用于二值神经网络训练与正则化的二次无约束二进制优化

Jonas Christoffer Villumsen[a]，Yusuke Sugita[b]

[a]日立欧洲有限公司研发组，丹麦孔斯灵比[b]日立有限公司研发组，日本东京国分寺

arXiv:2601.00449v1 [math.OC] 1 Jan 2026

摘要

人工智能（AI）和深度学习的发展引发了人们对能源消耗不断增加的担忧，与此同时，将人工智能部署于移动设备和边缘设备的需求也在增长。二值神经网络（BNN）作为适用于资源受限环境、能效高且内存占用低的模型，近来备受关注；然而，由于其具有离散特性，精确训练二值神经网络在计算上颇具挑战。近期提出的一种基于二次无约束二进制优化（QUBO）的二值神经网络训练框架，以及用于求解 QUBO 问题的伊辛机设计进展，表明高效优化离散神经网络具有可行的路径。在本研究中，我们扩展了现有的用于训练二值神经网络的 QUBO 模型，使其能够适应任意网络拓扑，并提出两种新颖的正则化方法。第一种方法通过最大化神经元裕度，使训练过程倾向于产生预激活幅度较大的参数配置。第二种方法采用一种受丢弃法启发的迭代方案：训练规模缩小的子网络，并利用这些子网络调整网络参数上的线性惩罚项。我们将所提出的 QUBO 表述应用于一个小型二值图像分类问题，并在基于GPU的伊辛机上开展计算实验。数值结果表明，所提出的正则化项会改变训练行为，并提高模型对训练集之外数据的分类准确率。

关键词： 二次二进制规划，离散优化，模拟退火，伊辛模型，神经网络

1. 引言

近年来，深度学习的进展推动了人工智能（AI）的快速发展，使其能够广泛应用于社会关键领域，例如

电子邮箱地址： jonas-christoffer.villumsen@hitachi-eu.com (Jonas Christoffer Villumsen), yusuke.sugita.gt@hitachi.com (YusukeSugita)

自动驾驶、预测性维护和对话智能体。大语言模型和多模态生成系统的出现极大拓展了人工智能应用的范围，但这一进展也导致训练和推理期间的能源消耗大幅增加 [1]。与此同时，将人工智能能力整合到可穿戴健康设备和智能传感器等边缘环境中的需求日益增长，而这些设备的计算和能源资源有限 [2]。这一双重趋势凸显了对更高效人工智能架构的需求。

为减轻深度神经网络的计算与能耗负担，量化技术得到了广泛应用。量化将网络参数从高精度浮点表示（如 32 位）降低至较低精度（如 8 位或 4 位），从而减少内存和计算需求，并使得在资源受限设备上进行推理成为可能。在最极端的情况下，网络参数会被压缩为单个位，从而形成二值神经网络（BNN），旨在实现其在低功耗边缘设备上的高效部署 [3, 4, 5, 6]。

文献中的大多数二值神经网络实现都采用反向传播算法进行训练，并做出一些调整以适应参数二值化。在前向传播阶段，输入激活值和网络参数会先进行二值化，然后再计算输出激活值。这样便可使用同或和总体计数运算，高效计算预激活值，替代开销较大的矩阵乘法 [7]。大多数已提出的二值神经网络使用符号函数作为二值化函数；然而，该函数不可微，且几乎处处导数为 0。因此，使用反向传播和梯度下降训练二值神经网络时，通常会应用直通估计器，通过离散激活函数近似梯度 [8]。尽管这种方法使得非可微网络能够实际训练，但它会导致前向与反向传播之间出现梯度失配，进而可能使参数更新产生偏差，并降低泛化性能。另一种方法是采用基于模型的方法，例如混合整数和约束规划的建模形式 [9], ，无需依赖梯度信息即可直接优化参数；但由于将二元变量视为内生变量，这些方法容易面临潜在解数量的组合爆炸。

二值神经网络与早期神经计算基于能量的模型（如伊辛模型 [10], 、霍普菲尔德网络 [11], 和玻尔兹曼机 [12], ）共享基本原理。这些模型将信息处理描述为对表征系统状态的能量函数进行最小化。伊辛模型由一组自旋构成，每个自旋可以处于两种状态之一，通常用数值$\sigma = 1$ 或 $\sigma = -1$表示；相邻自旋之间存在由矩阵 J控制的两两相互作用；外场 h 作用于各个自旋；哈密顿量$E(\sigma) = -\sum_{i<j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i$则描述系统的能量。霍普菲尔德网络描述由任意方式连接的神经元组成的系统，其中每个神经元都有二元激活（表示其是否激活），并根据

输入的加权和是否超过阈值来异步更新。当神经元连接的权重呈对称时，可以证明该网络与伊辛模型同构，因此会向等价于伊辛模型能量函数的极小值演化 [11]。玻尔兹曼机通过引入概率性的（但仍为二值）神经元激活值，使霍普菲尔德网络得以泛化，从而跳出能量函数的局部极小值 [12]。与使用反向传播算法训练的现代深度神经网络和二值神经网络不同，基于能量的模型不要求网络拓扑必须是无环的，原则上支持任意拓扑结构。

伊辛模型的能量最小化问题与求解二次无约束二进制优化（QUBO）问题同构，后者提供了一种通用表述，涵盖了若干经典优化问题 [13]。尽管 QUBO 问题是 NP难 [14], ，但人们已开发出一系列称为伊辛机的受自然启发的计算系统，以快速且高能效地获得近似解 [15]。典型实现包括利用超导量子比特的量子退火机 [16], 、基于光学参量振荡器的光学伊辛机 [17], ，以及采用图形处理器或专用集成电路等大规模并行架构的数字退火系统 [18, 19, 20, 21]。将约束优化问题重新表述为适于在伊辛机上求解的 QUBO，这种方法被认为有望提高组合优化的效率，因此，伊辛机已应用于广泛的实际领域 [22]。在机器学习领域，应用包括玻尔兹曼机的训练 [23], 、聚类 [24], 、决策树结构优化 [25],、黑盒优化 [26, 27], 以及神经网络的量化 [28]。

研究者还提出了利用量子退火直接训练通用神经网络 [29]，尤其是二值神经网络 [30] 的 QUBO 公式。具体而言，Sasdelli 和 Chin 将分层二值神经网络的训练问题表述为二次约束二进制优化问题，通过确保神经元激活正确并最小化二进制训练损失来实现训练；他们还给出了 QUBO 松弛，并利用模拟退火和量子退火在一个小型网络上展示了这一思路 [30]。Georgiev 扩展了这一表述，提出了一种迭代算法，并在数字伊辛机上针对图像分类和情感分析的小规模实例进行了计算实验 [31]。根据所提出的模型，网络中允许的前驱神经元数量分别限制为 $2^n - 1$ 和 $2n - 1$，其中 n为正整数。鉴于现有方法对网络拓扑有所限制，或主要侧重于拟合训练数据，因此有必要进一步研究如何在基于 QUBO 的二值神经网络框架中纳入正则化并提升泛化能力。

当网络参数过度贴合训练数据时，就会出现过拟合，进而导致模型在未见数据上的性能下降。正则化技术旨在通过防止过拟合来提升模型的泛化性能，常见技术包括 L1 和 L2正则化、提前停止、间隔最大化以及 Dropout。在分类问题中，

模型的性能很大程度上取决于分离超平面的几何形状，而间隔最大化通过增大样本与决策边界之间的距离来提升稳定性。间隔最大化是支持向量机中的基本概念 [32] ，近年来也已应用于深度神经网络 [33, 34, 35]。另一方面，丢弃技术通过在训练过程中随机停用节点，提供了一种有效的正则化方式，从而避免模型过度依赖特定特征 [36, 37]。基于模型的方法也容易出现过拟合 [38], ，因此研究者提出了适用于二值神经网络的各种公式，将正则化理念纳入其中，例如间隔最大化和连接数量最小化 [9]。

本文研究基于 QUBO模型、可在伊辛机上求解的二值神经网络训练，并提出两种用于基于 QUBO 的二值神经网络正则化的新方法。本研究的贡献有四点。首先，我们为任意拓扑结构的二值神经网络训练提供了通用 QUBO模型，并证明了其有效性。与以往研究不同，我们不限制允许的前驱节点数量。其次，我们针对一个四分类图像分类问题，在多种网络架构上开展计算实验，验证了该模型的有效性。第三，我们提出了一种新的二次惩罚表达式，可最大化神经元间隔并提高训练所得网络的鲁棒性。计算结果表明，在测试数据上的准确率最高可显著提升 31%。最后，我们提出了一种类似 Dropout 的迭代算法，通过随机修改网络拓扑，在某些超参数设置下使测试准确率最高提升 9%。所有模型与算法方面的成果均通过数字伊辛机上的全面计算实验得到验证。

本文其余部分安排如下。第 2 节中，我们将模型推导并表述为一个二次约束可行性问题，并将其转化为 QUBO问题。第 3 节介绍实验设置，包括图像分类问题以及计算环境的详细信息。第4节中，我们将第 2 节开发的模型应用于不含正则化的分类问题，采用多种神经网络架构，并展示计算结果。在确立模型的有效性后，我们将在第 5 节引入一个新的二次惩罚项，以最大化神经元间隔，并展示该方法如何提高模型在训练数据之外的数据上的准确率。第6节中，我们提出一种迭代方法，通过随机修改网络拓扑结构进一步提高测试准确率。最后，第 7 节总结全文，并讨论进一步的扩展方向和潜在的未来研究方向。

2. 模型公式化

本研究关注的是这样的二值神经网络：所有网络参数和激活值均为 -1 或 1，且激活函数 f 由下式给出：

$$f(x) = \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x \leq 0 \end{cases}$$

设 $G = (\mathcal{N}, \mathcal{A})$ 是表示神经网络拓扑结构的有向图，其中每个节点 $i \in \mathcal{N}$ 表示一个激活值为 $x_i \in \{-1, 1\}$的神经元，以及固定输入 $\nu_i \in \mathbb{R}$ 或偏置 $b_i \in \{-1, 1\}$；并且每条边 $(i, j) \in \mathcal{A}$ 表示神经元 i 与 j 之间的一条连接，其权重为 $w_{ij} \in \{-1, 1\}$。此外，设 $P_j = \{i \in \mathcal{N} | (i, j) \in \mathcal{A}\}$ 为相对于 G而言节点 j的前驱节点集合，并设 $\mathcal{N}_0$ 和 $\mathcal{N}_f$ 分别为输入节点和输出节点集合。每个输入节点$i \in \mathcal{N}_0$ 都有一个固定输入 $\nu_i \in \mathbb{R}$，而所有非输入神经元 $i \in \mathcal{N} \setminus \mathcal{N}_0$ 都有一个可变偏置参数 $b_i \in \{-1, 1\}$。请参见图1，其中我们提供了单个节点和一个简单网络的示意图，以及所定义的符号。为简化表述，对于所有节点 $i \in \mathcal{N}$，我们用一个新参数 $u_i \in \{-1, 1\}$替代输入和偏置参数，使得当 $i \in \mathcal{N}_0$和 $u_i = b_i$时， $u_i = f(\nu_i)$，否则为.....。对于每个节点 $j \in \mathcal{N}$， 节点 j 的激活值定义为:

$$x_j = f\left(u_j + \sum_{i \in P_j} w_{ij} x_i\right) \tag{1}$$

请参阅表 1，其中总结了本文引入并在全文中使用的符号。请注意，我们使用术语双极和二值来描述分别可取值为 $\{-1, 1\}$ 和 $\{0, 1\}$的参数和变量。

本节余下部分将遵循与先前研究 [30, 31]类似的方法，介绍一种用于训练二值神经网络的 QUBO 模型。不过，与先前工作不同，我们不限制前驱节点的数量 $|P_j|$，并证明该模型适用于任意数量的前驱节点。请注意，我们的公式不假设采用分层架构；事实上，原则上任意两个节点之间都可以建立连接，不过本文仅展示标准分层网络上的实验结果。

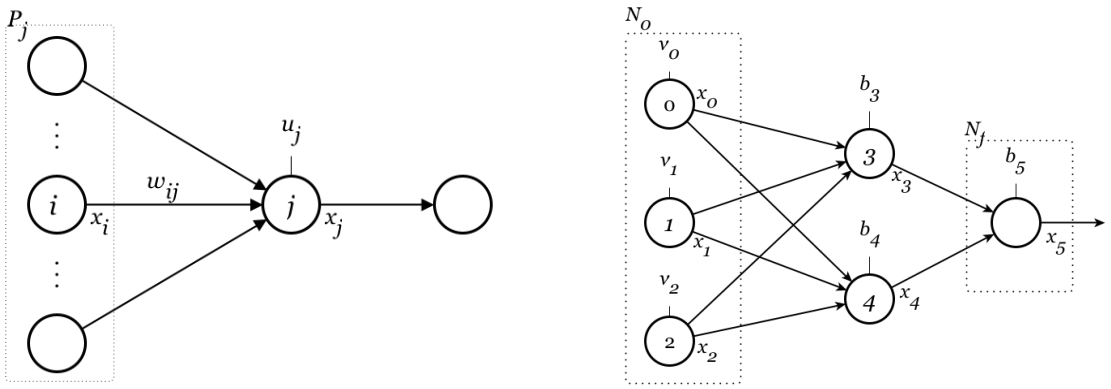
在第2.1节中，我们通过对预激活进行二进制展开，并将其最高有效位作为激活指示器，将激活函数编码为一个等式约束。在第2.2节中，我们通过针对各个数据样本复制激活约束来构建训练问题，并展示如何将其转换为 QUBO形式。

2.1. 激活函数

为了推导激活约束 (1) 的 QUBO 表述，我们采用以下定理。

表 1：符号表。

符号	领域	描述
$\mathcal{N}$		神经网络图表示中的节点集合
$\mathcal{N}_0$		神经网络图表示中的输入节点集合
$\mathcal{N}_f$		神经网络图表示中的输出节点集合
$\mathcal{A}$		神经网络图表示中的连接集合
P_j		节点 $j \in \mathcal{N}$ $P_j = \{i \in \mathcal{N} \mid (i,j) \in \mathcal{A}\}$ 的前驱节点集合
$\mathcal{K}$		训练数据点集合
x_i (x_i^k)	{-1,1}	节点 $i \in \mathcal{N}$ 的双极激活（对于训练数据点 $k \in \mathcal{K}$）
y_i (y_i^k)	{0,1}	节点 $i \in \mathcal{N}$ 的二元激活（对于训练数据点 $k \in \mathcal{K}$）
$\hat{x}_i^k$	{-1,1}	节点 $i \in \mathcal{N}_f$ 对训练数据点 $k \in \mathcal{K}$ 的固定输出值
ν_i (ν_i^k)	$\mathbb{R}$	节点 $i \in \mathcal{N}_0$ 的输入（训练数据点 $k \in \mathcal{K}$）
b_i	{-1,1}	节点 $i \in \mathcal{N} \setminus \mathcal{N}_0$ 的双极偏置
u_i (u_i^k)	{-1,1}	节点 i, $u_i = f(\nu_i)$ 的辅助参数，如果 $i \in \mathcal{N}_0$ 且 $u_i = b_i$，否则（针对 $k \in \mathcal{K}$）
d_i (d_i^k)	{0,1}	节点 i, $d_i = (u_i + 1)/2$ 的辅助参数（针对训练数据点 $k \in \mathcal{K}$）
w_{ij}	{-1,1}	连接 (i, j) 的双极权重 $\in \mathcal{A}$
v_{ij}	{0,1}	连接 (i, j) 的二进制权重 $\in \mathcal{A}$, $v_{ij} = (w_{ij} + 1)/2$
ψ_{ij}^k	{0,1}	用于线性化激活约束的辅助变量
π_i	$\mathbb{Z}$	节点 i 的预激活值；式 (3) 中双极值的和
ρ_i	$\mathbb{Z}_0$	节点 i 的预激活值；式 (4) 中二值的和
$s_l^{(i)}$ ($s_l^{(k,i)}$)	{0,1}	ρ_i 的二进制展开中的第 l 位（针对训练数据点 $k \in \mathcal{K}$）
n_j	$\mathbb{Z}_0$	一个非负整数，用于确定展开的位长 公式 (5) 中的节点 j
κ_j	$\mathbb{Z}_0$	用于调整公式 (6) 中位展开的非负整数
χ_j (χ_j^k)	$\mathbb{Z}_0$	激活约束中使用的辅助非负整数变量 针对节点 j （训练数据点 k）



(a) (b)

图1：单个节点 j 及其前驱节点（a）和一个包含六个节点的小型分层网络（b）的示意图。

定理1。 给定非负整数 m 以及 $x_i \in \{-1, 1\}$，其中 $i = 1$，……，m，定义非负整数

$$n = \lfloor \log_2(m) \rfloor,$$
$$\kappa = 2^{n+1} - (m+1),$$

其中 $\lfloor \cdot \rfloor$表示下取整函数。

通过<旋style id='3'>(n+1)比特展开定义比特 s_0，……，$s_n \in \{0, 1\}$

$$\sum_{i=1}^{m} \frac{x_i + 1}{2} + \left\lfloor \frac{\kappa}{2} \right\rfloor = \sum_{l=0}^{n} 2^l \, s_l. \tag{2}$$

那么，双极变量 x_i 之和为正，当且仅当最高有效位 s_n 为1，即

$$\sum_{i=1}^{m} x_i \ \geq \ 1 \quad \Longleftrightarrow \quad s_n = 1.$$

证明。 为简化记号，我们引入

$$Y = \sum_{i=1}^{m} \frac{x_i + 1}{2} \ \in \{0, \ldots, m\} \quad \text{and} \quad \delta = \frac{\kappa}{2} - \left\lfloor \frac{\kappa}{2} \right\rfloor \ \in \{0, 0.5\}.$$

注意到 $0 \leq Y \leq m = 2^{n+1} - 1 - \kappa$，因此

$$0 \leq Y + \left\lfloor \frac{\kappa}{2} \right\rfloor \leq 2^{n+1} - 1 - \kappa + \left\lfloor \frac{\kappa}{2} \right\rfloor = 2^{n+1} - 1 - \left\lceil \frac{\kappa}{2} \right\rceil \leq 2^{n+1} - 1,$$

因此，公式（2）中的（n+1）位展开存在且唯一。

由于 $\sum_{i=1}^{m} x_i = 2Y - m$，我们有

$$\begin{aligned}
\sum_{i=1}^{m} x_i \geq 1 &\Longleftrightarrow \ 2Y - m - 1 \geq 0 \\
&\Longleftrightarrow \ 2Y + \kappa - 2^{n+1} \geq 0 \\
&\Longleftrightarrow \ 2\left(Y + \left\lfloor \frac{\kappa}{2} \right\rfloor + \delta - 2^n\right) \geq 0 \\
&\Longleftrightarrow \ Y + \left\lfloor \frac{\kappa}{2} \right\rfloor \geq 2^n - \delta.
\end{aligned}$$

由于 $Y + \lfloor \kappa/2 \rfloor$ 是整数且 $0 \leq \delta < 1$，最后一个不等式等价于 $Y + \lfloor \kappa/2 \rfloor = \sum_{l=0}^{n} 2^l s_l \geq 2^n$。此外，由于 $0 \leq \sum_{l=0}^{n-1} 2^l s_l \leq 2^n - 1$，这等价于 $s_n = 1$。 □

对于每个节点 $j \in \mathcal{N}$，我们将预激活值记为 π_j，并将由相应二进制项组成的对应表达式记为 ρ_j，如下所示，

$$\pi_j = u_j + \sum_{i \in P_j} w_{ij} x_i, \tag{3}$$

$$\rho_j = \frac{u_j + 1}{2} + \sum_{i \in P_j} \frac{w_{ij} x_i + 1}{2}. \tag{4}$$

推论1。 对于某个节点 $j \in \mathcal{N}$ 及其前驱节点集合 P_j，定义非负整数

$$n_j = \lfloor \log_2(|P_j| + 1) \rfloor, \tag{5}$$

$$\kappa_j = 2^{n_j+1} - |P_j| - 2, \tag{6}$$

以及比特 $s_0^{(j)}$、.....、 $s_{n_j}^{(j)}$，使得，

$$\frac{u_j + 1}{2} + \sum_{i \in P_j} \frac{w_{ij} x_i + 1}{2} = \sum_{l=0}^{n_j} 2^l s_l^{(j)} - \left\lfloor \frac{\kappa_j}{2} \right\rfloor, \tag{7}$$

那么，根据定理*1*，

$$u_j + \sum_{i \in P_j} w_{ij} x_i \geq 1 \quad \textit{if and only if} \quad s_n^{(j)} = 1.$$

对于以下形式化表达，将双极变量替换为对应的二元变量会更方便，使得对于每个节点 $j \in \mathcal{N}$, $d_j = (u_j + 1)/2$,$y_j = (x_j + 1)/2$，以及每条连接 $(i, \ j) \in \mathcal{A}$, $v_{ij} = (w_{ij} + 1)/2$。现在我们可以用二元变量重写 ρ_j，

$$\begin{aligned} \rho_j &= d_j + \sum_{i \in P_j} \frac{(2v_{ij} - 1)(2y_i - 1) + 1}{2} \\ &= d_j + \sum_{i \in P_j} (2v_{ij} y_i - v_{ij} - y_i + 1). \end{aligned} \tag{8}$$

具体而言，推论1已指出， $\pi_j \geq 1$ 当且仅当 $s_{n_j}^{(j)} = 1$。由于 $y_j = {}^{x_j+1}_{2}$ 对应于节点 j的激活，因此为保持一致性，必须满足限制条件 $y_j = s_{n_j}^{(j)}$。因此，根据方程(7)和(8)，通过引入

$$\chi_j = \begin{cases} 0 & \text{if } P_j = \emptyset, \\ \sum_{l=0}^{n_j - 1} 2^l s_l^{(j)} & \text{otherwise} \end{cases} \tag{9}$$

并将 $s_{n_j}^{(j)}$ 替换为 y_j，我们得到

$$d_j + \sum_{i \in P_j} (2v_{ij}y_i - v_{ij} - y_i + 1) = 2^{n_j} y_j + \chi_j - \left\lfloor \frac{\kappa_j}{2} \right\rfloor. \tag{10}$$

因此，若约束 (10) 得到满足， y_j 就表示节点 j的激活。请注意，约束 (10) 在 $P_j = \emptyset$的情况下也成立；此时我们有 $n_j = 0$和$\kappa_j = 0$ ，因此 $d_j = y_j$。特别地，当 $j \in \mathcal{N}_0$ 是输入节点时， $y_j = d_j = {}^{u_j+1}_{2} = {}^{f\ (\nu_j)}{}_{2}{}^{+1}$ 是固定的。

图 2 展示了所提出的激活编码的一个简单示例，其中神经元有两个前驱节点，且所有权重均为 1。表 2 列举了前驱节点激活值与神经元偏置的全部八种组合，以展示预激活、其二进制展开与最终激活之间的精确对应关系。预激活 $\pi_3 = x_1 + x_2 + b_3$ 是三个双极项之和，其对应的二进制量 $\rho_3 = y_1 + y_2 + d_3$ 则统计正贡献的数量。 $\rho_3 = 2s_1^{(3)} + s_0^{(3)}$ 的位展开中，最高有效位唯一确定该双极和是否严格为正。事实上，表 2 显示， $\pi_3 \leq 0$ 的情况对应于 $\rho_3 \in \{0, 1\}$ 和 $s_1^{(3)} = 0$， 而 $\pi_3 > 0$ 的情况对应于 $\rho_3 \in \{2, 3\}$ 和 $s_1^{(3)} = 1$。

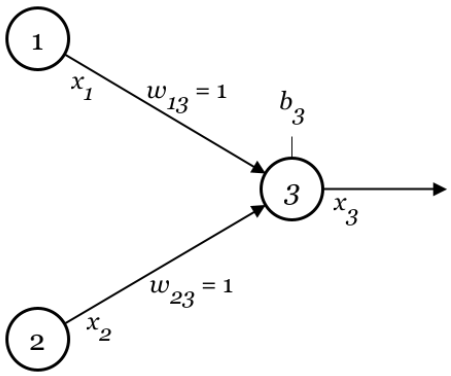


图 2：神经元示意图 $j = 3$ ，该神经元有两个前驱神经元 $P_j = \{1, 2\}$ ，并带有偏置b_3。

2.2. 训练模型

对于给定的训练集 $\mathcal{K}$（见表 1），训练模型中引入了按数据索引的符号。每个数据点 $k \in \mathcal{K}$ 由一个输入向量 ν^k 和相应的输出向量 $\hat{}x^k$组成。对于该数据集，任务是确定权重 $w \in \{-1, 1\}^{|\mathcal{A}|}$ 和偏置 $b \in \{-1, 1\}^{|\mathcal{N} \setminus \mathcal{N}_0|}$。下文中，如果对每个$k \in \mathcal{K}$ 和每个节点 $j \in \mathcal{N}$，激活约束均得到满足，我们就称赋值 $(w,\ b)$ 可行 。此外，对每个输出节点 $j \in \mathcal{N}_f$，要求激活与训练标签一致，

表 2：如图 2（$m=3$、 $n=1$、 $\kappa=0$）所示，具有两个前驱神经元的神经元的激活和预激活变量示例。表中展示了前驱神经元激活值（x_1 和 x_2）与偏置（b_3）的全部八种可能组合，以及相应的预激活值（π_3 和 ρ_3）和激活值（x_3和 y_3）。

x_1	x_2	b_3	y_1	y_2	d_3	$\pi_3 = x_1 + x_2 + b_3$	$\rho_3 = y_1 + y_2 + d_3$	$s_1^{(3)} s_0^{(3)}$	$y_3 = s_1^{(3)}$	x_3
-1	-1	-1	0	0	0	-3	0	0 0	0	-1
-1	-1	1	0	0	1	-1	1	0 1	0	-1
-1	1	-1	0	1	0	-1	1	0 1	0	-1
-1	1	1	0	1	1	1	2	1 0	1	1
1	-1	-1	1	0	0	-1	1	0 1	0	-1
1	-1	1	1	0	1	1	2	1 0	1	1
1	1	-1	1	1	0	1	2	1 0	1	1
1	1	1	1	1	1	3	3	1 1	1	1

即，对于所有 $k \in \mathcal{K}$， $x_j^k = \hat{x}_j^k$。为方便表示，令 d_j^k表示节点 j在数据点 k处的二值化输入/偏置参数；若满足 $j \in \mathcal{N}_0$，则设 $d_j^k = (f(\nu_j^k) + 1)/2$，否则设为 $d_j^k = (b_j + 1)/2$。

采用上述符号表示，激活约束变为

$$d_j^k + \sum_{i \in P_j} \big(2\, v_{ij} y_i^k - v_{ij} - y_i^k + 1\big) = 2^{n_j} y_j^k + \chi_j^k - \left\lfloor \frac{\kappa_j}{2} \right\rfloor, \quad \forall j \in \mathcal{N},\ k \in \mathcal{K}, \tag{11}$$

其中，与公式（9）类似，对于每个 $j \in \mathcal{N}$ 和 $k \in \mathcal{K}$，当 $P_j = \emptyset$时定义 $\chi_j^k = 0$，否则定义为 $\chi_j^k = \sum_{l=0}^{n_j - 1} 2^l s_l^{(k,j)}$，其中 $s_l^{(k,j)} \in \{0,1\}$ 为辅助二元变量。这里， y_j^k和 v_{ij} 分别是与激活值和权重对应的二元决策变量。通过以下方式强制实现输出一致性：

$$x_j^k = \hat{x}_j^k \quad \Longleftrightarrow \quad y_j^k = \frac{\hat{x}_j^k + 1}{2}, \quad \forall j \in \mathcal{N}_f,\ k \in \mathcal{K}. \tag{12}$$

由于约束（11）是二次的，并且为便于构造 QUBO问题，我们通过为每个连接（i, j）$\in \mathcal{A}$以及每个训练数据点 $k \in \mathcal{K}$ 引入一个辅助二元变量 ψ_{ij}^k，并令

$$\psi_{ij}^k = v_{ij} y_i^k, \quad \forall j \in \mathcal{N}, i \in P_j, k \in \mathcal{K}, \tag{13}$$

现在，我们可以将训练具有由 $G = (\mathcal{N}, \mathcal{A})$ 给定的底层拓扑结构的二值神经网络定义为非线性可行性问题，

$$\begin{aligned}&\text{Find vectors} \quad d \in \{0,1\}^{|\mathcal{N}|}, v \in \{0,1\}^{|\mathcal{A}|}, y \in \{0,1\}^{|\mathcal{N}||\mathcal{K}|}, \chi \in \mathbb{Z}_0^{|\mathcal{N}||\mathcal{K}|}, \psi \in \{0,1\}^{|\mathcal{A}||\mathcal{K}|} \\ &\text{such that} \quad d_j^k + \sum_{i \in P_j} \left(2\,\psi_{ij}^k - v_{ij} - y_i^k + 1\right) = 2^{n_j} y_j^k + \chi_j^k - \left\lfloor \frac{\kappa_j}{2} \right\rfloor, \quad \forall j \in \mathcal{N},\ k \in \mathcal{K}, \\ &\text{and} \quad \text{约束（13）} \\ &\text{得到满足}\end{aligned} \tag{14}$$

为了将可行性问题（13）–（14）转换为 QUBO 形式，我们注意到，通过最小化二次惩罚项，可以得到满足（14）的可行解，

$$H_1 = \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{N}} \left(d_j^k + \sum_{i \in P_j} (2\psi_{ij}^k - v_{ij} - y_i^k + 1) - 2^{n_j} y_j^k - \chi_j^k + \left\lfloor \frac{\kappa_j}{2} \right\rfloor \right)^2$$

并且 约束（13）可通过标准惩罚项 [39] 强制满足，给定一个 s

$$v_{ij} y_i^k - 2(v_{ij}\psi_{ij}^k + y_i^k \psi_{ij}^k) + 3\psi_{ij}^k, \quad \forall j \in \mathcal{N}, i \in P_j, k \in \mathcal{K} \tag{15}$$

表3：当且仅当 $\psi = xy$ 为真时，函数 $xy - 2(x\psi + y\psi) + 3\psi$ 恰好为 0；否则严格大于 0。

x	y	ψ	xy	$\psi = xy$	$xy - 2(x\psi + y\psi) + 3\psi$
0	0	0	0	True	0
0	0	1	0	False	3
0	1	0	0	True	0
0	1	1	0	False	1
1	0	0	0	True	0
1	0	1	0	False	1
1	1	0	1	False	1
1	1	1	1	True	0

如表3所示，当式（15）中的惩罚项达到其最小值 0 时，可以确认约束（13）成立。因此，我们可以将可行性问题（13）-（14）重新表述为：寻找使下列哈密顿量最小化的二进制向量 d、 v、 y、 χ和 ψ ，

$$H_{train} = H_1 + \alpha H_2 \tag{16}$$

其中， H_1 确保神经元正确激活，并且

$$H_2 = \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}}\sum_{i\in P_j} v_{ij}y_i^k - 2(v_{ij}\psi_{ij}^k + y_i^k\psi_{ij}^k) + 3\psi_{ij}^k$$

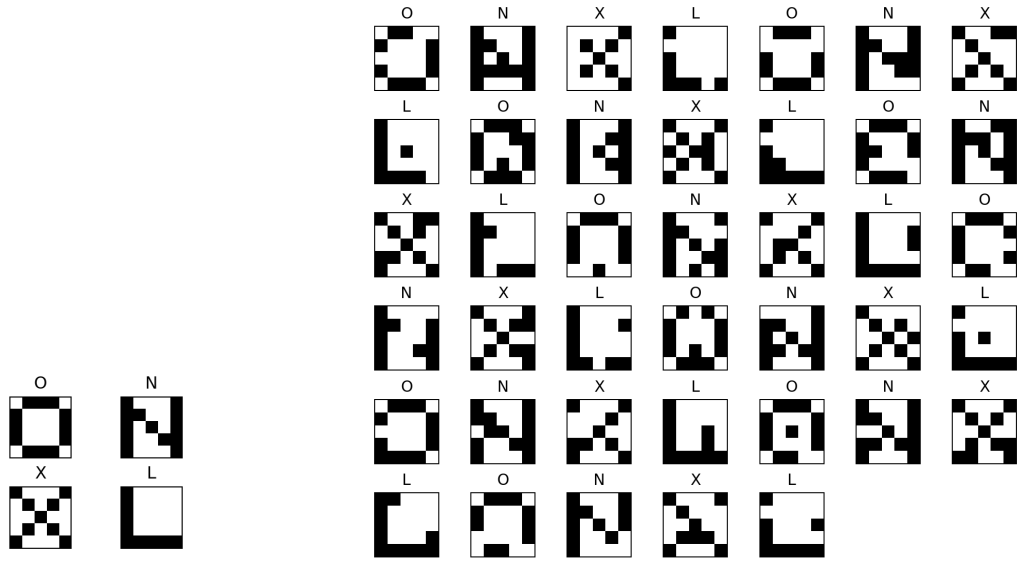
确保 $\psi_{ij}^k = v_{ij}y_i^k$和 α 是适当的缩放因子。权重和偏置的可行赋值可由满足所有激活约束（14）和辅助约束（13）的解来表征，其中 $H_{train} = H_1 = H_2 = 0$与之对应。只需固定输出变量，即可满足式（12）中的约束。

我们对该 formulation 的若干实际方面作几点说明。首先，不必像式（12）那样固定输出值，也可以添加一个损失函数，以 H_{train} 最小化输出激活值与训练输出之间的差异。然而，这会增加变量和哈密顿量中项的数量，从而可能更难找到近似最优解。其次，请注意，变量和约束的数量会随着数据集大小而变化，因为对于每个训练数据点$k \in \mathcal{K}$，我们都要为每个节点 $j \in \mathcal{N}$ 以及每个连接（i， j） $\in \mathcal{A}$引入诸如 y_j^k、 ψ_{ij}^k和 χ_j^k 这样的变量。因此，变量和约束的数量会增长为$\mathcal{O}(|\mathcal{K}| \cdot |\mathcal{N}|)$或 $\mathcal{O}(|\mathcal{K}| \cdot |\mathcal{A}|)$。

3. 实验设置

我们将第2节中开发的训练模型应用于一个四分类图像分类问题，并采用单次训练，即从每个类别的一张图像中学习网络参数。该数据集由44张二值图像组成，每张图像包含 5×5个像素，形状类似字母O、N、L和X。每张图像相应地被分为四个类别之一：O、N、L和X。数据集如图3所示，每个类别包含十一张图像，其中一张用于训练，其余十张用于测试。每张测试图像都是通过将同类别的训练图像中恰好两个像素取反得到的。由于每个图像像素仅限于两种颜色之一，因此无需在激活变换中损失信息，即可将每个像素直接映射到一个输入节点。我们指出，更高精度的输入（例如多位灰度或彩色图像）也可以轻松处理，只需为每个像素使用多个输入节点即可，例如，可以用八个输入节点表示一张8位图像中的每个像素。

在我们的实验中，使用伊辛机，通过最小化公式（16）中的哈密顿量及后续章节介绍的扩展形式来训练网络。公式（16）中的缩放因子 α 始终固定为 $\alpha = 1$。我们采用了一种针对图形处理器 [19] 等多核架构优化的、结合马尔可夫链蒙特卡罗的高度并行模拟退火方法作为伊辛机。代码以单精度运行于一台工作站上，该工作站配备 Intel Core i9-9900K CPU（3.60 GHz）和 NVIDIA GeForce RTX 2080 GPU。一次退火计算包含



(a) 训练图像。 (b) 测试图像。

图3：数据集由44张单色图像组成，每张图像都标注了相应的类别（O、N、L 或 X）。其中，4张图像（a）用于训练，40张图像（b）用于测试。

1,000 个独立副本，并进行 $N_{\mathrm{MC}} = 1{,}000$ 次蒙特卡洛步以更新变量。退火算法中的温度参数 T 在每次蒙特卡洛步后，按照几何冷却计划 $T = T_{\max}(T_{\min}/T_{\max})^{t/(N_{\mathrm{MC}}-1)}$ 进行更新，步数为 $t = 0, \ldots, N_{\mathrm{MC}} - 1$。温度参数 $T_{\min}$ 和 $T_{\max}$ 通过 Nelder-Mead 方法 [40] 进行调优。对每个二值神经网络进行训练时，都会独立重复整个退火过程。

除非另有说明，我们将在第4、5和6节报告使用此设置进行计算实验的结果。

4. 网络架构分析

本节将前文介绍的训练公式直接应用于多种网络架构。我们指出，这些结果为后文讨论的正则化方法提供了基线。共采用了十八种不同的网络架构：八种卷积网络（包含一个或两个隐藏卷积层）和十种全连接网络（包含一个全连接隐藏层）。这些网络总结于表4。图4展示了一个卷积网络（网络 2）的架构示意图。卷积网络使用尺寸为 2×2、3×3 或 4×4 的滤波器，并采用共享权重。对于所有网络，输出层均包含两个神经元，因此任意一种输出组合都恰好映射到四个标签中的一个。例如，输出 $(-1,\ -1)$ 对应标签 O，而输出 $(1,\ -1)$ 对应标签 X。针对每种网络架构进行了 200 次测试，每次测试均包含一次训练运行，并对测试数据集中的每张图像进行推理

测试数据集中的 40 张图像。下文将介绍并分析上述实验的详细情况。

表4：所研究的网络拓扑总结。每种网络架构由一个或两个隐藏层组成，每个隐藏层要么是卷积层（conv），要么是全连接层（fc）。fc(a)表示包含 a 个神经元的全连接层，而conv($a \times b$)表示包含一个 $a \times b$滤波器的卷积层。conv($a \times b \times 2$)表示在同一层中应用 2 $a \times b$ 个滤波器。使用两个隐藏层时，+运算符表示两个隐藏层之间的分隔。神经元 和连接分别表示网络中的神经元数量和连接数量。约束表示激活约束（ 14 ）和辅助替换约束（ 13 ）的总数；变量分别表示训练问题的 QUBO 表述中使用的二元变量 v_{ij}、 y_i^k和 ψ_{ij}^k 以及整数变量 χ_j^k的总数。

网络	架构	神经元	连接	变量		约束
		$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{A}\rvert$	二值	整数	
0	卷积层(2×2)	43	96	246	72	200
1	卷积层(2×2) +全连接层(4)	47	136	466	88	376
2	卷积层(3×3)	36	99	146	44	116
3	卷积层(3×3×2)	45	198	290	80	224
4	卷积层(3×3) +全连接层(4)	40	125	296	60	236
5	卷积层(4×4)	31	72	78	24	56
6	卷积层(4×4×2)	35	144	154	40	104
7	卷积层(4×4) +全连接层(4)	39	168	294	56	216
8	全连接层(1)	28	27	42	12	20
9	全连接层(2)	29	54	82	16	32
10	全连接层(3)	30	81	122	20	44
11	全连接层(4)	31	108	162	24	56
12	全连接层(5)	32	135	202	28	68
13	全连接层(6)	33	162	242	32	80
14	全连接层(7)	34	189	282	36	92
15	全连接层(8)	35	216	322	40	104
16	全连接层(9)	36	243	362	44	116
17	全连接层(10)	37	270	402	48	128

表5和图5按网络架构总结了结果。表现最佳的网络是网络 10、11 和 5，平均测试准确率分别为 57%、55% 和 55%。这三个网络的平均训练准确率均为 1.0。网络 10 和 11 都是全连接网络，其中间隐藏层分别有三个和四个神经元；网络 5 则是卷积网络，其中间层有四个神经元。表现最佳的网络规模都相对较小，神经元和连接数量适中。网络 5 是卷积网络（0–7）中规模最小的；网络 10 和 11 则是全连接网络（8–17）中，在中间层至少有三个神经元的网络里规模最小的。所有中间层至少有三个节点的全连接网络，平均训练准确率都达到或接近 1。

值得指出的是，所有哈密顿量存在最优解的网络，具体而言，网络 5 和网络 10–15，都满足所有激活约束和辅助约束，并达到 1.0 的训练准确率，这与预期一致。相比之下，表现最差的网络（例如网络 1、3、4、7、8 和 9）在训练问题的解中都有很大比例的约束未得到满足。这直接导致训练准确率低，因为许多神经元激活值未能正确应用。表现不佳的卷积网络（网络 1、3、4、7）都是规模相对较大的模型，每个模型的约束都超过 200 个。对于大型模型，性能不佳很可能源于退火算法无法收敛到满足所有约束的解。相反，全连接网络（网络 8 和 9）的规模很小，约束也很少。这些情况下，性能不佳是因为网络无法正确编码输入信号。例如，网络 8 的中间层只有一个神经元（且激活值为 1 比特），而将输入分类为四个类别之一需要两比特。因此，中间层会丢失一半所需信息，导致平均训练准确率仅为 0.5。

图 6 展示了网络 5 和网络 10 的测试准确率直方图，它们分别代表性能最佳的卷积架构和全连接架构。这些分布均为单峰分布，并集中在各自均值附近：网络 5 约为 0.55，网络 10 约为 0.57。请注意，退火计算为两个网络的所有训练运行都找到了最优解，但测试准确率的变化表明，即使在最优解中，不同解对应的网络参数配置对未见数据的泛化能力也有高下之分。这一观察促使我们在后续章节中探讨基于 QUBO 的二值神经网络的正则化方案，以鼓励获得泛化性能更好的解。

5. 最大化边际总和

在分类问题中，超平面定义类别之间的决策边界，并决定如何对输入进行分类。间隔表示任意数据点到决策边界的最小距离，因此反映了模型对输入扰动的鲁棒性。将这一概念扩展到神经网络，神经元裕度量化了翻转神经元输出所需的预激活输入变化量。由于已有研究表明，最大化神经元间隔可以提升泛化能力和鲁棒性，因此人们已着手将基于边际的损失函数纳入神经网络训练中 [33, 34, 35, 9]。基于这些观察，我们提出一种基于 QUBO 的公式化表达，以提升二值神经网络中的神经元间隔。

5.1. 初步分析与模型公式化

接下来，神经元 $j \in \mathcal{N}$ 的神经元裕度定义为其在所有训练样本 $k \in \mathcal{K}$ 上的预激活绝对值的最小值。

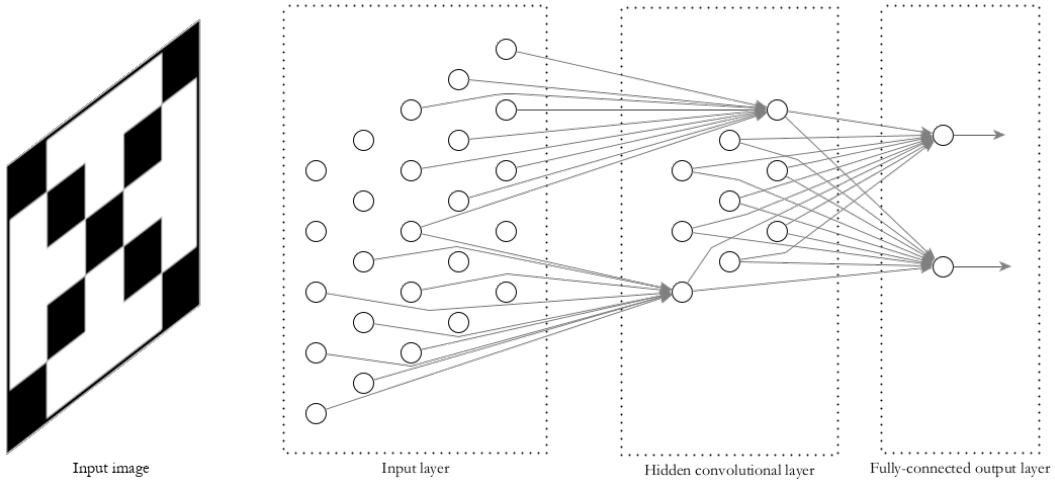


图4：表4中网络 2 的示意图，该网络是一个卷积神经网络，包含5×5 个输入节点、 3×3 个卷积滤波器，形成一个包含 3×3 个节点的隐藏卷积层，以及一个包含 2 个节点的全连接输出层。请注意，为清晰起见，图中仅显示了隐藏层中两个节点的输入连接。

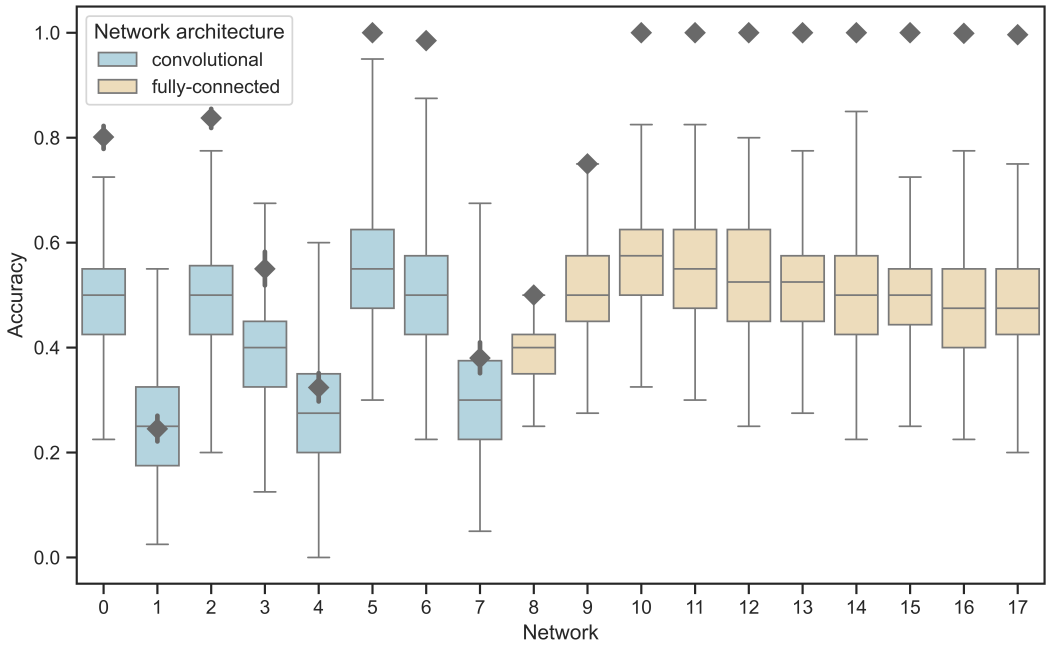
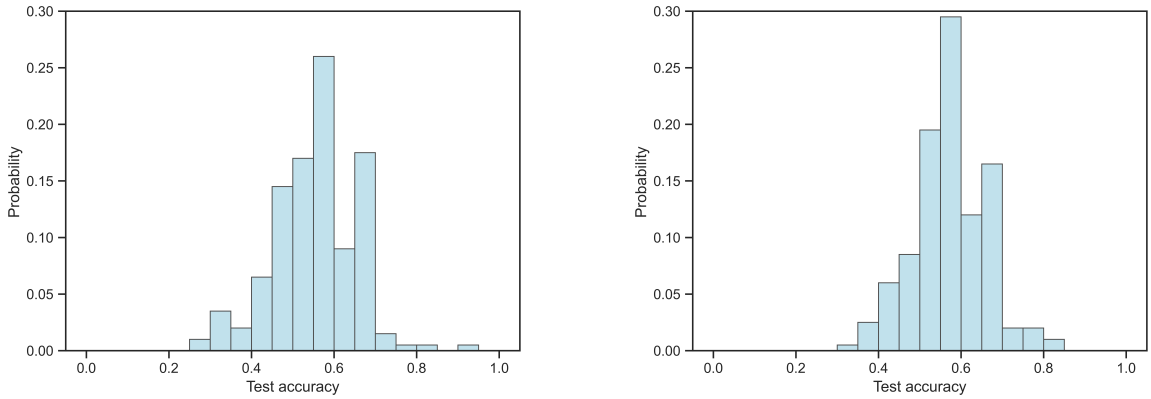


图5：18 种网络架构的测试准确率和训练准确率统计。每个箱线图表示第一四分位数、中位数和第三四分位数，须线表示测试准确率分布的范围（最小值至最大值）。灰色方块表示平均训练准确率，附带的灰色竖线表示训练准确率的范围（最小值至最大值）。

表5：所研究的18种网络架构的测试准确率和训练准确率汇总统计量。每个条目均为四分类图像分类任务中200次独立训练和推理运行所得的结果。表中列出测试准确率的最小值、最大值、均值和中位数，以及相应的平均训练准确率和约束 (14) 与 (13) 的未满足比例均值。以粗体标出的网络5和网络10，分别表示卷积架构和全连接架构中测试准确率最高的网络。

网络	测试准确率				训练准确率	未满足约束
	min.	max.	mean	中位数	mean	平均比例（%）
0	0.225	0.725	0.487	0.500	0.801	0.6875
1	0.025	0.550	0.254	0.250	0.245	4.6782
2	0.200	0.775	0.495	0.500	0.838	0.7845
3	0.125	0.675	0.390	0.400	0.550	2.8549
4	0.000	0.600	0.274	0.275	0.324	2.6483
5	**0.300**	**0.950**	**0.550**	**0.550**	**1.000**	**0.0000**
6	0.225	0.875	0.505	0.500	0.985	0.0865
7	0.050	0.675	0.301	0.300	0.380	2.2917
8	0.250	0.500	0.392	0.400	0.500	10.0000
9	0.275	0.750	0.509	0.500	0.750	3.1250
10	**0.325**	**0.825**	**0.569**	**0.575**	**1.000**	**0.0000**
11	0.300	0.825	0.552	0.550	1.000	0.0000
12	0.250	0.800	0.533	0.525	1.000	0.0000
13	0.275	0.775	0.512	0.525	1.000	0.0000
14	0.225	0.850	0.505	0.500	1.000	0.0000
15	0.250	0.725	0.500	0.500	1.000	0.0000
16	0.225	0.775	0.482	0.475	0.999	0.0086
17	0.200	0.750	0.486	0.475	0.996	0.0273



(a) 网络 5。

(b) 网络 10。

图 6：网络 5 (a) 和网络 10 (b) 各自进行 200 次独立训练和推理运行所得测试准确率的直方图。

$$\min_{k\in\mathcal{K}}\left|b_j+\sum_{i\in P_j}w_{ij}x_i^k\right|$$

该值衡量改变神经元激活所需的最小扰动。将所有非输入节点的间隔相加，可得到一个网络级指标，用于衡量给定训练数据时网络的鲁棒性，

$$S_1=\sum_{j\in\mathcal{N}\setminus\mathcal{N}_0}\min_{k\in\mathcal{K}}\left|b_j+\sum_{i\in P_j}w_{ij}x_i^k\right|.$$

然而，由于 S_1 涉及嵌套最小化，因此无法使用 QUBO表达式进行优化。为得到一个易于处理的表达式，我们定义了另一种代理指标 S_2：不再对所有数据点取最小值，而是按如下方式求和，

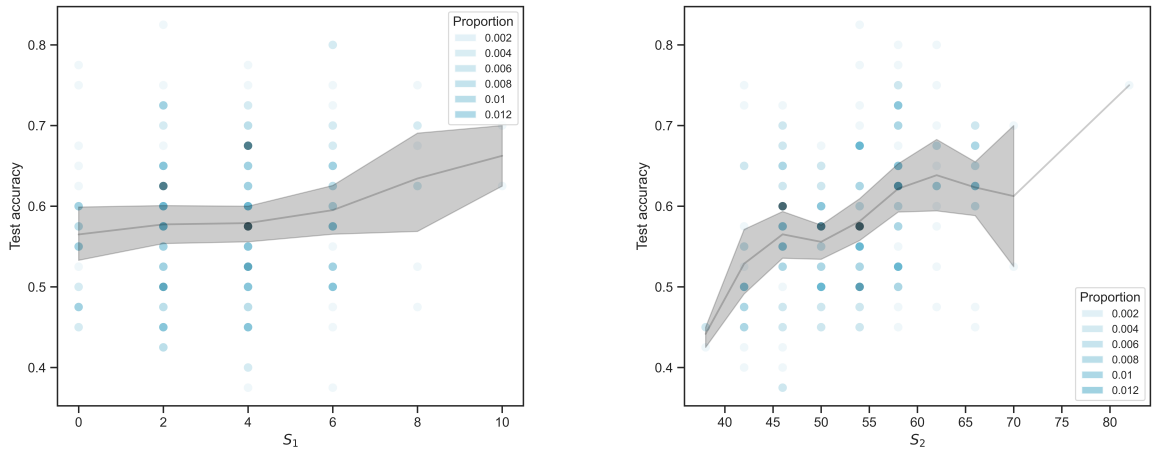
$$S_2=\sum_{j\in\mathcal{N}\setminus\mathcal{N}_0}\sum_{k\in\mathcal{K}}\left|b_j+\sum_{i\in P_j}w_{ij}x_i^k\right|.$$

在图7中，我们绘制了网络 10 在未进行正则化的情况下（如第4节所述）经过算法 200 次运行后的测试准确率与 S_1 和 S_2 的关系。这可以帮助我们了解这两个指标作为泛化预测指标的表现。由于 S_1 依赖于对预激活值绝对值取最小值，其可能结果的数量相对有限，在本例中仅限于六个值（0、2、4、6、8 和 10）；而另一方面， S_2 的取值则更为精细。此外，从图7可以看出， S_2 似乎与测试准确率呈明显的正相关关系——也就是说，在本实验中， S_2 的值越高，似乎表示在未见数据上的准确率越高。接下来，我们将展示如何将代理间隔指标 S_2 表示为 QUBO表达式，并将其纳入训练哈密顿量。

我们首先注意到，对于每个输入数据点 $k\in\mathcal{K}$ 和非输入节点 $j\in\mathcal{N}\setminus\mathcal{N}_0$，根据激活约束（14）以及 ρ_j 的定义，有：

$$\frac{b_j+1}{2}+\sum_{i\in P_j}\frac{w_{ij}x_i^k+1}{2}=2^{n_j}y_j^k+\chi_j^k-\left\lfloor\frac{\kappa}{2}\right\rfloor$$

因此，非输入节点 j 的预激活值为



(a) (b)

图7：使用未经正则化的网络 10 进行 200 次独立训练和推理运行后，评估测试准确率与边际总和 S_1 (a) 和 S_2 (b) 的关系。点表示观测值，灰线表示平均准确率，阴影带表示使用均值标准误计算的 95% 置信区间。点的明暗表示 200 次测试中具有相同测试准确率以及 S_1-值的比例； S_2-值亦然。

$$b_j + \sum_{i \in P_j} w_{ij} x_i^k = 2\left(2^{n_j} y_j^k + \chi_j^k - \left\lfloor \frac{\kappa}{2} \right\rfloor\right) - |P_j| - 1 \tag{17}$$

由于当预激活值严格为正时 $y_j = 1$ ，否则 $y_j = 0$ ，我们可以将数据点 k对应的节点 $j \in \mathcal{N} \setminus \mathcal{N}_0$ 的预激活值绝对值写为

$$\left| b_j + \sum_{i \in P_j} w_{ij} x_i^k \right| = (2y_j^k - 1)\left(b_j + \sum_{i \in P_j} w_{ij} x_i^k \right) \tag{18}$$

根据方程 (17) 和 (18)，我们可以将预激活 a 的绝对值重写为 s,

$$\left| b_j + \sum_{i \in P_j} w_{ij} x_i^k \right| = (2y_j^k - 1)\left(2\left(2^n y_j^k + \chi_j^k - \left\lfloor \frac{\kappa}{2} \right\rfloor\right) - |P_j| - 1\right) \tag{19}$$

式（19）右侧是关于 y_j 的二次式，可以很容易地纳入 QUBO 表述。因此，我们引入以下 QUBO惩罚项,

$$H_{som} = \sum_{j \in \mathcal{N} \setminus \mathcal{N}_0} \sum_{k \in \mathcal{K}} (2y_j^k - 1) \left(2 \left(2^n y_j^k + \chi_j^k - \left\lfloor \frac{\kappa}{2} \right\rfloor \right) - |P_j| - 1 \right)$$

为了最大化 S_2，我们从原始哈密顿量 H_{train} 中减去 H_{som} ，得到增广能量函数，

$$H^*_{train} = H_1 + \alpha H_2 - \gamma H_{som} \tag{20}$$

其中 α 和 γ 是缩放因子。

5.2. 计算结果

为考察最大化 S_2 对基于 QUBO 的二值神经网络的影响，我们在网络 5 和网络 10 上进行了计算实验。实验按照第4节的方法进行，但将目标函数替换为 H^*_{train} (20)。在所有实验中， $\alpha = 1$保持不变，同时改变缩放因子 γ的值。对于 γ的每个取值，我们均进行 200 次训练和推理。结果汇总于表 6 和图 8。采用 $\gamma = 0$ 的实验等同于第4节中未使用正则化的实验，可作为比较基线。表 6 和图 8 表明，引入最大化 S_2的基于间隔的正则化后，二值神经网络在分类任务上的性能显著提升。对于网络 5，使用 $\gamma = 0.03$时结果最佳，平均准确率从 54% 提高到 71%，性能提升了 31%。同样，对于网络 10，使用 $\gamma = 0.02$时平均准确率最高，从 58% 提高到 73%；与未使用正则化的基线相比，准确率相对提升了 26%。

当缩放因子 γ 过大时，性能往往会下降。如表 6 所示，对于 $\gamma \geq 0.03$ ，退火算法有时会生成不满足激活约束或辅助约束（即 $H_1 > 0$ 或$H_2 > 0$）的不可行解。此外，随着γ增加，不可行解的比例也会增加。这一现象源于 H_{som} 在哈密顿量中占主导地位，从而强调间隔最大化，而非约束满足。由于不可行解可能违反激活约束，由此类解得到的网络训练准确率较低，进而拉低所有试验的整体平均性能。然而，仅考虑可行解时，测试准确率会单调上升，最高约为 $\gamma \approx 0.1$（图 9）。具体而言，网络 10 的平均准确率接近 90%。

这些结果表明，最大化间隔能够获得具有更强泛化能力的解。图 10 和图 11 分别展示了网络 10 在 $\gamma = 0.00$、0.01 和 0.02 条件下的测试准确率和间隔指标 S_1 以及 S_2，其中

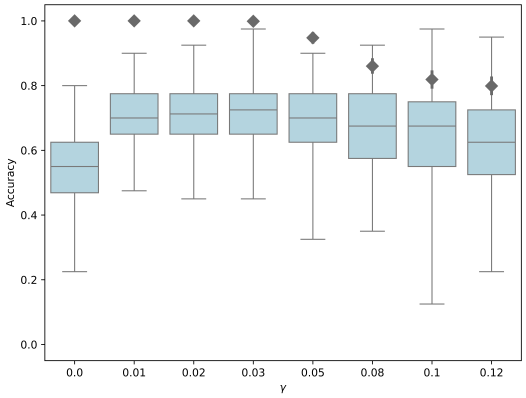
退火运行仅获得可行解。尽管图 7a 中 $\gamma = 0$ 的结果未显示测试准确率与S_1之间存在明确的相关性，但图 10 表明，纳入通过 $\gamma = 0.01$ 和 0.02 获得的结果后，呈现出明显的正相关。这与先前研究关于 S_1 作为正则化度量有效性的结论一致。同时，如图 11 所示，与 S_2 相应的强正相关表明， S_2 可作为 QUBO 表述中的一种有效近似。不过，这可能是由数据集的特性以及模型规模较小所致，因此还需要进一步研究，以确认其在其他数据实例上的有效性。

表 6：网络 5（卷积）和网络 10（全连接）在不同间隔因子之和下的测试准确率和训练准确率汇总统计量 γ。每个条目表示在四分类图像分类任务中进行 200 次独立训练和推理运行所获得的结果。表中列出了测试准确率的最小值、最大值、均值和中位数、相应的平均训练准确率、未满足约束 (13) 和 (14) 的平均比例，以及平均 S_1-和S_2-值。

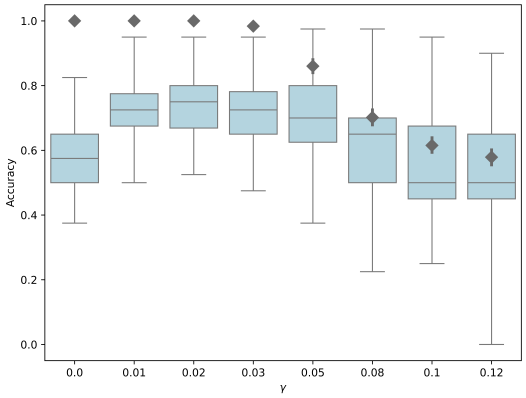
网络	γ	测试准确率				训练准确率	未满足约束	S_1	S_2
		min.	max.	mean	中位数	mean	平均比例（%）	mean	mean
5	0.00	0.225	0.800	0.540	0.550	1.000	0.00	7.05	59.98
5	0.01	0.475	0.900	0.706	0.700	1.000	0.00	10.04	88.48
5	0.02	0.450	0.925	0.704	0.713	1.000	0.00	10.34	89.81
5	0.03	0.450	0.975	0.714	0.725	0.999	0.01	9.91	89.57
5	0.05	0.325	0.900	0.694	0.700	0.948	0.54	10.84	95.60
5	0.08	0.350	0.925	0.673	0.675	0.860	1.40	11.45	102.67
5	0.10	0.125	0.975	0.644	0.675	0.819	2.12	11.32	106.46
5	0.12	0.225	0.950	0.620	0.625	0.799	2.58	10.94	107.66
10	0.00	0.375	0.825	0.583	0.575	1.000	0.00	3.47	52.92
10	0.01	0.500	0.950	0.728	0.725	1.000	0.00	6.86	91.32
10	0.02	0.525	0.950	0.734	0.750	1.000	0.00	7.45	93.86
10	0.03	0.475	0.950	0.724	0.725	0.984	0.17	7.54	96.52
10	0.05	0.375	0.975	0.703	0.700	0.860	1.39	9.93	112.06
10	0.08	0.225	0.975	0.617	0.650	0.701	3.11	12.50	127.94
10	0.10	0.250	0.950	0.557	0.500	0.615	4.38	14.29	137.02
10	0.12	0.000	0.900	0.532	0.500	0.579	5.57	14.79	141.66

6. 一种 dropout 算法

受第 5 节中正则化方法所带来改进的启发，我们进一步研究另一种正则化方法。第 4 节和第 5 节所采用的方法依赖于通过求解一个一次性纳入所有训练数据的单一优化问题来训练神经网络。当训练数据集规模较大时，随着模型 (16) 和 (20) 的规模随训练数据集规模增加，在单次退火运行中找到高质量的解将变得难以实现。为缓解这一可扩展性问题，例如，Georgiev [31] 提出了一种迭代算法，该算法

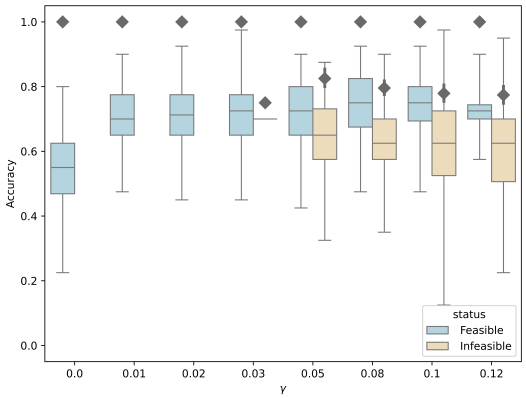


(a) 网络 5。

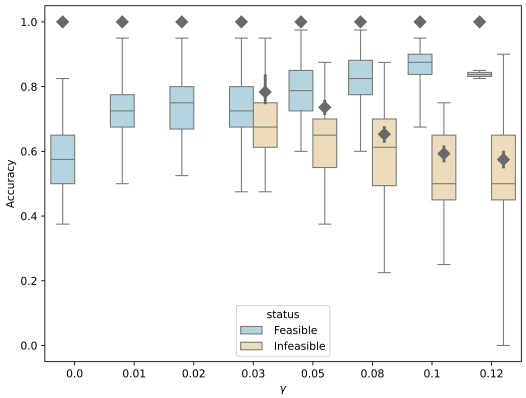


(b) 网络 10。

图 8：网络 5 (a) 和网络 10 (b) 在每个间隔缩放因子 γ 取值下，经过 200 次训练和推理运行后测得的测试准确率分布。每个箱线图表示第一四分位数、中位数和第三四分位数，须线表示该分布的最小值和最大值。灰色方块及相应的竖线分别表示对应 γ 的训练准确率均值和范围（最小值至最大值）。



(a) 网络 5。



(b) 网络 10。

图 9：网络 5 (a) 和网络 10 (a) 在每个间隔缩放因子 γ取值下，按所获得解的可行性（状态）分组，并经过 200 次独立训练和推理运行后测得的测试准确率分布。每个箱线图表示第一四分位数、中位数和第三四分位数，须线表示该分布的最小值和最大值。灰色方块及相应的竖线分别表示对应 γ 的训练准确率均值和范围（最小值至最大值）。

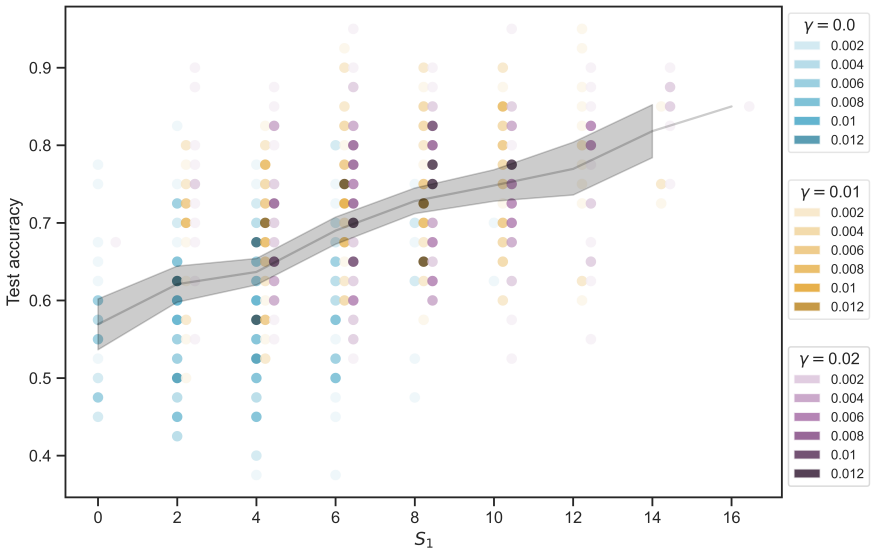


图 10：使用网络 10 进行的所有测试中，测试准确率与边际总和 S_1 的关系，参数为 $\gamma = 0.00$, 0.01，0.02。圆点表示观测值，灰色线条（带）表示观测值的均值（95%置信区间）。对于每个 γ值，圆点的明暗表示具有相同测试准确率和 S_1值组合的测试所占比例。为清晰起见， $\gamma \geq 0.01$ 的图稍向右移，以避免重叠。

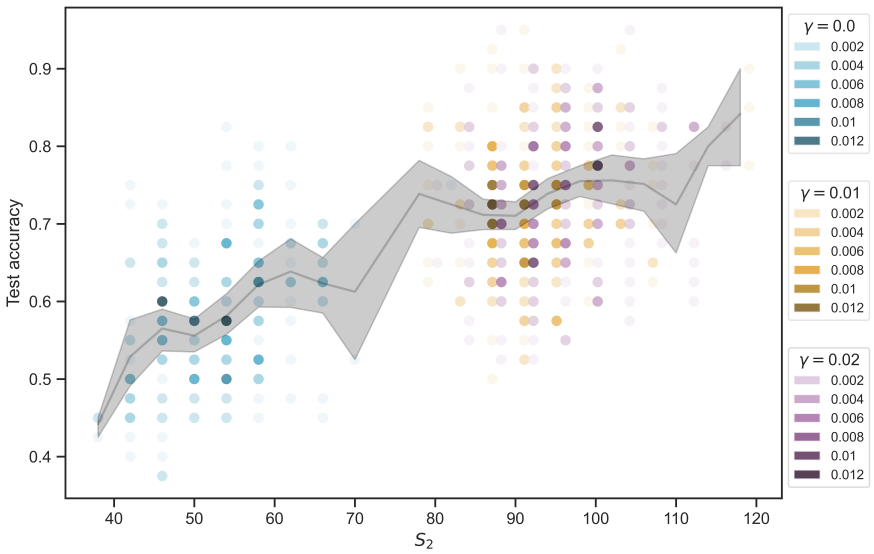


图 11：使用网络 10 进行的所有测试中，测试准确率与边际总和 S_2 的关系，参数为 $\gamma = 0.00$, 0.01，0.02。圆点表示观测值，灰色线条（带）表示观测值的均值（95%置信区间）。对于每个 γ值，圆点的明暗表示具有相同测试准确率和 S_2值组合的测试所占比例。为清晰起见， $\gamma \geq 0.01$ 的图稍向右移，以避免重叠。

通过迭代地从批次输入数据中学习，逐步间接更新网络参数。在这项工作中，我们提出一种类 dropout 正则化方法，该方法会随机移除神经元并利用批量训练数据，从而在保留正则化优势的同时缩小问题规模。我们指出，这种方法会动态改变网络结构，并利用我们的 QUBO 公式的灵活性，以支持任意网络拓扑。

6.1. 模型与算法

下面，我们提出一种迭代算法，通过依次求解若干 QUBO 问题，逐步更新网络参数。为此，我们对训练哈密顿量进行增广，使其能够在权重和偏置上加入额外的（线性）惩罚项。我们将增广哈密顿量定义为：

$$\text{minimise } H^{+}_{train} = H_1 + \alpha H_2 - H_{ext}$$

其中 $H_{ext} = \sum_{j\in\mathcal{N}} \left(c^b_j b_j + \sum_{i\in P_j} c^w_{ij} w_{ij} \right)$。

哈密顿量 H_{ext} 充当外部权重偏置项，因此，对于某个神经元 j，若 c^b_j 取较大的正值，则倾向于选择 $b_j = 1$这一偏置值；若其取较大的负值，则倾向于选择 $b_j = -1$，权重参数也同理。该算法依赖于权重因子向量 c^w 和偏置因子向量 c^b；二者均初始化为零向量，因此初始时 $H_{ext} = 0$。请注意， H_{ext} 相对于 H_1 和 H_2 的缩放由系数向量 c^b 和 c^w隐式定义。

在算法的每次迭代中，通过从原始网络 G 中移除随机选取的一部分节点 $\mathcal{N}_{drop}$，得到网络的一种新配置 $G' = (\mathcal{N}', \mathcal{A}')$，使得 $\mathcal{N}' = \mathcal{N} \setminus \mathcal{N}_{drop}$ and $\mathcal{A}' = \{(i, j) \in \mathcal{A} | i, j \notin \mathcal{N}_{drop}\}$。随后，在缩减后的网络 G' 上，通过最小化函数H^{+}_{train}，使用一批训练数据对新网络进行训练。请注意，从网络中移除节点会影响变量的数量，因为我们不再计算被移除节点的偏置和激活值，也不再计算与这些节点相连的连接的权重。

通过此优化得到的网络参数（权重和偏置）用于指导与缩减后的网络相关联的权重因子和偏置因子 c^w and c^b 的更新，使得对于每个神经元 $j \in \mathcal{N}'$, $c^b_j \leftarrow g(c^b_j, b_j)$, 以及每条连接 $(i, j) \in \mathcal{A}'$,$c^w_{ij} \leftarrow g(c^w_{ij}, w_{ij})$，均采用某个更新函数 g。本文中，我们假设使用以下更新函数：

$$g(a, b) = a + \eta \beta^{n^{USC}} b$$

其中， η 是一个常数学习因子， $\beta \leq 1$ 是一个常数因子， n^{USC} 是当前解中未满足约束的数量。该更新函数

旨在平衡适应性与稳定性。学习率 η 控制更新的步长，决定新的参数值对偏置和权重因子的影响程度。指数项 $\beta^{n^{USC}}$ 作为自适应权重，在存在未满足约束时按指数规律抑制更新幅度。因此，偏置和权重因子的增减与学习率成正比，并随未满足约束的数量按指数规律衰减。请注意，该更新规则是 [31] 中所述规则的推广版本，明确考虑了当前解与可行解之间的距离；当 $\beta = 1$时，两种更新规则等价。迭代次数 M 由用户设定。最后一次迭代后，完整网络 $\mathcal{N}$ 将在不移除任何节点的情况下完成训练。该算法概述见算法1。

算法1 一种用于基于 QUBO 的神经网络的 Dropout 算法。

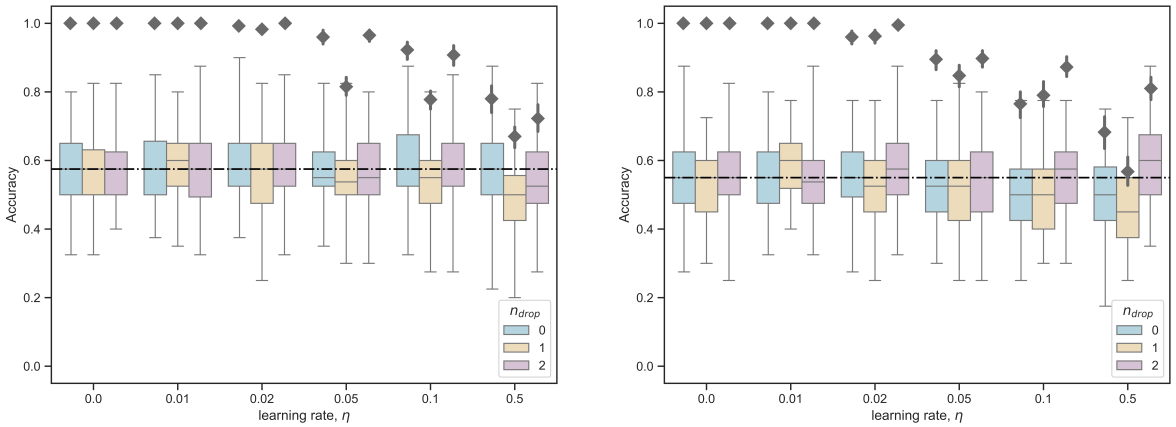
$c^w \leftarrow 0$ ▷ 初始化权重因子
$c^b \leftarrow 0$ ▷ 初始化偏置因子
对于 $i \leftarrow 1$ 到 M **执行**
  $\mathcal{K}_i \leftarrow$ 获取训练批次(i) ▷ 获取训练数据批次 i
  $G' \leftarrow$ 修改网络(G) ▷修改网络，例如移除节点
  $w,\ b \leftarrow$ 最小值$_{G',\ \mathcal{K}_i} H^+_{tr\ ain}(c^w,\ c^b)$ ▷训练网络
  $c^w \leftarrow g(c^w,\ w)$ ▷ 更新权重因子
  $c^b \leftarrow g(c^b,\ b)$ ▷更新偏置因子
结束循环
$w,\ b \leftarrow$ 最小化$_{\mathcal{N}\ ,\mathcal{K}_{M+1}} H^+_{train}(c^w,\ c^b)$ ▷ 训练最终批次时不使用 Dropout

6.2. 计算结果

我们在第 3 节所述并在图3中展示的数据集上进行计算实验。与之前的实验一样，训练数据仅限于前四张图像，而测试在其余 40 张图像上进行。训练批次 $\mathcal{K}_i$ 包含全部四张训练图像，并且在每次迭代中都相同i。因此，我们在这些实验中不采用小批量处理，迭代之间的唯一变化是算法1中 ModifyNetwork过程所移除的神经元。

我们在两种全连接网络架构上进行实验（表4中的网络 10 和网络 12），它们的隐藏层分别包含三个和五个神经元。在每次迭代中，我们从输入层随机移除五个神经元（20 %），并从中间隐藏层移除 $n_{drop} = 0$、1 个和 2 个神经元。输出神经元从不移除。学习率取值为 $\eta = 0.01$、0。02、0。05、0。1 和 0。5，而 $\beta = 0.1$保持不变。所有实验的迭代次数均保持不变，为$M = 10$。

针对每种丢弃超参数组合，分别进行了 100 次独立的训练和测试运行。每次退火计算所用的副本数和蒙特卡洛步数



(a) 网络 10。

(b) 网络 12。

图 12：网络 10 (a) 和网络 12 (b) 在 100 次独立训练和推理运行中的测试准确率分布，涵盖不同学习率 η 和 Dropout 水平 n_{drop}。每个箱线图显示第一四分位数、中位数和第三四分位数，须线表示最小值和最大值。水平虚线表示未使用 Dropout 时获得的测试准确率中位数。灰色方块和竖线分别表示训练准确率的均值和范围。

均与第 3 节所述设置相同。在中间的 Dropout 迭代期间，温度参数固定为 $T_{\max}^{-1} = 0.2$和 T^{-1} 最小值$= 8.6$；而在最终训练阶段，则使用 Nelder-Mead 方法 [40] 优化温度参数。作为对比，对于每个网络，还额外进行了 100 次不应用 Dropout 的测试（即不进行迭代训练），以此作为基线。实验结果汇总于表 7，其中列出了每组实验的测试准确率的最小值、最大值、均值和中位数，以及平均训练准确率。

图 12 展示了所研究的两个网络在不同学习率和 Dropout 率下的测试准确率。在本实验中，Dropout 算法对这两个网络的影响都相当有限。对于网络 12，当 η 和 n_{drop} 均增大时，偶尔会观察到轻微的改善。具体而言，当学习率为 $\eta = 0.5$且 $n_{drop} = 2$时，测试准确率均值从 54%（未使用 Dropout）提高到 59%。相比之下，在相同条件下，网络 10 的准确率略有下降。这种性能下降可能是由于隐藏层中的神经元数量较少（3 个），使该架构不太适合使用 Dropout。此外，训练准确率通常会随着 η 增大而下降。这种现象可能是因为在最后一次退火运行中，额外的惩罚项 H_{ext} 变得比 H_1 和 H_2 更占主导地位，从而导致违反约束 (14) 和 (13) 的不可行解。不过，测试准确率随 η 增大而出现的下降仍相对较小。这或许可以作为证据，表明 Dropout 过程能在训练期间提升网络的泛化能力。为了进一步验证批处理带来的模型尺寸缩减效果及 Dropout 的泛化益处，还需使用更大的数据集和网络架构开展额外实验。

表 7：Network 10 和 Network 12 在改变学习率 η 以及隐藏层中丢弃的神经元数量 n_{drop} 时，测试准确率和训练准确率的汇总统计量。每个条目表示在四分类图像分类任务中进行 100 次独立训练和推理运行所得的结果。该表列出了测试准确率的最小值、最大值、均值和中位数、相应的平均训练准确率，以及未满足约束 (13) 和 (14) 的平均比例。

网络	η	n_{drop}	测试准确率				训练准确率	未满足约束
			min.	max.	mean	中位数	mean	平均比例（%）
10	0.00	-	0.400	0.825	0.574	0.575	1.000	0.00
10	0.01	0	0.375	0.850	0.583	0.575	1.000	0.00
10	0.01	1	0.350	0.800	0.587	0.600	1.000	0.00
10	0.01	2	0.325	0.875	0.577	0.575	1.000	0.00
10	0.02	0	0.375	0.900	0.592	0.575	0.993	0.07
10	0.02	1	0.250	0.825	0.568	0.575	0.983	0.16
10	0.02	2	0.325	0.850	0.586	0.575	1.000	0.00
10	0.05	0	0.350	0.825	0.575	0.550	0.960	0.55
10	0.05	1	0.300	0.825	0.546	0.537	0.815	1.84
10	0.05	2	0.300	0.800	0.578	0.550	0.965	0.32
10	0.10	0	0.325	0.875	0.590	0.575	0.922	0.91
10	0.10	1	0.275	0.800	0.540	0.550	0.777	2.48
10	0.10	2	0.275	0.850	0.576	0.575	0.907	1.07
10	0.50	0	0.225	0.875	0.571	0.575	0.780	2.20
10	0.50	1	0.200	0.750	0.493	0.500	0.670	3.52
10	0.50	2	0.275	0.825	0.537	0.525	0.723	2.77
12	0.00	-	0.275	0.875	0.542	0.550	1.000	0.00
12	0.01	0	0.325	0.800	0.546	0.550	1.000	0.00
12	0.01	1	0.400	0.775	0.579	0.600	1.000	0.00
12	0.01	2	0.325	0.875	0.545	0.537	1.000	0.00
12	0.02	0	0.275	0.775	0.551	0.550	0.960	0.28
12	0.02	1	0.250	0.775	0.529	0.525	0.963	0.25
12	0.02	2	0.325	0.875	0.573	0.575	0.995	0.06
12	0.05	0	0.300	0.775	0.524	0.525	0.895	0.96
12	0.05	1	0.250	0.825	0.518	0.525	0.848	1.25
12	0.05	2	0.250	0.800	0.534	0.550	0.897	0.96
12	0.10	0	0.250	0.775	0.492	0.500	0.765	2.24
12	0.10	1	0.300	0.775	0.499	0.500	0.790	2.18
12	0.10	2	0.300	0.775	0.551	0.575	0.873	1.25
12	0.50	0	0.175	0.750	0.496	0.500	0.682	5.35
12	0.50	1	0.250	0.725	0.463	0.450	0.568	5.93
12	0.50	2	0.350	0.875	0.589	0.600	0.810	5.00

7. 结论

基于 QUBO 的公式化表达近期已实现了在伊辛机上训练二值神经网络。在本研究中，我们展示了该公式化表达可以扩展到任意网络拓扑，而不一定是分层或无环的。我们在数字伊辛机上针对一个小型四分类图像分类问题开展了计算实验，并提出了两种用于正则化训练问题的新方法。其中一种是通过最大化神经元裕度之和来实现的二次惩罚项。计算结果表明，该方法显著提升了测试准确率，并在裕度大小与泛化之间产生了明显的正相关性。另一种是类丢弃算法，在训练过程中随机修改网络拓扑。实验结果未显示性能提升，但即使在较高的 Dropout 率下，测试准确率仍保持稳健，表明其具有一定的泛化益处。

尽管取得了这些结果，仍存在若干局限性。该训练方法受到算法寻找满足所有激活约束的解的能力限制。随着网络规模和训练数据集的增大，这一过程变得越来越困难，并会影响性能。因此，本文所呈现的实验结果仅限于一个相对较小的构造数据集。还需要进一步研究，以确认这些方法对更大且更贴近现实的数据集的适用性，以及潜在的算法和模型改进方向。

未来研究应通过算法和模型改进，提升性能、可扩展性和适用性。首先，将训练问题表述为 QUBO问题时，需要构建一个能够由退火算法高效探索的能量景观，以提升性能 [41]。从这一角度来看，确定合适的缩放因子 α 和 γ至关重要，并应基于理论和实验分析进一步完善表述本身。其次，进一步改进和定制退火算法，例如改进对约束 [42], 的处理，或有助于找到高质量的训练问题解。第三，本研究假设模型参数为二值，且始终能够获得带标签训练数据；在实践中，这些假设可能过于严格，从而限制性能和适用性。不过，只需进行少量修改即可通过引入并行位级连接来支持多比特权重；此外，将该表述扩展到基于无标签数据的无监督学习，也为未来研究指明了一个值得探索的方向。最后，探索可能受益于本文所提出模型所支持的循环网络结构的任务，也是一个潜在的研究方向。

在稿件准备过程中使用生成式人工智能和人工智能辅助技术的声明

在准备这项工作期间，作者使用 ChatGPT 改进语言表达和可读性。作者已对所提供的建议进行审阅和编辑

as n，并对已发表文章的内容承担全部责任 e.

致谢

感谢 Alireza Ahrabian、Takuya Okuyama、Kento Hasegawa、Kazuo Ono 和 Masanao Yamaoka 富有成效的讨论与建议，并感谢他们为我们的实验提供计算环境。

参考文献

[1] A. de Vries，《人工智能不断增长的能耗足迹》，Joule 7 (10) (2023) 2191–2194。[2] J. Chen，X. Ran，《边缘计算中的深度学习：综述》，IEEE 论文集 107 (8) (2019) 1655–1674。[3] M. Courbariaux，Y. Bengio，J.-P. David，《BinaryConnect：在传播过程中使用二进制权重训练深度神经网络》，神经信息处理系统进展 28 (2015)。[4] M. Courbariaux，I. Hubara，D. Soudry，R. El-Yaniv，Y. Bengio，《二值化神经网络：将深度神经网络的权重和激活值约束为 +1 或 -1 进行训练》（2016）。arXiv:1602.02830。[5] I. Hubara，M. Courbariaux，D. Soudry，R. El-Yaniv，Y. Bengio，《二值化神经网络》29 (2016)。[6] M. Rastegari，V. Ordonez，J. Redmon，A. Farhadi，《XNOR-Net：使用二值卷积神经网络进行 ImageNet 分类》，载于：欧洲计算机视觉会议，Springer，2016，第 525–542 页。[7] C. Yuan，S. S. Agaian，《二值神经网络综述》，人工智能评论 56 (11) (2023) 12949–13013。[8] H. Qin，R. Gong，X. Liu，X. Bai，J. Song，N. Sebe，《二值神经网络：综述》，模式识别 105 (2020) 107281。doi:10.1016/j.patcog.2020. 107281。[9] R. Toro Icarte，L. Illanes，M. P. Castro，A. A. Cire，S. A. McIlraith，J. C. Beck，《使用混合整数规划和约束规划训练二值化神经网络》，载于：《约束规划原理与实践：第 25 届国际会议 CP 2019》，美国康涅狄格州斯坦福，2019 年 9 月 30 日至 10 月 4 日，论文集 25，Springer，2019，第 401–417 页。[10] E. Ising，《铁磁性理论的贡献》，《物理学杂志》31 (1925) 253–258。doi:10.1007/BF02980577。

[11] J．J．霍普菲尔德，神经网络与涌现集体现象的物理系统
计算能力。《美国国家科学院院刊》79 (8) (1982) 2554–2558。[12]D. H. Ackley、G. E. Hinton、T. J. Sejnowski，玻尔兹曼机的学习算法，《认知科学》9 (1) (1985) 147–169。doi:10.1207/ s15516709cog0901_7。[13] A. Lucas，许多NP问题的伊辛模型表述，《物理前沿》2 (2014年) 5。[14] F. Barahona，关于伊辛自旋玻璃模型的计算复杂性，《物理学杂志A：数学与通用》15 (10) (1982) 3241。[15] N. Mohseni、P. L. McMahon、T. Byrnes，作为组合优化问题硬件求解器的伊辛机，《自然综述：物理学》4 (2022) 363–379。[16] M. Johnson、M. Amin、S. Gildert、T. Lanting、F. Hamze、N. Dickson、R. Harris、A. Berkley、J. Johansson、P. Bunyk 等，利用人造自旋进行量子退火，《自然》473 (2011) 194–198。[17] T. Inagaki、Y. Haribara、K. Igarashi、T. Sonobe、S. Tamate、T. Honjo、A. Marandi、P. L. McMahon、T. Umeki、K. Enbutsu 等，用于求解 2000 节点优化问题的相干伊辛机，《科学》(2016) 603–606。[18] M. Yamaoka、C. Yoshimura、M. Hayashi、T. Okuyama、H. Aoki、H. Mizuno，采用 CMOS 退火技术、用于组合优化问题的 20k 自旋伊辛芯片，载于：IEEE国际固态电路会议，2015。[19] T. Okuyama、T. Sonobe、K. Kawarabayashi、M. Yamaoka，通过动量退火进行二元优化，《物理评论E》100 (1) (2019) 012111。[20] M. Aramon、G. Rosenberg、E. Valiante、T. Miyazawa、H. Tamura、H. G. Katz-graber，利用数字退火器进行受物理启发的二次无约束问题优化，《物理前沿》7 (2019) 48。[21] H. Goto、K. Tatsumura、A. R. Dixon，通过模拟非线性哈密顿系统中的绝热分岔实现组合优化，《科学进展》5 (4) (2019) eaav2372。doi:10.1126/sciadv.aav2372。[22] S. Yarkoni、E. Raponi、T. Bäck、S. Schmitt，面向工业应用的量子退火：介绍与综述，《物理学进展报告》85 (2022) 104001。[23] S. Niazi、S. Chowdhury、N. A. Aadit、M. Mohseni、Y. Qin、K. Y. Camsari，使用稀疏伊辛机训练深度玻尔兹曼网络，《自然·电子学》7 (7) (2024年) 610–619。

[24] P. Date、D. Arthur、L. Pusey-Nazzaro，用于训练机器学习模型的 QUBO公式，《科学报告》11 (1) (2021年) 10029。[25] K. Yawata、Y. Osakabe、T. Okuyama、A. Asahara，QUBO决策树：退火机拓展决策树划分，载于：IEEE国际知识图谱会议 (ICKG)，IEEE，2022，第 355–364 页。[26] K. Kitai、J. Guo、S. Ju、S. Tanaka、K. Tsuda、J. Shiomi、R. Tamura，利用量子退火和因子分解机设计超材料，《物理评论研究》2 (2020) 013319。doi:10.1103/PhysRevResearch.2.013319。[27] S. Izawa、K. Kitai、S. Tanaka、R. Tamura、K. Tsuda，利用伊辛机和随机子空间编码进行连续黑盒优化，《物理评论研究》4 (2022) 023062。doi:10.1103/PhysRevResearch.4.023062。[28] M. Nagel、R. A. Amjad、M. Van Baalen、C. Louizos、T. Blankevoort，向上还是向下？用于训练后量化的自适应舍入，载于：国际机器学习会议，PMLR，2020，第 7197–7206 页。[29] S. Abel、J. C. Criado、M. Spannowsky，完全量子神经网络，《物理评论A》106 (2) (2022) 022601。[30] M. Sasdelli、T.-J. Chin，用于二值神经网络的量子退火表述，载于：2021年数字图像计算：技术与应用 (DICTA)，IEEE，2021，第 1–10 页。[31] G. Georgiev，量子计算机上的基于 QUBO 的神经网络，技术报告，卡尔斯鲁厄理工学院 (2023)。[32] C. Cortes、V. Vapnik，支持向量网络，《机器学习》20 (3) (1995) 273–297。[33] S. Sun、W. Chen、L. Wang、X. Liu、T.-Y. Liu，大间隔深度神经网络：理论与算法 (2015)。arXiv:1506.05232v1。[34] G. Elsayed、D. Krishnan、H. Mobahi、K. Regan、S. Bengio，用于分类的大间隔深度网络，《神经信息处理系统进展》31 (2018)。[35] Y. Jiang、D. Krishnan、H. Mobahi、S. Bengio，利用间隔分布预测深度网络中的泛化差距，arXiv预印本 arXiv:1810.00113 (2018)。[36] G. E. Hinton、N. Srivastava、A. Krizhevsky、I. Sutskever、R. R. Salakhutdinov，通过防止特征检测器共同适应来改进神经网络，arXiv预印本 arXiv:1207.0580 (2012)。

[37] N. Srivastava、G. Hinton、A. Krizhevsky、I. Sutskever、R. Salakhutdinov，《Dropout：一种防止神经网络过拟合的简单方法》，机器学习研究期刊 15 (1) (2014年) 1929–1958.[38] C. Gambella、B. Ghaddar、J. Naoum-Sawaya，《机器学习优化问题：综述》，《欧洲运筹学杂志》290 (3) (2021年) 807–828.[39] E. Boros、P. L. Hammer，《伪布尔优化》，Discrete applied mathematics 123 (1-3) (2002年) 155–225.[40] J. A. Nelder、R. Mead，《一种用于函数最小化的单纯形法》，《计算机杂志》7 (4) (1965年) 308–313.[41] D. Dobrynin、A. Renaudineau、M. Hizzani、D. Strukov、M. Mohseni、J. P. Strachan，《伊辛机中组合优化的能量景观》，《物理评论E》110 (2024年) 045308. doi:10.1103/PhysRevE.110.045308.[42] H. Kameyama、Y. Takita、M. Parizy、S. Sazawa、P. Caballero，《具有二次约束二进制优化问题的数字退火器基准测试》，载于：遗传与进化计算会议伴会论文集，GECCO '24 伴会，计算机协会，纽约，NY，美国，2024年，第1954–1957页。doi:10.1145/3638530.3664114.