# 量子深度学习

Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore
*Microsoft Research, Redmond, WA (USA)*

付容天 (学号 2020211616), 王子晗 (学号 2020211596), 吴清柳 (学号 2020211597), and 汤润清 (学号 2020211595)
计算机学院 (国家示范性软件学院), 北京邮电大学, 海淀, 中国北京

近年来，深度学习已经对深度学习和人工智能产生了深远影响。与此同时，用于量子计算机的算法已经被证明可以有效地处理那些传统的、经典的计算机难以处理的问题。我们说明了，量子计算不仅减少了训练深度受限玻尔兹曼机所需的时间，而且还为深度学习提供了比经典计算方法更丰富、更全面的框架，并使底层目标函数优化得到了显著改进。我们的量子方法还允许有效训练完整的玻尔兹曼机器、多层感知机和完全连接模型，并且没有传统意义上的经典对应方法。

## Introduction

我们提出了量子算法来执行深度学习，在训练效率和模型质量方面都优于传统的、最先进的经典算法。深度学习是最近用于机器学习的技术，它极大地影响了分类、推理和人工智能任务的建模方式 [1–4]。它以执行诸如语音和视觉识别等复杂的 AI 任务为前提，因此让机器学习包含原始输入数据的多层抽象的模型可能是必要的。例如，一个为了检测汽车而训练的模型可能首先接受以像素为单位的原始图像的输入，然后在随后的层中将数据抽象为简单的形状。在再下一层，可以进一步将基本形状抽象为聚合形式，例如保险杠或车轮。而在更高的层次上，形状可能会用“轮胎”或“引擎盖”之类的词来标记。因此，深度网络会自动学习复杂的、嵌套的原始数据表示，类似于我们大脑中的神经元处理层，理想情况下，学习到的概念层次结构是（人类）可以理解的。

一般来说，深度网络可能包含许多抽象级别，编码为高度连接的复杂图形网络。训练这样的图形网络属于深度学习的范畴。

玻尔兹曼机（BM）就是这样一类深度网络，其在形式上是一类具有无向边的递归神经网络，因此为数据提供了生成模型。从物理角度来看，玻尔兹曼机使用处于热平衡状态的伊辛（Ising）模型对训练数据进行建模。模型中的角动量在机器学习文献中被称为单位（unit），并且这些单位对特征和概念进行编码，伊辛模型交互图中的边表示这些特征的统计依赖关系。对观察数据和输出进行编码的节点集称为 units($v$)，而用于对潜在概念和特征空间进行建模的节点称为隐藏的 units($h$)。两类重要的玻尔兹曼机包括将底层图视为完全二分图的受限玻尔兹曼机（RBM），和由多层 RBM 组成的深度受限玻尔兹曼机（见图 1）。出于讨论的目的，我们假设单元是二元的，即只有可见单元和隐藏单元两种单元类型。

玻尔兹曼机通过吉布斯分布（逆温度为 1）模拟可见和隐藏单元的给定配置的概率：

$$P(v,h) = e^{-E(v,h)}/Z, \tag{1}$$

其中 $Z$ 是称为配分函数（partition function）的归一化因子，可见单元和隐藏单元的给定配置 $(v,h)$ 的能量 $E(v,h)$ 由下式给出：

$$E(v,h) = -\sum_i v_i b_i - \sum_j h_j d_j - \sum_{i,j} w_{ij}^{vh} v_i h_j - \sum_{i,j} w_{i,j}^{v} v_i v_j - \sum_{i,j} w_{i,j}^{h} h_i h_j. \tag{2}$$

其中向量 $b$ 和 $d$ 称为偏置（bias），这些偏置可以对于取值为 1 的单位提供能量惩罚（energy penalty）。并且 $w_{i,j}^{v,h}$、$w_{i,j}^{v}$ 和 $w_{i,j}^{h}$ 构成了权重（weight），这些权重可以对取值为 1 的可见单元与隐藏单元提供能量惩罚。我们记 $w = [w^{v,h}, w^{v}, w^{h}]$，并且分别记 $n_v$ 和 $n_h$ 为可见单元与隐藏单元的数量。

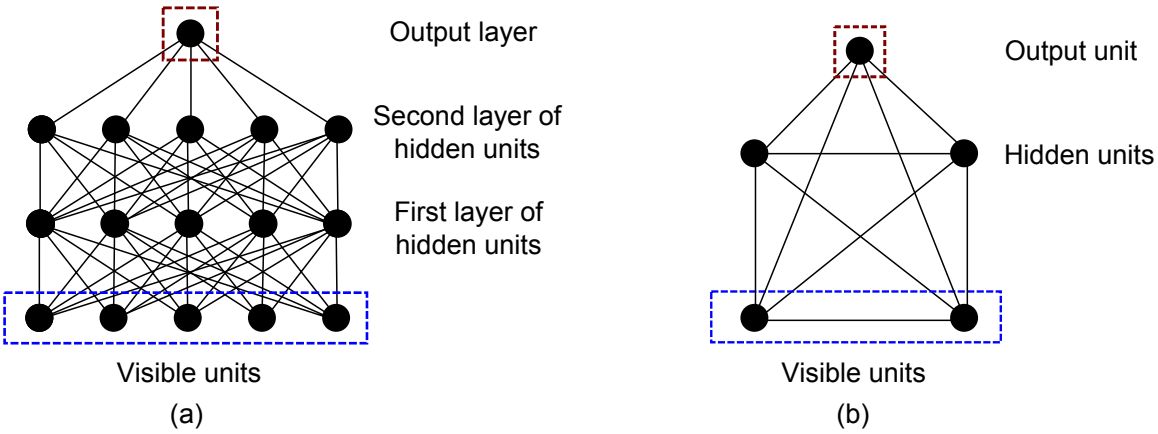



图 1: 两种玻尔兹曼机的图形表示。(a)4 层深度受限玻尔兹曼机（dRBM），其中每个黑色圆圈代表一个隐藏或可见单元，每条边代表相应交互的非零权重。输出层通常被视为可见层，以在图表底部的可见单元中提供数据输入的分类。(b)5 单元全玻尔兹曼机的示例。由于相同类型的单元之间的连接，隐藏单元和可见单元不再占据不同的层。

给定一些先验观察数据（称为训练集），这些模型的学习通过修改图中交互作用的强度来进行，从而最大化玻尔兹曼机产生给定观察结果的可能性。因此，训练过程使用梯度下降来找到优化最大似然目标（ML-objective）的权重和偏置：

$$O_{\mathrm{ML}} := \frac{1}{N_{\mathrm{train}}} \sum_{v \in x_{\mathrm{train}}} \log \left( \sum_{h=1}^{n_h} P(v,h) \right) - \frac{\lambda}{2} w^T w, \tag{3}$$

其中 $N_{\mathrm{train}}$ 是训练集的大小，$x_{\mathrm{train}}$ 是训练向量的集合，$\lambda$ 是一个用于对抗过拟合的 L2 正则化项。$O_{ML}$ 关于权重的导数是

$$\frac{\partial O_{\mathrm{ML}}}{\partial w_{i,j}} = \langle v_i h_j \rangle_{\mathrm{data}} - \langle v_i h_j \rangle_{\mathrm{model}} - \lambda w_{i,j}, \tag{4}$$

其中括号表示玻尔兹曼机的数据和模型的期望值。其需要求解的导数采用和 [5] 类似的形式。

直接从 (1) 和 (4) 计算这些梯度会导致在 $n_v$ 和 $n_h$ 方面呈现出指数级困难。因此，经典方法求助于近似值，例如对比散度（contrastive divergence） [3, 5–8]。但是，对比散度不提供任何真实目标函数的梯度 [9]，且已知会导致次优解决方案的出现 [10–12]，并在存在某些正则化函数 [9] 的情况下不能保证收敛，并且它不能直接用于训练完整的玻尔兹曼机。我们通过为基本分析方法提供有效的替代方案来说明量子计算为深度学习提供了一个更好的框架，这一框架可以加速学习过程并为训练数据带来更好的模型。

## GEQS 算法

我们提出了两种量子算法：通过量子采样的梯度估计（GEQS）和通过量子幅度估计的梯度估计（GEQAE）规模。这些算法首先为玻尔兹曼机提供了吉布斯状态的相干模拟（coherent analog），然后从结果状态中抽取样本并计算 (4) 中的期望值。算法的正式描述在附录中给出。用于制备这些状态的现有算法 [13–17] 对于机器学习应用程序往往效率不高，或者没有提供量子加速的明确证据。 [15, 17] 的低效率是因为均匀初始状态与吉布斯状态重叠很小。 Table I给出了这些之前的算法以及我们自己的算法的复杂性。

我们的算法通过对每个配置的概率使用非均匀先验分布来解决这个问题，这是因为我们从权重和偏差中先验地知道某些配置相比其他配置来讲不太可能。我们通过对配置概率使用平均场（MF）近似来获得此分布。这种近似是典型有效的，通常可以很好地近似于实际机器学习问题中观察到的吉布斯状态 [7, 18, 19]。我们的算法

利用这些先验知识从平均场状态的副本中细化吉布斯状态。这允许在两个状态足够接近的情况下有效且准确地准备吉布斯分布。

平均场估计（MF approximation）$Q(v,h)$ 被定义为最小化 KL 散度（Kullback–Leibler divergence）$\mathrm{KL}(Q||P)$ 的结果分布。它是一个乘积分布这一事实意味着它可以被有效地计算，也可以用来找到配分函数 $Z$ 的经典易处理的估计：

$$Z_Q := \sum_{v,h} Q(v,h) \log\left(\frac{e^{-E(v,h)}}{Q(v,h)}\right)$$

其中 $Z_Q \leq Z$ 并且当且仅当 $\mathrm{KL}(Q||P) = 0$ 时等号成立。上式中 $Q(v,h)$ 不需要是 MF 近似值。如果 $Q(v,h)$ 被另一个有效的近似值代替，类似的公式也适用，例如结构化平均场理论计算 [20]。

让我们假设一个常数 $\kappa$ 是已知的，从而使得

$$P(v,h) \leq \frac{e^{-E(v,h)}}{Z_Q} \leq \kappa\, Q(v,h), \tag{5}$$

并且定义如下的配置的“标准化”概率为

$$\mathcal{P}(v,h) := \frac{e^{-E(v,h)}}{\kappa\, Z_Q\, Q(v,h)}. \tag{6}$$

注意到

$$Q(v,h)\mathcal{P}(v,h) \propto P(v,h), \tag{7}$$

这意味着如果状态

$$\sum_{v,h} \sqrt{Q(v,h)}|v\rangle|h\rangle, \tag{8}$$

被制备出来且每个振幅都用 $\sqrt{\mathcal{P}(v,h)}$ 相乘，那么结果将会正比于我们想要的状态。

上述过程可以通过添加一个额外的量子寄存器来计算 $\mathcal{P}(v,h)$ 并使用量子叠加来制备状态

$$\sum_{v,h} \sqrt{Q(v,h)}|v\rangle|h\rangle|\mathcal{P}(v,h)\rangle\left(\sqrt{1-\mathcal{P}(v,h)}|0\rangle + \sqrt{\mathcal{P}(v,h)}|1\rangle\right). \tag{9}$$

这样，如果最右侧的量子比特测量结果为 1，那么就可以得到目标吉布斯状态。制备 (9) 是有效的，因为 $e^{-E(v,h)}$ 和 $Q(v,h)$ 可以在可见和隐藏单元的数量上以多项式的时间计算。以这种方式制备状态的成功概率为

$$P_{\text{success}} = \frac{Z}{\kappa\, Z_Q} \geq \frac{1}{\kappa}. \tag{10}$$

在实际情况中，如果 (10) 很小，我们的算法使用量子幅度放大 [21] 来二次提高成功的概率。

算法的复杂度由梯度计算所需的量子运算次数决定。由于能量的评估需要许多由对数因子所决定的操作，而这些操作的次数随着模型中的边的总数而进行线性变化，所以估计梯度的综合成本是

$$\tilde{O}\left(N_{\text{train}} E(\sqrt{\kappa} + \max_{x \in x_{\text{train}}} \sqrt{\kappa_x})\right), \tag{11}$$

此处 $\kappa_x$ 是 $\kappa$ 的值对应于可见单元被限制为 $x$ 的情况。估计 $Q(v,h)$ 和 $Z_{ML}$ 的成本是 $\tilde{O}(E)$（见附录），因此不会对结果产生渐近的影响。相比之下，使用贪婪逐层优化 [3] 经典地估计梯度所需的操作数量为

$$\tilde{O}(N_{\text{train}} \ell E), \tag{12}$$

此处 $\ell$ 是深度受限玻尔兹曼机（dRBM）的层数，$E$ 则为玻尔兹曼机（BM）的连接数。假定 $\kappa$ 是一个常数，那么量子采样方法为训练深度网络提供了渐近优势。我们在附录中提供了数值证据，表明可以通过增加 $n_h$ 和正则化参数 $\lambda$ 来保持 $\kappa$ 不变。

| | Operations | Qubits | Exact |
|---|---|---|---|
| ML | $\tilde{O}(N_{\text{train}} 2^{n_v+n_h})$ | 0 | Y |
| CD–k | $\tilde{O}(N_{\text{train}} \ell E k)$ | 0 | N |
| GEQS | $\tilde{O}(N_{\text{train}} E(\sqrt{\kappa} + \max_x \sqrt{\kappa_x}))$ | $O(n_h + n_v + \log(1/\mathcal{E}))$ | Y |
| GEQAE | $\tilde{O}(\sqrt{N_{\text{train}}} E^2(\sqrt{\kappa} + \max_x \sqrt{\kappa_x}))$ | $O(n_h + n_v + \log(1/\mathcal{E}))$ | Y |
| GEQAE (QRAM) | $\tilde{O}(\sqrt{N_{\text{train}}} E^2(\sqrt{\kappa} + \max_x \sqrt{\kappa_x}))$ | $O(N_{\text{train}} + n_h + n_v + \log(1/\mathcal{E}))$ | Y |

表 I: 我们用于具有 $\ell$ 层、$E$ 条边和 $N_{\text{train}}$ 条训练向量的深度受限玻尔兹曼机的资源规模。如果采样是唯一的错误来源，则算法是精确的。如果违反 (5)，则 GEQS 和 GEQAE 不准确。我们假设 QRAM 允许以单位成本同时执行不同量子位上的操作。

与现有的量子机器学习算法相比，我们的算法所需的量子比特数量最少 [22–25]。这是因为训练数据不需要存储在量子数据库中，否则需要 $\tilde{O}(N_{\text{train}})$ 个逻辑量子位 [26, 27]。相反，如果 $\mathcal{P}(v,h)$ 是用 $\lceil \log(1/\varepsilon) e \rceil$ 位精度计算的，并且可以作为 oracle 访问，那么只有

$$O(n_h + n_v + \log(1/\mathcal{E}))$$

个逻辑量子比特在 GEQS 算法中被使用。如果使用可逆运算计算 $\mathcal{P}(v,h)$，所需的量子比特数将会增加，但量子算术的最新发展可以大大降低此类成本 [28]。

此外，我们不需要知道 $\kappa$ 的确切值。如果选择的 $\kappa$ 值对于所有配置都不满足 (5)，那么如果 $\mathcal{P}(v,h)$ 被裁剪到区间 $[0,1]$，我们的算法仍然能够近似梯度。因此，通过在玻尔兹曼机的大小增加时保持 $\kappa$ 不变，算法总是可以变得高效，但代价是在结果概率分布中引入误差。这些误差的出现是因为状态制备算法会低估违反 (5) 的配置的相对概率；然而，如果这些违规的概率之和很小，那么一个简单的连续性论证表明近似吉布斯状态和正确状态的保真度（fidelity）很高。特别是，如果我们将“坏”定义为违反 (5) 的配置集，那么连续性论证表明，如果

$$\sum_{(v,h)\in \text{bad}} P(v,h) \leq \epsilon$$

则结果状态与吉布斯状态的保真度至少为 $1-\epsilon$，这在附录中正式说明。

我们的算法预计不会对所有玻尔兹曼机都既准确又高效。如果它们可以用来学习非平面的伊辛模型的基态能量，那么意味着 NP $\subseteq$ BQP，而这被广泛认为是错误的。因此，存在某种玻尔兹曼机使我们的算法无法有效和精确地模复杂度理论假设。我们不知道这些困难的例子在实际情况中有多普遍。然而，由于观察到平面场近似对训练过的玻尔兹曼机 [7, 8, 18, 19] 的有效性，而且训练模型中使用的权重往往很小，因此它们不太可能普遍存在。

## GEQAE 算法

在通过量子预言机（quantum oracle）提供训练数据的情况下，新的训练形式（例如我们的 GEQAE 算法）是可能的，这些新的训练形式允许以叠加方式而不是顺序方式访问训练数据。GEQAE 算法背后的想法是通过幅值估计（amplitude estimation） [21] 来利用数据叠加，这使得估计梯度的方差在 GEQS 算法上以二次趋势进行减小。因此，GEQAE 算法显著改进了大型训练集的性能。此外，允许以量子方式访问的训练数据使其可以被使用量子聚类和数据处理的算法进行预处理 [22–25, 29]。

在 GEQAE 算法中使用的量子预言机抽象了训练数据的访问模型。这个预言机可以被认为是量子数据库或生成训练数据的有效量子子程序的替代品（例如预训练的量子玻尔兹曼机或量子模拟器）。由于训练数据必须直接（或间接）存储在量子计算机中，GEQAE 通常需要比 GEQS 更多的量子比特；然而，量子叠加允许在一个

步骤中对整个训练向量集进行训练，而不是按顺序对每个训练示例进行学习，这一事实减轻了这种情况。这使我们可以在至多访问 $O(\sqrt{N_{\text{train}}})$ 次训练数据时准确估计出梯度。

记 $U_O$ 为对于所有下标 $i$ 都有如下操作的量子预言机：

$$U_O|i\rangle|y\rangle := |i\rangle|y \oplus x_i\rangle \tag{13}$$

其中 $x_i$ 为训练向量。这个预言机可以用来准备第 $i$ 个训练向量状态下的可见单元。然后，对该预言机的单个查询就足以计算出对所有训练向量统一叠加的结果，该结果可以通过重复 (9) 中给出的状态制备方法被转换为

$$\frac{1}{\sqrt{N_{\text{train}}}} \sum_{i,h} \sqrt{Q(X_i,h)}|i\rangle|x_i\rangle|h\rangle \left(\sqrt{1-\mathcal{P}(x_i,h)}|0\rangle + \sqrt{\mathcal{P}(x_i,h)}|1\rangle\right) \tag{14}$$

GEQAE 算法通过估计 (14) 中最右边的量子位为 1 的概率 $P(1)$ 和 (14) 中最右边量子位为 1 并且 $v_i = 1, h_j = 1$ 的概率 $P(11)$ 计算数据和模型的期望值，例如 $\langle v_i, h_j\rangle$。然后就有

$$\langle v_i h_j\rangle = \frac{P(11)}{P(1)} \tag{15}$$

这两个概率可以通过采样来估计，但是一个更有效的方法是使用幅值估计 [21]——一种使用 Grover 算法的相位估计（phase estimation）来直接以量子比特串输出这些概率的量子算法。如果我们要求采样错误在 $1/\sqrt{N_{\text{train}}}$ 的规模（与前一个例子大致类似），那么 GEQAE 算法的查询复杂度为：

$$\tilde{O}\left(\sqrt{N_{\text{train}}} E(\kappa + \max_x \kappa_x)\right) \tag{16}$$

每次能量计算需要 $\tilde{O}(E)$ 次代数运算，因此 (16) 非查询操作的次数规模为

$$\tilde{O}\left(\sqrt{N_{\text{train}}} E^2(\kappa + \max_x \kappa_x)\right) \tag{17}$$

如果成功概率已知在一个常数因子内，则幅值放大（amplitude amplification） [21] 可用于在估计成功概率之前提高成功概率。然后根据放大后的概率计算原始成功概率。这将 GEQAE 算法的查询复杂度降低到了

$$\tilde{O}\left(\sqrt{N_{\text{train}}} E(\sqrt{\kappa} + \max_x \sqrt{\kappa_x})\right) \tag{18}$$

这样，在 $\sqrt{N_{\text{train}}} \gg E$ 的条件下，GEQAE 算法的表现就要比 GEQS 算法更好。

### 并行算法

贪婪的对比散度算法（CD-k）的训练是尴尬的并行过程，意思是说这个算法的几乎所有部分都可以分布在并行处理节点上。然而用于训练 CD-k 中每一层的 k 轮采样却不容易并行化。这意味着在某些情况下，简单但易于并行化的模型（例如 GMM）可能更可取 [30]。相比之下，GEQS 和 GEQAE 可以利用容错量子计算机中预期的并行性来更有效地训练深度受限玻尔兹曼机。要理解这一点，需要注意能量是每一层的能量之和，可以在 $\log(M) = O(\max(n_v, n_h)\log(\max(n_v, n_h)))$ 深度处进行计算（见图 1）并在 $O(\log(\ell))$ 深度处进行求和。$O(\sqrt{\kappa + \max_x \kappa_x})$ 平面场状态的制备可以同时执行，并且可以通过对数深度计算定位正确的样本。因此 GEQS 算法的深度为

$$O\left(\log([\kappa + \max_x \kappa_x] M \ell N_{\text{train}})\right). \tag{19}$$

由于 GEQAE 输出的每个导数都可以独立计算，因此 GEQAE 的深度为

$$O\left(\sqrt{N_{\text{train}}[\kappa + \max_x \kappa_x]}\log(M\ell)\right). \tag{20}$$

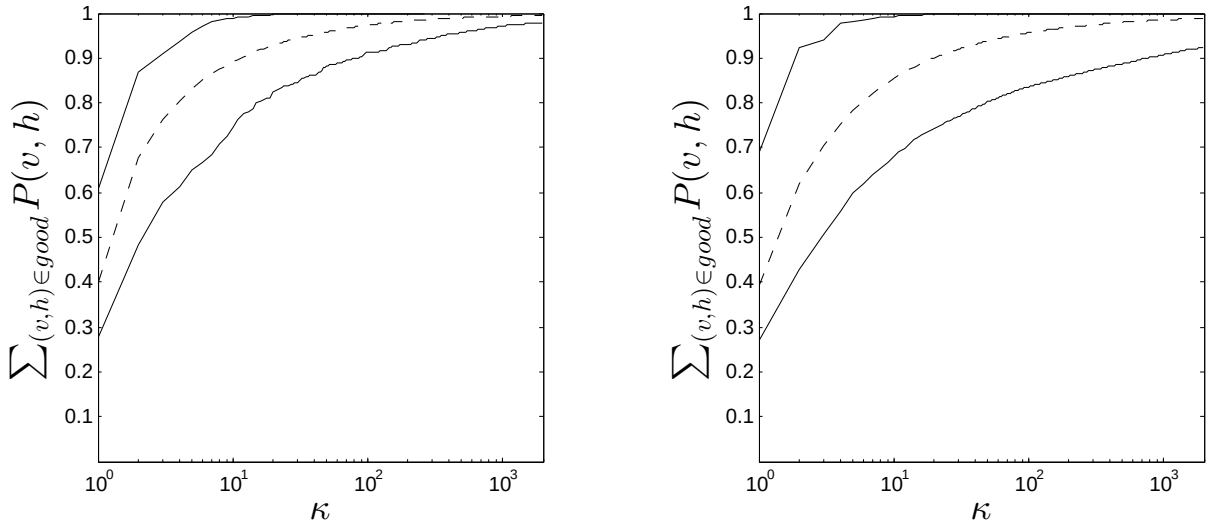



图 2: 具有参数 $n_h = 8$ 和 $n_v = 6$（左）和 $n_v = 12$（右）的在 (22) 上训练的受限玻尔兹曼机的 $\mathcal{P}(v,h) \leq 1$ vs $\kappa$ 的概率质量。虚线给出平均值；实线给出 95% 的置信区间。

通过将训练集划分为小批量并对得到的导数进行平均，可以以增加电路尺寸为代价来减小深度。

使用 $k$ 步的对比散度（CD-$k$）训练需要的深度为

$$O(k\ell^2 \log(MN_{\text{train}})). \tag{21}$$

$O(\ell^2)$ 缩放的出现是因为 CD-$k$ 是一种前馈算法，而 GEQS 和 GEAQE 不是。

### 数值结果

我们解决了以下有关我们算法行为的问题：

1. $\kappa$ 的典型值是什么？

2. 使用 CD-1 训练的模型与使用 GEQS 和 GEQAE 训练的模型有何不同？

3. 完整的玻尔兹曼机是否会产生比深度受限玻尔兹曼机更好的模型？

要回答这些问题，需要计算 $\mathcal{P}(v,h)$ 和 $O_{ML}$，该操作的时间随着 $\max n_v, n_h$ 而指数增长。因此，计算限制严重限制了我们可以通过数值实验研究的模型大小。在实践中，我们在计算上仅限于研究具有最多 20 个单位的模型。我们训练下面的具有 $\ell \in \{2,3\}$ 层、$n_h \in \{2,\ldots,8\}$ 个隐藏单元以及 $n_v \in \{4,\ldots,12\}$ 个可见单元的深度受限玻尔兹曼机。

由于计算限制，用于基准机器学习的大规模传统数据集（例如 MNIST[31]）在这里是不切实际的。因此，我们专注于由四个不同功能组成的合成训练数据：

$$\begin{aligned} [x_1]_j &= 1 \text{ if } j \leq n_v/2 \text{ else } 0 \\ [x_2]_j &= j \bmod 2, \end{aligned} \tag{22}$$

以及它们的按位取反。我们将伯努利噪声 $\mathcal{N}\,[0, 0.5]$ 添加到位串中的每个位以增加训练集的大小。特别地，我们采用 (22) 中的四种模式中的每一种，并以概率 $\mathcal{N}$ 翻转每个位。我们在每个数值实验中使用 10000 个训练示例；

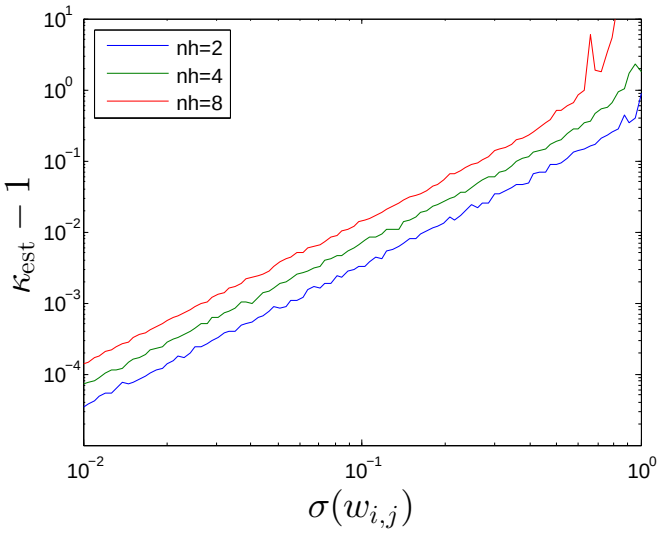



图 3: 在 $n_v = 4$ 和具有零均值和方差 $\sigma^2(w_{i,j})$ 的高斯权重的合成受限玻尔兹曼机中，作为函数权重标准差的估计的 $\kappa$ 的规模。偏差被设置为从具有零均值和单位方差的高斯中得到。

每个向量包含 $4, \ldots, 12$ 个二进制特征。我们的任务是为这四个向量推断一个生成模型。我们在附录中提供了对子采样 MNIST 数字数据的数值实验。结果在性质上是相似的。

图2显示，尽管它们的希尔伯特空间维度相差 $2^6$ 倍，但对于该数据集（$\mathcal{N} = 0$），可见单元数量并不会显着增加 $\kappa$ 倍。这说明 $\kappa$ 主要取决于平均场近似值的质量，而不是 $n_v$ 和 $n_h$。如附录所示，对于完整的玻尔兹曼机，我们可以观察到类似的行为。此外，$\kappa \approx 1000$ 通常会导致非常接近真实的吉布斯状态。即使 $\kappa = 1000$ 并不算过分，但我们在附录中引入的“对冲策略”（hedging strategy）可以将 $\kappa$ 降低到大约 50。

我们通过以下方法进一步检查随机（未经训练的）受限玻尔兹曼机的 $\kappa$ 缩放

$$\kappa_{\rm est} = \sum_{v,h} P^2(v,h)/Q(v,h). \tag{23}$$

图3显示了对于小的、随机的受限玻尔兹曼机，对于 $\sigma^2(w_{i,j}) \ll 1$ 有 $\kappa - 1 \in O(\sigma^2(w_{i,j})E)$。

这引出了第二个问题：确定实际玻尔兹曼机的权重分布。图 4 显示，对于使用对比散度训练的大型受限玻尔兹曼机，其权重会随着 $n_h$ 的增加而迅速缩小。对于 $\mathcal{N} = 0$，经验缩放为 $\sigma^2 \in O(E-1)$，这表明 $\kappa - 1$ 不会随着 $n_h$ 的增长而发散。尽管取 $\mathcal{N} = 0.2$ 会显着降低 $\sigma^2$，但缩放也会降低。这可能是正则化对两个训练集具有不同效果的结果。无论哪种情况，这些结果都与图3的结果相结合，表明 $\kappa$ 对于大型网络应该是可管理的。

我们可以通过比较在对比散度和我们的量子算法下发现的深度受限玻尔兹曼机的 $O_{\rm ML}$ 平均值来评估 GEQS 和 GEQAE 的优点。即使是小的受限玻尔兹曼机，找到的最优值之间的差异也是显著的；深度网络的差异可能在 10% 左右。表II中的数据表明，机器学习训练可以显着提高结果模型的质量。我们还观察到，在高度受限的情况下，对比散度可以优于机器学习目标的梯度下降，这是因为对比散度近似的随机性使其对局部最小值不那么敏感。

就目标函数的质量而言，完整玻尔兹曼机的建模能力可以显着优于深度受限玻尔兹曼机。事实上，我们在附录中展示了 $n_v = 6$ 和 $n_h = 4$ 的完整玻尔兹曼机可以实现 $O_{\rm ML} \approx -1.84$。具有相当数量的边的深度受限玻尔兹曼机可以得到 $O_{\rm ML} \approx -2.3$（见表II），比完整玻尔兹曼机少 25%。由于我们的量子算法可以有效地训练除深度受限玻尔兹曼机之外的完整玻尔兹曼机，因此量子框架支持的机器学习形式不仅比经典的易处理方法更丰富，而且还可能使数据模型得到改进。

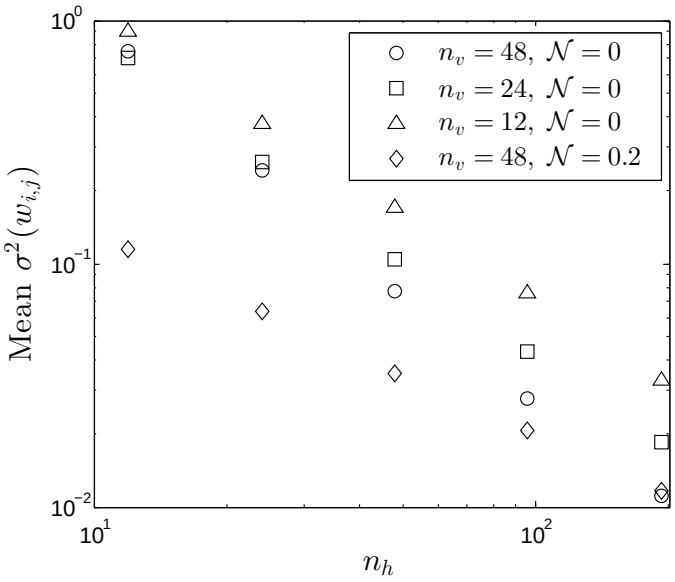



图 4: 使用 $\lambda = 0.01$ 的对比散度在 (22) 上训练的大型受限玻尔兹曼机的权重标准差。

| $n_v$ | $n_{h1}$ | $n_{h2}$ | CD | ML | % Improvement |
|---|---|---|---|---|---|
| 6 | 2 | 2 | $-2.7623$ | $-2.7125$ | 1.80 |
| 6 | 4 | 4 | $-2.4585$ | $-2.3541$ | 4.25 |
| 6 | 6 | 6 | $-2.4180$ | $-2.1968$ | 9.15 |
| 8 | 2 | 2 | $-2.8503$ | $-3.5125$ | $-23.23$ |
| 8 | 4 | 4 | $-2.8503$ | $-2.6505$ | 7.01 |
| 8 | 6 | 4 | $-2.7656$ | $-2.4204$ | 12.5 |
| 10 | 2 | 2 | $-3.8267$ | $-4.0625$ | $-6.16$ |
| 10 | 4 | 4 | $-3.3329$ | $-2.9537$ | 11.38 |
| 10 | 6 | 4 | $-2.9997$ | $-2.5978$ | 13.40 |

表 II: 使用 $O_{\mathrm{ML}}$ 的对比散度和梯度下降优化对 $\mathcal{N} = 0$ 的 3 层深度受限玻尔兹曼机进行贪心训练找到的 $O_{\mathrm{ML}}$ 的平均值。

## 结论

我们工作的一个基本结果是训练玻尔兹曼机可以简化为量子态制备的问题。该状态制备过程不需要使用对比散度近似或关于图形模型拓扑的假设。我们证明了量子算法不仅可以显着改进数据模型，还可以提供一个更优雅的框架来训练玻尔兹曼机。该框架丰富了在量子信息和凝聚态物理学中关于制备和近似吉布斯状态的知识，这些知识能够在训练期间得到利用。

我们的量子深度学习框架能够将平均场估计细化为接近（或等效于）所需吉布斯状态的状态。这种状态制备方法允许使用不明确依赖于深度受限玻尔兹曼机中的层数的许多操作来训练玻尔兹曼机。它还允许二次减少必须访问训练数据的次数，并可以训练完整的玻尔兹曼机。我们的算法还可以在多个量子处理器上更好地并行化，解决了深度学习的一个主要缺点 [30]。

虽然在拥有可扩展的量子计算机之前，小规模样例的数值结果令人鼓舞，但未来需要使用量子硬件的实验研究来评估我们算法的泛化性能。鉴于我们的算法能够提供比对比散度更好的梯度，很自然地期望它通过使用当前用于训练深度玻尔兹曼机的相同方法使在该设置中表现良好 [3]。无论如何，量子计算为深度学习提供的无数优势不仅表明了量子计算机的重要近期应用，而且强调了从量子角度思考机器学习的价值。

## 附录 A: 状态准备的量子算法

首先我们展示量子计算机如何从吉布斯分布中提取无偏样本，从而能够通过采样（或量子采样）计算概率 $P(v,h)$。基于该算法，我们的设想是准备一个近似于模型或数据的理想概率分布的量子分布。然后使用拒绝采样将该近似分布细化为量子分布，即在数值误差范围内的目标概率分布 [17]。如果我们从吉布斯态振幅的均匀先验开始，那么通过量子排斥采样来制备该态很可能是低效的。这是因为其成功概率取决于初始状态和吉布斯状态的配分函数之比 [15]，而这在实践中对于机器学习问题来说微乎其微。相反，我们的算法在吉布斯态的联合概率上使用了平均场近似，而不是均匀先验。数据显示，通过这些额外的信息可以将成功的概率提高到可接受的水平。然后可以通过从量子分布中采样来找到所需的期望值。我们发现，通过使用振幅估计的量子算法，可以二次减少达到固定采样误差所需的样本数 [21]。

首先我们讨论了将初始量子分布细化为量子相干吉布斯态（通常称为相干热态或 CTS）的过程。然后，我们讨论如何使用平均场理论或其推广来为量子计算机提供合适的初始状态，以细化为 CTS。我们假设下文中玻尔兹曼机器的所有单位都是二进制值。其他有值单位，如高斯单位，可以在这个框架内通过从几个量子位的字符串中形成一个单个单位来近似。

首先,我们将联合概率分布的平均场近似定义为 $Q(v,h)$。有关平均场近似的更多详细信息,请参见Section G。我们还使用平均场分布来计算我们算法所需的配分函数的变分近似值。这些近似值可以有效地计算（因为概率分布分解），定义如下。

**Definition 1.** 设 $Q$ 为吉布斯分布 $P = e^{-E}/Z$ 的平均场近似值，则

$$Z_Q := \sum_{v,h} Q(v,h) \log\left(\frac{e^{-E(v,h)}}{Q(v,h)}\right).$$

此外，对于任何 $x \in x_{\text{train}}$，设 $Q_x$ 为可见单位设定为 $x$ 的玻尔兹曼机器的吉布斯分布的平均场近似值，则

$$Z_{x,Q} := \sum_{h} Q_x(x,h) \log\left(\frac{e^{-E(x,h)}}{Q_x(x,h)}\right).$$

为了使用量子算法从 $Q$ 制备 $P$，我们需要知道从 $P(v,h) \approx e^{-E(v,h)}/Z_Q$ 与 $Q(v,h)$ 之比的近似值的上界 $\kappa$，定义如下。

**Definition 2.** 设 $\kappa > 0$ 为满足所有可见和隐藏配置 $(v,h)$ 的常数

$$\frac{e^{-E(v,h)}}{Z_Q} \leq \kappa Q(v,h), \tag{A1}$$

其中 $Z_Q$ 是*Definition 1*中给出的配分函数的近似值。

我们还定义了一个类似的量，适用于可见单元被设定为一个训练向量的情况。

**Definition 3.** 设 $\kappa_x > 0$ 是满足 $x \in x_{\text{train}}$ 和所有隐藏配置 $h$ 的常数

$$\frac{e^{-E(x,h)}}{Z_{x,Q}} \leq \kappa Q_x(x,h), \tag{A2}$$

其中 $Z_{x,Q}$ 是*Definition 1*中给出的配分函数的近似值。

**Lemma 1.** 假设 $Q(v,h)$ 是玻尔兹曼机器的平均场概率分布，那么对于隐藏和可见单元的所有配置，有

$$P(v,h) \leq \frac{e^{-E(v,h)}}{Z_Q} \leq \kappa Q(v,h).$$

证明. 平均场近似值也可用于提供对数配分函数的下限。例如 Jensen 不等式

$$\begin{aligned}\log(Z) &= \log\left(\sum_{v,h} \frac{Q(v,h)e^{-E(v,h)}}{Q(v,h)}\right), \\ &\geq \sum_{v,h} Q(v,h)\log\left(\frac{e^{-E(v,h)}}{Q(v,h)}\right) = \log(Z_Q). \end{aligned} \tag{A3}$$

这表明 $Z_Q \leq Z$，因此

$$P(v,h) \leq e^{-E(v,h)}/Z_Q, \tag{A4}$$

其中 $Z_Q$ 是使用平均场分布得出的 $Z$ 的近似值。然后根据(A4) 和Definition 2得出结果。 □

Lemma 1的结果可证明下面的引理，它给出了从平均场态准备吉布斯态的成功概率。

**Lemma 2.** 玻尔兹曼机器吉布斯态的相干模拟可以用成功概率 $\frac{Z}{\kappa Z_Q}$。类似地，与被设定为配置 $x$ 的可见单元对应的吉布斯态可以用成功几率 $\frac{Z_x}{\kappa_x Z_{x,Q}}$ 来制备。

证明. 算法的第一步是使用(G3)计算平均场参数 $\mu_i$ and $\nu_j$。这些参数唯一地指定了平均场分布 $Q$。平均场参数用于近似配分函数 $Z$ 和 $Z_x$。然后，通过执行一系列单量子比特旋转，这些平均场参数用于制备 $Q(v,h)$ 的相干模拟，表示为 $|\psi_Q\rangle$：

$$|\psi_Q\rangle := \prod_i R_y(2\arcsin(\sqrt{\mu_i}))|0\rangle \prod_j R_y(2\arcsin(\sqrt{\nu_j}))|0\rangle = \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle. \tag{A5}$$

剩余的步骤使用拒绝采样来将该粗略近似值细化为 $\sum_{v,h}\sqrt{P(v,h)}|v\rangle|h\rangle$.

对于紧凑性，我们定义

$$\mathcal{P}(v,h) := \frac{e^{-E(v,h)}}{\kappa Z_Q Q(v,h)}. \tag{A6}$$

$\mathcal{P}(v,h)$ 可以根据平均场参数有效地计算，因此也存在一种有效的量子算法（量子电路）来计算 $\mathcal{P}(v,h)$。Lemma 1还保证 $0 \leq \mathcal{P}(v,h) \leq 1$。

由于量子运算（测量除外）是线性的，如果我们将算法应用于状态 $\sum_v\sum_h\sqrt{Q(v,h)}|v\rangle|h\rangle|0\rangle$，我们得到 $\sum_v\sum_h\sqrt{Q(v,h)}|v\rangle|h\rangle|\mathcal{P}(v,h)\rangle$。然后我们添加一个额外的量子比特，称为 ancilla 量子比特，并根据 $R_y(2\sin^{-1}(\mathcal{P}(v,h)))$ 进行以下转换：

$$\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|\mathcal{P}(v,h)\rangle|0\rangle \mapsto \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|\mathcal{P}(v,h)\rangle\left(\sqrt{1-\mathcal{P}(v,h)}|0\rangle + \sqrt{\mathcal{P}(v,h)}|1\rangle\right). \tag{A7}$$

然后，通过反向应用用于准备 $\mathcal{P}(v,h)$ 的相同操作，将包含量子位串 $\mathcal{P}(v,h)$ 的寄存器恢复到 $|0\rangle$ 状态。这个过程是可能的，因为除了测量之外，所有量子操作都是可逆的。由于 $\mathcal{P}(v,h) \in [0,1]$，则(A7)是适当归一化的量子态，而其平方又是有效的概率分布。

如果测量(A7)中最右边的量子比特并得到的结果为 1（回想一下，投影测量总是产生单位向量），那么状态的剩余部分将与下式成比例

$$\sum_{v,h}\sqrt{Q(v,h)\mathcal{P}(v,h)} = \sqrt{\frac{Z}{\kappa Z_Q}}\sum_{v,h}\sqrt{\frac{e^{-E(v,h)}}{Z}}|v\rangle|h\rangle = \sqrt{\frac{Z}{\kappa Z_Q}}\sum_{v,h}\sqrt{P(v,h)}|v\rangle|h\rangle, \tag{A8}$$

**Algorithm 1** 用于生成可测量状态的量子算法，以估计模型上的期望值。

**Input:** 模型权重 $w$、可见偏差 $b$、隐藏偏差 $d$、边缘集 $E$ 和 $\kappa$。
**Output:** 可以测量的量子态，以获得（深层次）玻尔兹曼机器的正确吉布斯态。

**function** qGenModelState($w, b, d, E, \kappa$)
根据 $w$、$b$ 和 $d$ 计算平均场参数 $\mu$ 和 $\nu$ 的矢量。
计算平均场配分函数 $Z_Q$。
制备状态 $\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle := \left(\prod_{i=1}^{n_v} e^{-i\sqrt{\mu_i}Y}\,|0\rangle\right)\left(\prod_{j=1}^{n_h} e^{-i\sqrt{\nu_j}Y}\,|0\rangle\right)$
添加量子位寄存器以存储能量值并初始化为零：$\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle \to \sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|0\rangle$
**for** $i = 1 : n_v$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,\left|E(v,h) + v_i b_i + \ln\left(\mu_i^{v_i}(1-\mu_i)^{1-v_i}\right)\right\rangle$.
▷ 这里包括一个能量惩罚，用来处理偏差和可见单元对 $Q(v,h)^{-1}$ 的贡献。
**end for**
**for** $j = 1 : n_h$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,\left|E(v,h) + h_j d_j + \ln\left(\nu_j^{h_j}(1-\nu_j)^{1-h_j}\right)\right\rangle$.
**end for**
**for** $(i,j) \in E$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h) + v_i h_j w_{i,j}\rangle$.
**end for**
$\sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|E(v,h)\rangle\left(\sqrt{\frac{e^{-E(v,h)}}{Z_Q\kappa}}\,|1\rangle + \sqrt{1-\frac{e^{-E(v,h)}}{Z_Q\kappa}}\,|0\rangle\right)$.
**end function**

这是归一化因子的期望状态。测量 1 的概率是这个比例常数的平方

$$P(1|\kappa, Z_Q) = \frac{Z}{\kappa Z_Q}. \tag{A9}$$

注意，这是一个有效的概率，因为Lemma 1给出了 $\sum_{v,h}\kappa Z_Q Q(v,h) \geq \sum_{v,h} e^{-E(v,h)} \Rightarrow \kappa Z_Q \geq Z$。

制备一个可以用来估计数据期望值的量子态需要对这个算法稍加修改。首先，对于每个 $x \in x_{\text{train}}$ 为了获得期望值，我们将 $Q(v,h)$ 替换为约束平均场分布 $Q_x(x,h)$。然后用这个数据计算量子态

$$\sum_h \sqrt{Q_x(x,h)}\,|x\rangle\,|h\rangle\,, \tag{A10}$$

可以制备。然后，我们遵循完全相同的协议，使用 $Q_x$ 代替 $Q$，$Z_x$ 代替 $Z$，以及 $Z_{x,Q}$ 代替 $Z_Q$。该算法成功的概率为

$$P(1|\kappa, Z_{x,Q}) = \frac{Z_x}{\kappa_x Z_{x,Q}}. \tag{A11}$$

□

Lemma 2中使用的状态制备问题的方法类似于 [15] 的方法，只是我们使用平均场近似值而不是无限温度吉布斯状态作为初始状态。初始状态的选择很重要，因为状态制备过程的成功概率取决于初始状态和目标状态之间的距离。对于机器学习应用，吉布斯 12 态和无限温度吉布斯态之间的内积通常微乎其微；而我们在Section E 3中发现，平均场和吉布斯态通常有很大的重叠。

下面的引理是Lemma 2的更常用的版本，它表明如果 $\kappa$ 的值不够大，那么仍然可以使用状态制备算法，但代价是与理想相干吉布斯状态的保真度降低。

**Algorithm 2** 用于生成可测量的状态的量子算法，以估计数据的期望值。

**Input:** 模型权重 $w$、可见偏差 $b$、隐藏偏差 $d$、边缘集 $E$ 和 $\kappa_x$、训练向量 $x$。

**Output:** 可以测量的量子态，以获得（深层次）玻尔兹曼机器的正确吉布斯态，可见单位被设定为 $x$。

**function** QGENDATASTATE($w, b, d, E, \kappa_x, x$)

计算 $w$、$b$ 和 $d$ 的平均场参数 $\nu$ 的矢量，可见单位被设定为 $x$。

计算平均场配分函数 $Z_{x,Q}$。

**for** $i = 1 : n_v$ **do**

$\sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h)\rangle \to \sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h) + x_i b_i\rangle$.

**end for**

**for** $j = 1 : n_h$ **do**

$\sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h)\rangle \to \sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,\left|E(x,h) + h_j d_j + \ln\left(\nu_j^{h_j}(1-\nu_j)^{1-h_j}\right)\right\rangle$.

**end for**

**for** $(i,j) \in E$ **do**

$\sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h)\rangle \to \sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h) + x_i h_j w_{i,j}\rangle$.

**end for**

$\sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h)\rangle \to \sum_h \sqrt{Q(x,h)}\,|x\rangle\,|h\rangle\,|E(x,h)\rangle\left(\sqrt{\frac{e^{-E(x,h)}}{Z_{x,Q}\kappa_x}}\,|1\rangle + \sqrt{1-\frac{e^{-E(x,h)}}{Z_{x,Q}\kappa_x}}\,|0\rangle\right)$.

**end function**

**Lemma 3.** 如果我们放松*Lemma 2*的假设，使得对所有 $(v,h) \in \text{good}$ 有 $\kappa Q(v,h) \geq e^{-E(v,h)}/Z_Q$，对所有 $j \in \text{bad}$ 有 $\kappa Q(v,h) < e^{-E(v,h)}/Z_Q$，并且 $\sum_{(v,h)\in\text{bad}}\left(e^{-E(v,h)} - Z_Q\kappa Q(v,h)\right) \leq \epsilon Z$，则当一个状态的保真度至少为 $1-\epsilon$ 时，目标吉布斯态的概率至少为 $Z(1-\epsilon)/(\kappa Z_Q)$。

证明. 假设将Lemma 2中的协议改为仅当 $e^{-E(v,h)}/Z_Q\kappa \leq 1$. 这意味着在测量将状态投影到成功或失败分支的寄存器之前，状态为

$$\sum_{(v,h)\in\text{good}} \sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\left(\sqrt{\frac{e^{-E(v,h)}}{Z_Q\kappa Q(v,h)}}\,|1\rangle + \sqrt{1-\frac{e^{-E(v,h)}}{Z_Q\kappa Q(v,h)}}\,|0\rangle\right) + \sum_{(v,h)\in\text{bad}} \sqrt{Q(v,h)}\,|v\rangle\,|h\rangle\,|1\rangle \quad \text{(A12)}$$

成功准备近似状态的概率为

$$\sum_{(v,h)\in\text{good}} \frac{e^{-E(v,h)}}{\kappa Z_Q} + \sum_{(v,h)\in\text{bad}} Q(v,h) = \frac{Z - \left(\sum_{(v,h)\in\text{bad}} e^{-E(v,h)} - \sum_{(v,h)\in\text{bad}} \kappa Z_Q Q(v,h)\right)}{\kappa Z_Q} \geq \frac{Z(1-\epsilon)}{\kappa Z_Q}. \quad \text{(A13)}$$

合成态与理想态 $\sum_{v,h}\sqrt{e^{-E(v,h)}/Z}\,|v\rangle\,|h\rangle$ 的保真度为

$$\frac{\sum_{(v,h)\in\text{good}} e^{-E(v,h)} + \sum_{(v,h)\in\text{bad}} \sqrt{Q(v,h)Z_Q\kappa e^{-E(v,h)}}}{\sqrt{Z\left(\sum_{(v,h)\in\text{good}} e^{-E(v,h)} + \sum_{(v,h)\in\text{bad}} Z_Q\kappa Q(v,h)\right)}} \geq \frac{\sum_{(v,h)\in\text{good}} e^{-E(v,h)} + \sum_{(v,h)\in\text{bad}} Z_Q\kappa Q(v,h)}{\sqrt{Z\left(\sum_{(v,h)\in\text{good}} e^{-E(v,h)} + \sum_{(v,h)\in\text{bad}} Z_Q\kappa Q(v,h)\right)}}, \quad \text{(A14)}$$

因为对于所有的 $(v,h) \in \text{bad}$ 有 $Q(v,h)Z_Q\kappa \leq e^{-E(v,h)}$。现在使用(A14)中使用的相同方法，并假设 $\sum_{(v,h)\in\text{bad}}\left(e^{-E(v,h)} - Z_Q\kappa Q(v,h)\right) \leq \epsilon Z$，可得保真度的下限为

$$\frac{Z(1-\epsilon)}{Z\sqrt{1-\epsilon}} = \sqrt{1-\epsilon} \geq 1-\epsilon. \quad \text{(A15)}$$

□

Algorithm 1和Algorithm 2中概述了相应的算法，用于分别准备计算模型期望值和数据期望值所需的状态。

## 附录 B: 抽样梯度计算

我们估计 $O_{\mathrm{ML}}$ 梯度的第一个算法包括从平均场状态准备吉布斯状态，然后从结果分布中提取样本，以估计梯度表达式中所需的期望值。我们在正文中将该算法称为 GEQS（通过量子采样的梯度估计）。我们还通过使用称为振幅放大的量子算法 [21]（Grover 搜索算法的推广 [32]））来优化 GEQS 算法，该算法使用Lemma 2或Lemma 3中的方法二次减少了从吉布斯分布中提取样本所需的平均重复次数。

关键是，该算法准备的分布与平均场分布没有直接关系。选择平均场分布是因为它是一种有效的可计算分布，接近真实的吉布斯分布，从而提供了准备状态的捷径。替代选择，如均匀分布，理想情况下会产生相同的终态分布，但可能需要比使用平均场近似值作为起点时更多的操作。我们在以下定理中陈述了 GEQS 算法的性能。

**Theorem 1.** *存在一种量子算法，该算法可以使用含 $E$ 条边的的连通图上的玻尔兹曼机的 $N_{\mathrm{train}}$ 样本来估计 $O_{\mathrm{ML}}$ 的梯度。算法计算梯度所需的平均量子运算次数为*

$$\tilde{O}\left(N_{\mathrm{train}} E\left(\sqrt{\kappa}+\sqrt{\max_v \kappa_v}\right)\right),$$

*其中 $\kappa_v$ 是当可见单元被设定为 $v$ 和 $f \in \tilde{O}(g)$ 时，对应于吉布斯分布的 $\kappa$ 值 $f \in O(g)$ 直到多对数因子。*

*证明.* 我们使用Algorithm 3来计算所需的梯度。从Lemma 2可以看出，Algorithm 3从玻尔兹曼机器中提取 $N_{\mathrm{train}}$ 样本，然后通过从这些状态中提取样本来估计计算对数似然梯度所需的期望值。Algorithm 1和Algorithm 2中给出的生成这些状态的子例程 `qGenModelState` 和 `qGenDataState` 表示该算法中唯一的量子处理。在检测到成功之前，平均必须调用子程序的次数由Lemma 2给出的成功概率至少为

$$\min\left\{\frac{Z}{\kappa Z_Q}, \min_x \frac{Z_x}{\kappa_x Z_{x,Q}}\right\}. \tag{B1}$$

Lemma 1给出了 $Z > Z_Q$，因此成功概率满足

$$\min\left\{\frac{Z}{\kappa Z_Q}, \min_v \frac{Z_x}{\kappa_x Z_{x,Q}}\right\} \geq \frac{1}{\kappa + \max_v \kappa_v}. \tag{B2}$$

通常，(B2)意味着吉布斯态的制备平均需要 $O(\kappa+\max_v \kappa_v)$ 调用Algorithm 1和Algorithm 2，但量子振幅放大算法 [21] 将成功之前所需的平均重复次数减少到 $O(\sqrt{\kappa+\max_v \kappa_v})$。因此，Algorithm 3需要调用 `qGenModelState` 和 `qGenDataState` 的平均次数为 $O(N_{\mathrm{train}}\sqrt{\kappa+\max_v \kappa_v})$。

Algorithm 1和Algorithm 2需要准备平均场状态，计算配置 $(v,h)$ 的能量，并执行受控旋转。假设图是连通的，则隐藏和可见单元的数量为 $O(E)$。由于合成单个量子比特旋转的成本在误差范围内 $\epsilon$ 是 $O(\log(E/\epsilon))$[33–35]，计算能量的成本为 $O(E\ \mathrm{polylog}(E/\epsilon))$，这些算法的成本为 $\tilde{O}(E)$。因此，Algorithm 3的预期成本为 $\tilde{O}(N_{\mathrm{train}} E\sqrt{\kappa+\max_v \kappa_v}) \in \tilde{O}(N_{\mathrm{train}} E(\sqrt{\kappa}+\sqrt{\max_v \kappa_v}))$。 □

相比之下，使用贪婪的逐层优化估计梯度所需的 $U_O$ 操作和查询数量为 [3]

$$\tilde{O}(N_{\mathrm{train}}\ell E), \tag{B3}$$

其中 $\ell$ 是深度玻尔兹曼机器中的层数。假设 $\kappa$ 是常数，那么量子采样方法为训练深度网络提供了渐近优势。在实践中，这两种方法很难直接比较，它们优化了不同的目标函数，因此得到的训练模型的质量将不同。然而，可以合理地预期，量子方法将倾向于找到更好的模型，因为它优化了最大似然目标函数，直到由于采用有限 $N_{\mathrm{train}}$ 而导致的采样误差。

需要关注的是，Algorithm 3比许多现有的量子机器学习算法具有重要优势 [22–25]：它不需要将训练向量存储在量子存储器中。如果在计算 $E(v,h)-\log(Q(v,h))$ 时需要 $\mathcal{E}$ 的数值精度，则只需要 $n_h+n_v+1+\lceil\log_2(1/\mathcal{E})\rceil$

**Algorithm 3** 用于估计 $O_{\mathrm{ML}}$ 梯度的 GEQS 算法。

**Input:** 初始模型权重 $w$、可见偏差 $b$、隐藏偏差 $d$、边缘集 $E$ 和 $\kappa$、一组训练向量 $x_{\mathrm{train}}$、正则化项 $\lambda$ 和学习率 $r$。
**Output:** 三个数组包含权重梯度、隐藏偏差和可见偏差：$\mathtt{gradMLw}, \mathtt{gradMLb}, \mathtt{gradMLd}$.

**for** $i = 1 : N_{\mathrm{train}}$ **do**
  $\mathtt{success} \leftarrow 0$
  **while** $\mathtt{success} = 0$ **do**
    $|\psi\rangle \leftarrow \mathtt{qGenModelState}(w, b, d, E, \kappa)$
    $\mathtt{success} \leftarrow |\psi\rangle$ 中最后一个量子位的测量结果
  **end while**
  $\mathtt{modelVUnits}[i] \leftarrow |\psi\rangle$ 中可视化量子位寄存器的测量结果
  $\mathtt{modelHUnits}[i] \leftarrow$ 使用振幅放大法测量 $|\psi\rangle$ 中隐藏单位寄存器的结果
  $\mathtt{success} \leftarrow 0$
  **while** $\mathtt{success} = 0$ **do**
    $|\psi\rangle \leftarrow \mathtt{qGenDataState}(w, b, d, E, \kappa, x_{\mathrm{train}}[i])$.
    $\mathtt{success} \leftarrow$ 用振幅放大法测量 $|\psi\rangle$ 中最后一个量子位的结果
  **end while**
  $\mathtt{dataVUnits}[i] \leftarrow |\psi\rangle$ 中可见量子位寄存器的测量结果
  $\mathtt{dataHUnits}[i] \leftarrow |\psi\rangle$ 中隐藏单位寄存器的测量结果
**end for**
**for** 每个可见单元 $i$ 和隐藏单元 $j$ **do**
  $\mathtt{gradMLw}[i,j] \leftarrow r\left(\frac{1}{N_{\mathrm{train}}}\sum_{k=1}^{N_{\mathrm{train}}}(\mathtt{dataVUnits}[k,i]\mathtt{dataHUnits}[k,j] - \mathtt{modelVUnits}[k,i]\mathtt{modelHUnits}[k,j]) - \lambda w_{i,j}\right)$.
  $\mathtt{gradMLb}[i] \leftarrow r\left(\frac{1}{N_{\mathrm{train}}}\sum_{k=1}^{N_{\mathrm{train}}}(\mathtt{dataVUnits}[k,i] - \mathtt{modelVUnits}[k,i])\right)$.
  $\mathtt{gradMLd}[j] \leftarrow r\left(\frac{1}{N_{\mathrm{train}}}\sum_{k=1}^{N_{\mathrm{train}}}(\mathtt{dataHUnits}[k,j] - \mathtt{modelHUnits}[k,j])\right)$.
**end for**

量子比特。这意味着，假设 32 位的精度足以满足能量要求，可以用少于 100 个量子位来演示这种无法经典模拟的算法。但实际上，在量子计算机上实现所需的运算可能需要额外的量子比特。然而，量子旋转合成的最新发展可以用于消除能量明确存储为量子比特串的要求 [28]，这可以大大减少该算法的空间要求。下面我们考虑相反的情况：量子计算机可以通过预言机连贯地访问训练数据的数据库。该算法需要更多的量子位（空间），但在某些情况下，它可以二次减少学习所需的样本数。

### 附录 C: 通过量子幅度估计进行训练

我们现在考虑一个不同的学习环境，其中用户可以通过量子预言机访问训练数据，量子预言机可以表示提供训练数据的有效量子算法（例如用作生成模型的另一个玻尔兹曼机器）或通过二元访问树存储存储器的量子数据库 [26, 27]，例如量子随机存取存储器（qRAM）[27]。

如果我们将训练集表示为 $\{x_i | i = 1, \ldots, N_{\mathrm{train}}\}$，则 oracle 定义为酉运算，如下所示：

**Definition 4.** *$U_O$ 是对任何计算基状态 $|i\rangle$ 和任何 $y \in \mathbb{Z}_2^{n_v}$ 执行的酉运算*

$$U_O |i\rangle |y\rangle := |i\rangle |y \oplus x_i\rangle ,$$

*其中 $\{x_i | i = 1, \ldots, N_{\mathrm{train}}\}$ 是训练集并且 $x_i \in \mathbb{Z}_2^{n_v}$。*

对 $U_O$ 的单量子访问足以准备所有训练数据的均匀分布

$$U_O\left(\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle|0\rangle\right)=\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle|x_i\rangle\,. \tag{C1}$$

状态 $\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle|0\rangle$ 可以使用量子技术有效地制备 [25]，因此整个过程是高效的。

乍一看，准备叠加训练集中所有数据的能力似乎是一种强大的资源。然而，通过选取随机训练向量，也可以使用一个查询生成类似的概率分布。如果我们希望利用训练数据的这种量子叠加，就需要更复杂的方法。Algorithm 4利用这种叠加来在某些情况下为计算梯度提供优势。该算法的性能在以下定理中给出。

**Theorem 2.** *对于恒定的学习率 $r$ 和任意 $(i,j)$，在误差 $\delta$ 内使用对 $U_O$ 的预期查询数，有一种量子算法可以计算 $r\frac{\partial O_{\text{ML}}}{\partial w_{ij}}$、$r\frac{\partial O_{\text{ML}}}{\partial b_i}$ 或 $r\frac{\partial O_{\text{ML}}}{\partial d_j}$ 对应于带有 $E$ 条边的连接图上玻尔兹曼机器的可见/隐藏单元对，其规模为*

$$\tilde{O}\left(\frac{\kappa+\max_v\kappa_v}{\delta}\right),$$

*以及许多量子运算，其规模为。*

$$\tilde{O}\left(\frac{E(\kappa+\max_v\kappa_v)}{\delta}\right),$$

Algorithm 4需要使用振幅估计 [21] 算法，该算法提供了学习事件发生概率所需样本数量的二次减少，如以下定理所述。

**Theorem 3** (Brassard、Høyer、Mosca 和 Tapp)**.** *对于任何正整数 $L$，振幅估计算法将不使用测量且具有成功概率 $a$ 的量子算法作为输入，并输出 $\tilde{a}$ $(0\le\tilde{a}\le 1)$，即*

$$|\tilde{a}-a|\le\frac{\pi(\pi+1)}{L}$$

*其概率至少为 $8/\pi^2$。该定理使用了 Grover 算法的 $L$ 次迭代。如果 $a=0$，则可确定 $\tilde{a}=0$，如果 $a=1$ 并且 $L$ 为偶数，则可确定 $\tilde{a}=1$。*

这个结果是下面给出的Theorem 2的证明的核心。

*Proof of Theorem 2.* Algorithm 4计算 $O_{\text{ML}}$ 相对于权重的导数。该算法可适于计算关于偏差的导数。算法的第一步准备所有训练数据的均匀叠加，然后对其应用 $U_O$。根据要求，其结果为

$$\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle\,, \tag{C2}$$

任何不使用测量的量子算法都是线性的，因此将 `qGenDataState`（Algorithm 2）应用于(C2)会产生

$$\begin{aligned}&\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle\sum_h\sqrt{Q(x_p,h)}\,|h\rangle|\mathcal{P}(x_p,h)\rangle\left(\sqrt{1-\mathcal{P}(x_p,h)}\,|0\rangle+\sqrt{\mathcal{P}(x_p,h)}\,|1\rangle\right)\\ &:=\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle\sum_h\sqrt{Q(x_p,h)}\,|h\rangle|\mathcal{P}(x_p,h)\rangle|\chi(x_p,h)\rangle\,.\end{aligned} \tag{C3}$$

如果我们认为测量 $\chi=1$ 是成功的，那么Theorem 3给出了 $\tilde{O}((\kappa+\max_v\kappa_v)/\Delta)$（(C3)）的准备工作需要学习 $P(\texttt{success})=P(\chi=1)$ 到相对误差 $\Delta/8$ 以内具有高概率。这是因为 $P(\texttt{success})\ge 1/(\kappa+\max_v\kappa_v)$。同

**Algorithm 4** 用于估计 $O_{\mathrm{ML}}$ 梯度的 GEQAE 算法。

**Input:** 初始模型权重 $w$、可见偏差 $b$、隐藏偏差 $d$、边缘集 $E$ 和 $\kappa$、一组训练向量 $x_{\mathrm{train}}$、正则化项 $\lambda$、$1/2 \geq \Delta > 0$、学习率 $r$ 和规范边 $(i,j)$。

**Output:** 在误差 $2r\Delta$ 范围内计算 $r\frac{\partial O_{\mathrm{ML}}}{\partial w_{ij}}$

调用 $U_O$ 一次以制备状态 $|\psi\rangle \leftarrow \frac{1}{\sqrt{N_{\mathrm{train}}}} \sum_{p \in x_{\mathrm{train}}} |p\rangle |x_p\rangle$。

$|\psi\rangle \leftarrow \texttt{qGenDataState}(w, b, d, E, \kappa, |\psi\rangle)$. ▷ 使用 $x_p$ 的叠加而不是单个值应用到Algorithm 2上。

在 $|\psi\rangle$ 的状态准备过程中在误差 $\Delta/8$ 范围内使用振幅估计来学习 $P([x_p]_i = h_j = \texttt{success} = 1)$。

在 $|\psi\rangle$ 的状态准备过程中在误差 $\Delta/8$ 范围内使用振幅估计来学习 $P(\texttt{success} = 1)$。

$\langle v_i h_j \rangle_{\mathrm{data}} \leftarrow \frac{P([x_p]_i = h_j = \mathrm{success} = 1)}{P(\mathrm{success} = 1)}$.

在 $\texttt{qGenModelState}(w, b, d, E, \kappa)$ 上以完全相同的方式使用振幅估计来学习 $\langle v_i h_j \rangle_{\mathrm{data}}$。

$\frac{\partial O_{\mathrm{ML}}}{\partial w_{ij}} \leftarrow r\left(\langle v_i h_j \rangle_{\mathrm{data}} - \langle v_i h_j \rangle_{\mathrm{model}}\right)$

样，我们也可以将 success 视为第 $i^{\mathrm{th}}$ 个可见单位为 1、第 $j^{\mathrm{th}}$ 个隐藏单位为 1 的事件，并测量 success 的状态准备。这个标记过程与前面的情况完全相同，但需要一个 Toffoli 门。因此，可以在相对误差 $\Delta/8$ 范围内学习 $P(v_i = h_j = \chi = 1)$ 使用 $\tilde{O}((\kappa + \max_v \kappa_v)/\Delta)$ 准备工作。然后，根据条件概率定律

$$\langle v_i h_j \rangle_{\mathrm{data}} = P([x_p]_i = h_j = 1 | \chi = 1) = \frac{P([x_p]_i = h_j = \chi = 1)}{P(\chi = 1)}, \tag{C4}$$

可以根据这些值计算。

为了确保 $\langle v_i h_j \rangle_{\mathrm{data}}$ 中的总误差不超过 $\Delta$，我们需要限制(C4)中商的误差。可以看出，对于 $\Delta < 1/2$，

$$\left| \frac{P([x_i]_j = h_k = \chi = 1)(1 \pm \Delta/8)}{P(\chi = 1)(1 \pm \Delta/8)} - \frac{P([x_i]_j = h_k = \chi = 1)}{P(\chi = 1)} \right| \leq \frac{\Delta P([x_i]_j = h_k = \chi = 1)}{P(\chi = 1)} \leq \Delta. \tag{C5}$$

因此，该算法给出了误差 $\Delta$ 范围内的 $\langle v_i h_j \rangle_{\mathrm{data}}$ 数据。

可以使用Algorithm 1代替Algorithm 2来重复完全相同的步骤，作为振幅估计中使用的状态准备子程序。这允许我们在误差 *Delta* 范围内使用 $\tilde{O}(1/\Delta)$ 准备计算工作 $\langle v_i h_j \rangle_{\mathrm{data}}$。三角形不等式表明，近似 $\langle v_i h_j \rangle_{\mathrm{data}} - \langle v_i h_j \rangle_{\mathrm{model}}$ 产生的最大误差最多为 $2\Delta$ 因此，由于在Algorithm 4中使用了 $r$ 的学习率，因此导数的总误差最多为 $2\Delta r$。如果我们选择 $\Delta = \delta/(2r)$，则我们看到整个算法需要为常数 $r$ 准备 $\tilde{O}(1/\delta)$ 状态。

每个状态准备都需要对 $U_O$ 和 $\tilde{O}(E)$ 操作进行一次查询，假设构成玻尔兹曼机器的图是连通的。这意味着该算法的预期查询复杂度为 $\tilde{O}((\kappa + \max_v \kappa_v)/\delta)$，所需电路元件的数量为 $\tilde{O}((\kappa + \max_v \kappa_v)E/\delta)$。 □

该方法与Algorithm 3的方法有两个定性差异。第一个是算法提供了关于梯度的一个分量的详细信息，而Algorithm 3的样本提供了关于每个方向的有限信息。如果振幅估计用于学习 $\langle v_i h_j \rangle$，那么 $\langle v_k h_\ell \rangle$ 可以通过从剩余的量子位采样来估计，这似乎是合理的。问题是振幅估计会在这些测量中产生偏差，因此使用这些证据来更新用户对梯度剩余分量的信心的正确方法尚不清楚。

一般来说，盲目地使用振幅放大来提供 $\kappa$ 缩放的二次改进是不可行的，因为在成功概率未知的情况下通常采用随机算法。由于随机化算法使用测量值，因此不能与幅度估计一起使用。然而，如果成功概率的上限是已知的，则可以确定地使用振幅放大来导致放大系统的成功概率。然后可以对原始电路的放大版本采用幅度估计，然后可以回溯推断的成功概率。这个过程在下面的推论中解释。

**Corollary 1.** *已知 $P_u \in \Theta(\frac{Z}{\kappa Z_Q})$，假设 $P_u : 1 \geq P_u > \frac{Z}{\kappa Z_Q}$，则对于恒定的学习速率 $r$ 和任意的 $(i,j)$，在误差 $\delta \leq P_u$ 内使用对 $U_O$ 的预期查询数，有一种量子算法可以计算 $r\frac{\partial O_{\mathrm{ML}}}{\partial w_{ij}}$，$r\frac{\partial O_{\mathrm{ML}}}{\partial b_i}$ 或 $r\frac{\partial O_{\mathrm{ML}}}{\partial d_j}$ 对应于带有 $E$ 条边的*

连接图上玻尔兹曼机器的可见/隐藏单元对，其规模为

$$\tilde{O}\left(\frac{\sqrt{\kappa}+\max_v\sqrt{\kappa_v}}{\delta/P_u}\right),$$

以及许多量子运算，其规模为

$$\tilde{O}\left(\frac{E(\sqrt{\kappa}+\max_v\sqrt{\kappa_v})}{\delta/P_u}\right).$$

证明. 这个推论背后的想法是，在估计振幅之前，应用 Grover 搜索的 $m$ 次迭代来提高成功概率，然后使用最终的成功概率来反向计算 $P(11):=P([x_p]_i=h_j=\chi=1)$ 或 $P(1):=P(\chi=1)$。

注意到学习 $P(1)$。$P(11)$ 的情况是相同的。应用 $m$ 步振幅放大（不测量）会导致成功概率变为

$$P_s=\sin^2([2m+1]\sin^{-1}(\sqrt{P(1)})). \tag{C6}$$

这个方程不能倒转来求 $P_s$，除非 $[2m+1]\sin^{-1}(P(1))\le\pi/2$。然而，如果选择 $m$ 使得 $[2m+1]\sin^{-1}(P_u)\le\pi/2$，则我们保证存在逆。在此假设下

$$P(1)=\sin^2\left(\frac{\sin^{-1}(\sqrt{P_s})}{2m+1}\right). \tag{C7}$$

现在我们需要证明，估计 $P_s$ 中的小误差不会传播到 $P(1)$ 中的大误差。一般来说，由于(C7)相对于 $P_s$ 的导数在 $P_s=1$ 时发散，所以会传播较大的误差。让我们假设选择 $m$ 使 $P_s\le 1/4$。如果这样的 $m$ 不存在，那么成功概率已经是 $O(1)$，因此复杂性与 $\kappa$ 无关，因此我们可以安全地假设这是最坏的情况假设。然后存在 $P(1)$ 的导数为

$$\frac{\partial P(1)}{\partial P_s}=\frac{\sin\left(\frac{2\sin^{-1}(\sqrt{P_s})}{2m+1}\right)}{2\sqrt{P_s}\sqrt{1-P_s}(2m+1)}. \tag{C8}$$

由于 $\sin(x)\le x$，则这个方程变成

$$\frac{\sin\left(\frac{2\sin^{-1}(\sqrt{P_s})}{2m+1}\right)}{2\sqrt{P_s}\sqrt{1-P_s}(2m+1)}\le\frac{\sin^{-1}(\sqrt{P_s})}{\sqrt{P_s}\sqrt{1-P_s}(2m+1)^2}. \tag{C9}$$

根据泰勒公式，$\sin^{-1}(x)$ 只有正项，且 $\sin^{-1}(x)/x=1+O(x^2)$, $\sin^{-1}(\sqrt{P_s})/\sqrt{P_s}\ge 1$。Ergo(C9)是 $P_s$ 在 $(0,1)$ 上的单调递增函数。因此，根据极值定理得

$$\frac{\partial P(1)}{\partial P_s}\le\frac{2\pi}{3^{3/2}(2m+1)^2}. \tag{C10}$$

泰勒余数定理表明，如果精确地使用相位估计 $\Delta_0:-P_s\le\Delta_0\le 1/4-P_s$，则

$$\left|\frac{\sin\left(\frac{2\sin^{-1}(\sqrt{P_s})}{2m+1}\right)}{2\sqrt{P_s}\sqrt{1-P_s}(2m+1)}-\frac{\sin\left(\frac{2\sin^{-1}(\sqrt{P_s+\Delta_0})}{2m+1}\right)}{2\sqrt{P_s+\Delta_0}\sqrt{1-P_s-\Delta_0}(2m+1)}\right|\le\frac{2\pi\Delta_0}{3^{3/2}(2m+1)^2}. \tag{C11}$$

因此，结果误差为 $O(\Delta_0/m^2)$。则如果总体误差为 $\Delta/8$，那么取 $\Delta_0\in O(m^2\Delta)$ 即可。这里是 $m\in\Theta(\sqrt{1/P_u})$ 意味着 $\Delta_0\in O(\Delta/P_u)$。因此，振幅估计需要 $O(P_u/\Delta)$ 用于产生 $P(1)$ 和 $P(11)$ 的放大电路的重复。假设 $P_u\in\Theta(\frac{Z}{\kappa Z_Q})$，振幅放大导致 $O(\sqrt{\kappa}+\max_v\sqrt{\kappa_v})$ 的开销。然后，总成本就是这两个成本的乘积，导致了所谓的复杂性。 □

可以对梯度向量的每个分量重复上述过程，以便更新玻尔兹曼机的权重和偏差。

**Corollary 2.** 假设 $N_{\text{op}}$ 是使用 *Algorithm 4*或 *Corollary 1*计算连通图上的玻尔兹曼机的 $O_{\text{ML}}$ 梯度分量所需的量子运算与预言调用的数量，其规模为

$$\tilde{O}\left(EN_{\text{op}}\right)$$

如果学习率 $r$ 为常数。

证明. 该证明是使用Theorem 2 $O(E)$ 次的结果来计算梯度向量的每个分量的一个微不足道的结果。 □

与先前的算法不同，很难有意义地将Corollary 2中的成本与对比差异下的训练成本进行比较。这是因为Algorithm 4使用量子叠加来同时使用所有训练数据来计算相关期望值。因此，使用整个数据集计算导数算子的每个分量，最好将运行时间视为估计误差的函数，而不是训练向量的数量。比较的一个自然指标是假设训练集是从可以考虑的更大的训练数据集中提取的。在这种情况下，在 $O(1/\sqrt{N_{\text{train}}})$。因此，取 $\delta \in O(1/\sqrt{N_{\text{train}}})$ 比较两种方法的合理选择，但这绝不是对两种成本进行有意义比较的唯一方法。

尽管查询复杂度与训练向量的数量无关，但为了在实际示例中评估该算法的成本，我们还需要包括实例化预言机的成本。我们考虑三种情况。如果每个预言机都实现了一个有效的可计算函数，那么在 $N_{\text{train}}$ 中实现预言机的空间和时间复杂性是多对数的。另一方面，如果数据只能通过查找表访问（大多数机器学习问题都是如此），那么允许并行执行的量子计算机可以实现使用存储器 $O(N_{\text{train}})$ 在时间 $O(\text{polylog}(N_{\text{train}}))$ 中存储预言。另一方面，如果量子计算机只能串行处理信息，则需要 $\Theta(N_{\text{train}})$ 空间和时间来使用量子计算机中作为量子比特串存储的训练向量数据库来实现预言查询。下界来自奇偶函数的下界，该下界表示需要对该数据库中的位进行 $\Theta(N)$ 查询，以确定 $N$ 个量子位串的奇偶性。这表明，依赖于训练向量的数量取决于特定于问题和体系结构的问题。

具有 $E$ 的二次缩放意味着对于学习所有权重，Algorithm 4可能不优于Algorithm 3。另一方面，Algorithm 4可用于改进使用先前方法估计的梯度。其想法是从使用直接梯度估计方法的初步梯度估计步骤开始，同时使用 $O(\sqrt{N_{\text{train}}})$ 随机选择的训练向量。然后通过将结果分成更小的组并计算每个子组上梯度向量的每个分量的平均值和方差来估计梯度。然后通过将结果分成更小的组并计算每个子组上梯度向量的每个分量的均值和方差来估计梯度。然后可以学习具有最大不确定性的梯度分量，其误差与在对比散度训练中仅使用 $N_{\text{train}}$ 训练示例所产生的采样误差相当，通过使用 $\delta \sim 1/\sqrt{N_{\text{train}}}$ 的Algorithm 4来估计它们。由于这两种成本是渐近可比的，因此这种方法允许在梯度中的大部分不确定性来自少量分量的情况下使用这两种方法的优点。

## 附录 D: 对冲策略

$\kappa$ 中的较大数值可能是面对精确状态准备的障碍。这个问题的出现是因为 $Q(v,h)$ 可能比 $P(v,h)$ 给构型分配的概率小显著的数量级。例如，我们发现玻尔兹曼机的例子要求 $\kappa$ 的值大于 $10^{20}$，这就是为即便是小的玻尔兹曼机也能精确地准备吉布斯态。这个问题的根源在于，取 $Q(v,h)$ 为平均场分布并不总是能充分反映 $P(v,h) \approx Q(v,h)$ 的不确定性。我们引入了“对冲策略”来解决这个问题。我们的策略是引入一个对冲参数 $\alpha : 1 \geq \alpha \geq 0$，可以调整该参数以减少对平均场状态的偏差。如果 $\mu_i$ 的平均场期望值 $v_i$，在没有对冲的情况下，则选择 $\alpha < 1$，结果为 $\mu_i \to \alpha\mu_i + (1-\alpha)/2$，对于隐藏单元也是如此。$Z_Q$ 保持相同的值，不管 $\alpha$ 取什么值。

这种策略也可以从贝叶斯的角度认为是参数化先验分布，从平均场 ansat 的完全置信度过渡到均匀先验。从这个状态准备吉布斯状态则对应于这个先验的更新，其中 $\mathcal{P}(v,h)$ 是这种语言中的似然函数。

通过使用对冲，状态准备基本上没有改变，因为唯一的区别是指定 $Q(v,h)$ 的平均场参数被改变了。使用 [36] 方法制备平均场状态和均匀分布的线性组合也可以获得类似的效果，但为了简单起见，我们只关注前者的方法。我们在下面看到，对冲不会大幅增加算法的开销，但在 MF 近似开始崩溃时，可以大幅降低 GEQS 和 GEQAE 的复杂性。

| | First Optimizer | Second Optimizer |
|---|---|---|
| CD–ML | CD−1 | Gradient ascent on ML objective |
| ML–CD | BFGS on ML objective | CD−1 |
| ML–ML | BFGS on ML objective | Noisy gradient ascent on ML objective. |

表 III: Numerical experiments.

## 附录 E: 数值实验

在本节中，我们量化了使用对比散度训练玻尔兹曼机（参见Section H节对对比散度的简要回顾）和通过使用Algorithm 3或Algorithm 4来优化 $O_{\mathrm{ML}}$ 来训练它们之间的差异。在这里，我们提出了额外的数据，这些数据检查了我们的算法在更广泛的条件下的性能，而不是在主体中考虑的那些条件。特别是，我们做了对玻尔兹曼机和受限玻尔兹曼机的 $O_{\mathrm{ML}}$ 和 $\kappa$ 的详细调查。我们还提出证据表明，主体中的结果（涉及对合成数据集的训练）与使用从 MNIST 数据库中提取的次采样手写数字进行训练获得的结果相当。

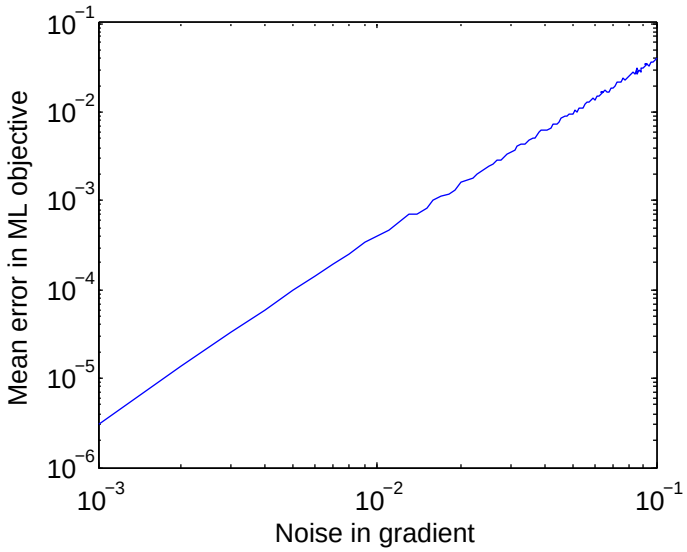
### 1. 数据和方法

我们使用梯度上升训练 dRBM 模型，使用 (a) 对比发散来近似梯度和 (b) 使用Algorithm 3或Algorithm 4的目标优化。这两种情况下的目标函数都是 $O_{\mathrm{ML}}$。由于梯度的不同近似会导致学习不同的局部最优，即使在两个实例中使用相同的初始条件，直接比较使用两种训练技术发现的最优向量可能是不公平的。我们考虑了两种比较方法。首先，我们通过使用 Donmez、Svore 和 Burges [37] 的具有高概率方法验证我们的结果大约位于最优值 $O_{\mathrm{ML}}$ 处。对于每个提出的最优点，我们对该点进行多次摄动，在摄动的同时固定所有参数，并且对每个参数重复多次，然后比较目标函数在原始点和摄动点的值之差。这让我们可以在固定置信度的情况下说，目标函数在随机选择的一个方向上递减的概率小于一个引值。我们在大小为 $10^{-3}$ 的扰动下重复这个过程 459 次，这足以保证对于大小为 $10^{-3}$ 的步骤，目标函数不会在所有随机选择的方向的 99% 上增加。

其次，我们使用了类似于 [38] 中使用的方法。我们通过运行一个算法，直到找到一个局部最优，然后使用这个局部最优作为第二个算法的初始配置来执行我们的实验。这样我们就可以比较类似局部最优的位置和质量。我们在Table III中列出了训练组合，我们将其表示为 CD–ML、ML–CD 和 ML–ML，分别对应于比较的第一步和第二步中使用的优化器。考虑这类比较的一个微妙之处是收敛性的确定。对比发散和其他有噪声的梯度上升算法（意思是在梯度计算中加入噪声的梯度上升）不会收敛到单个最优点，而是在一个近似的不动点上下波动。处于这个原因，我们认为当 $O_{\mathrm{ML}}$ 的运行平均值在学习率为 $r = 0.01$ 的至少 $10,000$ 训练时期后变化小于 0.001% 时算法已经收敛。我们不仅在我们的对比散度计算中应用了这个停止条件，而且在我们确定 ML 目标函数的梯度中引入采样噪声的效果时也应用了这个停止条件。我们通常使用 Broyden–Fletcher–Goldfarb–Shanno 算法（BFGS）在无噪声的情况下优化 ML 目标函数，也使用离散导数的梯度上升方法。在这两种情况下，我们选择当 ML 目标函数的绝对误差为 $10^{-7}$ 时对应的停止条件。

### 2. 梯度中噪声的影响

我们首先确定是否可以从一个量子设备的少量样本中获得足够准确的梯度估计，例如在使用Algorithm 3或Algorithm 4进行训练时。这里，我们使用 ML–ML 训练一个有 6 个可见单元和 4 个隐藏单元的单层受限玻尔兹曼机。然后，我们继续计算 ML 目标函数的梯度，并在梯度向量上添加零均值高斯噪声。训练数据由 $10,000$ 个训练示例组成。每个数据点至少使用 $10,000$ 个训练周期，在满足停止条件




(a)

Fit to Error $= a \times \text{Noise}^b$.

| Number of hidden units | a | b |
|---|---|---|
| 4 | 1.4044 | 2.0161 |
| 6 | 1.4299 | 2.0121 |
| 8 | 1.5274 | 2.0086 |

(b)

图 5: (a) 通过在 ML–目标处初始化并对计算出的梯度的每个分量引入高斯噪声，得到 ML–目标与 ML 之间的差异。对于 $\mathcal{N} = 0$ 和 100 个样本的 $6 \times 4$ 单位的受限玻尔兹曼机所取的数据，用于每个数据点的平均值。(b)$6 \times n_h$ 单位的受限玻尔兹曼机时的结果在 $n_h = 6, 8$ 上是定性相同的，并强烈支持梯度评价中误差与噪声的二次关系。

(在Section E 1中给出）之前，通常需要 $20,000$ 个训练周期。

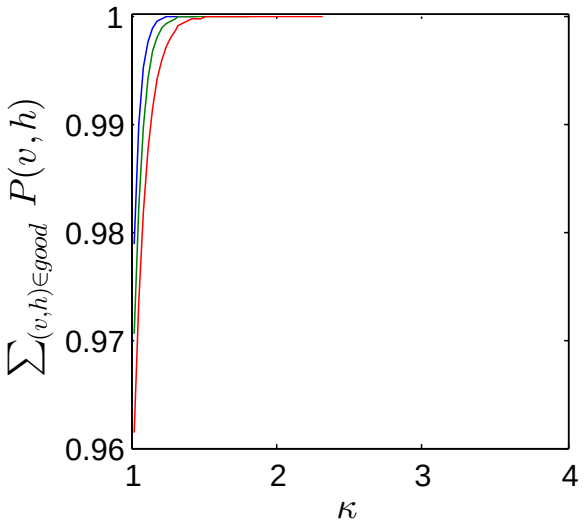
Figure 5显示，目标函数值的平均误差随梯度中的噪声呈二次比例增长。这意味着梯度上升算法对梯度分量中的采样噪声具有很强的弹性，一个相对较小的 $\delta$ 值，例如 $\delta = 0.01$ 就足以产生接近于 $\delta = 0$ 时发现的局部最优值。因此，小的采样误差对Algorithm 4来说不会是灾难性的。

如果抽样误差是零，那么学习率总是可以调整，以减少这种抽样误差。这意味着，如果需要改进的梯度，那么就没有必要增加 GEQS 中训练集的大小。然而，在 GEQAE 中并不能保证错误是无偏的，因此降低学习率可能并不总是足以减少此类错误。无论如何，有多种策略可以将这种错误减少到 $10^{-2}$ 在训练过的模型的质量中实现可忽略误差的经验所需的阈值。

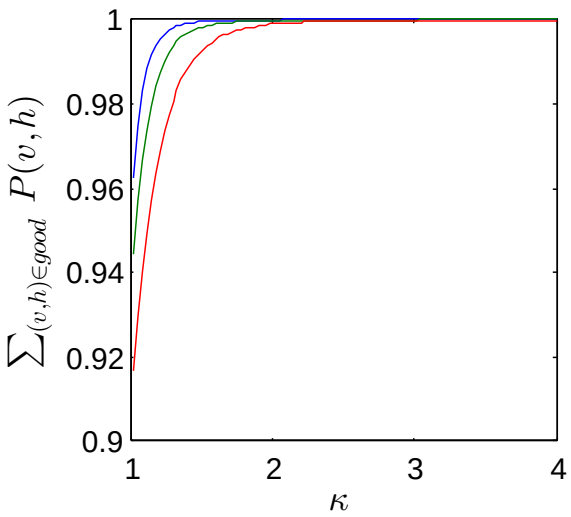
### 3. 平均场近似和 $\kappa$ 的缩放导致的误差

我们的量子算法依赖于从平均场或相关近似准备吉布斯态近似的能力。Lemma 2和Lemma 3表明，算法的成功概率强烈依赖于选择的 $\kappa$ 值，如果使用的 $\kappa$ 太小或 $Z_Q$ 与 $Z$ 过于不同，则导数的准确性将受到影响。我们分析了具有 $4, 6, 8$ 个可见单元和 4 个隐藏单元的随机单层受限玻尔兹曼机的结果。标准差的值是一个重要的问题，因为众所周知，如果更强的权重（即，更强的相关性）被引入到模型 [39]，那么平均场分布的估计的质量将会下降。我们在波尔兹曼机中取正态分布的权重，其平均值为零，标准差为 0.1325 的倍数，我们选择该值来匹配一个 884 单元的受限玻尔兹曼机的权重分布的方差，该受限玻尔兹曼机被训练为使用对比散度执行面部识别任务。对于本节中的所有数值实验，偏差均根据具有零均值和单位方差的高斯分布来随机设置。

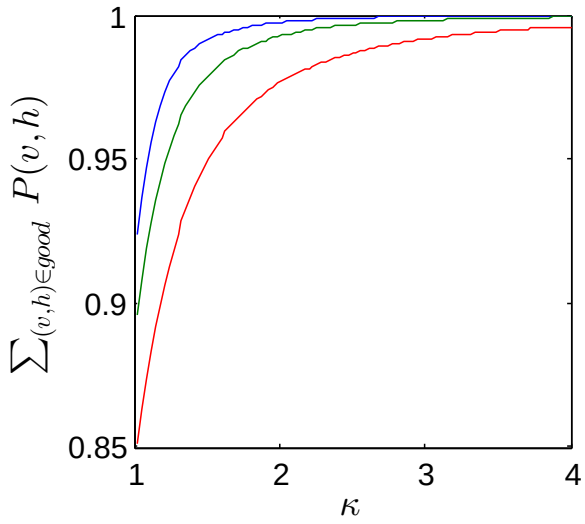
从Figure 6可以看出，$\kappa$ 的值并不是特别地大。事实上，对于所有考虑的情况，$\kappa < 10$ 足以产生接近零误差的状态。此外，我们还看到，$\kappa$ 的值是受限玻尔兹曼机中可见单位的数量和合成模型中使用的权重的标准差的一个缓慢递增的函数。然而，$\kappa$ 是标准差的递增函数这一事实不一定有问题，因为在实际的机器学习任务中，正则化往往会导致权重的标准差小于 1。由于选择的值敏感地依赖于为残差概率选择的截止点，因此很难从这些数




(a) Standard deviation = 0.1325




(b) Standard deviation = 0.2650




(c) Standard deviation = 0.5300

| $\sigma(w_{i,j})$ | $n_v$ | Mean KL$(Q\|\|P)$ | Mean ln$(Z)$ |
|---|---|---|---|
| 0.1325 | 4 | 0.0031 | 4.182 |
| 0.1325 | 6 | 0.0045 | 5.550 |
| 0.1325 | 8 | 0.0055 | 6.942 |
| 0.265 | 4 | 0.0115 | 4.211 |
| 0.265 | 6 | 0.0170 | 5.615 |
| 0.265 | 8 | 0.0245 | 7.020 |
| 0.53 | 4 | 0.0423 | 4.359 |
| 0.53 | 6 | 0.0626 | 5.839 |
| 0.53 | 8 | 0.0827 | 7.295 |

(d)

图 6: 对于合成 $n_v \times 4$ 的有着具有不同标准差的正态分布随机选择的权重的受限玻尔兹曼机的 $\mathcal{P}(v,h) \leq 1$ 对比 $\kappa$ 的概率分数，每张图从上到下 $n = 4, 6, 8$。表 (d) 显示了 (a)、(b) 和 (c) 中所示数据的平均场分布和吉布斯分布之间的 KL–散度值。在 100 个随机实例中计算得到期望值。

据中提取出 $\kappa$ 的缩放值。

量 $\kappa$ 很小，因为如果边权值很小，平均场近似非常接近真实的吉布斯态。这可以从Figure 6(d) 和Figure 7的表中看出，它们给出了随机受限玻尔兹曼机的平均场近似和真正的吉布斯状态之间的 KL–散度的平均值。对于现实的权值分布，KL$(Q||P)$ 往往小于 0.1，这意味着平均场近似往往会非常接近实际分布。KL–散度也是 log–配分函数变分逼近中的松弛（见Section G）。这意味着Figure 6中的数据也表明对于这些小型合成模型，$Z_Q$ 将会非常接近 $Z$。

在成功概率上有两个相互竞争的趋势。当平均场逼近开始失效时，我们预计 $\kappa$ 将会发散。另一方面，我们也期望 $Z/Z_Q$ 随着 $Q$ 和 $P$ 之间的 KL–散度的增加而增加。我们可以通过将 $Z/Z_Q$ 的规模纳入考虑来更好地理

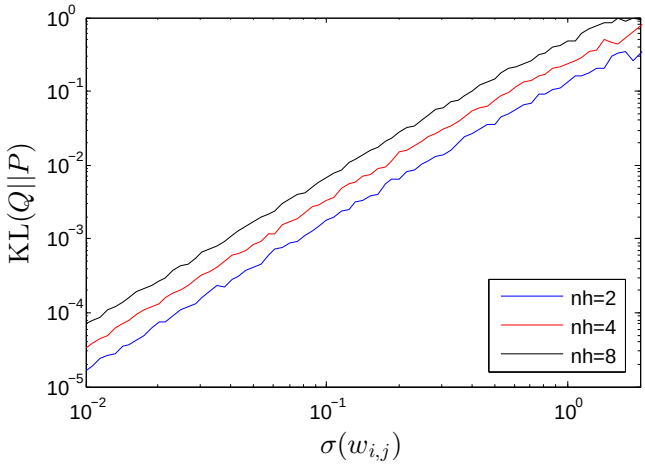



图 7: 含 $n_v = 4$ 以及具有零均值和 $\sigma^2(w_{i,j})$ 方差的高斯分布的合成受限玻尔兹曼机的 KL–散度的平均值。偏差设定为从均值和单位方差为零的高斯分布中得出。每个数据点是 100 个随机受限玻尔兹曼机的平均值，并且数据规模和 $O(\sigma^2(w_{i,j})E)$ 是一致的。

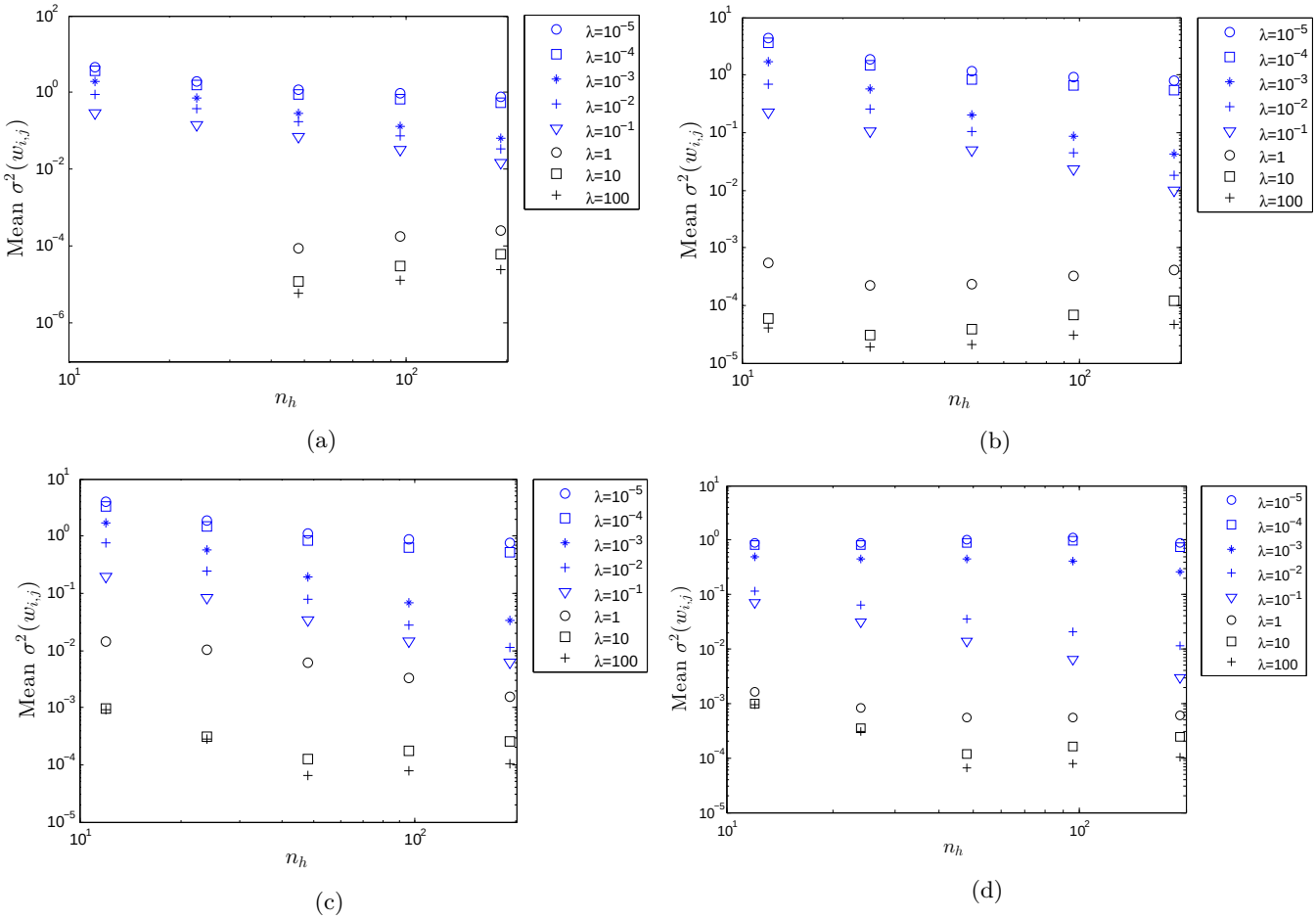



图 8: 带有 (a)$n_v = 12$、(b)$n_v = 24$、(c)$n_v = 48$ 和 (d)$n_v = 48$ 以及 $n_h = \{12, 24, 48, 96, 192\}$ 的受限玻尔兹曼机的方差和权重的平均值。每个受限玻尔兹曼机通过 CD1 在我们的合成数据集上进行 200,000 轮次的训练，其中学习率为 0.01。(a)–(c) 在训练集中使用零噪声，然而 (d) 给每个比特翻转一个 20% 的机会。对图中每个点取 100 个不同的局部最优值的平均值。

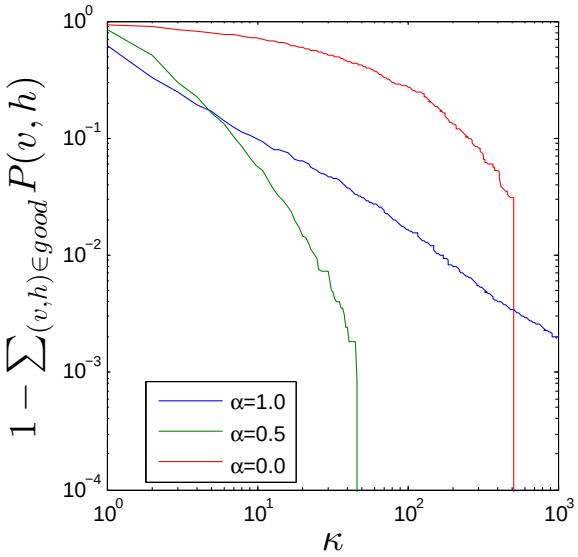



图 9: 配置的中值概率质量使得对于具有 $n_v = 6$ 和 $n_h = 8$ 的、使用具有 $\mathcal{N} = 0$ 的主体中描述的合成数据集进行训练的受限玻尔兹曼机具有 $P(v, h) \geq 1$。

解误差的规模。$Z_Q$ 是配分函数的近似，它服从 $\log(Z_Q) = \log(Z) - \mathrm{KL}(Q||P)$，这意味着

$$P_{\text{success}} \geq \frac{Z}{Z_Q \kappa} = \frac{e^{\mathrm{KL}(Q||P)}}{\kappa} \geq \frac{1}{\kappa}. \tag{E1}$$

Figure 6和Figure 7的数据表明，$\mathrm{KL}(Q||P)$ 的经验比例为 $O(\sigma^2(w_{i,j})E)$，其中 $E$ 为图中的边数并且 $\sigma(w_{i,j})$ 为合成模型权重的标准差。因此，我们期望 (a)$\sigma^2(w_{i,j}) \in O(1/E)$ 那么有 $P_{\text{success}} \approx 1/\kappa$ 并且 (b) 对于 $\sigma^2(w_{i,j})E \ll 1$ 有 $\kappa - 1 \in O(\sigma^2(w_{i,j})E)$。因此如果普遍出现在训练过程中的模型的 $\sigma^2(w_{i,j})E$ 很小，我们的算法应该既准确又高效。

我们进一步通过计算具有 $12, 24$ 和 $48$ 个可见单元和可变数量的隐藏单元的受限玻尔兹曼机的典型权重分布来研究这个问题，如Figure 8所示。这使我们能够检查使用对比散度训练的相对较大的受限玻尔兹曼机的边缘数量的规模。尽管通过对比散度学习到的权重不同于使用我们的量子算法学习到的权重，我们在Section E 4中看到这些受限玻尔兹曼机的差异通常很小，因此对比散度让我们很好地估计了 $\sigma(w_{i,j})$ 在训练过程中自然产生的大型模型的尺度。我们从Figure 8中注意到，随着更多的隐藏单元被添加到模型中，权重的标准差会迅速下降。这是因为正则化（即，$\lambda > 0$）为模型添加边提供了一个惩罚。$\lambda = 0.1$ 到 $\lambda = 0.001$ 的权重中的方差衰减速度快于 $\Theta(1/E)$ 缩放，该缩放预计会导致高成功概率以及对于 (a)-(c) 的 $Z_Q \approx Z$。(d) 中的结果在性质上是相似的，但必要的缩放只适用于 $\lambda = 0.1$ 到 $\lambda = 0.01$。正则化常数的这些值是实际算法中使用的典型值，因此我们不期望 $\kappa$ 的缩放和取 $Z \approx Z_Q$ 所产生的误差将成为将我们的方法（或其自然泛化）应用于实际机器学习问题的障碍。

可以使用几种策略来对抗吉布斯状态准备的低成功概率。在成功概率低得不可接受的情况下，则可在算法中使用比 $Z_Q$ 更准确的配位函数估计 [8, 20, 40, 41]。可以使用Algorithm 3来代替Algorithm 4在成功概率上提供二次的优势。选择的 $\kappa$ 也可以减小，如Lemma 3所示。在极端情况下，还可以调整正则化常数，以对抗训练过程中出现的大权重；然而，这有产生一个模型严重不符合数据的风险。

对冲策略也可以用来解决由 $\kappa$ 值可能过大带来的问题。我们在Figure 9中观察到这一点，其中研究了 $\mathcal{P}(v, h) \geq 1$ 的状态的概率质量。我们发现，在这种情况下，如果想要一个精确的状态，且 $\kappa$ 值很小，则不存在对冲（$\alpha = 1$）。适量的对冲（$\alpha = 0.5$）可以显著提高状态制备的准确性：相对于没有对冲的 $\kappa > 1000$，

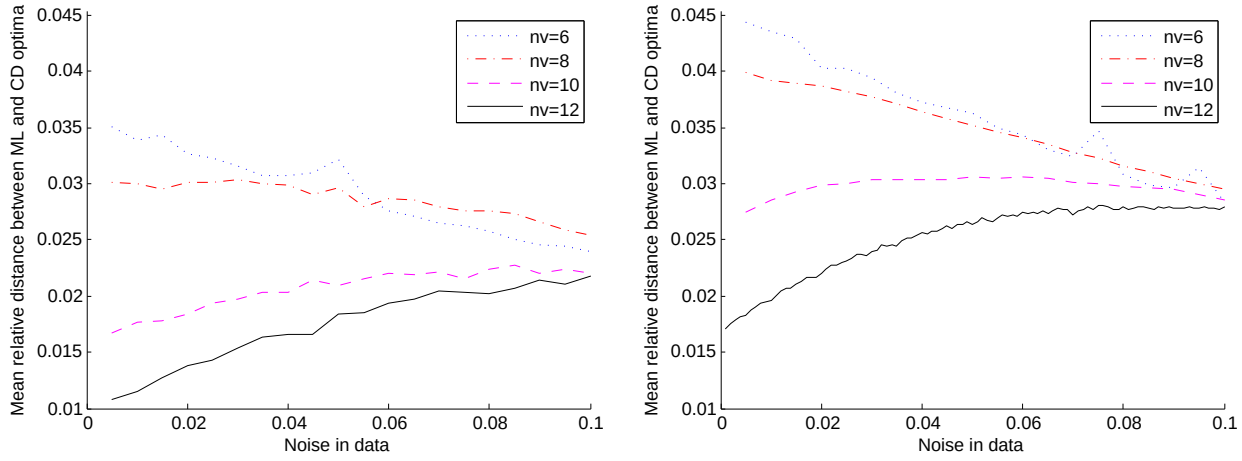



图 10: 具有 $n_v$ 可见单元和 (a)4 个隐藏单元和 (b)6 个隐藏单元的受限玻尔兹曼机的 CD 最优值和 ML 最优值之间的相对距离。

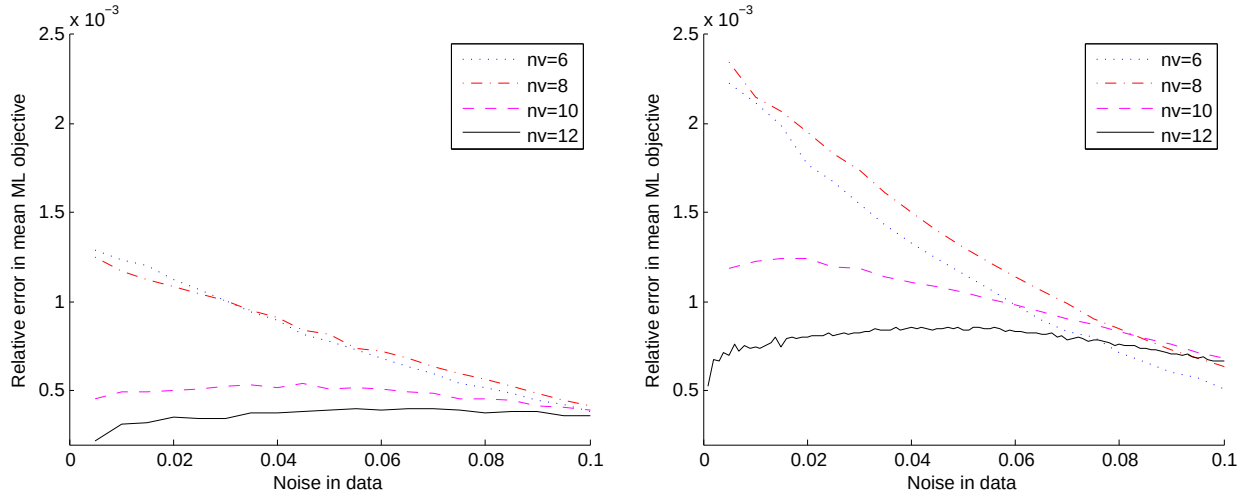



图 11: 对于具有 $n_v$ 可见单元和 (a)4 个隐藏单元和 (b)6 个隐藏单元的受限玻尔兹曼机，在 CD 最优值和 ML 最优值下计算 ML 目标值的相对差异。

$\kappa = 50$ 足以实现完美的状态制备。$\alpha = 1$ 的情况下不如 $\alpha = 0$ 或 $\alpha = 0.5$，除了这一事实：在这些情况下，不再需要连贯地计算 $Q(v, h)$（传统上仍然需要平均场计算来估计 $Z$）然而，正如我们后来看到的对冲策略对于更大的问题往往不会那么成功，而使用平均场状态作为初始分布 ($\alpha = 1$) 往往会导致 $\kappa$ 的值随着系统规模的增加而几乎恒定不变。

### 4. CD−1 和 ML 学习的比较

我们的量子算法的一个重要优势是，它们为训练深度受限玻尔兹曼机提供了一种替代对比发散的方法。我们现在研究的问题是，对比散度发现的最优值与 ML 目标优化发现的最优值之间是否存在实质性差异。我们在单层受限玻尔兹曼机上使用 CD–ML 进行训练，其中最多 12 个可见单元和最多 6 个隐藏单元，并且从使用 CD 发现的最优点开始，计算首次使用 CD−1 训练后发现的最优点与使用 ML 训练后发现的最优点之间的距离。我

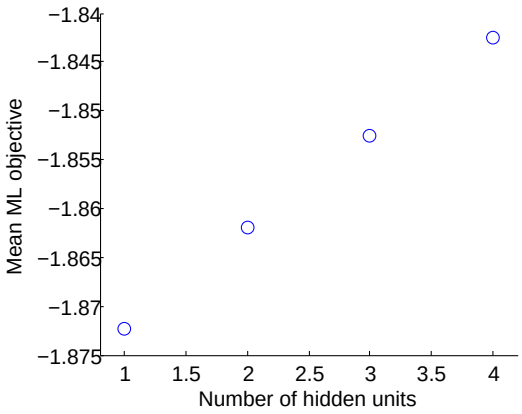



图 12: $O_{\mathrm{ML}}$ 用于具有 6 个可见单元和 1 到 4 个隐藏单元的完全链接的玻尔兹曼机。

们发现，使用两种方法发现的最优点的位置存在本质差异。

Figure 10说明了对比散度最优和相应 ML 最优之间的距离是相当显著的。通过 ML 训练和 CD 训练找到的模型之间的距离是通过将权重矩阵扁平化到一个向量，将结果与偏差向量连接，并计算两个向量之间的欧氏距离得到的。在无噪声的极限下，观察到几个百分点量级的差异。这表明使用 CD 优化和 ML 优化学习的模型可以有很大的不同。我们看到，随着更多的隐藏单元被添加到模型中，这些差异往往会增加，并且向训练数据中添加伯努利噪声往往会导致因 $n_v$ 的变化而产生的相对距离差异缩小。

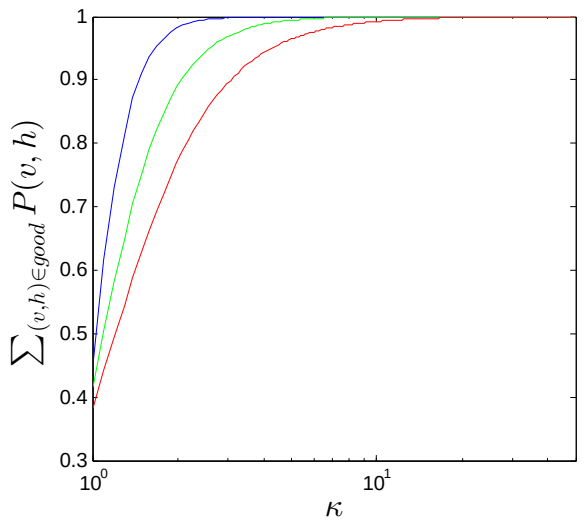
Figure 11表明，作为伯努利噪声添加到训练数据中的函数，发现的 ML 最优值的质量存在差异，其中质量是被在该点的 $O_{\mathrm{ML}}$ 的值所决定。对于这些示例，观察到的相对误差往往在 0.1 的数量级，，这个数量级很小但不可忽略，因为在当代机器学习应用程序中，这个数量级的分类误差差异非常显著。$O_{\mathrm{ML}}$ 值的差异遵循与Figure 10中的数据相似的趋势。

这些结果表明，即使是在小的例子中，CD 和 ML 最优值的位置和质量之间叶存在显著差异。因此，我们的非常接近 ML 训练的量子算法，如果 $\kappa$ 的值足够小，很可能得到比当前基于对比发散的最先进的经典方法更加改进的模型。这一点对经典机器学习方法也很重要，其中使用更昂贵的对比散度变体（例如 $k > 1$ 的 CD$-k$）也可能得到模型 [7] 质量上的显著差异。

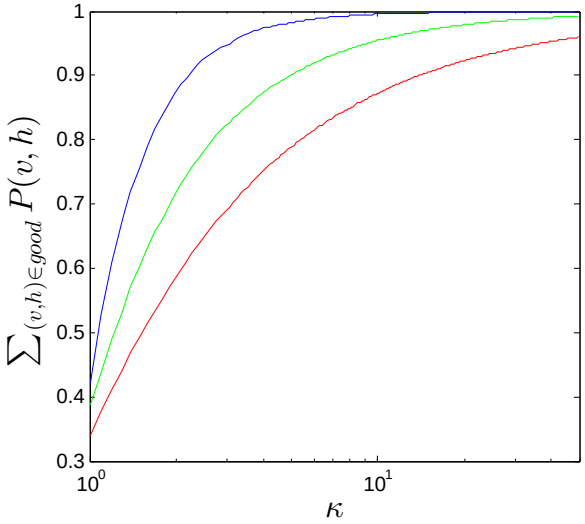
### 5. $O_{\mathrm{ML}}$ 下训练的玻尔兹曼机

前面的例子考虑了基于 ML 的学习在单层和多层受限玻尔兹曼机上的性能。在这里，我们检验了在训练任意两个单元之间具有任意连接的完整玻尔兹曼机时发现的 ML 最优的质量。虽然使用对比发散的经典训练需要在分层二部图（dRBM）上学习，但我们的量子算法不需要计算条件分布，因此可以高效地训练全玻尔兹曼机，因为吉布斯态的平均场近似与真实吉布斯态只有多项式般的小重叠。剩下的主要问题是，是否存在重叠是使用量子计算机来训练这样完整的图形模型的优势，如果这样的模型表现出优于 dRBM 的性能。

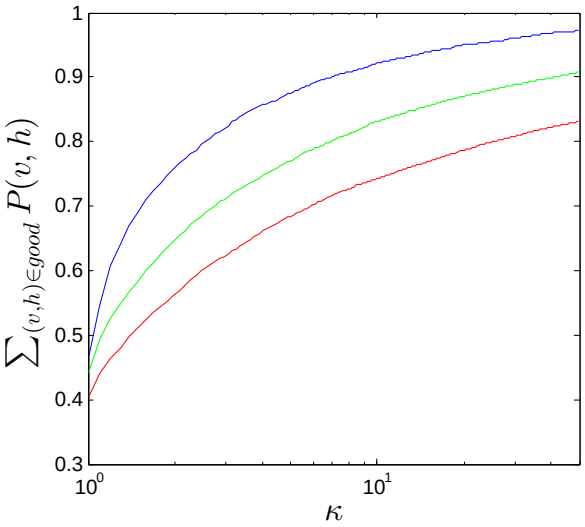
Figure 12显示，通过训练一个完整的玻尔兹曼机发现的 ML 目标函数，随着可见单元数量的增加，慢慢地提高了学习到的最优的质量。虽然这种增加在所考察的 $n_h$ 范围内是缓和，但是必须注意到，在具有 6 个可见单元和 4 个隐藏单元的受限玻尔兹曼机上，通过 BFGS 优化获得的平均 ML 目标的值约为 $-2.33$。即使是带有 7 个单元和 21 条边的完整玻尔兹曼机，也比带有 10 个单元和 24 条边的受限玻尔兹曼机提供了更好的模型。尽管这个数值示例相当小，但它证明了向玻尔兹曼机引入完全连通性（即层内连接）的好处，因此表明我们的量子学习算法可能会产生比使用现有方法可以有效学习的模型更好的模型。




(a) Standard deviation = 0.25




(b) Standard deviation = 0.5




(c) Standard deviation = 1

| $\sigma(w_{i,j})$ | $n_v$ | Mean KL$(Q\|\|P)$ | Mean ln$(Z)$ |
|---|---|---|---|
| 0.25 | 8 | 0.0371 | 6.673 |
| 0.25 | 14 | 0.1102 | 12.020 |
| 0.25 | 20 | 0.2193 | 17.287 |
| 0.5 | 8 | 0.1222 | 7.125 |
| 0.5 | 14 | 0.3100 | 13.701 |
| 0.5 | 20 | 0.5528 | 20.228 |
| 1 | 8 | 0.3209 | 8.807 |
| 1 | 14 | 0.5187 | 19.200 |
| 1 | 20 | 0.7822 | 29.883 |

(d)

图 13: $\mathcal{P}(v,h) \leq 1$ 与 $\kappa$ 在 $n$ 个量子位上的合成完整玻尔兹曼机的概率分数，其中 $n = 8, 14, 20$（从上到下），权重根据高斯分布随机初始化，均值和方差为 $\sigma^2$，偏差设置类似但均值为 0 和单位方差... 数据表明对于具有大权重的玻尔兹曼机，平均场近似开始迅速失效。所有数据点都是使用 100 随机实例计算的。

这种方法的一个重要限制是，对于完整图上的 Ising 模型，平均场近似往往比对于分层网络 [39] 要差得多。这意味着对于这些系统，所需的 $\kappa$ 值也可能更大。虽然 [39] 中的结果在小边权的情况下显示了可接受的性能，但还需要进一步的工作来研究量子算法的模型质量和训练时间之间的权衡。

剩下的问题是，成功概率如何作为玻尔兹曼机中标准偏差和单元数的函数进行缩放？我们在Figure 13中对此进行了检验，并在Figure 13发现了与Figure 6中所见的结果在质量上相似的结果。最显著的区别是，我们考虑了更广泛的可见单位范围和更大的标准差，以说明在平均场近似不再适用的情况下，算法如何会失败。这种行为在Figure 13(c) 中最为显著，其中需要 $\kappa > 50$ 的值来进行精确的状态制备。相比之下，我们在 (a) 中看到，更小的权重往往会导致状态准备算法更有效，且 $\kappa < 20$ 足以对隐藏单位为 20 的网络进行完美的状态制备。

尽管期望值似乎表明，成功概率作为 $\sigma$ 的函数系统地缩小，但这并不一定是正确的。在我们的抽样中，我们还发现有证据表明，对于状态制备，有简单的例子，也有困难的例子。这一点可以在Figure 14中看到，我们

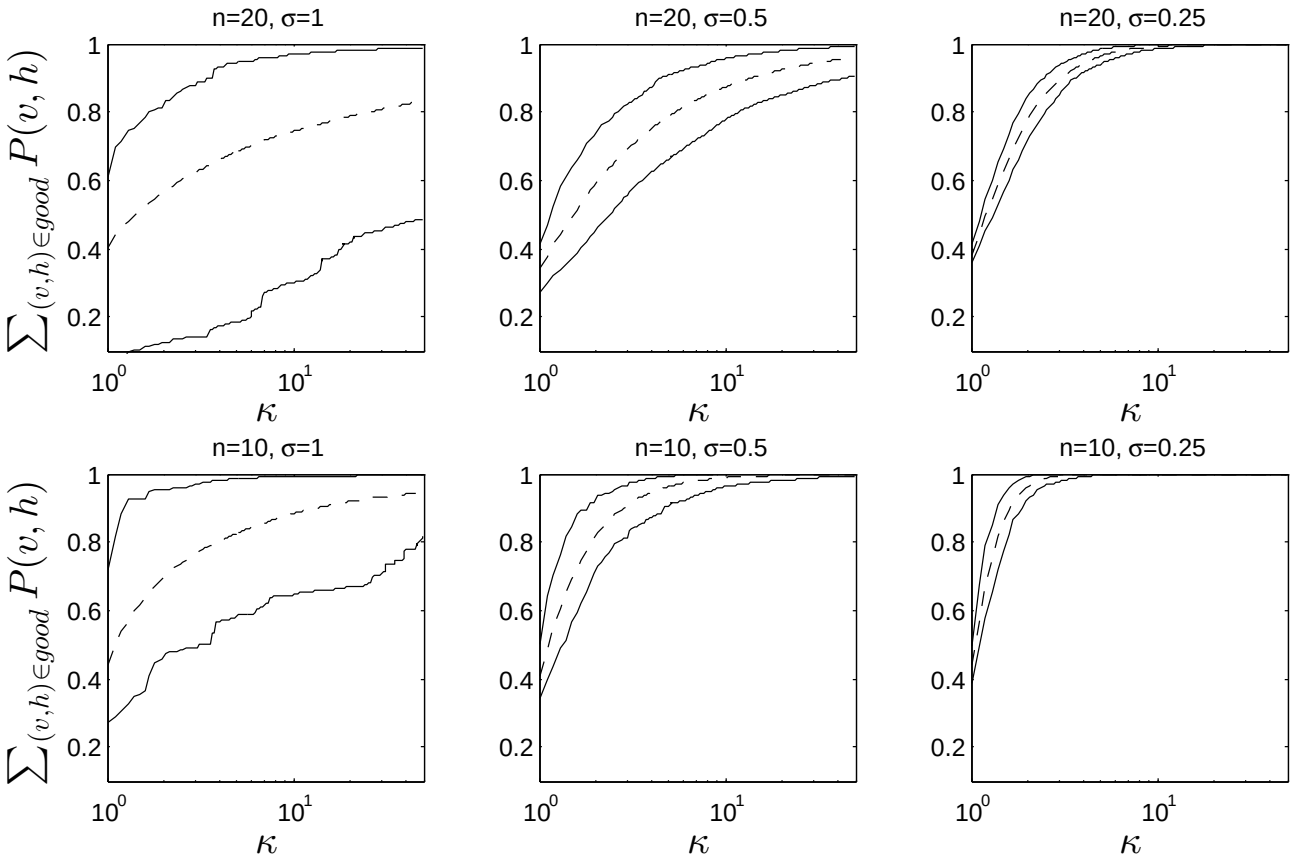



图 14: 对于不受限制的玻尔兹曼机，用固定的 $\kappa$ 作为 $\kappa$ 的函数不能正确准备的概率分布的分数的期望值（虚线）和 95% 置信区间。数据表明，$\sigma \geq 1$ 时，分布不是强烈集中在平均值附近。

绘制了一个 95% 置信区间，并看到，有些令人惊讶的是，许多模型符合 $\sigma = 1$ 数据的平均场分布。事实上，对于小 $\kappa$，$\sigma = 1$ 的 $95^{\text{th}}$ 百分位数实际上提供了比 $\sigma = 0.5$ 的相应百分位数更准确的吉布斯状态近似值。相反，$\sigma = 1$ 数据的 $5^{\text{th}}$ 百分位的保真度很差，并且似乎和考虑的其余数据的 $\kappa$ 的定性缩放方式不同。

这表明，虽然这里的情况与受限玻尔兹曼机在性质上类似，但完整的玻尔兹曼机需要更大的 $\kappa$ 值才能达到受限玻尔兹曼机所能达到的同样保真度。不幸的是，小型数值研究不足以得出结论性的结论，即通常在训练期间出现的模型是否符合这些简单案例或困难案例。

对于某函数 $f(n,\sigma)$，Figure 14中 $\sum_{(v,h)\in\text{good}} P(v,h)$ 的均值可以缩放为 $1-\kappa^{f(n,\sigma)}$。我们通过拟合 $n = 6, 8, \ldots, 20$ 和 $\sigma = 0.25, 0.5, 0.75, 1$ 的数据发现 $\sum_{(v,h)\in\text{good}} P(v,h) - 1 \in \kappa^{\Theta(-1/\sigma^{1.5}n)}$。因此，如果寻求导致最终吉布斯状态中的误差为 $\delta$ 的 $\kappa$ 值，则只要采用 $\kappa \in \delta^{-\Theta(\sigma^{1.5}n)}$。因此，这些小型数值实验表明，如果 $\sigma^2 \in o(n^{-4/3})$，完整玻尔兹曼机的状态制备可能是高效和准确的，否则它可能效率低下或近似。

### 附录 F: 使用子采样 MNIST 数据进行训练

对之前数值结果的一个重要批评是，它们只检查合成数据集。为了提供一些关于自然数据集是否与合成示例在质量上不同的见解，我们现在考虑从 MNIST 手写数字数据库中提取的训练示例。MNIST 数字是 $16 \times 16$ 灰度图像，因此我们不能直接计算 $O_{\text{ML}}$，因为计算 $P(v,h)$ 需要计算 $e^{-E(v,h)}$，至少需要 $2^{256}$ 个不同的配置。相反，我们关注一个更简单的问题，它由原始图像的 $3 \times 3$ 个粗粒度版本组成。由于这些示例的分辨率足够低，以

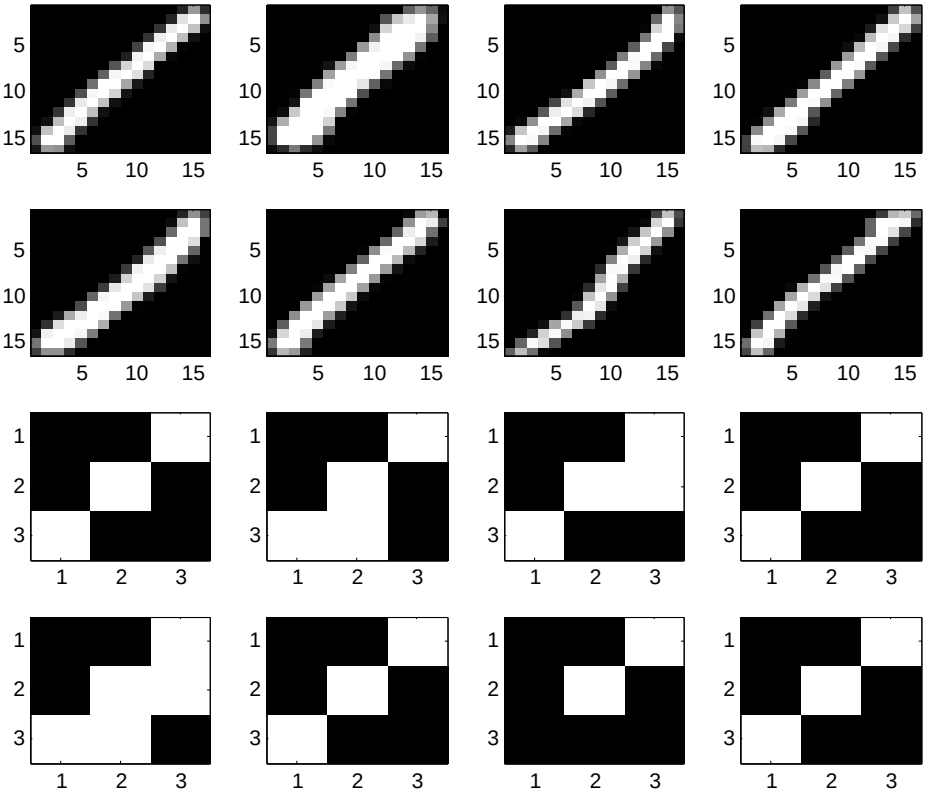


图 15: 原始（顶部）和子采样（底部）MNIST 个位数字的示例。

至于数字可能非常难以区分，我们将重点放在被分配为“1”标签的训练示例上，以避免与在 9 像素网格上可能出现相似的其他数字混淆。通过将图像分成三份，计算该三分之一图像内所有像素的期望值，并将 $3 \times 3$ 图像中相应的像素设置为该值，就可以找到结果的 400 个训练样例。然后我们将像素四舍五入，取像素的平均值作为阈值，将像素四舍五入为二值。这个子采样过程的结果如Figure 15所示。

Figure 16比较了通过 CD–1 和 $O_{\mathrm{ML}}$ 上发现的最优质量和和 CD–ML 实验的 OML 梯度上升作为隐藏单元数量的函数（$n_h \in \{4, 6, 8, 10\}$）。通过对训练过程使用 1000 随机重启找到期望值。我们观察到，在这些实验中发现的最佳位置之间的相对差异变化了几个百分点，而 $O_{\mathrm{ML}}$ 的差异变化了半个百分点。这些差异与观察到的训练模型的相对差异和主体中使用的综合训练集的最终最优质量的差异相当。

我们看到有证据表明，对比发散训练和 ML 训练之间的差异随着图中隐藏单元的数量近似线性增长。我们不能自信地说这构成了渐近状态中的线性缩放，因为差异随着 $n_h$ 适度增长，并且不能排除多项式或指数缩放。

接下来，我们检查对作为 $\kappa$ 和 $n_h$ 的函数准备的吉布斯状态质量的依赖性，用于使用 ML 优化训练的模型。我们选择这些模型是因为它们是在训练过程结束时推断出的模型的典型示例，而为随机模型找到的 $\kappa$ 的缩放代表了在训练开始时出现的模型。我们在Figure 17中获得的结果显示与合成示例观察到的行为在性质上相似。我们看到，在所有情况下，平均场拟设最初在预测配置概率方面都做得非常好：它低估了大约 $10 - 15\%$ 的概率质量。此外，适度的对冲会显着提高系统准确准备吉布斯状态的能力，在所考虑的绝大多数情况下，所需的 $\kappa$ 值小于 1000。对于所考虑的情况，$\alpha = 0.5$ 的值不太可能是最优的。在实践中，扫描 $\alpha$ 的值以找到合适的值可能比选择Figure 17中给出的 $\alpha$ 的三个值中的任何一个更可取。

Figure 17中的数据表明，随着我们向这些图形网络添加更多节点，平均场近似值保持更加稳定。特别是，不良配置的中值概率质量在 $\kappa = 2000$ 时考虑的所有数据几乎是一个常数。相比之下，我们看到 $\kappa = 1$ 时中位数略

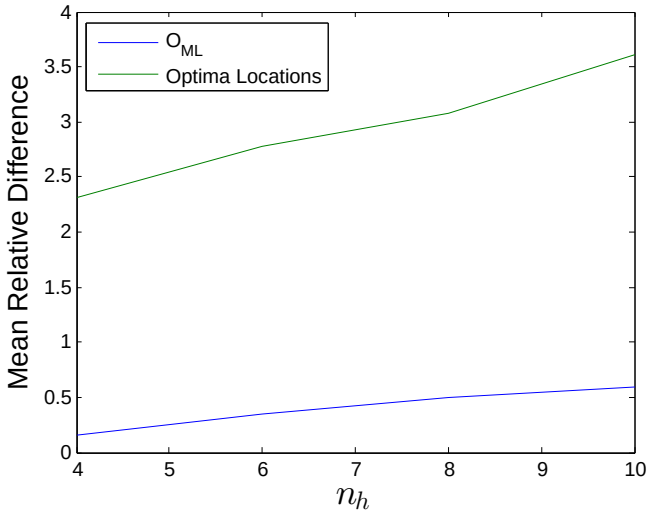



图 16: 在使用 MNIST 示例训练的受限玻尔兹曼机上进行 CDML 实验的 ML 最优值与 $O_{\mathrm{ML}}$ 中相应相对差异的平均相对距离（以百分比差异衡量）。在所有这些实验中都使用 $\lambda = 0.01$ 的正则化常数。

有变化的证据。即使对于这些小示例，$\kappa$ 值的缩放也不清楚，其中系统从不完美的状态准备过渡到 $\alpha = 0.5$ 的精确状态准备；数据与单位数量的幂律比例和指数比例一致。可能需要更大的数值实验来区分这两种可能性，但在任何一种情况下，$\alpha = 0.5$ 的缩放在性质上都不同于平均场初始状态（$\alpha = 1$）的缩放。最重要的是，结果表明，当我们从合成训练数据过渡到二次采样 MNIST 训练数据时，没有任何质的变化。

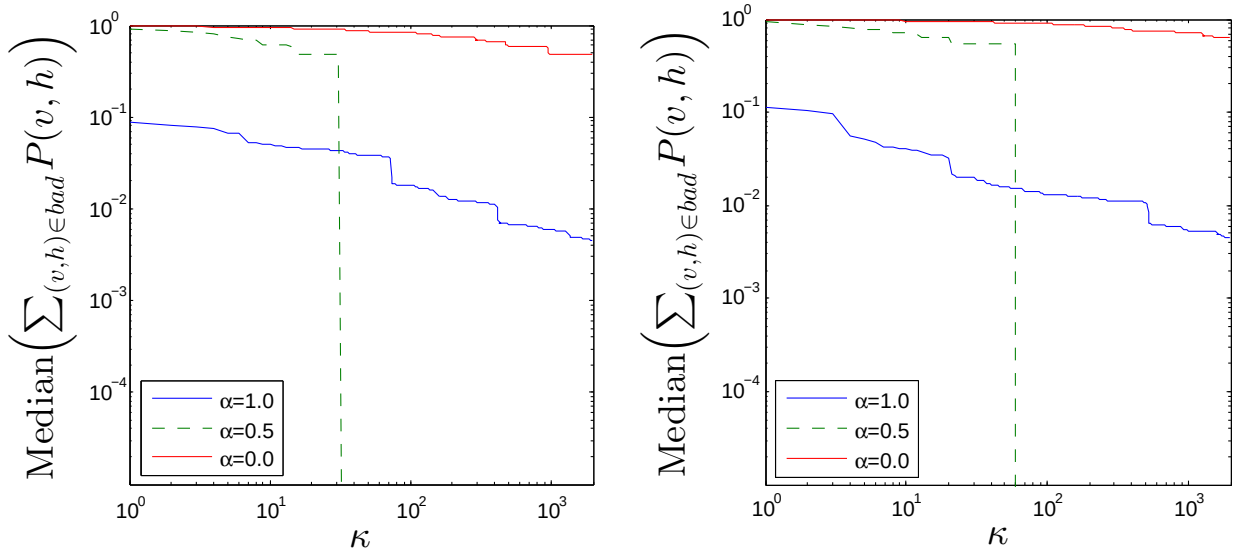
所考虑的真实示例和合成示例都受到以下事实的影响：由于我们的训练数据中存在很强的模式，因此模型中会出现非常强的相关性。这种相关性可能对平均场近似是有害的，因此结构化的平均场近似可能会在可以使用经典计算机模拟的小示例中产生更好的保真度。如何为系统配置的真实可能性选择初始先验分布的问题仍然是该领域的一个重要问题，并提供了一种重要的方法，可以优化我们的量子深度学习算法以提高训练速度和结果模型的质量。

### 附录 G: 平均场理论回顾

平均场近似是一种变分方法，它找到一个不相关的分布 $Q(v,h)$，它与吉布斯分布给出的联合概率分布 $P(v,h)$ 具有最小的 KL–散度。使用 $Q$ 而不是 $P$ 的主要好处是 $\langle v_i h_j \rangle_{\mathrm{model}}$ 和 $\log(Z)$ 可以使用均值场近似有效地估计 [18]。第二个好处是可以使用单量子位旋转有效地准备平均场状态。更具体而言，平均场近似是一种分布，即

$$Q(v,h) = \left( \prod_i \mu_i^{v_i} (1-\mu_i)^{1-v_i} \right) \left( \prod_j \nu_j^{h_j} (1-\nu_j)^{1-h_j} \right), \tag{G1}$$

其中选择 $\mu_i$ 和 $\nu_j$ 以最小化 $\mathrm{KL}(Q||P)$。参数 $\mu_i$ 和 $\nu_j$ 称为平均场参数。




(a) $n_h = 4$ (b) $n_h = 6$

(c) $n_h = 8$ (c) $n_h = 10$

图 17: 在具有不同数量的隐藏单元和不同对冲参数的截断 MNIST 数据上训练的受限玻尔兹曼机的 $\mathcal{P}(v,h) \geq 1$ 与 $\kappa$ 的概率分数。

使用伯努利分布的性质，很容易看出

$$\begin{aligned}
\mathrm{KL}(Q||P) &= \sum_{v,h} -Q(v,h)\ln(P(v,h)) + Q(v,h)\ln(Q(v,h)), \\
&= \sum_{v,h} Q(v,h)\left(\sum_i v_i b_i + \sum_j h_j d_j + \sum_{i,j} w_{i,j} v_i h_j + \ln Z\right) + Q(v,h)\ln(Q(v,h)) \\
&= \sum_i \mu_i b_i + \sum_j \nu_j d_j + \sum_{i,j} w_{i,j}\mu_i\nu_j + \ln(Z) \\
&\qquad + \sum_i \mu_i \ln(\mu_i) + (1-\mu_i)\ln(1-\mu_i) + \sum_j \nu_j \ln(\nu_j) + (1-\nu_j)\ln(1-\nu_j). \qquad \text{(G2)}
\end{aligned}$$

$\mu_i$ 和 $\nu_i$ 的最佳值可以通过将此方程对 $\mu_i$ 和 $\nu_i$ 微分并将结果设置为零来找到。解决办法是

$$\mu_i = \sigma(-b_i - \sum_j w_{i,j}\nu_j)$$
$$\nu_j = \sigma(-d_j - \sum_i w_{i,j}\mu_i), \tag{G3}$$

其中 $\sigma(x) = 1/(1+\exp(-x))$ 是 sigmoid 函数。这些方程可以通过定点迭代隐式求解，这涉及任意初始化 $\mu_i$ 和 $\nu_j$ 并迭代这些方程直到达到收敛。如果映射的雅可比行列式的范数以 1 为界，则可以保证收敛。通过不动点迭代求解平均场方程类似于吉布斯采样，不同的是这里有且只有多项式的配置需要采样，所以整个过程是有效的。将此过程推广到深度网络很简单，在 [3] 中进行了讨论。

可以使用完全相同的方法计算 $P(v,h) = \delta_{v,x}\exp^{-E(x,h)}/Z_x$ 等分布的平均场近似值。唯一的区别是在这种情况下，平均场近似中的可见单元仅接管隐藏单元。需要此类近似值来计算我们算法中估计 $O_{\mathrm{ML}}$ 的导数所需的数据期望值。

从上面的论证中也很容易看出，在所有乘积分布中，$Q$ 是对(A3)中的对数配分函数的近似误差最小的分布，这是因为

$$\log(Z_Q) = \log(Z) - \mathrm{KL}(Q||P), \tag{G4}$$

以及通过求解(G3)找到的平均场参数最小化所有产品分布中的 KL–散度。同样有趣的是，所有这些近似值都是对数分区函数的下界，因为 $\mathrm{KL}(Q||P) \geq 0$。

在实验上，均值场近似可以在小于 1% 误差 [42] 内估计对数配分函数，具体取决于权重分布和所用图形的几何形状。我们在Section E 3中进一步表明，配分函数的平均场近似对于小型受限玻尔兹曼机来说足够准确。如果需要，可以使用结构化平均场近似方法 [20]、TAP[40] 或 AIS[8, 41] 来减少此类误差，尽管经典计算成本更高。

这些结果还表明，当在模型中关联强度消失的极限情况下，我们所使用的状态准备方法的成功概率接近 1。

**Corollary 3.** *当 $\max_{i,j}|w_{ij}| \to 0$ 时Lemma 2中的成功概率接近 1。*

*证明.* 能量 $w$ 是一个连续函数，因此 $e^{-E(v,h)}/Z$ 也是 $w$ 的一个连续函数。因此，$\lim_{w\to 0} P(v,h) = e^{-\sum_i b_i v_i - \sum_j b_j h_j}/\sum_{v,h} e^{-\sum_i b_i v_i - \sum_j b_j h_j}$。在这种情况下，存在 $\tilde{\mu}_i$ 和 $\tilde{\nu}_j$ 使 $P(v,h)$ 因子化为

$$\lim_{w\to 0} P(v,h) = \left(\prod_i \tilde{\mu}_i^{v_i}(1-\tilde{\mu}_i)^{1-v_i}\right)\left(\prod_j \tilde{\nu}_j^{h_j}(1-\tilde{\nu}_j)^{1-h_j}\right). \tag{G5}$$

因此，根据(G1)，存在一个平均场解使得 $\mathrm{KL}(Q||\lim_{w\to 0} P) = 0$。由于(G3)的解在 $w_{i,j} = 0$ 时是唯一的，因此找到的平均场解必须是全局最优解，因此 $\mathrm{KL}(Q||P)$ 在 $\max_{i,j}|w_{i,j}| \to 0$ 时接近 0。因此(G4)意味着 $Z_Q \to Z_{x,Q}$ 在极限内。因此当 $\max_{i,j}|w_{i,j}| \to 0$ 时，我们取 $\kappa = 1$ 以及 $Z/Z_Q = 1$。因此如果 $\kappa$ 选取了最佳值，则成功概率接近 1。这一结论同样适用于 $Z_x/Z_{x,Q}$。 □

## 附录 H: 对比散度训练回顾

对比散度背后的思想是直截了当的。论文主体中所给出的平均对数似然梯度表达式中的模型平均值可以通过吉布斯分布 $P$ 的抽样来计算。这个过程在经典上不易处理，所以对比散度从通过应用有限轮数的吉布斯抽样所得的吉布斯分布的近似值中取样。理想情况下，从分布中抽取的结果样本是从接近真实吉布斯分布 $P$ 的分布中抽取的。

吉布斯采样的过程如下。首先，可见单元被设置为训练向量。然后将隐藏单元以 $P(h_j = 1|v) = \sigma(-d_j - \sum_i w_{i,j}v_i)$ 的概率设置为 1。一旦隐藏单元被设置，可见单元以 $P(v_i = 1|h) = \sigma(-b_i - \sum_j w_{i,j}h_j)$ 的概率重置

为 1。然后可以使用新生成的训练向量取代原来的 $v$ 以重复这一过程。随着吉布斯抽取轮数的增加，结果样本上的分布逐渐接近于真实的吉布斯分布。

CD−1 作为最简单的对比散度算法，只用一轮吉布斯抽样来重置可见单元。然后以 $P(h_j = 1|v) = \sigma(-d_j - \sum_i w_{i,j}v_i)$ 的概率将每个隐藏单元设置为 1。存储这些概率，并对每个训练向量重复吉布斯采样和概率计算的过程。然后对于每个 $w_{i,j}$ 而言，将模型平均所需的必要概率设定为在先前样本中计算的所有概率 $\Pr(v_i = 1, h_j = 1)$ 的平均值。通过使用更多步的吉布斯抽样可以找到更接近吉布斯分布的结果 [7, 11, 38]。例如，CD-10 采用 10 步吉布斯抽样而不是 1 步，结果倾向于给出更好地近似于目标函数的真实梯度。

对比散度这一名字的由来，是因为它不试图接近 ML 目标函数的梯度，相反，CD-$n$ 近似优化了 0 轮和 $n$ 轮吉布斯采样后的平均对数似然之间的差异：$\mathrm{KL}(p_0||p_\infty) - \mathrm{KL}(p_n||p_\infty)$[5]。同时，当 $n \to \infty$ 时，对比散度的目标变为平均对数似然，在没有正则化的情况下则是 $O_{\mathrm{ML}}$。这意味着渐进的 CD-$n$ 近似于正确的导数。虽然对比散度近似于对比散度目标函数的梯度，但算法产生的梯度并不精确是任何目标函数的梯度 [9]。因此，对比散度优化接近 $O_{\mathrm{ML}}$ 的目标函数的类比是不准确的。

尽管对比散度训练是有效的，但它仍存在一些缺点和不足。对比散度最主要的缺点是它不允许隐藏单元和可见单元之间的交互。这就限制了可允许的一类图形模型。除此之外，训练的过程可能持续数小时甚至数天 [7]。最后，这一方法不能直接适用于深度网络的训练。为例训练深度受限的玻尔兹曼机，通常采用分层训练，这打破了网络的无向结构，并有可能导致次优模型。并行性可以加速使用对比散度来训练深度受限玻尔兹曼机的训练过程，但也只能在有限的范围内，这是因为（现在有向）图上的更新具有顺序性。我们的工作表明，量子计算可以规避这些限制。

## 附录 I: 量子计算回顾

在量子信息处理（Quantum Information Processing，QIP）中信息被存储在一个量子位上，或称位量子比特，它类似于一个经典的比特。经典比特的状态值 $s \in \{0, 1\}$，而量子比特的状态 $|\psi\rangle$ 实际上是状态的线性叠加：

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{I1}$$

其中，$\{0, 1\}$ 基态向量分别用狄拉克符号（ket 向量）表示为 $|0\rangle = \begin{bmatrix}1 & 0\end{bmatrix}^T$ 以及 $|1\rangle = \begin{bmatrix}0 & 1\end{bmatrix}^T$。振幅 $\alpha$ 和 $\beta$ 是满足归一化条件的复数：$|\alpha|^2 + |\beta|^2 = 1$。在测量量子态 $|\psi\rangle$ 的时候，观察到 $|0\rangle$ 态和 $|1\rangle$ 态的概率分别为 $|\alpha|^2$ 和 $|\beta|^2$。注意，一个 $n$ 量子比特的量子态是 $2^n \times 1$ 维的状态向量，其中每个条目代表相应的基础状态的振幅。因此 $n$ 量子比特在一个 $2^n$ 维的希尔伯特空间中存在，我们可以将其表示为 $2^n$ 状态的叠加：

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i\,|i\rangle\,, \tag{I2}$$

其中 $\alpha_i$ 是满足 $\sum_i |\alpha_i|^2 = 1$ 的复振幅，并且 $i$ 的整数 $i$ 的二进制表示。注意，例如，态 $|0000\rangle$ 等价于写成四个态的张量积的形式：$|0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle = |0\rangle^{\otimes 4} = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$。只用线性数量的量子比特来表示指数级的状态叠加的能力是量子算法的基本要素之一，即固有的大规模并行性。

一个量子计算通过一个量子态的幺正演化来完成；反过来说，量子运算必然是可逆的。我们将量子幺正运算称为量子门。要注意是量子测量是不可逆的，测量使量子态坍缩到观测值，从而抹去了振幅 $\alpha$ 和 $\beta$ 的信息。

一个 $n$ 量子位的量子们是一个作用于 $n$ 量子位的量子态的 $2^n \times 2^n$ 的幺正矩阵。例如，Hadamard 门完成 $|0\rangle \to \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ 以及 $|1\rangle \to \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ 的映射。$X$ 门类似于一个经典的 NOT 门，完成 $|0\rangle \to |1\rangle$ 和 $|1\rangle \to |0\rangle$ 的映射。单位门用 $I$ 表示。两量子比特受控非门 $CX$，完成 $|x, y\rangle \to |x, x \oplus y\rangle$ 的映射。为了方便起

见，再引入一个门 $T$ 它被称为 $\pi/8$ 门使上述量子门组完备。对应的幺正矩阵如下：

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, X = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}, I = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}, CX = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}, T = \begin{bmatrix}1 & 0\\ 0 & e^{i\pi/4}\end{bmatrix}. \tag{I3}$$

单量子位旋转使量子计算中的一个重要操作。单量子位旋转，$R_y(2\theta)$，在 SO(3) 和 SU(2) 的同构下，对应于状态向量围绕 $y$ 轴的旋转，其中状态 $|0\rangle$ 和 $|1\rangle$ 是计算的基础状态。这一门的定义如下：

$$R_y(2\theta) = \begin{bmatrix}\cos\theta & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix}. \tag{I4}$$

与之前的门不同的是单量子位的旋转不是离散的。然而，它们可以通过一系列基本的（离散的）量子操作在任意小的误差内近似得到 [33–35]。


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

[2] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

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

[4] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 253–256. IEEE, 2010.

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

[6] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[7] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008.

[8] Ruslan Salakhutdinov and Geoffrey E Hinton. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009.

[9] Ilya Sutskever and Tijmen Tieleman. On the convergence properties of contrastive divergence. In *International Conference on Artificial Intelligence and Statistics*, pages 789–795, 2010.

[10] Tijmen Tieleman and Geoffrey Hinton. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1033–1040. ACM, 2009.

[11] Yoshua Bengio and Olivier Delalleau. Justifying and generalizing contrastive divergence. *Neural Computation*, 21(6):1601–1621, 2009.

[12] Asja Fischer and Christian Igel. Bounding the bias of contrastive divergence learning. *Neural computation*, 23(3):664–673, 2011.

[13] Daniel A Lidar and Ofer Biham. Simulating ising spin glasses on a quantum computer. *Physical Review E*, 56(3):3661, 1997.

[14] Barbara M Terhal and David P DiVincenzo. The problem of equilibration and the computation of correlation functions on a quantum computer. *arXiv preprint quant-ph/9810063*, 1998.

[15] David Poulin and Pawel Wocjan. Sampling from the thermal quantum gibbs state and evaluating partition functions with a quantum computer. *Physical review letters*, 103(22):220502, 2009.

[16] Misha Denil and Nando De Freitas. Toward the implementation of a quantum rbm. In *NIPS Deep Learning and Unsupervised Feature Learning Workshop*, 2011.

[17] Maris Ozols, Martin Roetteler, and Jérémie Roland. Quantum rejection sampling. *ACM Transactions on Computation Theory (TOCT)*, 5(3):11, 2013.

[18] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[19] Max Welling and Geoffrey E Hinton. A new learning algorithm for mean field boltzmann machines. In *Artificial Neural Networks—ICANN 2002*, pages 351–357. Springer, 2002.

[20] Eric P Xing, Michael I Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 583–591. Morgan Kaufmann Publishers Inc., 2002.

[21] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *arXiv preprint quant-ph/0005055*, 2000.

[22] Esma Aïmeur, Gilles Brassard, and Sébastien Gambs. Machine learning in a quantum world. In *Advances in Artificial Intelligence*, pages 431–442. Springer, 2006.

[23] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. *arXiv preprint arXiv:1307.0411*, 2013.

[24] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big feature and big data classification. *arXiv preprint arXiv:1307.0471*, 2013.

[25] Nathan Wiebe, Ashish Kapoor, and Krysta Svore. Quantum nearest-neighbor algorithms for machine learning. *QIC*, 15:318–358, 2015.

[26] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[27] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Physical review letters*, 100(16):160501, 2008.

[28] Nathan Wiebe and Martin Roetteler. Quantum arithmetic and numerical analysis using repeat-until-success circuits. *arXiv preprint arXiv:1406.2040*, 2014.

[29] Esma Aïmeur, Gilles Brassard, and Sébastien Gambs. Quantum clustering algorithms. In *Proceedings of the 24th international conference on machine learning*, pages 1–8. ACM, 2007.

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

[31] Yann LeCun and Corinna Cortes. The mnist database of handwritten digits, 1998.

[32] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.

[33] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. Fast and efficient exact synthesis of single-qubit unitaries generated by clifford and t gates. *Quantum Information & Computation*, 13(7-8):607–630, 2013.

[34] Neil J Ross and Peter Selinger. Optimal ancilla-free clifford+ t approximation of z-rotations. *arXiv preprint arXiv:1403.2975*, 2014.

[35] Alex Bocharov, Martin Roetteler, and Krysta M Svore. Efficient synthesis of universal repeat-until-success circuits. *arXiv preprint arXiv:1404.5320*, 2014.

[36] Andrew M Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *arXiv preprint arXiv:1202.5822*, 2012.

[37] Pinar Donmez, Krysta M Svore, and Christopher JC Burges. On the local optimality of lambdarank. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 460–467. ACM, 2009.

[38] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Proceedings of the tenth international workshop on artificial intelligence and statistics*, pages 33–40. Citeseer, 2005.

[39] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. A new class of upper bounds on the log partition function.

*Information Theory, IEEE Transactions on*, 51(7):2313–2335, 2005.

[40] Manfred Opper and Ole Winther. Tractable approximations for probabilistic models: The adaptive thouless-anderson-palmer mean field approach. *Physical Review Letters*, 86(17):3695, 2001.

[41] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879. ACM, 2008.

[42] Martijn AR Leisink and Hilbert J Kappen. A tighter bound for graphical models. In *NIPS*, volume 13, pages 266–272, 2000.