是吉布斯链和无线深度的有向图模型的联系（在 8.1 节中有解释）（这里与对数似然梯度的展开形式有关）；第二是链的收敛性解释了对比分歧的合理性（当吉布斯链的采样 $\tilde{\mathbf{x}}$ 来自于模型本身的概率分布时，等式 5.27 的期望和等式 5.29 等价）。具体来说，我们希望理解对比分歧的偏差，并将其与难以计算的对数似然梯度进行比较。

考虑一个由条件概率分布 $P(\mathbf{h}_t|\mathbf{x}_t)$ 和 $P(\mathbf{x}_{t+1}|\mathbf{h}_t)$ 定义的收敛马尔可夫链 $\mathbf{x}_t \Rightarrow \mathbf{h}_t \Rightarrow \mathbf{x}_{t+1} \Rightarrow \cdots$

其中$\mathbf{x}_1$采样自训练数据的经验概率分布。

下面的定理（出自[12]）表明当$t \geq 1$时，对数似然梯度可以如何展开。

————————————————————————————————

定理 5.1. 对于一个从数据点$\mathbf{x}_1$开始的收敛马尔科夫链，它的对数似然梯度可以表示为：

$$\frac{\partial \log P(\mathbf{x}_1)}{\partial \theta} = -\frac{\partial \text{FreeEnergy}(\mathbf{x}_1)}{\partial \theta} + E\left[\frac{\partial \text{FreeEnergy}(\mathbf{x}_t)}{\partial \theta}\right] + E\left[\frac{\partial \log P(\mathbf{x}_t)}{\partial \theta}\right] \tag{5.28}$$

当 t 趋向无穷时，最后一项收敛为 0。

————————————————————————————————

如定理所示，最后一项对着 t 的增大而减小，因此在第 k 步的时候截断马尔科夫链是有其合理性的。这样我们可以得到如下的约等式

$$\frac{\partial \log P(\mathbf{x}_1)}{\partial \theta} \simeq -\frac{\partial \text{FreeEnergy}(\mathbf{x}_1)}{\partial \theta} + E\left[\frac{\partial \text{FreeEnergy}(\mathbf{x}_{k+1})}{\partial \theta}\right]$$

而这正是在用一个采样取代了期望，$\tilde{\mathbf{x}} = \mathbf{x}_{k+1}$，以后得到的 CD-k 更新（等式（5.27））。这说明 CD-k 的偏置为$E[(\partial \log P(\mathbf{x}_{k+1}))/\partial\theta]$。实验和理论都显示相比于 CD-(k-1)，更小的偏置让 CD-k 有更好的更快的收敛特性（以循环次数为准）（尽管实际中这并不一定值得）。另一方面，在 k 比较的小的时候，尽管 CD-k 有较大的偏置，它仍然可以像对数似然梯度一样可以驱使模型参数到达合适的象限[12]。这与 k=1 情况下所得到的理想结果相对应。更加直观地解释的话：当我们用$\mathbf{x}_1$来初始化马尔科夫链的时候，第一步（向$\mathbf{x}_2$）就已经大概指向了相对于$\mathbf{x}_1$正确的方向，也就是指向比$\mathbf{x}_1$能量更低的区域。因为梯度取决于$\mathbf{x}_2$和$\mathbf{x}_1$之间的变化，我们可以得到梯度的方向。

所以 CD-1 对应于采样两次（一次根据$\mathbf{h}_1|\mathbf{x}_1$，另一次根据$\mathbf{x}_2|\mathbf{h}_1$）以后截断马尔科夫链。如果只从$\mathbf{h}_1|\mathbf{x}_1$采样一次就截断会怎样呢？我们可以从对数似然梯度的展开来分析[12]：

$$\frac{\partial \log P(\mathbf{x}_1)}{\partial \theta} = E\left[\frac{\partial \log P(\mathbf{x}_1|\mathbf{h}_1)}{\partial \theta}\right] - E\left[\frac{\partial \log P(\mathbf{h}_1)}{\partial \theta}\right]. \quad (5.29)$$

对于第一个项期望，我们做平均场近似：用$\mathbf{h}_1$的期望，$\hat{\mathbf{h}}_1 = E[\mathbf{h}_1|\mathbf{x}_1]$，来替代每次使用$P(\mathbf{h}_1|\mathbf{x}_1)$生成不同的$\mathbf{h}_1$，这样可以得到：

$$E\left[\frac{\partial \log P(\mathbf{x}_1|\mathbf{h}_1)}{\partial \theta}\right] \simeq \frac{\partial \log P(\mathbf{x}_1|\hat{\mathbf{h}}_1)}{\partial \theta}. \quad (5.30)$$

如果我们忽略掉等式（5.29）中的第二个期望项（尽管这样会引入对数似然梯度的另一偏差），我们可以利用约等式（5.30）中约等号右边的项来做更新。它也可以看作为重建误差的梯度取负：

$$-\log P(\mathbf{x}_1|\hat{\mathbf{h}}_1)$$

这样的更新规则一般用来训练自动编码器（见等式(4.7)并取$\mathbf{c}(\mathbf{x}) = E[\mathbf{h}|\mathbf{x}]$）。(footnote 脚注：在 5.30 中，如果将$\hat{\mathbf{h}}_1$看做θ的函数，那么在参数θ的梯度中就有一部分来自$\hat{\mathbf{h}}_1$的贡献。是否考虑这部分贡献在一般情况下是值得商榷的。但在直接与自动编码器类比的情况下，这一部分是必须的。）

所以我们可以看到截断吉布斯链的近似效应。平均场近似下的一次采样对粗略对应于重建误差训练。CD-k 则是用采样来近似期望，例如 CD-1 使用了采样两次之后的样本。重建误差的计算是确定性的并且和对数似然值有关，这也是为什么它被与监视使用 CD 来训练 RBM 时的进度。

5.4.4、标准集是一个反例

通过这种模型，我们认为一系列的区别于样例的分类问题能被成功的当做一个基于能量的模型。在玻尔兹曼机器学习法则和分歧对比中，从模型中提炼样例的能力也学是相似的。理解提高日志记录可能的例子的价值的有效方法在迅速发展中被提出[201]。我们先是通过非正式的方式解释这种想法，之后再通过正式的方式，按照从训练集中分离标准集方法得到训练的代际模型证明我们的法则。最大似然准则可能适用于训练集以及其他地方。如果我们已经有了一个模型和提高可能性的方法，出现高频率模型(通过标准模型体现的)的地方和训练集的地方的不同是如何去改变这个模型。如果我们能够通过表面现象从标准模型中大概分离出训练集的话，我们就有应该在减少表面判断(一方面，这里有很多的训练实例)的强大作用和增强其他方面(另一方面，这里有很多从模型中分离出来的标准实例)有很大的可能。用数学的方法，考虑到自由能(x)(或者如果我们找不到潜在变量，那就用 x 表示能量)参数的可能梯度时便可以得到方程(5.10)。现在考虑一个高度正规化的两类概率分类器，他将尝试从标准模型 P (x)中分离训练集 P (x)，并可以得到一个概率 q(x) = P (y = 1|x)，十分接近 1/2(希望正确的更多一些)。向自由能一样，把 q(x) = sigm(—a(x))，i.e.，—a(x)当成判别式或者非标准的条件概率。凭经验用 p 指示(x,y)对的话，当 y=i 是 p_i 就是 x。假设 $P(y = 1) = P(y = 0) = 1/2$,那么，

$$\forall f, E_{\tilde{p}}[f(x,y)] = E_{\tilde{p}1}[f(x,1)]\tilde{p}(y=1) + E_{\tilde{p}0}[f(x,0)]\tilde{p}(y=0) = \frac{1}{2}(E_{\tilde{p}_1}[f(x,1)]) + E_{\tilde{p}_0}[f(x,0)]$$

运用这个公式，分类器的平均记录概率梯度为：

$$E_{\tilde{p}}[\frac{\partial \log P(y|x)}{\partial \theta}] = E_{\tilde{P}}[\frac{\partial(ylongq(x)+(1-y)\log(1-q(x)))}{\partial \theta}] = \frac{1}{2}(E_{\tilde{p}}[(q(x)-1)\frac{\partial a(x)}{\partial \theta}] + E_{\tilde{p}_0}[q(x)\frac{\partial a(x)}{\partial \theta}]) \approx \frac{1}{4}(-E_{\tilde{p}_1}[\frac{\partial a(x)}{\partial \theta}] + E_{\tilde{p}_0}[\frac{\partial a(x)}{\partial \theta}])$$

（5.31）

当分类器高度规则化时最后部分是相等的：当输出的权重很小且 a(x)接近于 0，$q(x) \approx 1/2$ 时，那么 $(1-q(x)) \approx q(x)$。这个表达式非常精确地得到了自由能可能表现的能量基础模型，此时，我们就能解释来自于 $\tilde{p}_1$ 积极的(y=1)(i.e.，$\tilde{p}_1 = \hat{p}$)和消极的(y=0,i.e.，$\tilde{p}_0 = p$)训练集了。梯度结构的对比与发散梯度估计也很相似(等式(5.27))。一种解释这种结果的方法是如果我们能够提高从标准模型中得到训练模型的可能性，我们就能通过设置大量的可能训练集来提高标准模型的可能性。实际上，这可能是一个分类器的判别函数被定义为一个生成模型(一个乘法因子)的自由能，并可以假设能从模型中(可能)得到标准模型。这个思想的变种被用来证明升压增量算法，从而增加了产品的专业性[201]。

第 6 章 深度学习架构的贪婪分层训练

6.1 深度信仰网络的分层训练

一个有着 l 分层的深度信仰网路 [73] 构成了一个联合分布，这个分布，由可观察到的向量 x 和 *I 隐藏分层的* h^k 所组成，如下所示：

$$P(\mathbf{x},\mathbf{h}^1,\ldots,\mathbf{h}^\ell) = \left(\prod_{k=0}^{\ell-2} P(\mathbf{h}^k|\mathbf{h}^{k+1})\right) P(\mathbf{h}^{\ell-1},\mathbf{h}^\ell), \qquad (6.1)$$

其中 $x = h^0$, $P(h^{k-1}|h^k)$ 是一个可视化的给定的条件分布在 RBM 中，并且与 DBN 的 k 层有关。而 $P(h^{*-1}, h^*)$是在最高级的 RBM 中的联合分布。

这些都展示在图 6.1 中了。

条件分布 $P(h^k|h^{k+1})$ 和最高阶的联合分布 (RBM) $P(h^{*-1}, h^*)$ 定义所生成的模型. 在下文中, 我们将会介绍字母 *Q* :准确的或者近似的模型后验概率, 我们会用它来推断或者训练. 后验概率 *Q* 是除了最高级的 $Q(h^*|h^{*-1})$之外，所有的近似值 ， $Q(h^*|h^{*-1})$ 其实等于 $P(h^*|h^{*-1})$ 因为 (h^*,h^{*-1}) 构成一个 RBM, 这样精确的推断就是可行的了。

当我们在贪婪分层方法中，训练 DBN 时, 正如算法 2 的伪代码所展示的, 每一个分层都被初始化了。

算法 2

TrainUnsupervisedDBN(P, *e*, £, *W*, b, c, mean_field_computation)

在一个无监管的环境下，训练一个 DBN, 通过贪婪分层步骤，其中 每一个增加的层都被训练成一个 RBM (例如通过对比离散).

P 是指网络的输入训练分布

e 是指 RBM 训练的学习效率.

£ 是指训练的分层个数

W^k 是指 k 层的权矩阵, *k* 在 1 到 £之中

b^k 是指可视化单元的抵消了向量 RBM 在 k 阶上，k 是从 0 到 1 的。

c^k 是指隐藏的单元抵消向量 RBM 在 k 阶，k 是从 0 到 1 的。

mean_field_computation 属于布尔数学体系的是真的，如果训练数据在每个额外的程度上是是由平均场近似所获得的而不是随机抽样。

for k = 1 to _ **do**

• initialize Wk = 0, **b**k = 0, **c**k = 0

while not stopping criterion **do**

• sample **h**0 = **x** from P

for i = 1 to k - 1 **do**

if mean field computation then

• assign **h**i

j to Q(**h**i

j = 1/**h**i-1), for all elements j

of **h**i

else

• sample **h**i

j from Q(**h**i

j

/**h**i-1), for all elements j

of **h**i

end if

end for

• RBMupdate(**h**k-1,_,Wk,**b**k,**c**k) {thus providing Q(**h**k/**h**k-1) for

future use}

end while

end for

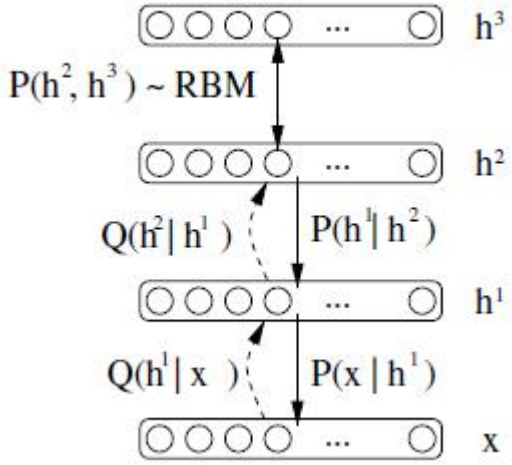


图. 6.1 深度信仰网络,作为一个有生长能力的模型.(通过 P 分布的生成路径,全弧) 并且是一种提炼多层的代表点(Q 分布的认知路径). 来自 RBM 前两个分层(他们联合分布). 低层次的分层构成一个定向的图像模型(s 形的信仰网络 h^2 ^ h^1 ^ x) 并且优先于 倒数第二层的 h^2 是被

最高层 RBM 提供的. $Q(h^{fc+1}|h^{fc})$ 近似 $P(h^{fc+1}|h^{fc})$ 但是可以跟简单地被计算.

作为一个 RBM, 并且我们用 $Q(h^k, h^{k-1})$ 代表 k-th RBM 用这种方式训练, 然而 $P(...)$ 代表 DBN 的概率. 我们使用 $Q(h^k|h^{k-1})$ 作为 $P(h^k|h^{k-1})$的近似, 因为他很容易来计算并且从 $Q(h^k|h^{k-1})$ 抽样, 并且不是来自于 $P(h^k|h^{k-1})$. 这些 $Q(h^k|h^{k-1})$ 同样可以被用作构造一个输入变量 x 的代表点. 为了获得这个近似的后验概率或者是所有层次的代表点,我们使用以下的步骤. 第一个样本 $h^1 \sim Q(h^1|x)$ 来自第一层 RBM, 或者是通过平均场方法, 令 $h^1 = \hat{}[h^1|x]$ 而非 h^1 的样本,其中期望是超过 RBM 分布 $Q(h^1|x)$的. 这个只是隐藏单位输出向量的概率, 通常的情况下他们是二项式单位: $h1 = sigm(b^1 + W/x)$. 选取平均场向量 h^1 或者样本向量 h^1 作为第二层的输入向量, 计算 h^2 或者 样本向量 h^2, 例如知道最后的分层. 一旦 DBN 被按照算法 2 训练,参数 W^l (RBM 权重) 和 c^l (RBM 隐藏抵消单位) 对于每个分层可以被用做最初化一个深度多层次神经网络. 这些参数可以之后被根据另一个标准微调(通常是一个监督式学习)。

一个 DBN 生成模型 x 的样本可以通过以下方式获得：

> 从最高级的 RBM 中取样一个可视的向量 h^{*-1}。这个可以近似地通过运行一个吉布斯序列在那个 RBM 中在 $h^* \sim P(h^*|h^{*-1})$ and $h^{*-1} \sim P(h^{*-1}|h^*)$交替变化来获得，正如在 5.3.1 所概述的。通过从一个由训练集例子的代表点 h^{*-1} 构造一个链条, 更少的 Gibbs 步骤可能被需要。

1. 对于 $k = £ — 1$ 到 1, 样本 h^{k-1} 给定 h^k 根据 k 阶的隐藏可视的条件分布 $P(h^{k-1}|h^k)$。
2. $x = h^0$ 是一个 DBN 样本。

6.2 训练堆放的自动编码器

自动编码器已经被用作构建模块来建造并且最初化一个深度多层的神经网络。这个训练的步骤是和深度信仰网络中的一种是一样的：希望是未监督的预锻炼在这个贪婪分层方法中已经赋予了参数。好的局部最优可以被达到通过局部下降法。这确实发生在一些任务中[17, 99, 153, 195]。

1. 将第一层作为一个自动编码器训练，从而最小化一些形式的自然输入的重构误差。这是纯粹无监督的。

2. 自动编码器的隐藏单位的输出量（例如：代码）是被用作另一层的输入量，同样被训练为一个自动编码器。再次强调，我们只需要未标记的例子。

3. 重复步骤（2）来最初化所想要的额外分层数

4. 选取最后的隐藏分层作为被监督的输入量并且最初化他的参数（无论是随机地还是通过监督训练，保持剩下的网络被修正）

5. 根据监督的标准，调整这个深度架构的所有参数。 或者，打开所有的自动编码器到一个非常深的自动编码器并且微调全局重构的误差，正如 [75]所示。

其中的原理是和之前为训练 DBNs 所提出的原理完全一样的, 但是是运用自动编码器而非 RBMs. 比较的实验结果表明了深度信仰网络典型地在堆放自动编码器上有优势。[17, 99, 195]. 这个可能是因为，相比起重构误差梯度，CD-k 是和对数似然函数梯度更接近。然而，因为重构误差梯度比 CD-k 有着更小的方差（因为没有抽样涉及）, 他也可能对结合两种标准, 至少在学习的最初阶段。 同样地要注意，DBN 的优势消失在试验中，其中普通的自动编码被去噪自动编码所取代[195]。并且是随机的 (见 7.2 节)。

使用自动编码而非 RBMs 作为无监督的深度架构的构成部分的优势是多数的分层的参数化方法是可能的，只要是训练的标准是连续的在这些参数上。另一方面，概率模型的等级对于 CD 或者其他已知的，易驾驭的对数似然函数的估计量可以被应用的当前是更加限制的。堆放的自动编码器的劣势是他们不符合于一个生成的模型：通过生成模型，例如 RBMs 和 DBNs，样本可以被定性的检查什么都被学习了。例如，通过观察图像和单词序列，这个模型看似是合理的。

6.3 半监督的和部分监督的训练

DBNs 和堆放的自动编码器，两种训练信号是可获得的, 并且可以被结合的: 局部分层非监管训练信号 (从 RBM 或者该层的自动编码器), 并且一个全局监督的训练 (从深度多层次的网络中 m 分享相同的参数作为 DBN 或者 堆放自动编码器). 在之前所展示的算法中, 这两个训练信息是被依次运用:第一个 非监督的阶段, 和第二个监督的微调的阶段. 其他的结合也是可能的.

一个可能是来结合两种信号在训练中, 并且这被叫做局部监督训练在 [17]中. 这个已经被发现十分的有用[17]当真实的输入分布 $P(X)$ 是被认为是不是与 $P(YX)$强烈相关. 来确信 RBM 保存了有关于 Y 的信息在其隐藏的代表点中, CD 跟新是和分类对数概率梯度结合的, 并且对于一些分布更好的预测从而被发现。

一个有吸引力的半监督学习的概括, 特别是在深度架构的环境下, 是自教学习的 [109, 148], 其中未标记的例子来自于层级而非标记的层级。这是更现实的相比起标准的半监督设置, 例如, 即使我们只对一些特定的对象阶层感兴趣, 可以更加简单的获得未标记的来自网络任意对象的例子(然而只选择与那些被选取的感兴趣的有关的阶层，会是很昂贵的。

第七章 受限波尔兹曼机和自编码器的变种

在本章我们介绍和讨论波尔兹曼机和自编码器的一些变种，这些变种是对这两个基本模型的进一步扩展和提升。我们已经提到过，将 RBM 中与可视单元或隐层单元相关的条件分布进行泛化是很直接的，如：将其泛化至指数函数族中的任意一个函数[200]。

高斯单元和指数或截顶指数单元在[17，51，99，201]中被提出和使用。考虑到此处的分析，三个等式可以很容易地被适配到这里，只需要简单地改变对 hi 和 xi 的求和(积分)域即可。对角二次项（如：生成高斯或截顶高斯分布）也可以被加入能量函数中，同时自由能量仍然可以被因子化。

7.1 自编码器和受限波尔兹曼机中的稀疏表示

近年来，人们开始对稀疏性这个概念表现出极大的兴趣，这种兴趣不仅存在于机器学习领域，还存在于统计和信号处理领域，特别是和压缩感知结合的领域[30,47]，但稀疏性这个概念早在计算神经科学中关于视觉系统稀疏编码的研究中就被提出了[139]。对于由自编码器变种构建而成的深度卷积网络，稀疏性一直是这些网络的一个关键要素[121,150,151,152,153]。此外，稀疏性还是 DBN（Deep Belief Networks，深度信念网络）的关键要素。

(译者注：原文该句词序可能有误，"and has been a key elelment deep convolutional networks exploiting of a variant of auto-encoders" 应为 "and has been a key elelment of deep convolutional networks exploiting a variant of auto-encoders")

7.1.1 为什么是稀疏表示

这里我们讨论的是，如果一个稀疏表征朝着固定大小的方向发展，考虑到每个样品的不同有效位数的话，那么稀疏表征相比于非稀疏表征在信息理论意义上更有效，根据学习理论，为了得到良好的泛化效果，全部训练集的比特值经过编码后，总的体量变小就足够了。在很多结构域中，感兴趣的不同例子经过压缩后往往要求不同的比特值。

另一方面，在降维算法中，不管是线性的 PCA 或者 ICA，亦或者非线性的 LLE 或者 isomap，都是为了让每个样本中的例子变换到低维度的空间。鉴于上面的讨论，将每个样品映射到可变长度的表示方法也是更有效率。为了简化讨论，我们这里假设这里的表示是一个二元的向量。如果我们被要求映射每个案例到一个固定长度的表征，一个比较好的解决办法可能是选择一个有足够多自由度可以展示尽量多的案例的表征，同时允许这个固定比特的向量可以真多大部分样例压缩为一个相对较小的可变代码的代码。这里我们现在有了两个表征：固定长度的，这个我们也许可以用来作为一个输入来作为预计，和做决策。

另外一个是更小的，可变可变大小的一个，原则上我们可以通过压缩固定长度的那个部分的步骤得到。距离来说，如果在我们固定长度固定长度的表征向量中 0 有一个相对是 0 的概率搞（比如，分布式的），那么对于大多部分来说压缩固定长度的向量是非常容易的，（稀疏数量的平均值）对于一个给定的稀疏水平，稀疏向量的立体基阵数目相对于较少稀疏（或者无稀疏）的强制话稀疏更小，所以，稀疏代码的熵更小。

另外一种赞成稀疏化矩阵的观点是固定长度表征可以被用来作为下一步分析的输入，所以它也比较容易说明。一个高度压缩的编码过程通常也是高度纠缠态的，所以没有一个子集的代码可以被解读出来，除非其他的比特也被考虑在内。反而，我们希望我们的固定长度的稀疏化表征可以有一种特性，就是它的每个比特或者这些比特的子集可以被解读出来，比如说，输入的相应的有意思的方面，和数据中捕获到的因素的变化。比如，演讲信号作为输入，如果一些比特编码演讲者的特征，另外一些比特编译发生因素的一般化特征，我们已经可以

分开一些数据中的变化因素，同时一些元素的子集也许足够解决一些特殊的问题。

另外一种证明稀疏化表征的方法[150]是在基于自编译的模型环境下提出的。这种观点其实解释了这样一种情况，在这种情况下，即使分类函数不能明确的最小化，或者接近最小化，只要其他的约束性条件。（如稀疏化）在学习表征中使用了就可以得到一个好的模型

假设表征被一个自编码学习到的是稀疏化，那么自编码器很可能不能被输入的模式重建，

因为稀疏化立体基阵的数目必须要比立体基阵小。

为了最小化训练数据中的重构误差，自动编码器需要找到一个能捕获数据分布的统计规律的表征。　首先，[150]要关联一种有重构误差的自由能（通过隐藏单元配置的最大化，用和取代以前的值）。因此，在训练数据集中最小化重建误差意味着最小化自由能，也就是说最大化基于能量似然模型的计数器(Equation (5.7)).。既然标准（分类函数）仅仅是所有可能输入配置的计数器的和，最大似然也就大致相当于是重建误差高于最可能的输入配置，同时又使它地域训练集。如果编码器（它映射一个输入到它的表征）在以一种他不能很好的代表大不能最可能的输入模式的条件下被约束，这种情况就可以实现。请注意，在代码远小于输入的时候，这个已经实现了。

另外一种方法是利用表征上的稀疏罚分法来实现，这种方法可以合并入训练标准里面。在这种方法中，这个属于的日志类似的梯度和分类函数的关联性是完全可以避免的，同时在隐藏层单元也可以通过稀疏罚分的方法来替代。有意思的是，这种方法在用来改进 CD-k　RBM 训练方面很有应用的潜力，这种方法只用日志梯度分类函数的一种似然估计就可以了。如果我们在隐藏表征上增加一个稀疏罚分，我们也许可以通过确定对大部分可能的输入配置增加自由能来对近似的弱点进行补偿，而不只是输入样例临近捕获到的对比差异负相位上的单元。

7.1.2 稀疏自编码器和稀疏编码

有许多方式可以强制让隐层表示获得某种形式的稀疏性。第一个成功利用稀疏表示的深度结构包含了自编码器[153]，该结构的稀疏性通过一种叫稀疏化逻辑的方法获得，BEGIN 通过近乎饱和的逻辑，适配其偏移使得编码结果只有在少数情况下为显著非 0 值 END。一年后，同一小组提出了一种更简单的变种[150]，该变种基于编码值的 Student-t 先验分布。在过去，Student-t 先验分布用于获取编码值 MAP 估计的稀疏性，该编码值在关于 V1 视觉皮层的计算神经科学模型中产生输入。另一种方法也和计算神经科学相关，该方法包含两层稀疏 RBM。通过增加正则项可获取稀疏性，该正则项将隐层的期望激活值的导数抑制在较低水平。在[139]可以看到，对图像进行单层稀疏编码，其效果和 V1 中滤波器效果非常相似，[110]发现，当训练稀疏 DBN（即两个 RBM 的级联）的时候，第二层网络学习到的特征与 V2 视觉皮层（即在灵长类动物视觉皮层主处理过程链中，跟随在在 V1 皮层之后的区域）中的特征类似。

在压缩感知领域，稀疏性通过对编码值施加 L1 惩罚获得，即，给定矩阵 W 的基（W 的每一列为基），我们寻找稀疏的 h 使得输入信号 X 以很低的 L2 误差得到重建：

$$\min_{h} ||\mathbf{x} - W\mathbf{h}||_2^2 + \lambda||\mathbf{h}||_1, \tag{7.1}$$

其中$||\mathbf{h}||_1 = \sum_i |\mathbf{h}_i|$.和有向图模型一样(比如在4.4中讨论过的sigmoid信念网络)，稀疏编码也进行了某种explaining away（译者注：原因转移）:（在多种关于隐节点值的配置中）选择其中一个，该配置能解释输入信号。这些不同的配置互相竞争，但其中一个配置被选中时，其他配置则完全被关闭。这种特性可同时被视作优点和缺点。优点在于，如果其中一个原因（译者注：即一个配置）比其他原因（译者注：即其他配置）更有可能，那么这个原因正是我们想凸显的原因(译者注：原文“than it is the one that we want to highlight”应为“then it is the one that we want to highlight”)；缺点在于，这样得到的编码值不太稳定，因为输入的微小波动都会导致完全不同的最优编码值h。如果更高层的转换或分类器将h作为其输入，那么这种不稳定会带来麻烦。如果非常相似的输入会导致稀疏编码层得到非常不同的表示，那么泛化是很困难的。这些方法在计算上还有一个弱点，有些作者试图消除该弱点：尽管能高效地对等式7进行优化，但仍比在普通自编码器或RBM上编码慢上几百倍，这使得训练和识别都很慢。另一个和稳定性相关的问题是基W和深度结构中更高层联合优化的问题。从编码精调(译者注：fine tuning没有对应的中文术语，暂译为 “精调”)目标的角度看，该问题显得尤其重要，精调的目的是使模型能够专注于信号最具有区分度的方面。如在9.1.2中所讨论的，如果有一个我们感兴趣的训练标准，该标准具有较好的区分度，则按此标准对深度结构的所有层进行精调，能够显著降低分类错误率。原则上可以通过编码值的优化计算梯度，但如果优化结果是不稳定的， 那么该梯度可能不存在或数值上不可信。为了同时解决稳定性问题和上述精调问题，[6]提出了用一个更“软”的近似物来代替L1惩罚，该近似物只会产生近似的稀疏系数(即很多接近于0的小值稀疏)。

请记住，稀疏自编码器和稀疏 RBM 不存在任何稀疏编码问题：(来自编码的)计算复杂度，编码值的稳定性，数值可靠性，以及深度结构全局精调时计算第一层的梯度的运算开销。稀疏编码系统只对译码器进行参数化：编码器隐含地定义为优化问题的解。而常规的自编码器或 RBM 有一个编码器(计算 P(h|k))和一个解码器(计算 p(x|h))。一系列关于稀疏自编码器的论文[150,151,152,153]提出了介于常规编码器和稀疏编码之间的模型,并将这些模型应用到模式识别和机器视觉任务中。这些文章提出让编码值 h 不受限(如同稀疏编码算法)，但包含一个参数化的编码器(如同常规自编码器和 RBM)，并对不受限的非参数化编码值 h 和参数化编码器输出之间的差异进行抑制。这样，最优编码值 h 试图满足两个目标：对输入进行重构（如同在稀疏编码中的做法），同时与编码器输出(因为编码器参数化比较简单，所有通过重构是稳定的)差异不会太大。在试验中，编码器只是仿射变换，再跟随一个类似 sigmoid 的非线性变换，译码器则是线性稀疏编码。实验表明，得到的编码值在深度结构上（伴随有监督精调）工作的非常好，并且比稀疏编码[92]得到的编码值更稳定(如，相对于输入图像的微小波动)。

7.2 去噪自编码器

去噪自编码器[195]是自编码器的随机版本，其输入被人为进行随机地破坏，但未被破坏的原始输入仍然作为模型的重建目标。从直觉上认为，一个去噪自编码器完成两件事：对输入进行编码（保留关于输入的信息）；对编码器输入进行去噪，即试图恢复被人为破坏的信号。第二个目标只能通过捕获输入信号间的统计依赖性来实现。事实上，在[195]中，对输入信号的随机破坏过程就是随机地将若干输入（大约一半的输入）置 0。因此对随机选择的缺失模式的子集，去噪自编码器试图从尚存的信号中预测缺失的信号。去噪自编码器的训练标准表示为重构对数似然度，

$$-\log P(\mathbf{x}/c(\tilde{\mathbf{x}})) \qquad (7.2)$$

其中 x 是未被破坏的原始输入，x 是被随机破坏后的信号，C(x)则是从 x 中获取的编码。因此解码器的输出可视作上述分布（关于未被破坏的原始输入信号的分布）的参数。在[195]的实验中，该分布是因子化的二项分布（每个像素 1bit），输入像素值可以认为是概率。注意更早的[174]提出了去噪自编码器的递归版本，其输入信号的破坏过程对应于某种形式的遮掩操作(对输入图像的某块矩形区域置 0)。利用自编码器去噪实际上在更早的[103,55]就被介绍了。因此[195]的主要创新在于展示了该策略如何成功地应用于深度结构的无监督预训练，并且将去噪自编码器和生成模型联系起来。

考虑一个随机 d 维向量 X，S 是一组 k 个索引，Xs = (Xs1，…,Xsk)是通过 S 选择的 X 的子元素集合，X_s 表示其余元素的集合。注意到，对于某些 S 子集，条件分布集合 P(Xs|X_s)完整刻画了联合分布 P(X)，并且该特性得到了利用，如吉布斯采样。如果|S| = 1,并且若干输入不完美地相关，则可能会存在以下问题：即时没有完全捕获联合分布，也能做到完美的预测，这

意味着吉布斯链不收敛。如果考虑随机大小的子集，并且坚持重构一切（就像常规自编码器），这类问题可以在去噪自编码器中被规避。

有趣的是，在一系列 8 个视觉任务的对比实验中，堆叠去噪自编码器构成的深度结构经过有监督精调后，其泛化性能整体好于堆叠常规自编码器，且与 DBN 性能相当或更好。

去噪自编码器的一个令人感兴趣的特征是其和生成模型相对应。它的训练标准是生成模型的对数似然度的边界。[195]讨论了若干可能的生成模型。一个简单的生成模型是半参数化的：采样一个训练样本，对其进行随机破坏，作为编码器的输入以获取隐层表示，再应用解码器（用于获取关于输入的分布的参数），然后选取一个输入。这么做不是十分让人满意，因为需要将整个训练集留作备用（类似于非参数化密度模型）。[195]还探索了其他可能的生成模型。

去噪自编码器的另一个令人感兴趣的特征是，其对部分数值缺失和有多个峰值(当多个峰值的子集对于任意特殊样本是可得到的)的数据有所增益。这是因为去噪自编码器在训练时就使用了部分数值缺失的训练输入数据（当输入的破坏过程就是随机隐藏了部分输入数值时）。

7.3 横向联系

受限玻尔兹曼机(RBM)可以通过引入交互项或可见单位间的“横向联系”来略微减少限制。在 P(h|x)中采样 h 依然很容易，但是在 P(x|h)中采样 x 现在普遍较为困难，并且相当于在马尔可夫随机场中取样，这也是一个被充分研究过的波尔兹曼机，其中偏移量等于 h 的值。[141]提出了这样一个模型用于捕捉图像的统计数据，其结果表明基于这个模型的深信度网(DBNs)生成比基于普通 RBMs 的深信度网更加真实的图像块。其结果还显示所得到的边缘成对的像素强度数据的分布类似于那些观察到的真实图像块。

这些横向联系可以比使用隐藏单位更容易获得成对的依赖，大大节省了用于获得高阶依赖的隐藏单位。在第一层的情况下，可以看出这意味着一种增白的形式，在图像处理系统中作为一个预处理步骤非常有用[139]。[141]中提出的想法是在各级 DBN（现在可以看作是马尔可夫随机场的层次结构）中使用横向连接。这类近似的普遍优势在于通过隐藏单位表示的更高级别要素不必编码低层次横向联系获得的局部信息。例如，在生成面部图像时，嘴和鼻子的大致位置在高级层面上可能被指定而他们的确切位置可能为了满足编码在横向连接较低层面上的成对偏好。这似乎有别于在部分的相对位置生成更尖锐边缘和一般更准确的图像，而不必扩展大量更高级别的单位。

为了在 P(x|h)中取样，我们可以在现有的例子中启动一个马尔可夫链（大概已经拥有近似与那些通过模型表示的像素的共同依赖，以便快速衔接），并且只在 x 上运行一个短链（保持 h 一定）。依据式(5.15)中一般玻尔兹曼机能量方程，用 U 表示可见-可见方阵间的联系。为了

减少在 CD 中对于这种模型的抽样误差，[141]在 x 上使用了 5 个衰减的平均场步骤而不是一个普通的吉布斯链：7.3　横向联系

受限玻尔兹曼机(RBM)可以通过引入交互项或可见单位间的“横向联系”来略微减少限制。在 P(h|x)中采样 h 依然很容易，但是在 P(x|h)中采样 x 现在普遍较为困难，并且相当于在马尔可夫随机场中取样，这也是一个被充分研究过的波尔兹曼机，其中偏移量等于 h 的值。[141]提出了这样一个模型用于捕捉图像的统计数据，其结果表明基于这个模型的深信度网(DBNs)生成比基于普通 RBMs 的深信度网更加真实的图像块。其结果还显示所得到的边缘成对的像素强度数据的分布类似于那些观察到的真实图像块。

这些横向联系可以比使用隐藏单位更容易获得成对的依赖，大大节省了用于获得高阶依赖的隐藏单位。在第一层的情况下，可以看出这意味着一种增白的形式，在图像处理系统中作为一个预处理步骤非常有用[139]。[141]中提出的想法是在各级 DBN（现在可以看作是马尔可夫随机场的层次结构）中使用横向连接。这类近似的普遍优势在于通过隐藏单位表示的更高级别要素不必编码低层次横向联系获得的局部信息。例如，在生成面部图像时，嘴和鼻子的大致位置在高级层面上可能被指定而他们的确切位置可能为了满足编码在横向连接较低层面上的成对偏好。这似乎有别于在部分的相对位置生成更尖锐边缘和一般更准确的图像，而不必扩展大量更高级别的单位。

为了在 P(x|h)中取样，我们可以在现有的例子中启动一个马尔可夫链（大概已经拥有近似与那些通过模型表示的像素的共同依赖，以便快速衔接），并且只在 x 上运行一个短链（保持 h 一定）。依据式(5.15)中一般玻尔兹曼机能量方程，用 U 表示可见-可见方阵间的联系。为了减少在 CD 中对于这种模型的抽样误差，[141]在 x 上使用了 5 个衰减的平均场步骤而不是一个普通的吉布斯链：x t = αx t-1 + (1 - α)sigm(b + U x t-1 + W ‘ h)，α

7.4 条件 RBM 和暂时 RBM

条件 RBM 是有一些参数不是自由的，而是使用有条件的随机变量的函数。例如，假设 RBM 是观察向量 x 和隐藏向量 h 的联合分布 P(x, h)，所带的参数(b,c,W)是等式（5.15），分别代表输入偏移量 b，隐藏单位偏移量 c，权值矩阵 W。文献[182,183]介绍为这种内容关联的 RBM 方法,在这种方法里隐藏单位偏移量 c 是关于内容变量 z 的仿射变换。因此 RBM 表示为 P(x,h|z)，或者忽视 h，表示成 P（x|z）。一般情况下，RBM 的参数 $\theta = (\mathbf{b}, \mathbf{c}, W)$ 可以写作参数化函数 $\theta = f(\mathbf{z}; \omega)$，例如，带条件变量 z 的有条件 RBM，其真实自由参数是 w。从 RBM 归纳出有条件的 RBM 需要建立深层结构，每层的隐藏变量是以其他变量的值为条件的（一般代表一些内容形式）。

RBM 的对比差异算法可以被简单归纳为条件 RBM 的事例。参数 θ 的 CD 梯度估计 $\Delta\theta$，可以被简单向后传播（back-propagated）为包含 w 的梯度估计：

$$\Delta\omega = \Delta\theta \frac{\partial\theta}{\partial\omega}. \qquad (7.3)$$

在仿射例子 $\mathbf{c} = \beta + M\mathbf{z}$（c，$\beta$，和 z 是列向量，M 是矩阵）被文献【183】研究，CD 简化了条件参数：

$$\begin{aligned} \Delta\beta &= \Delta\mathbf{c}, \\ \Delta M &= \Delta\mathbf{c}\,\mathbf{z}', \end{aligned} \qquad (7.4)$$

最后的乘法是输出结果（依据导数的链式法则），$\Delta\mathbf{c}$ 是根据 CD-K 在隐藏单位偏移的更新。

这个方法已经被成功运用到对人类运动的序列数据的条件分布函数 $P(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_{t-2},\mathbf{x}_{t-3})$ 的建模，Xt 是一个从人类运动例如走路、跑步捕捉到的数据中计算得到的包含有角度和其他几何特性的向量。有趣的是，这些允许产生真实的人类运动序列，通过先前抽样的 k 帧能够成功采样到第 t 帧，例如采用相似逼近

$$P(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T) \approx P(\mathbf{x}_1,\ldots,\mathbf{x}_k) \prod_{t=k+1}^{T} P(\mathbf{x}_t|\mathbf{x}_{t-1},\ldots,\mathbf{x}_{t-k}). \qquad (7.5)$$

初始帧可以通过使用特殊的空数据作为文本产生或者使用单独的模型 $P(\mathbf{x}_1,\ldots,\mathbf{x}_k)$。

正如文档【126】所证明的那样，它不仅仅可以用来产生偏移量，而且也可以对文本变量进行有条件性地分配权重。在那种条件下，我们可以极大增加自由度的数量，通过交互参数 ζ_{ijk}，使得对输入单元 xi，隐藏单元 hj，和文本单元 zk 三个数据之间的相互关系进行建模也成为可能。这种逼近方法在视频处理中使用 x 为一副图片，z 为之前的图片，这种模型学习捕捉流域【126】。

带有隐藏变量 ht（称为状态）的序列数据的概率模型能够获得很多增益，通过捕捉隐藏状态在序列中不同时刻 t 的暂时依赖关系。这允许隐马尔科夫模型（HMMs）捕捉长观测序列 x1，x2，。。。的依赖关系，即使模型只考虑隐状态序列 h1，h2，。。。作为序列 1（直接关系只有 ht 和 ht+1 的马尔科夫链）。在 HMMs 的隐状态表达 ht 是本地的（ht 的所有可能数值是列举的和与每一个数值的特定的参数），暂时 RBMs 已经被文献【180】建议构造一个分布式的状态表达式。这个方法是代表以上方法的条件 RBM 的延伸，但是内容不只包含过去的输入，也包含过去状态值，例如，我们构造一个模型

$$P(\mathbf{h}_t,\mathbf{x}_t|\mathbf{h}_{t-1},\mathbf{x}_{t-1},\ldots,\mathbf{h}_{t-k},\mathbf{x}_{t-k}), \qquad (7.6)$$

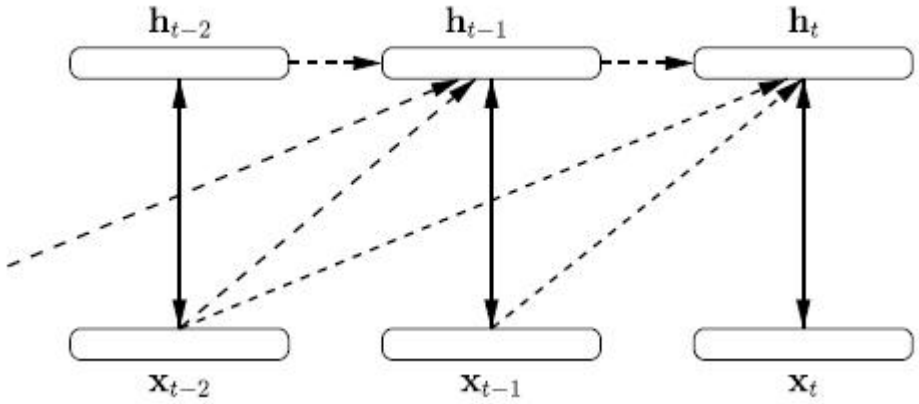


图 7.1 构造序列数据的暂时 RBM 的例子，包含隐状态之间的依赖。双箭头表示没有直接的关联，例如，RBM。单箭头虚线表示条件依赖：（xt，ht）RBM 被过去的输入值和过去的隐藏状态矢量所条件化。

内容是 $\mathbf{z}_t = (\mathbf{h}_{t-1}, \mathbf{x}_{t-1}, \ldots, \mathbf{h}_{t-k}, \mathbf{x}_{t-k})$，如图 7.1 所示。虽然暂时 RBM 产生的采样序列在条件 RBM 中可以完成（使用同样的 MCMC 逼近从 RBM 抽样，每次一步），隐状态序列的每一个推断给定一个输入序列不再是可追踪的。取而代之的是，文献【180】建议使用中值滤波来逼近隐序列尾部。

7.5 因式分解的 RBMs

在一些概率语言模型中， 学习每一个词的分布式表示已经被提及到过[15, 16, 37, 43, 128, 130, 169, 170, 171, 207]。对于一个词序列进行建模的 RBM，参数化是很方便的，它能够自动学习词汇表里每个词的分布式表示。这也是文献[129]提及到的。

对于一个 RBM 的输入 x，它是在固定规模的词序列 *(w1,w2,...,w*k)中每个词 *wt* 的一连串独热（one-hot）向量 vt ，也就是说在位置 wt，vt 包含除 1 以外所有的 0,在这里 x = (v1,v2,...,*v*'k)'.把 RBM 权重矩阵 W 的因式分解应用于两个因子上，其中一个因子取决于输入序列里的位置 t，另一因子则不会。在给定输入序列(v1, *v2,...,v*k)的情况下来计算隐藏单元的概率。不是直接应用矩阵 W 于 X,而是按下面的步骤：

首先，每个词符号 wt 通过矩阵 R 被映射到一个 d-维的向量 R .,wt= Rvt，这里的 *t* £{1 ...k}。

其次，一个连接的向量组 *(R' wi,R' W2,...,R'* ,Wk)'与矩阵 B 相乘。因此 *W* = BDiag(R)，其中 Diag(R) 是一个在对角线上由 R 填充的块-对角矩阵。这个模型产生了更好的对数似然 n 元语言模型[129, 130]，在平均预测最先进的 n 元语言模型时也进一步提高了泛化能力[129]。

7.6 推广 RBMs 和对比分歧

让我们试着推广 RBM 的定义以便于能够将我们前面讨论的方法和学习算法（例如对比分歧）直接用在一大类参数求解中。我们按照下面的方法来推广 RBMs:一个广义的 RBM 是一种基于能量的概率模型，含有输入向量 x 和隐藏向量 h，它们的能量函数使 P(h|x)和 P(x|h)均能够因数分解。可以通过能量函数求解参数的过程正式化这个定义，[73]里面对此也有过描述：

定理 7.1 如果一个具有方程 5.5 形式的模型满足 $P(h|x) = \prod_i P(h_i|x)$和 $P(x|h) = \prod_j P(x_j|h)$，则其对应的能量函数一定为下面的形式：

$$\text{Energy}(x, h) = \sum_j \phi_j(x_j) + \sum_i \xi_i(h_i) + \sum_{i,j} \eta_{i,j}(h_i,x_j). \qquad (7.7)$$

这里直接使用了 Hammersley-Clifford 理论[33,61]。[73]也证明上面的形式是得到互补先验的充分必要条件。合适的互补先验 P(h)能够让后验分布 P(h|x)因式分解。

当隐藏值和输入值都为二元变量时，这个新的公式实际上并没有额外的描述能力。事实上，由于(hi, xj)具有 2 × 2 的组态，因此 ηi,j (hi, xj)最多能取 4 个不同的值，这时 ηi,j (hi, xj)总可以表示为一个二阶多项式：a + bxj + chi + dhixj。其中，b 和 c 可以当作偏置项，a 可以当作一个全局可加的常量，对模型没有影响（因为 a 可以被配分函数抵消）。

另一方面，当 x 或者 h 是实向量的时候，我们可能会想更好地建模(hi, xj)的相互作用，可能是无参的，例如对 ηi,j 逐渐增加一些项使得更好的建模相互作用。另外，从条件密度 P(xj|h)或 P(hi|x)中抽样将是可行的，尽管 ηi,j 是一个复杂函数，这是因为它们是一维的概率密度，有效的近似抽样方法和数值积分都很容易计算（例如通过计算密度在嵌套的子区间或者窗口累计的和）。

这种分析突出了 RBMs 基本的限制，即它的参数求解过程只考虑了变量两两的相互作用。是因为 h 是隐藏的，还有我们可以选择隐藏单元的数目，我们才仍可以完整表示 x 的边缘分布（事实上，我们可以表示任何的离散分布）。在 7.4 节讨论的其他的 RBMs 变体允许三元相互作用[126]。

对于这个广义的 RBM 公式，对比分歧怎么更新呢？注意到方程 7.7 中的 ϕ_j's 和 ξ_i's 可以合并到$\eta_{i,j}$'s 中，为了简化符号，我们在后续的分析中将忽略掉他们。利用定理 5.1

$$\text{FreeEnergy}(x) = -\log\sum_h \exp\left(-\sum_{i,j} \eta_{i,j}(h_i,x_j)\right).$$

因此，一个样本 x 的自由能的梯度为

$$\frac{\partial \text{FreeEnergy}(x)}{\partial\theta} = \sum_h \frac{\exp\left(-\sum_{i,j} \eta_{i,j}(h_i,x_j)\right)}{\sum_{\tilde{h}} \exp\left(-\sum_{i,j} \eta_{i,j}(\tilde{h}_i,x_j)\right)} \sum_{i,j} \frac{\partial \eta_{i,j}(h_i,x_j)}{\partial\theta}$$

$$= \sum_h P(h|x) \sum_{i,j} \frac{\partial \eta_{i,j}(h_i,x_j)}{\partial\theta}$$

$$= E_h\left[\sum_{i,j} \frac{\partial \eta_{i,j}(h_i,x_j)}{\partial\theta} | x\right]$$

根据定理 7.1，我们仍然可以使用 Gibbs 链。在 k 步 Gibbs 链之后截断对数-似然函数梯度的展开式，然后用这条链的样本近似期望值，我们可以仅仅根据 Gibbs 样本 h1，hk+1，和 xk+1 得到训练样本 x1 处一个近似的对数-似然梯度：

$$\frac{\partial \log P(x_1)}{\partial \theta} \approx -\frac{\partial \mathrm{FreeEnergy}(x_1)}{\partial \theta} + \frac{\partial \mathrm{FreeEnergy}(x_{k+1})}{\partial \theta},$$

$$\approx \left(-\sum_{i,j} \frac{\partial \eta_{i,j}(h_{1,i}, x_{1,j})}{\partial \theta} + \sum_{i,j} \frac{\partial \eta_{i,j}(h_{k+1,i}, x_{k+1,j})}{\partial \theta}\right) \propto \Delta\theta,$$

其中$\Delta\theta$是模型参数θ的更新方式，对应于一个广义 RBM 的 CD-k。在多数参数求解中，我们都会有一个特定的参数θ以一种不需要加和的方式依赖于$\eta_{i,j}$'s。比如（在 hk+1 上求期望而非抽样的样本）我们重新写算法 1：

$$\eta_{i,j}(h_i, x_j) = -W_{ij} h_i x_j - \frac{b_j x_j}{n_h} - \frac{c_i h_i}{n_x},$$

其中 nk 和 nx 分别是隐藏单元与可视单元的数目，我们也重新得到了[200, 17]中描述的具有不同形式能量函数的和允许不同输入输出值的其他变体。

第八章　DBN 层次联合优化的随机变化边界

在本节中，我们讨论 DBN(深层信念网络)训练算法的数学基础。DBN 的对数似然度可以由 Jensen 不等式给出其下界，正如我们下面将讨论到的，这可以用于验证文献【73】中所介绍的 greedy layer-wise 训练策略，本文的 6.1 节也介绍了这一策略。我们将使用针对 DBN 联合分布的公式（6.1），将 h1（第一层隐藏向量）写作 h 以简化标记，并引入一个任意的条件分布 Q（h|x)。首先将 log(P(x))乘以 1=sum(Q(h|x))，之后使用 P(x)=P(x,h)/P(h|x)，并乘以 1=Q(h|x)/Q(h|x)，之后展开得到

$$\begin{aligned}
\log P(\mathbf{x}) &= \left(\sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x})\right) \log P(\mathbf{x}) = \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) \log \frac{P(\mathbf{x},\mathbf{h})}{P(\mathbf{h}|\mathbf{x})} \\
&= \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) \log \frac{P(\mathbf{x},\mathbf{h})}{P(\mathbf{h}|\mathbf{x})} \frac{Q(\mathbf{h}|\mathbf{x})}{Q(\mathbf{h}|\mathbf{x})} \\
&= H_{Q(\mathbf{h}|\mathbf{x})} + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) \log P(\mathbf{x},\mathbf{h}) + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) \log \frac{Q(\mathbf{h}|\mathbf{x})}{P(\mathbf{h}|\mathbf{x})} \\
&= KL(Q(\mathbf{h}|\mathbf{x})||P(\mathbf{h}|\mathbf{x})) + H_{Q(\mathbf{h}|\mathbf{x})} \\
&\quad + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) (\log P(\mathbf{h}) + \log P(\mathbf{x}|\mathbf{h})), \qquad (8.1)
\end{aligned}$$

其中 H_Q(h|x)是是分布 Q(h|x)的熵。由 KL 散度的非负性可得到如下的不等式

$$\log P(\mathbf{x}) \geq H_{Q(\mathbf{h}|\mathbf{x})} + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) (\log P(\mathbf{h}) + \log P(\mathbf{x}|\mathbf{h})), \qquad (8.2)$$

当 P 和 Q 相等时，该不等式中的等号成立，例如，在单层的情况下（即受限玻尔兹曼机，RBM）。鉴于我们选择了使用 P 来指代 DBN 中的概率，这里我们使用 Q 来指代 RBM 中的概率（第一层的 RBM）；而在方程中我们选择 Q（h|x）来作为第一层 RBM 的 hidden-given-visible 条件分布。我们对第一层 RBM 进行定义，使得 Q(x|h)=P(x|h)。而一般情况下，这两者并不相等。这是因为尽管首层隐藏向量 h1=h 的边界分布 P(h)是由 DBN 中的上一层决定的，但 RBM 的边界分布 Q(h)则仅由 RBM 中的参数所决定。

8.1 分解波兹曼机为无限有向深度信念网络（DBNs）

在使用上述似然的分解来说明对于 DBNs 的贪心式训练过程之前，我们需要建立在一个 DBN 中的 P(h1)和它所对应的由第一层有限波兹曼机产生的边缘分布 Q(h1)之间的联系。一个有趣的观测结果是，只要 h2 的维数与 h0 的维数相等都为 x，则存在一个 DBN，这个 DBN 的 h1 的边缘分布等于第一个有限波兹曼机的 h1 的边缘分布，即，P (h1)= Q(h1)。要看到这点，考虑一个第二层的有限波兹曼机，它的权重矩阵是第一层的有限波兹曼机的转置（这就是我们需要确保维数一致的原因）。于是，由于在一个有限波兹曼机的联合分布中的可见的和隐形的角色的对称性（当将权重矩阵进行转置），第二个有限波兹曼机的可见向量的边缘分布与第一个有限波兹曼机的隐向量的边缘分布 Q(h1)相等。

另一种来看到这点的有趣的方式由[73]给出：考虑始于 t = —to，结束于 t = 0 的无限 Gibbs 抽样马尔科夫链，对于第一个有限波兹曼机在 x 和 h1 之间交替抽样，其中可见向量抽样于偶数的 t，隐向量于奇数的 t。这条链可以被视为一条具有固定参数（所有的偶数步使用权重矩阵 W'，而所有奇数步使用权重矩阵 W）的无限有向信念网络。或者，如图 8.1 所示，根据 t 的对称性，我们可以获得任一具有权重矩阵 W 或 W'的有限波兹曼机所对应的从 t = —^到 t = t 的子链，并且得到一个有 1 — t 层 （不包括输入层）的 DBN。这个论点也说明了，一个两层的 DBN，如果它的第二层的权重是第一层的权重的转置，那么这个 DBN 就相当于一个简单有限波兹曼机。

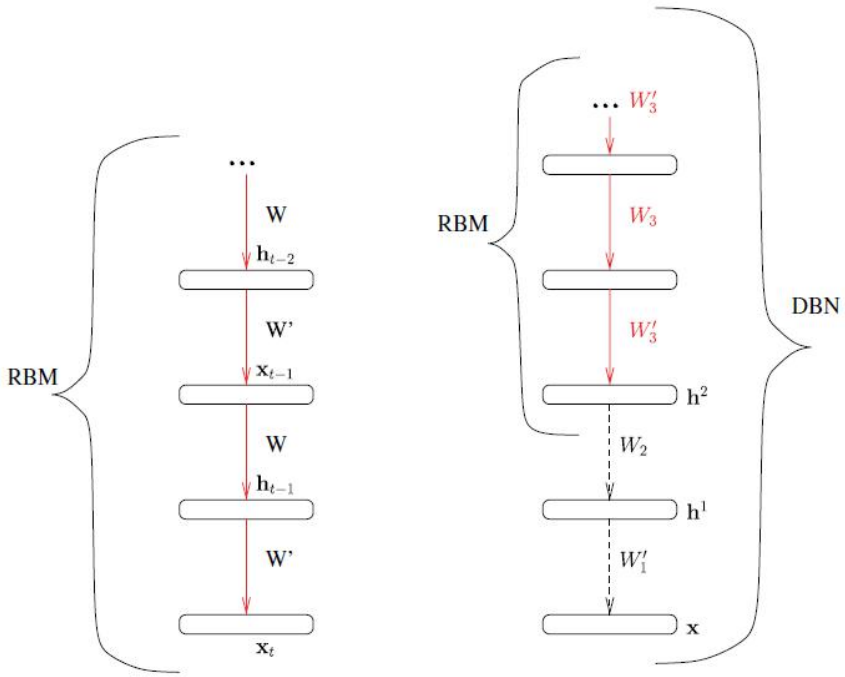


图 8.1：一个有限波兹曼机可以被分解为一条具有绑定参数的无限有向信念网络（详见文中）。左图，权重矩阵 W 或者它的转置的使用依赖于层索引的对称性。这个随机变量序列对应于一条 Gibbs 马尔科夫链来产生 xt（对于 t 很大）。右图，在一个 DBN 中的顶层的有限波兹曼机也可以以相同的方式分解，说明一个 DBN 是一个无限有向图模型，在这个模型中，某些层是固定的（实际上是除了底部的部分层之外所有层）。

8.2 逐层贪婪训练的可变理由

在这里，我们讨论了[73]提出的论点，即增加一个 RBM 层提高了 DBN 的可能性。让我们假设我们已经训练了 RBM 模型 x，它为我们提供了一个模型 Q（X），通过两个条件语句 Q(h1|x) and Q(x|h1)表达。

利用在上一小节的说法，现在让我们初始化一个同等的两层 DBN，即 P（X）=Q（X），通过将 P(x|h1) = Q(x|h1) 和 P(h1,h2) 和 P（H1，H2），它由一级 RBM 的权重转置得到二级 RBM 所给定。现在让我们回到等式（8.1），目的是通过改变 P(h1)提高 DBN 相似度，例如，固定 P(x|h1)和 Q(h1|x)不变，而允许第二级 RBM 发生变化。有趣的是，增加 KL 散度项能增加该可能性。从 P(h1|x) = Q(h1|x)开始，对 KL 项是零（即，只能增加），并且在方程式（8.1）中的熵项不依赖于 DBN P(h1),，使得 P(h1) 的少量增加保证 log P（x）的增加。我们也保证了 P（H1）项的进一步改善（即再培训第二 RBM 的，下文详述）不能带来对数相似度比第二 RBM 加上之前更低。这仅仅是因为 KL 和熵方面为阳性：进一步培训第二层 RBM 提高了对数似然度的下界（公式(8.2)），如[73]所述。这证明培训第二个 RBM 最大化的第二项，即预期超过ˆhi Q(h1|x)log$P$(h1)训练集。

因此，训练第二层 RBN 用以最大化

$$\sum_{\mathbf{x},\mathbf{h}^1} \hat{P}(\mathbf{x})Q(\mathbf{h}^1|\mathbf{x})\log P(\mathbf{h}^1), \qquad (8.3)$$

对于 P(h1)。这是最大似然判据一个模型，视实例 h1 为来自联合分布..P(x)Q(h1|x).的边际样品。如果我们保持一级 RBM 固定的，那么二级 RBM 因此可以用如下方式训练：训练集中的样本 x，然后是样本 h1 ~ Q(h1|x 的，并把 h1 作为第二级 RBM 的训练样本（即，作为观察其“可见的”载体）。如果有对 P (h1)无约束，上述训练标准的最大化将是它的“经验”或目标分配

$$P^*(\mathbf{h}^1) = \sum_{\mathbf{x}} \hat{P}(\mathbf{x})Q(\mathbf{h}^1|\mathbf{x}). \qquad (8.4)$$

同样的论点可证明增加了第三层，等等。我们获得第 6.1 节所列的逐层贪婪训练过程。在实践中该层尺寸改变的要求未被满足，因此用前一层[73, 17]权重的转置来初始化新添加的 RBM 也不是常见的做法，虽然这将是有趣的验证实验（在该尺寸约束施加）无论用前一层转置来初始化是否有利于加速训练

需要注意的是，我们将继续训练模型的顶部部分(这包括添加额外的层)，也不能保证 log$P$(x)（以训练集中的平均）将单调增加。随着我们下界的不断增加，实际对数似然度可以随之下降。让我们更仔细地审查这是如何发生的。这将要求当第二 RBM 继续培训，KL(Q(h1|x)||P(h1|x))项减少。然而，这一般是不可能的：随着 DBN 的 P(h1)越来越偏离 h1 上第一 RBM 的边际 Q(h1)，很可能后验的 P(h1|x)（来自 DBN）和 Q(h1|x)（来自 RBM）偏离越来越多（因为 P(h1|x)的一个 Q(x|h1)P(h1)P（H1）和 Q(h1|x)的一个 Q(x|h1)Q(h1)），使得方程（8.1）中的 KL 项增加。随着第二个 RBM 训练相似度的增加，P(h1)平滑地从移动顺利地从 Q(h1) 移向 P*(h1)。因此，这似乎很合理：第二 RBM 的持续培训是要增加 DBN 的似然性（不只是最初）并且通过传递性，增加更多的层也可能会增加 DBN 的似然性。然而，这是不正确的：用任何参数配置初始化的第二 RBM，增加其训练似然性保证了 DBN 似然性增加，因为至少可以发现一个病理反例，（I. Sutskever,个人通讯)。考虑其中第一 RBM 的具有非常大的隐蔽偏置的情况下,使 Q(h1|x) = Q(h1) = 1hi=h = P*(h1)，但大的权重和小的可见偏移，使得 P(xj|h) = 1Xi=hi,，即隐藏的向量已被复制到可见的单位。当以第一 RBM 的权重转置来转初始化第二 RBM，第二 RBM 的培训似然性不能得到改善，也不能增加 DBN 似然性。然而，如果第二 RBM 的是从一个“糟糕”配置得来的（其训练似然性更差，并且 DBN 似然性也更坏），那么 P(h1)将朝着 P*(h1) = ^(h1)，从而使所述第二 RBM 的似然性提高，同时对 KL 项将减少，DBN 似然性将会降低。正确（用第一 RBM 复印件）初始化第二 RBM 时，这些情况不可能发生。因此，我们是否能找到（除了以上提到之外）条件，以确保当第二 RBM 似然度增加时 DBN 似然度也同样增加，这个问题悬而未决。

另一种说法来解释为什么贪婪程序的工作原理如下所示（Hinton, NIPS’2007 教程)。用于第二 RBM 的（样品 P*(h1)的 h1 样品）的训练分布，比起原始训练分布-P(x)，看起来更像由 RBM 生成的数据。这是因为 P*(h1)是通过应用 RBM 吉布斯链上的一个分步骤从 P(x)中的例子获得的，而我们知道，很多应用吉布斯步骤会从那个 RBM 产生的数据。

不幸的是，当我们这个贪婪逐层过程中训练的 RBM 不会是一个 DBN 的顶层（top-level level?），我们没有考虑到这样一个事实：随后容量会增加以提高对隐单元的优先级。 [102]提出了考虑对比分歧（Contrastive Divergence）的替代方案，用以训练 RBM 注定要初始化 DBN 的中间层。我们的想法是要考虑，以具有非常高容量的模型（DBN 的上级）对 P(h)进行建模。在这个充满无限容量的极限情况下，可以写下了最佳的 P(h)将是：这仅仅是经验分布的随机变换，通过随机映射 Q(h|x)的第一 RBM（或先前的数个 RBM），例如，第二层情况中方程（8.4）中 P*。

实验[102]证实，这个标准得到了更好的 DBN（以这个 RBM 初始化）优化。不幸的是，这个标准是不是易处理的，因为它涉及到了所有隐藏向量 h 配置的总和。它的易处理近似值可能被考虑了，因为这个标准看起来像是在随机自动编码形式（类似一个人提出去噪自动编码器一个生成模型[195]）上重建误差的形式。另一个有趣的替代方案——这将在接下来的部分中探讨——是直接作用于一个 DBN 的所有层的联合优化。

8.3 Joint Unsupervised Training of All the Layers

We discuss here how one could train a whole deep architecture such as a DBN in an unsupervised way, i.e., to represent well the input distribution.(该句译者未翻译，请审校补充)

8.3.1 睡醒算法（死去活来算法）

睡醒算法被介绍用来训练反曲信网络的（即，上层单位因式分解分配）。它是基于一个用来扮演生成模型 P(h,x)的变分近似的可“识别”模型 Q(h|x)（其中，Q(x)用来训练集分配）。在这里，我们用 h 表示所有隐藏层。在一个 DBN 中，Q(h|x)在上面的章节中被定义（章节 6.1），每一层获得从输入层向更高层的向上随机传播样本。在睡醒算法中，我们解耦了识别参数（向上权重，用来计算 Q(h|x)）到生成参数（向下权重，用来计算 P(x|h)）的过程。算法基本思想比较简单：

1.醒阶段：训练集样本 x，生成 h ~ Q(h|x)以及使用(h, x)作为全观察数据来训练 P(x|h)和 P(h)。这和做一步随机梯度一致并遵从下式：

$$\sum_{h} Q(h \mid x) \log P(x, h) \tag{8.5}$$

2.睡阶段：来自模型 P(x,h)的样本(h,x)，使用这一对作为全观察数据来训练 Q(h|x).这和做一步随机梯度一致并遵从下式：

$$\sum_{h,x} P(h, x) \log Q(h \mid x) \tag{8.6}$$

在参考文献[73]中提及睡醒算法的使用是为了 DBNs，正如前面探讨的，每一层联系的权重已经被训练位 RBMs。对于一个层数为($h^1,...,h^\zeta$)的 DBN，为了更新顶端 RBM(从 $h^{\zeta-1}$到 h^ζ)的权重，其醒阶段被执行考虑到样本 $h^{\zeta-1}$（从 Q(h|x)中获得）作为顶端 RBM 的训练数据。

一个变分近似可以用来辩证睡醒算法。log 似然分解参照方程(8.1)

$$\log P(\mathbf{x}) = KL(Q(\mathbf{h}|\mathbf{x})||P(\mathbf{h}|\mathbf{x})) + H_{Q(\mathbf{h}|\mathbf{x})} + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x})\left(\log P(\mathbf{h}) + \log P(\mathbf{x}|\mathbf{h})\right), \qquad (8.7)$$

此式表明 log 似然可以从下面与 Helmholtz 自由能量 F 的相反面而被定界：

$$\log P(\mathbf{x}) = KL(Q(\mathbf{h}|\mathbf{x})||P(\mathbf{h}|\mathbf{x})) - F(\mathbf{x}) \geq -F(\mathbf{x}), \qquad (8.8)$$

其中

$$F(\mathbf{x}) = -H_{Q(\mathbf{h}|\mathbf{x})} - \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x})\left(\log P(\mathbf{h}) + \log P(\mathbf{x}|\mathbf{h})\right), \qquad (8.9)$$

当 Q = P 时，不等式是封闭的。变分方法基于当尝试使边界封闭时来最大化下界-F。即，最小化 KL(Q(h|x)||P(h|x))。当边界封闭式，-F(x)的增量很可能产生 logP(x)的增量。因为我们解耦了 Q 与 P 的参数，我们看到了睡醒算法的两个阶段所做的。在醒阶段，我们通过训练集样本 x 来考虑固定 Q 以及做一步随机梯度来最大化 F(x)中最期望的值，遵从 P 的参数（即，我们不关心 Q 的熵）。在睡阶段，理想情况下，我们尽可能使 Q 接近 P，从某种意义来说应该最小化 KL(Q(h|x)||P(h|x))（即，让 Q 做参考），但是，我们用最小化 KL(P(h,x)||Q(h,x))来代替，让 P 做参考，这是因为 KL(Q(h|x)||P(h|x))很难处理。

8.3.2 将 DBN 转换为玻尔兹曼机（Boltzmann Machine）

最近有人提出了另一种方法，在评估中得到的结果优于使用 Wake-Sleep 算法[161]。

最近有人提出了另一种方法，在评估中得到的结果优于使用 Wake-Sleep 算法。已经在 6.1 节讨论过将每一层初始化为 RBM，这样初始化之后，DBN 被转换为相应深度的玻尔兹曼机。因为在一个玻尔兹曼机中，每一个单元接受来自上层和下层的输入，所以初始化分层的 RBMs 的深玻尔兹曼机时，推荐使用 RBM 一半的权重。要注意到一点，深度玻尔兹曼机的 RBM 初始化，对于得到好的结果很重要。

之后，这些作者提出了玻尔兹曼机（见 5.2 节和式 5.16）的正相和负相梯度的近似值。对于正相梯度（原则上需要 x 固定，从 P(h|x)抽样），他们提出了一种对应于均场松弛的变分逼近方法,对于负相梯度(原则上需要从联合 P(h,x)抽样),他们推荐使用在 5.4.1 节讨论及在[187]介绍的回归 MCMC 链（马尔卡夫链蒙特卡罗理论）的方法。

这种方法要保存（h,x）状态（）的集合，而(h,x)状态要根据当前模型（例如，根据在前

一步骤给定的概率，进行每一个单元抽样）以一个 Gibbs 步长更新。

即使这些参数保持变化（十分缓慢），我们仍用相同的马尔科夫链替代新的（就像在旧的玻尔兹曼机算法[77,1,76]一样）这种方法似乎效果很好。基于 DBNs 在 MNIST 数据集上取得了进步，在对数似然数据方面（用退火抽样估算）和分类误差（经过监督微调），也将错误率从 1.2%下降到 0.95%。最近，[111]也将训练的 DBN（有卷积结构的 DBN）转换为一个深度玻尔兹曼机，用来从中进行抽样。

第九章 展望未来

9.1 全区优化策略

正如4.2章节讨论的，较好的泛化会观察到深度架构中的局部层次无监督预训练，对此的部分解释是通过使用较好的无监督模型相关联的参数空间来初始化监督训练，更进一步的帮助优化底部层次（近输入）。同样的，想要获得更好的结果，也需要初始化深度波尔兹曼机的每个层作为一个限制波尔兹曼机（RBM），这是非常重要的。在这两种处理中，我们都是在对整个深度架构细微调整前，优化了局部层次的代理规则。

在此，我们已经画出现有的工作和方法之间的联系，并可以采用延拓法的原理来帮助处理困难的优化问题。虽然，他们无法提供保证获得的是全局性的最优化方法，但是这些方法对于计算化学中寻找困难的优化分子结构问题的近似解特别有用。基本思想是先解决容易的、被平滑版本的问题，再逐渐考虑没有被平滑的，直觉上讲平滑版本的问题可以揭开全局画面，就像模拟退火算法。一个定义单参数族的价值函数 $C_\lambda(\theta)$，C_0 能够被更容易的优化（可能是 θ 的凸点），C_1是我们确实期望的最小准则。第一个最小化$C_0(\theta)$,逐步增加 λ,保持 θ 在$C_\lambda(\theta)$局部最小化的区间内。通常C_0是高于C_1平滑版本，以至于 θ 逐渐的移入C_1显著最小值的吸引力的底部。

9.1.1 作为可持续方法的深度信念网络 DBNs（Deep Belief Nets）的贪婪逐层训练

在第6.1章节中已经描述了深度信念网络DBNs的贪婪逐层训练算法，而下面的介绍将这种算法视为一种近似持续的方法。首先回忆一下（第8.1章节），深度信念网络（DBN）的顶层限制波尔兹曼机（RBM）可以展开成一个带有参数的无限有向图模型。在贪婪逐层处理的每一步过程

中，我们都是从倒数第二层参数来解开顶层限制波尔兹曼机（RBM）的参数。逐层处理的过程正如下面所讲的。模型的结构是保持相同的，都是一个S型自信层的无限链，但是我们改变了逐层处理每一步的参数约束。最初所有层之间都是紧张的。在训练第一个限制波尔兹曼机（RBM）后（例如，在这个约束下优化），我们就会从松散的层里解开第一层参数。在训练第二个限制波尔兹曼机（RBM）后（例如，在略微松散约束下优化），我们从松散的层里解开第二层参数。这并不是一个连续的训练准则，我们这里存在一个逐渐的、更困难的优化问题的离散序列。在他们已经训练和仅仅优化了k＋1层之后，我们通过使用过程贪婪确定了k层的参数，例如训练一个限制波尔兹曼机（RBM）。做一个严格的比喻，我们相当于用前一个的转置阵初始化新加入的RBM的权重。注意不是优化所有的参数，贪婪的逐层方式仅仅优化新加入的。但是，即使有这些相近性，分析也是指出这是在寻找更好解的期间内一个具有很好性能的逐层训练方式的解释。

9.1.2 非监督向监督的过渡

很多论文中的实验表明，对于深度结构，先非监督的训练再监督精化可以取得非常好的效果。然而之前的非监督和监督规则结合的工作[100]侧重于非监督准则的正则化效果（），4.2节中的讨论表明深度网络非监督预训练所得到的效果的改善部分是由于深度结构中较低层的优化。

许多近期的工作主要先用非监督的表示学习算法（比如稀疏编码）然后用区分准则或者结合区分准则和非监督准则来精化这些表示[6, 97, 121]。

在[97]中，RBM通过一个两部分的可见性向量来训练，该向量同时包含输入x和目标类y。这样的RBM可以通过训练来建模联合概率$P(x,y)$也可以建模条件概率$P(y|x)$（条件对数概率的准确梯度是可知的）。已有的最好结果[97]结合了两种准则，但是模型初始化采用了非区分准则。

在[6, 121]中，在稀疏编码系统中训练解码基的任务和在这些稀疏编码之上训练一个分类器的任务相结合。在使用非区分学习初始化了解码基之后，可以用同时应用于表示参数和分类参数的区分准则来精化。根据[121]，试图直接优化监督准则而不首先用非区分训练初始化会产生很差的结果。事实上，他们提出了一种从非区分准则向区分准则的平滑过渡，从而采用一种连续方法来优化区分准则。

9.1.3 控制温度

即使优化一个RBM的对数似然函数，可能已经是一个困难的优化问题了。使用随机的梯度（比如从CD-k得到的）和较小的初始权重就会比较接近连续方法，并且很容易转化成一个连续方法。考虑对应于RBM的正则化路径[64]的一类优化问题。例如参数L2正则化，以$\lambda \in (0,1]$为参数表示的训练规则：

$$C_\lambda(\theta) = -\sum_i \log P_\theta(x_i) - \|\theta\|^2 \log \lambda \qquad (9.1)$$

当$\lambda \to 0$时，我们有$\theta \to 0$，并且可以看出RBM对数似然函数变成θ的凸函数。当$\lambda \to 1$时，不存在正则化（注意到如果训练集较小的话，某些中间的λ值可能通用性会更好）。控制RBM中偏移和权重的大小等价于空on控制Boltzmann机中的温度（能量函数的尺度系数）。高的温度对应着随机性高的系统，极限状况就是在输入上的分形的均匀的分布。低的温度对应于更确定性的系统，只有一小部分的设定是可变的。

有趣的是，通常观察到随机梯度下降从较小的权重开始，让权重指数级地增大，从而近似遵从正则化路径。提早终止是周知的一种有效控制计算能力的技巧，在训练过程中通过一个验证集监测性能，并且保留使得验证集误差最小的参数。提早终止和L2正则化之间的数学联系已经建立了[36, 211]：从较小的参数开始进行梯度下降会得到逐步增大的参数，对应于正则化程度渐弱的训练规则。然而，依靠常规的随机梯度下降（无显式的正则化项），无法保证我们追踪的是依据公式9.1的λ序列所对应的局部最小的序列。稍微修改随机梯度算法也或许可以使之更好地追踪正则化路径（亦即使之更接近连续方法）：显式控制λ，当优化已经足够接近当前λ对应的局部最小时，逐渐增大λ的值。注意到同样的技术也可以加以扩展用于机器学习领域的其他复杂非线性优化问题，诸如训练一个深度监督神经网络。我们希望从一个全局最优解开始并逐渐追踪局部最小，从强正则化开始缓慢变为弱正则或非正则。

9.1.4 定型：采用课程的训练

另一种连续方法的实现，可以通过逐步变化训练任务，从简单的任务（其中的样本表达简单的概念）过渡到目标任务（有更复杂的样本）。人类被训练成完全适应社会的成年人需要大约20年。这种训练是高度组织化的，依靠教育系统和一套课程，在不同的时间引入不同的概念，充分利用之前学习的概念来使得学习新的抽象变得简单。通过一套课程来训练学习机器的思想最晚在[49]就有了。基本的思路是循序渐进，先学习任务的简单部分或者较简单的子任务，接着逐渐提高难度等级。从构造表示的观点，这种思想是先学习对低层抽象的表示，然后通过对他们进行组合来学习稍微高一些的抽象，并为更复杂的结构做准备。通过选择展示什么样的样例以及采用何种顺序进行展示，我们可以对学习过程进行指导并显著加快学习者的速度。这种思想叫做定型，在动物训练中被广泛采用[95, 144, 177]。

定型和课程的使用也可以看成是一种连续方法。我们设想这样一个学习问题：对来自于训练分布$\hat{P}$的数据进行建模。主要思路是根据某个给定的课程，从最简单的样本逐渐向描述更加抽象的概念的样本过度，重新计算从训练分布采样的样本概率。在课程中的t点，我们采用分布P_t进行训练。令$P_1 = P$，P_0选取使之易于学习。和其他的连续方法一样，当学习者在课程中的当前点t达到局部最小时（即已经足够掌握了P_t中的样本），沿着课程移动。根据训练样本的概率分布的平滑变化，来微调t，从而可以构造一条连续的路径，从一个简单的学习问题逐渐到所需的训练分布。[20]进一步发展了这种思想，对比了仅通过目标分布，和通过一个逐渐逼近目标分

布的课程，分别对视觉和语言的任务进行实验，结果表明采用课程的方法具有更好的推广性。

定型或者说采用课程的思想和贪心逐层思想之间存在着相通之处。这两者都是我们想要通过先学习适当的底层抽象来简化高层抽象的学习。在逐层的方式中，我们通过在之前学习的概念之上进行构造，来逐渐增加学习者的能力。通过在课程的思想中，我们控制训练样本从而确保在处理高阶难度的样本之前，已经学习了简单的概念。人类在没有先理解新思想所需的概念的情况下试图掌握新的思想所面临的苦难呢，说明过早处理更加复杂的概念的样本可能是浪费时间。

依据课程的思想，我们在学习者和训练分布之外，引入了教师的角色。教师可以使用两个信息源来决定课表：a)一系列的概念以及可以简化学习过程的顺序的先验知识b)通过观察学习者的进度来决定何时进入课程的下一阶段。教师必须选择一个合适的难度等级的新样本，在过于简单和过于困难之间取得平衡。过于简单，学习者无需调整模型便可适应新的样本；过于困难的话学习者无法通过渐进的学习来适应新的样本从而导致新样本可能被当作误差点。

9.2 为什么无监督学习如此重要

本文认为：对于旨在实现人工智能的深度架构，要实现成功的学习算法，强大的无监督或半监督（或自学习）学习是至关重要的组件。

这里我们简要给出支持这一假设的论据：

- 带标样本的稀缺性，以及相比之下未标记样本的可用性（可能不限于自学习[148]中举例的那些感兴趣的类别）。
- 未知的未来任务：如果一个学习agent不知道将来它要处理什么样的学习任务，但是知道该任务将被定义在现在可观察的一个世界（比如，随机变量）范围内，那么收集和集成关于这个世界的尽可能多的信息以用于学习其运行的机制，就是十分合理的。
- 一旦学习到了一个很好的高层级表示，其他学习任务（例如，监督学习或增强学习）可能会容易得多。我们知道，例如，如果使用合适的核（亦即合适的特征空间），核机器的方法会非常强大。同样，我们知道，在基本上是通过合适特征的线性组合来获取动作的情况下，强大的增强学习算法是可以得到保证的。我们不知道合适的表示应该是什么样的，但如果能够捕获输入数据变量的显著因素并将其分离出来，那就很靠谱了。
- 逐层无监督学习：这在第4.3节已有讨论。多数学习可以利用结构中在一个层级或子层局部可用的信息来完成，从而避免有监督梯度的假设问题带着大扇入元素传播至很长的链条。
- 将前述两点联系起来，可以得出，无监督学习可以把有监督或增强学习机的参数放入一个区域，从这个区域梯度下降（局部优化）将产生优解。这已在若干场景中被经验方法证实，特别是在图4.2及 [17 , 98 , 50]中的实验。

- 施加于优化过程的额外约束，要求模型不仅捕获输入-目标依赖性，还要有输入分布的统计规律，这有助于避免生成明显的局部最小值（不能对应于输入分布的良好模型）。注意到，一般情况下额外的约束也可能会产生局部最小值，不过我们通过实验观察到[17]，训练和测试的误差都可以通过无监督的预训练来降低，说明无监督的预训练使空间区域中的参数更接近于对应学到较佳表示（在较低层级上）的局部最小值。有人[71] 认为（但值得商榷），无监督学习比有监督学习更不容易过拟合。深度架构通常用于构造有监督的分类器，并且在此情况下，无监督学习组件可以明确作为一个正则项或先验项[137 , 100 , 118 , 50] ，使得结果参数不仅对于给定类别的输入的建模有意义，还对于捕获输入分布的结构也有意义。

9.3 待解决的问题

深层架构的研究还很初期，许多问题仍没有答案。

以下是可能的有意思的问题。

1. 关于回路中计算深度的作用，我们的研究成果可否泛化至超越逻辑门和线性阈值单元？

2. 对于各种人工智能任务，要达到人类水平的性能，是否存在一个基本上够用的计算深度？

3. 对于大小固定的输入，其回路深度的理论结果，可否泛化到带上下文及递归计算可能性的操作时动态回路？

4. 对于从随机初始化开始的深度神经网络，为什么基于梯度的训练往往是不成功的？

5. 通过CD训练的RBMs是否可以很好的保持与其输入一致的信息（因为它们未以自动编码器进行训练，可能会损失一些输入信息，这在之后会变得很重要），如果不能，如何弥补？

6. 深度架构的监督训练条件（对于深度玻尔兹曼机及DBNs可能为对数似然）是否真的满是实际不良局部最小值，或者仅仅是该条件对所尝试的优化算法（如梯度下降和共轭梯度）而言太复杂了？

7. 训练RBMs的时候，局部最小值的出现是否是一个重要的问题？

8. 我们可否将RBMs和自动编码器替换为擅于提取良好表示但又涉及较为简单的优化问题的算法，甚至可能是凸的？

9. 目前的深度架构训练算法涉及许多阶段（每层一个阶段，另加上全局微调）。这在纯在线情形下不太实际，因为一旦我们进入微调，可能会陷入明显的局部最小值。是否有可能得到一个完全在线的训练深度架构的程序，可以一直保持无监督的组件？注意到这正是[202]引人关注的理由。

10. 对比散度(CD)中Gibbs步骤的数量，是否应该在训练的过程中调整？

11. 考虑上计算时间，我们能否显著改善对比散度？最近有些新的替代方案被提出，值得进一步研究[187 , 188]。

12. 除了重建误差，是否有其它更合适的方法来监测RBMs和DBNs训练的过程？等效地，RBMs和DBNs是否有可控的近似划分函数？这个方向的近期工作 [163 , 133] 采用了退火重要性采样，令人鼓舞。

13. RBMs 和自动编码器可否通过对其所习得的表示实施某种形式的稀疏惩罚，从而加以改进？这样做的最好方式是什么？

14. 在不增加隐单元数量的情况下，RBM的容量可否使用其能量方程的非参数形式来增加？

15.由于我们对单一的降噪自动编码器只有一种生成模型，是否存在一个概率解释，以描述在堆叠自动编码器或堆叠降噪自动编码器中学到的模型？

16. 贪婪逐层算法对于深信度网络训练（在最大化训练数据的似然度方面）的有效性如何？会不会过于贪婪？

17. 对于深信度网络及相关的深度生成模型，我们可否得到其对数似然梯度的低方差低偏向的评估值，也就是说，我们可否联合训练所有的层（对于无监督的目标）？

18. 这里所讨论的无监督层级训练过程有助于训练深度架构，但实验表明，训练仍然会陷于明显的局部最小值，且不能在非常大的数据集中发掘出所有信息。果真如此吗？我们可否开发出更为强大的深度架构优化策略，来突破这些限制？

19. 基于连续方法的优化策略，可否带来深度架构训练的显著改进？

20. 除了深信度网络、堆叠自动编码器和深度玻尔兹曼机之外，是否还有其他有效的可训练的深度架构？

21. 是否需要一个课程表，来学习各种高层级抽象，就像人类花几年乃至几十年来学习的那样？

22. 在训练深度架构中发现的原理，可否应用或泛化于训练循环网络或动态信度网络，这些网络要学习上下文或长项依赖的表示。

23. 深度架构如何泛化至可表示那些看起来不容易用向量来表示的信息，因为这些信息由于其性质，具有可变的大小及结构（例如，树、图）？

24. 虽然深信度网络原则上适用于半监督和自学习的学习场景，但什么是使当前深度学习算法适应这些场景的最佳方法，这些方法相比已存在的半监督算法表现如何？

25. 当有可用的标注样本时，监督和无监督的条件应该如何结合，方可学习表示输入的模型？

26. 我们可否从大脑中找到类比，以说明对比散度和深信度网学习计算的必要性？

27. 大脑皮层与前馈申请网络不一样，存在明显的反馈连接（例如，从视觉处理的较晚阶段返回到其较早阶段），而这可能不仅在学习（如RBMs）中起作用，还将上下文先验与视觉证据融合起来[112]。什么样的模型可以引发深度架构中的这种交互，并正确学习这种交互？

第十章 结论

本文始于以下几点动机：首先对于人工智能我们采用学习的方法，其次,一个问题在直觉上可以被分解为多个层次的计算和表征，再者理论结果也表明，缺乏足够此类层次的计算结构需

要大量的计算元素，并且我们也观察到依赖于局部普遍化的学习算法在学习高度差异化的函数时难以普遍化。

对于结构和算法，我们首先考虑数据的分布式表征，这让输入的抽象特征的大量可能性构造具有可行性，使得一个系统可以紧凑地表征每个例子，从而让形式丰富的一般化成为可能。本文也着眼于成功训练可以学习多个层次的分布式表征的深度结构的困难。虽然在这种情况下标准的基于梯度的方法并不可行的原因还有赖于考察，但近几年来引进的一些算法与曾经可行的，使用简单的，基于梯度的优化的算法相比，也表现出了更佳的性能，而我们也试图着眼于理解导致这些算法成功的原理。

虽然本文的大部分都关注于深度神经网络和深度图模型结构，但探索针对深度结构的学习算法的思考理应超越神经网络的框架。举例来说，考虑决策树的扩展和多层次的增强算法也是很有意义的。

核学习算法是另一种值得探索的方法。因为捕获与有意义的分布相关的抽象性的特征空间，将会是在其上应用核方法的正确空间。这个方向的研究应该考虑所习得的核可以非局部性的普遍化的方法，来避免在3.3节中提及的当学习一个高度差异化的函数时所产生的维数灾难问题。

本文着眼于深度信念网络这一族特殊的算法，和它们组成元素有限波兹曼机，以及其近亲：不同种类的可以堆栈形成一种深度结构自动编码机。我们研究并联系起有限波兹曼机的对数似然的估计量，帮助证明训练有限波兹曼机中的对立发散更新原则。我们强调一种在深度信念网络上表现良好的优化原则及其相关算法，譬如基于贪心，单层和非监督的在模型的每一层上进行初始化的堆叠自动编码机。我们发现这种优化法则是一种更普遍化的优化法则的近似，在所谓的延拓法中得到应用，从而此中一系列逐渐难度加大的优化问题得到解决。这暗示了优化深度结构的新途径，既不是在正则化的途径中寻找解决方法，也不是类似于学生或者动物接受训练一样，用一系列选择的例子说明逐渐复杂化的概念来表征系统。

声明

作者由衷感谢来自这些人的启发和建设性建议：

Yann LeCun

Aaron Courville

Olivier Delalleau

Dumitru Erhan

Pascal Vincent

Geoffrey Hinton

Joseph Turian

Hugo Larochelle

Nicolas Le Roux

Jerome Louradour

Pascal Lamblin

James Bergstra

Pierre-Antoine Manzagol

Xavier Glorot.

感谢以下组织对于本研究的资金支持：

NSERC（加拿大自然科学与工程技术研究理事会）

MITACS（拿大信息技术与综合系统数学组织）

Canada Research Chairs（加拿大科学院）