张量网络与神经网络相遇：综述与未来展望

Maolin Wang*、Yu Pan*、Zenglin Xu**、*IEEE* 高级会员、Guangxi Li、Xiangli Yang 和 Andrzej Cichocki，*IEEE* 会士

摘要——张量网络（TNs）和神经网络（NNs）是两种基础的数据建模方法。张量网络最初用于解决大规模张量中的维数灾难，其通过将指数级数量的维度转换为多项式复杂度来实现这一目标。因此，张量网络在量子物理和机器学习领域受到了广泛关注。与此同时，神经网络在计算机视觉、自然语言处理和机器人学研究等多种应用中展现出了卓越的性能。有趣的是，尽管这两类网络源自不同的观察，它们却通过共同的多线性结构而内在关联，该结构是张量网络和神经网络的基础，从而推动了大量关于二者结合的研究发展。在本文中，我们将这些结合称为张量神经网络（TNNs），并从三个主要方面介绍张量神经网络：网络压缩、信息融合和量子电路模拟。此外，本文还探讨了改进张量神经网络的方法，考察了用于实现张量神经网络的灵活工具箱，记录了张量神经网络的发展，并强调了其潜在的未来方向。据我们所知，这是首篇全面梳理神经网络、张量网络与量子电路之间联系的综述。我们整理了一份张量神经网络列表，详见 https://github.com/tnbar/awesome-tensorial-neural-networks。

索引词—张量　　网络、深度神经网络、网络压缩、信息融合、量子电路模拟

arXiv:2302.09019v2 [cs.LG] 8 May 2023

1 引言

张量 TENSORS 是表示多个模态源之间多路交互的高阶数组。相比之下，向量（即一阶张量）和矩阵（即二阶张量）分别只沿一个或两个模态访问。作为一种常见的数据类型，张量已广泛出现在多种场景中 [1], [2], [3], [4]。例如，功能性磁共振成像（fMRI）样本本质上是由随时间变化的三维体素组成的四阶张量 [5], [6], [7]。在量子物理学中，用于研究多体量子系统的变分波函数同样是高阶张量 [8], [9]。在时空交通分析中，从多条道路收集数周的道路流量/速度信息，也可以构造为三阶张量（道路路段×天×一天中的时刻） [10]。然而，对于高阶张量，当模态数量增加时，张量中的元素总数会呈指数增长，从而给张量的存储和处理带来灾难性问题，这也被称为“维数灾难” [11]。张量网络是解决这一问题的常见且有效的方法。

* 共同贡献 ** 通讯作者

• *M. Wang* 就职于中国香港特别行政区的香港城市大学。电子邮箱：*morin.w98@gmail.com*
• *Y. Pan* 就职于中国深圳的哈尔滨工业大学（深圳）。电子邮箱：*i perryuu@gmail.com*
• *Z. Xu* 就职于中国深圳的哈尔滨工业大学（深圳）和鹏城实验室。电子邮箱：*zenglin@gmail.com*
• *G. Li* 就职于澳大利亚新南威尔士州的悉尼科技大学。电子邮箱：*gxli2017 @gmail.com*
• *X. Yang* 就职于中国成都的电子科技大学。电子邮箱：*xlyang@std.uestc.edu.cn*
• *A. Cichocki* 任职于波兰科学院系统研究所，地址为波兰华沙 *Newelska 6* 号，邮编 *01-447*；同时任职于理化学研究所人工智能项目组，地址为日本东京 *103-0027*。电子邮箱：*a.cichocki@riken.jp*

张量网络（TN）。张量网络 [8], [11], [12]通常是由一组数量可数的小规模张量构成的集合，这些张量通过张量收缩相互连接。这些小规模张量被称为“组件”“块”“因子”或“核心”。通过张量网络，可以用极度压缩的分布式格式近似表示超大规模张量。因此，可以实现分布式存储，并高效处理以往无法处理的高阶张量。采用张量网络方法，可以缓解或彻底克服维数灾难 [11]。常用的张量网络格式包括 CANDECOMP/PARAFAC（CP分解） [13], [14], [15], Tucker分解 [16], [17], 块项 Tucker（BTT）分解 [18], 矩阵乘积态（MPS）/张量列车（TT）分解 [19], [20], [21], 矩阵乘积算子（MPO）/矩阵张量列车（mTT）分解 [19], [20], [21], 张量环（TR）分解 [22],树张量网络/分层 Tucker（HT）分解 [23], 投影纠缠对态（PEPS）/张量网格分解 [8],[24], 多尺度纠缠重整化 [25], 等。为了理解张量网络的互连结构，人们设计了张量网络图，将其作为一种直观的图形示意图（第2.2节将对此进行讨论）。张量网络能够为分析某些计算上不可接受的任务提供理论和计算框架。例如，Pan 等人基于张量网络的低秩结构 [26]，使用 512 个图形处理器（GPU）在 15 小时内解决了量子随机线路采样问题；此前，人们认为该问题需要在最强大的经典电子超级计算机上运行超过 10,000 年才能解决，并且它实际上对 Google 的量子计算机“Sycamore”所体现的量子霸权构成了挑战。其他应用还包括脑分析 [27]，、降维 [28]，、子空间学习 [29]，等。

神经网络（神经网络）。神经网络是一种学习范式，使机器能够通过反向传播从观测数据中学习知识 [122], [123]。神经网络堆叠有

表 1：TNN 及其效用概览。我们首先在第 3 节介绍不同基本 NN 结构中紧凑型 TNN 的构建，包括卷积神经网络、循环神经网络、Transformer、图神经网络（GNN）和受限玻尔兹曼机。接着，我们在第 4 节探讨基于张量融合和多模态融合的高效信息融合方法中张量网络的应用。然后，我们在第 5 节讨论涉及量子电路和量子张量神经网络的张量网络应用。此外，我们在第 6 节解释张量神经网络的一些训练和实现技术。最后，我们在第 7 节介绍一些用于处理张量神经网络的通用且功能强大的工具箱。

类别		子类别	详细模型/技术	章节
TNNs	网络压缩	卷积神经网络	CP-CNN [30], [31], [32], [33], [34], Tucker-CNN [33], [35], [36], TT-CNN [37], [38], [39], [40],TR-CNN [41], BTT-CNN [42], TC-CNN [43], [44], HT-TT-CNN [45], T-Net [46], TTMT/TMT [47], PMT [48], CP-HOConv [49]	3.1
		循环神经网络	TT-RNN [50], BTT-RNN [42], [51], TR-RNN [52], HT-RNN [53], CP/Tucker-RNN [35], TC-RNN [43], [44] Kronecker-RNN [54], Tucker/CP/TT-RNN [55], HT-TT-RNN [45], KCP-RNN [56], Conv-TT-LSTM [57]	3.2
		Transformer	BTT-Transformer [58], MPO-Transformer [59], MMT [60] Tuformer [61],Tucker-Bert [62], Hypoformer [63], 因果张量网络 [64]	3.3
		图神经网络	TGNNs [65], TGCN [66], DSTGNN [67], RTGNN [68]	3.4
		受限玻尔兹曼机	TT-RBM [69], TR-RBM [70], Tv-RBM [71], Mv-RBM [72]	3.5
	信息融合	张量融合层	TFL [65], [73], [74], LMF [75], PFN [76]	4.1
		多模态池化层	MUTAN [77], MCB [78], MLB [79] CIT [80]	4.2
	量子电路模拟	量子嵌入	图像嵌入 [81], 语言嵌入 [82], [83], [84]	5.1
		量子数据处理	监督矩阵乘积态 [81], 广义张量网络 [85], 树张量网络 [86], 均匀矩阵乘积态 [84],LPS [87], 自回归矩阵乘积态 [88]	5.2
		量子张量神经网络	ConvAC [83], [89], TLSM [90]	5.3
效用 of TNNs	训练策略	稳定训练	混合精度 [91], Yu初始化 [92]	6.1
		秩选择	PSTRN [93], TR-RL [94], CP-Bayes [95], PARS [96] TT-Bayes [97], 自适应TR [98], TT-ADMM [99], BMF [100], [101]	6.2
		硬件加速	TIE [102], LTNN [103], TT-Engine [104], Fast CP-CNN [105]	6.3
	工具箱	基本张量运算	Tensorly [106], TensorTools [107],Tensor Toolbox [108], TenDeC++ [109], OSTD [110], TensorD [111], TT-Toolbox [112], Tntorch [113],TorchMPS [114],ITensor [115],T3F [116], TensorNetwork [117],Scikit-TT [118]	7.1
		深度模型实现	Tensorly-Torch [106], TedNet [35]	7.2
		量子张量模拟	Yao [119], TensorNetwork [117], lambeq [120], ITensor [115], TeD-Q [121]	7.3

多层，即深度神经网络 [124], [125], 由于能够从深层结构中捕获丰富信息，深度神经网络在人工智能领域得到了广泛应用。深度神经网络的典型类型包括受限玻尔兹曼机 [126], 卷积神经网络 [125], [127], 循环神经网络 [128], [129], 和 Transformer [130], [131]。目前，深度神经网络在计算机视觉 [132]和自然语言处理 [133]等众多应用领域达到最先进的性能。例如，AlexNet [134], VGGNet [135], GoogLeNet [136] 和 ResNet [137]在 ImageNet数据集上夺冠 [138], , 展现出解决图像分类任务的巨大潜力。尤其是，AlphaFold 2 [139], [140] 能够在几天内识别蛋白质结构，而这项工作过去需要研究人员花费数年时间。AlphaFold2 [139],[140] 已经预测了几乎所有已知蛋白质的结构，平均达到原子精度。最近，同样基于 Transformer 架构的 ChatGPT [141],在各种专业和学术基准数据集上展现出人类水平的性能，并将极大地改变各种现有工作的模态。深度学习技术仍在推动多个学科的发展，包括语音识别 [142],DNA突变检测 [143],结构生物学 [140], 等。

张量网络遇见神经网络。 如上所述，张量网络和神经网络是两类源自不同领域、并从不同方面取得成功的网络。有趣的是，它们并非彼此正交，而是通过多线性数学性质紧密联系在一起 [11], [144]。因此，一种有前景的方法是通过多线性将二者整合起来，以实现“整体大于部分之和”的目标。张量网络的主要优势在于其紧凑的结构、多重条目以及与量子力学的密切关系，而神经网络则因其广泛的应用而广为人知 [8], [12]。基于这些观察，可以通过三种方式将张量网络与神经网络结合起来。

（1）网络压缩。神经网络在各种任务中已取得许多成功 [124], [125]。然而，神经网络仍然面临涉及海量维度的过多线性乘积计算以及维数灾难 [51]。解决这一问题的一种有前景的方法是利用张量网络的轻量化和多线性特征 [41], [51], [52]。具体而言，张量网络可以将神经网络中的任意张量分解为更小的块，从而将维数降低至线性复杂度 [30], [31]。例如，与使用朴素长短期记忆（LSTM）执行动作识别任务相比，TR-LSTM [52] 模型

利用张量网络技术分解权重张量，可以将参数数量压缩约34,000倍，同时性能优于朴素LSTM。

（2）信息融合。在现实世界的数据分析场景中，为了取得更好的性能，对来自多模态来源的数据中的高阶交互进行建模十分重要 [8]。然而，神经网络通常用于处理单模态向量形式的输入，因此缺乏对这类高阶交互进行建模的足够表达能力 [77]。为解决这一问题，一种有前景的方法是将张量网络嵌入神经网络中，利用多入口属性作为高效的融合单元来处理多模态数据 [73],[74], [76]。以视觉问答（VQA）任务 [145]为例，多模态 Tucker 融合（MUTAN） [77] 可以通过 Tucker 格式框架学习文本表示与视觉表示之间的高阶交互。因此，MUTAN 凭借参数化高效的低秩结构，取得了当前最先进的性能。

（3）量子电路模拟。张量网络可以充当模拟器，并成为经典神经网络与量子电路之间的桥梁。首先，许多研究表明，可以将神经网络部署在量子电路上，借助量子计算方案的超高并行性来提升其运行速度 [146], [147]。然而，目前量子计算机的能力还不足以直接部署神经网络，这使得验证量子神经网络（QNNs）的潜在性能变得困难 [146]。幸运的是，由于张量网络与量子电路之间具有等价性，张量网络可以在电子计算机中充当有效的量子模拟器 [8], [148]。具体而言，量子电路中的输入量子比特和幺正操作门都可以视为张量。门之间的连接也可以视为张量网络方案中的张量收缩 [148]。通过利用张量网络为神经网络实现量子电路模拟，可以在真正具备强大能力的量子计算机制造出来之前，开启对量子神经网络的探索新时代。

我们将这类连接张量网络与神经网络的方法称为**张量神经网络（TNNs）**。据我们所知，这是首篇全面梳理神经网络、张量网络与量子电路之间联系的综述。表1展示了张量神经网络及其用途的概览。

本综述的其余部分安排如下。
第 2 节介绍张量记号、张量图和张量网络格式的基础知识。第 3 节讨论使用张量网络构建紧凑型张量神经网络。第 4 节探讨使用张量神经网络实现高效的信息融合过程。第5节讨论张量网络在量子电路和张量神经网络中的一些基本应用。第6节介绍张量神经网络的一些训练和实现技术。第 7节介绍可用于处理张量神经网络的通用且功能强大的工具箱。

2 张量基础

2.1 张量记号

张量 [149], [150], 也称为多维数组，可以视为向量（即一阶张量）或矩阵（即二阶张量）的高阶扩展。与矩阵中的行和列类似， N阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$具有 N 个模态（即方式、阶或指标），其长度（即维度）分别表示为 I_1 到 I_N。如表2所示，小写字母表示标量，例如 a；粗体小写字母表示向量，例如$\mathbf{a}$；粗体大写字母表示矩阵，例如$\mathbf{A}$；粗体花体字母表示

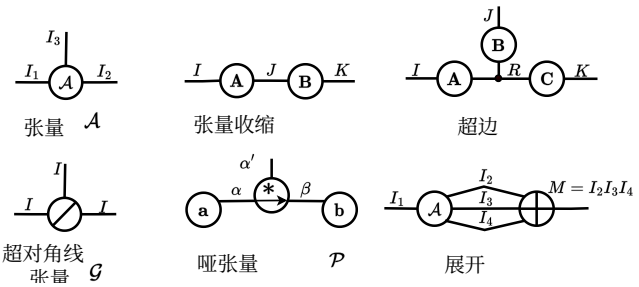


图1：张量网络图的基本符号。有关张量网络的更多基础知识，请参阅 [8]和 [11]。

表2：张量记号。

符号	说明
a	标量
$\mathbf{a}$	向量
$\mathbf{A}$	矩阵
$\mathcal{A}$	张量
A	维数
$\circledast$	卷积运算
$\circ$	外积操作
$<\cdot,\ \cdot>$	两个张量的内积
$\lvert\cdot\rangle$	量子态 bra 向量（单位列复向量）
$\langle\cdot\rvert$	量子态 ket 向量（单位行复向量）
$\langle\cdot\rvert\cdot\rangle$	两个量子态向量的内积

高阶张量，例如 $\mathcal{A}$。本文将“张量”定义为一个更广义的概念，其中包括标量、向量和矩阵。

2.2 张量图

在本小节中，我们将介绍张量网络图及其对应的数学运算。张量网络图最初由 Roger Penrose [151] 于 20 世纪 70 年代初提出，如今常用于描述量子算法 [8], [9] 和机器学习算法 [12], [30], [81]。在这些图中，张量以图形方式表示为带有边的节点 [20]。张量网络图是直观展示和便捷表示复杂张量的实用工具。因此，张量图在张量领域得到了广泛应用。由于深度学习领域中的数据和权重都是张量，张量图也有望作为该领域通用的网络分析工具。图中展示了张量的基本符号概览。1。

2.2.1 张量节点

张量表示为带有边的节点，如图 1 所示。边的数量表示张量的模态，边上的数值表示相应模态的维度。例如，单边节点表示向量 $\mathbf{a} \in \mathbb{R}^{I}$，双边节点表示矩阵 $\mathbf{A} \in \mathbb{R}^{I \times J}$，三边节点表示张量 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$。

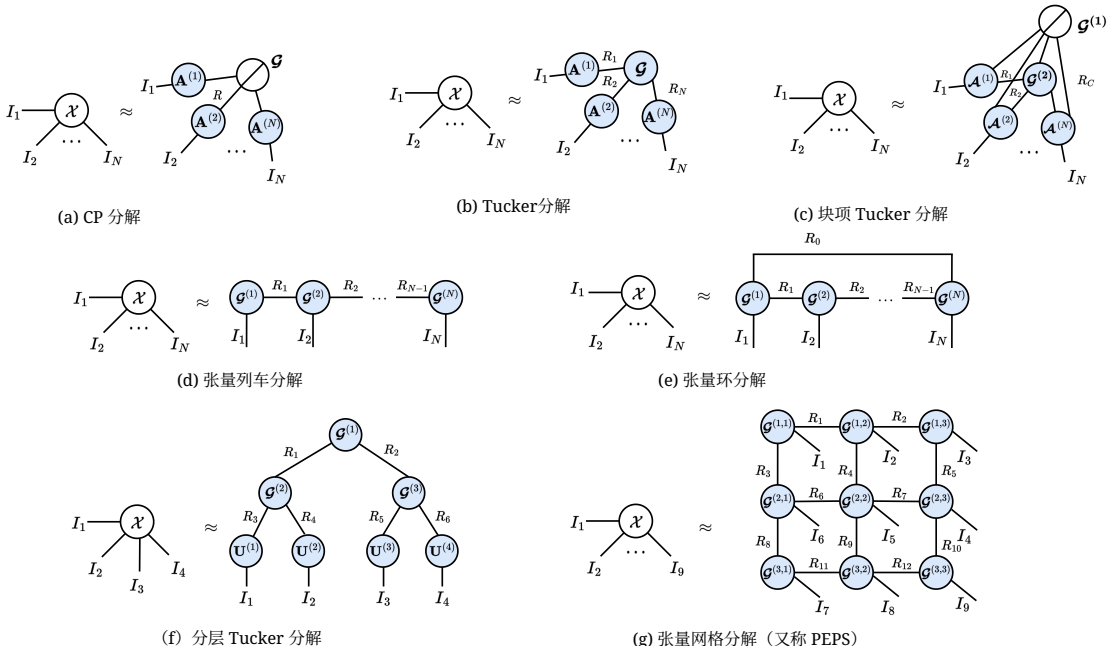
2.2.2 张量收缩

张量收缩是指沿着两个张量对应的指标对，将两个张量收缩为一个张量。收缩后，相应的连接边消失，而悬挂边保留。张量收缩可以表示为张量积：

$$\mathcal{C} = \mathcal{A} \times_{M+1,M+2,\ldots M+N}^{1,2,\ldots N} \mathcal{B} \tag{1}$$

且 $\mathcal{C}$ 的元素通过以下方式计算：

$$\mathcal{C}_{p_1,\ldots,p_{K+M}} = \sum_{i_1,i_2,\cdots i_N} \mathcal{A}_{i_1,i_2,\cdots i_N,*} \quad \mathcal{B}_{*,i_1,i_2,\cdots i_N}, \tag{2}$$



(a) CP 分解　(b) Tucker分解　(c) 块项 Tucker 分解

(d) 张量列车分解　(e) 张量环分解

(f) 分层 Tucker 分解　(g) 张量网格分解（又称 PEPS）

图 2：一些常见分解的张量网络图。（a）CP格式的张量网络图。它将张量 $\mathcal{X}$ 分解为若干秩一张量之和 $\boldsymbol{a}_{:,r}^{(1)} \circ \boldsymbol{a}_{:,r}^{(2)} \circ \cdots \circ \boldsymbol{a}_{:,r}^{(N)}$。（b）Tucker分解的张量网络图。它将张量 $\mathcal{X}$ 分解为一个核心张量 $\mathcal{G}$ 与一个矩阵相乘 $\boldsymbol{A}^{(n)}$ 沿第 n 个模态。（c）块项分解的张量网络图。它将张量 $\mathcal{X}$ 分解为若干个具有低 Tucker 秩的 Tucker分解之和（右侧）。（d）TT分解的张量网络图。它将张量 $\mathcal{X}$ 分解为一组三阶核心张量 $\mathcal{G}^{(2)} \cdots \mathcal{G}^{(N-1)}$ 与两个矩阵 $\mathcal{G}^{(1)}$，$\mathcal{G}^{(N)}$的线性乘积。（e）张量环分解的张量网络图。它将张量 $\mathcal{X}$ 分解为一组三阶核心张量，并将它们收缩成环状结构。（f）HT分解的张量网络图。它将张量 $\mathcal{X}$ 表示为树状图。有关张量网络的更多基础知识，请参阅 [8] 和 [11]。

其中 $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots I_N \times P_1 \times \ldots P_K}$，$\mathcal{B} \in \mathbb{R}^{P_{K+1} \times \ldots P_{K+M} \times I_1 \times \ldots I_N}$，以及 $\mathcal{C} \in \mathbb{R}^{P_1 \times \ldots P_K \times P_{K+1} \ldots P_{K+M}}$。图1还展示了矩阵乘法运算的示意图，这是最经典的张量收缩情形。其方程表示为：

$$\mathbf{C} = \mathbf{AB} = \mathbf{A} \times_2^1 \mathbf{B}. \tag{3}$$

多个张量（例如张量网络）之间的张量收缩可以通过依次对每对张量进行张量收缩来实现。值得一提的是，必须确定收缩顺序，以提高计算效率 [152]。

2.2.3 哑张量

最近，Hayashi 等人提出了一种新设计的哑张量，用于表示卷积运算 [30]。如图 1 所示，带有星形和箭头符号的节点表示哑张量。该运算可表示为

$$\mathbf{y}_{j'} = \sum_{j=0}^{\alpha-1} \sum_{k=0}^{\beta-1} \mathcal{P}_{j,j',k} \mathbf{a}_j \mathbf{b}_k, \tag{4}$$

其中，$\mathbf{a} \in \mathbb{R}^{\alpha}$ 表示将由卷积权重$\mathbf{b} \in \mathbb{R}^{\beta}$处理的向量，$\mathbf{y} \in \mathbb{R}^{\alpha'}$ 为输出。$\mathcal{P} \in \{0,1\}^{\alpha \times \alpha' \times \beta}$ 是一个二值张量，其元素定义为$\mathcal{P}_{j,\ j',\ k} = 1$ 如果 $j = sj' + k - p$，否则为0，其中 s 和 p分别表示步幅和填充大小。因此，$\mathcal{P}$ 可以应用于任意两个张量，以形成卷积关系。

2.2.4 超边

如图 1所示，我们展示了 Hayashi 等人也曾介绍过的超边 [30]。一个大小为 R 的超边示例可表示为

$$\mathcal{Y}_{ijk} = \sum_{r=1}^{R} \mathbf{A}_{ir} \mathbf{B}_{jr} \mathbf{C}_{kr}, \tag{5}$$

其中 $\mathbf{A} \in \mathbb{R}^{I \times R}$，$\mathbf{B} \in \mathbb{R}^{J \times R}$ 和 $\mathbf{C} \in \mathbb{R}^{K \times R}$ 是三个矩阵。$\mathcal{Y} \in \mathbb{R}^{I \times J \times K}$ 表示对$\mathbf{A}$、$\mathbf{B}$和 $\mathbf{C}$应用超边后的结果。超边节点等价于一个对角元素均为 1 的张量。该张量表示对多个子结构（例如图 1 中的矩阵）执行的加法运算。Hayashi 等人 [30] 证明，通过引入哑张量和超边，张量图可以表示任意张量卷积神经网络。

2.2.5 超对角张量

超对角张量是指主对角线以外的元素均为 0，且各维数相同的张量。具体而言，

一个 N 阶超对角张量 $\mathcal{G} \in \mathbb{R}^{\overbrace{I \times I \times \ldots \times I}^{N}}$是一个张量，其元素定义为 $\mathcal{G}_{i_1,\ i_2,\ \ldots i_N} \in \mathbb{R}$ 若 $i_1 = i_2,\ \cdots = i_N$且0 否则。如图 1所示，在张量网络图中，超对角张量表示为带有斜线的节点。单位张量$\mathcal{J}$ 是一个特殊的超对角张量，其主对角线上的所有元素均为 1。超边可视为通过单位张量执行张量收缩运算。

2.2.6 张量展开

张量展开是一种将张量虚拟地展平为高维但低阶张量的运算。矩阵化是张量展开的一种特殊情况。具体而言，给定一个 N 阶张量 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$，其模态-$n$ 展开过程会生成一个矩阵 $\boldsymbol{A}_{(n)} \in \mathbb{R}^{I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N}$。这种运算也可视为与一个专门设计的张量进行张量收缩。图 1展示了一个四阶张量展开示意图。

2.3 张量分解格式

通常使用的术语“张量分解”（TD）在一定程度上等同于“张量网络”。此前，张量分解主要应用于信号处理领域 [153], [154]。张量网络最初主要应用于物理学和量子电路领域 [8], [151]。传统的张量分解模型，如 CP分解 [13], [14], [15]和 Tucker分解 [16], [17],，可以视为基本的张量网络。 在信号处理领域，人们也提出了几种用于量子分析的强大张量网络架构。例如，MPS分解 [155] 被定义为 TT分解 [19] ，并在多个应用中取得了巨大成功 [12]。经过不同研究领域多年的合作与发展，这两个术语之间已不存在显著区别。因此，本文将张量分解和张量网络统一处理。我们通过张量网络图简要介绍一些基本的张量分解。

2.3.1 CANDECOMP/PARAFAC

CP [13], [14], [15]将高阶张量分解为若干秩一张量组件之和。例如，给定一个 N阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots I_N}$，其在CP格式中的每个元素都可以表示为

$$\mathcal{X}_{i_1, i_2, \ldots, i_N} \approx \sum_{r=1}^{R} \mathcal{G}_r \prod_{n=1}^{N} \mathbf{A}_{i_n, r}^{(n)}, \tag{6}$$

其中 R 表示CP秩（定义为秩一张量 [153]的最小可能数量），$\mathcal{G}$ 表示 N阶超对角张量，$\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$ 表示一系列因子矩阵。CP的张量网络图如图2（a）所示。

计算CP格式时，首先需要解决的问题是如何确定秩一张量组件的数量，即CP秩 R。实际上，这是一个NP难问题 [156]。因此，在实践中，通常会预先假设一个数值（即作为超参数），以拟合各种基于CP的模型 [153]。之后，通过采用算法迭代可以直接求解对角核心张量 $\mathcal{G}$ 和因子矩阵$\mathbf{A}^{(n)}$，该过程通常涉及交替最小二乘法（ALS），该方法最初由 [13], [14]提出。

2.3.2 Tucker分解

Tucker分解 [16], [17] 将高阶张量分解为一个核心张量，并沿每个模态乘以相应的因子矩阵。具体而言，给定一个 N 阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots I_N}$，Tucker分解可按元素形式表示为

$$\mathcal{X}_{i_1, i_2, \ldots, i_N} \approx \sum_{r_1, \ldots, r_N=1}^{R_1, \ldots, R_N} \mathcal{G}_{r_1, r_2, \ldots, r_N} \prod_{n=1}^{N} \mathbf{A}_{i_n, r_n}^{(n)}, \tag{7}$$

其中，$\{R_1, R_2, \ldots, R_N\}$ 表示一系列 Tucker 秩，$\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \ldots R_N}$ 表示核心张量，$\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ 表示因子矩阵。Tucker分解的张量网络图如图2 (b)所示。需要注意的是，与 CP秩相比，R_1、R_2、.....、R_N可以取不同的数值。

Tucker分解通常被采用；将核心张量设置为 $\mathcal{G}$ 超对角张量，还可将其简化为 CP分解。此外，Tucker分解对其因子没有施加约束，因此分解结果并不唯一；由于缺乏可解释性，这通常不利于实际应用。因此，通常会对分量矩阵施加正交约束，从而得到著名的高阶奇异值分解（HOSVD）算法 [157]。

2.3.3 BTT 分解

CP分解和Tucker分解都会将张量分解为一个核心张量，并沿每个模态乘以一个矩阵；其中，CP分解对核心张量施加了额外的超对角约束，以简化核心张量的结构信息。一种更通用的分解方法——BTT 分解 [18]通过对 Tucker 核心张量施加块对角约束，在 CP分解和Tucker分解之间实现折衷。BTT 分解的张量网络图如图2(c)所示。

BTT 分解旨在将张量分解为若干个 Tucker 分解之和，且这些分解的 Tucker 秩较低。具体而言，四阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$ 的 BTT 分解可表示为由 6 个节点构成、带有特殊收缩的结构。这里，$\mathcal{G}^{(1)}$ 表示5阶超对角张量，$\mathcal{G}^{(2)} \in \mathbb{R}^{R_C \times R_T \times R_T \times R_T \times R_T}$ 表示 Tucker 分解的 R_C 核心张量，而每个 $\mathcal{A}^{(n)} \in \mathbb{R}^{R_C \times I_n \times R_T}$表示 Tucker 分解中 R_C 相应的因子矩阵。此外，$\mathcal{X}$ 的每个元素计算如下:

$$\mathcal{X}_{i_1, i_2, i_3, i_4} \approx \sum_{r_C=1}^{R_C} \mathcal{G}_{r_C}^{(1)} \sum_{r_1, r_2, r_3, r_4=1}^{R_T, R_T, R_T, R_T} \mathcal{G}_{r_C, r_1, r_2, r_3, r_4}^{(2)} \mathcal{A}_{r_C, i_1, r_1}^{(1)} \mathcal{A}_{r_C, i_2, r_2}^{(2)} \mathcal{A}_{r_C, i_3, r_3}^{(3)} \mathcal{A}_{r_C, i_4, r_4}^{(4)}, \tag{8}$$

其中，R_T 表示 Tucker 秩（即 Tucker 秩等于 $\{R_T, R_T, R_T, R_T\}$），R_C表示 CP秩。两者合称为 BT 秩。

BTT 分解的优势主要在于兼具 CP 分解和 Tucker 分解的优点。这是因为，当 Tucker 秩等于 1 时，BTT 分解退化为 CP分解；当 CP秩等于 1 时，它退化为 Tucker分解。

2.3.4 TT分解

TT分解 [19], [20], 在量子物理学中也称为MPS分解 [155], [158], ，完全源自张量网络。TT分解将一个高阶张量分解为一系列三阶核心张量的线性乘积。例如，给定一个 N 阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots I_N}$，TT分解可以按元素形式表述为

$$\mathcal{X}_{i_1, i_2, \ldots, i_N} \approx \sum_{r_1, r_2, \ldots, r_{N-1}=1}^{R_1, R_2, \ldots, R_{N-1}} \mathcal{G}_{1, i_1, r_1}^{(1)} \mathcal{G}_{r_1, i_2, r_2}^{(2)} \mathcal{G}_{r_2, i_3, r_3}^{(3)} \cdots \mathcal{G}_{r_{N-1}, i_N, 1}^{(N)}, \tag{9}$$

其中 $\{R_1, R_2, \ldots, R_{N-1}\}$ 表示张量列车秩，$\mathcal{G}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$ 表示一个三阶核心张量，$R_0 = R_N =$

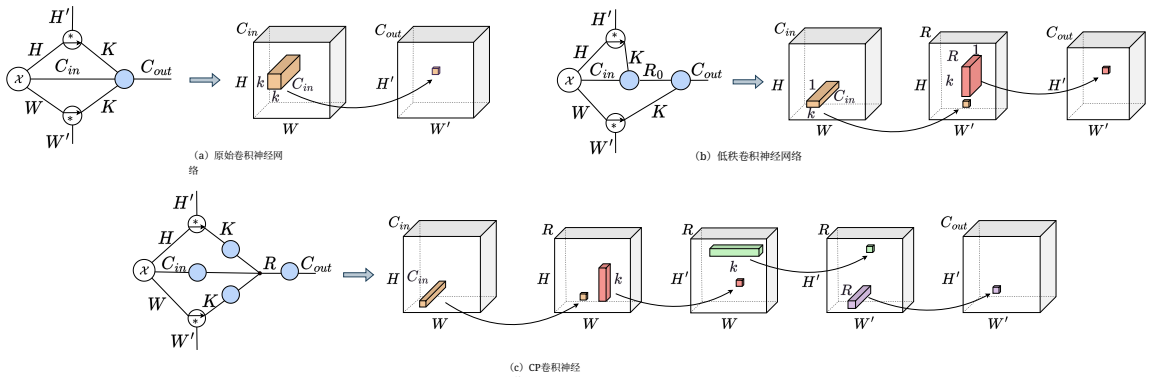


图 3：张量网络图与卷积过程之间的对应关系。在每个子图中，左侧为张量网络图，右侧为相应的常用特征表示。

1，这意味着 $\mathcal{G}^{(1)}$ 和 $\mathcal{G}^{(N)}$ 实际上是两个矩阵。TT分解的张量网络图如图2 （d）所示。

通过递归采用奇异值分解，可以轻松计算 TT分解。此外，作为现有张量网络中最简单的模型，TT分解被广泛应用于张量网络的理论与实践中 [9]。值得注意的是，式（9）和图 2 （d）采用矩阵乘积态格式。一些论文 [50], [63], [69] 还采用了矩阵乘积算子 [159]格式的 TT分解。给定一个 $2N$阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times J_1 \times I_2 \times J_2 \ldots I_N \times J_N}$，其矩阵乘积算子分解可数学表示为

$$\mathcal{X}_{i_1,j_1,i_2,j_2\ldots,i_N,j_N} \approx \sum_{r_1,r_2,\ldots,r_{N-1}=1}^{R_1,R_2,\ldots,R_{N-1}} \mathcal{G}^{(1)}_{1,i_1,j_1,r_1} \mathcal{G}^{(2)}_{r_1,i_2,j_2,r_2} \mathcal{G}^{(3)}_{r_2,i_3,j_3,r_3} \cdots \mathcal{G}^{(N)}_{r_{N-1},i_N,j_N,1}, \tag{10}$$

其中，$\{R_1, R_2, \ldots, R_{N-1}\}$ 表示秩，$\mathcal{G}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times I_n \times R_n}$ 表示一个四阶核心张量，$R_0 = R_N = 1$，这意味着 $\mathcal{G}^{(1)}$ 和 $\mathcal{G}^{(N)}$ 实际上是两个三阶核心张量。

2.3.5 TR分解

TT具有快速收敛的优势。然而，它受限于两个端点，这阻碍了基于张量列车的模型的表示能力和灵活性。因此，为了释放线性架构的潜力，研究人员将其端点连接起来，形成一种称为TR [22], [160], [161] $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots I_N}$ 的环形格式。张量的TR分解可表示为

$$\mathcal{X}_{i_1,i_2,\ldots,i_N} \approx \sum_{r_0,r_1,\ldots,r_{N-1}}^{R_0,R_1,\ldots,R_{N-1}} \mathcal{G}^{(1)}_{r_0,i_1,r_1} \mathcal{G}^{(2)}_{r_1,i_2,r_2} \mathcal{G}^{(3)}_{r_2,i_3,r_3} \cdots \mathcal{G}^{(N)}_{r_{N-1},i_N,r_0}, \tag{11}$$

其中 $\{R_0, R_1, \ldots, R_N\}$ 表示TR秩，每个节点 $\mathcal{G}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$ 都是一个三阶张量，并且 $R_0 = R_N$。与TT分解相比，TR分解在相乘其节点时不必遵循严格的顺序。TR分解的张量网络图如图 2 （e）所示。

2.3.6 HT分解

HT分解 [23]具有树状结构。通常，可以将张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 转移到一棵二叉树上，其中根节点与 $S_{set} = \{1, 2, \cdots, N\}$ 相关联，并以$\mathcal{X} = \mathcal{U}_{S_{set}}$ 作为根框架。 S_{set1}， $S_{set2} \subseteq S_{set}$ 定义为与左子节点 $\mathcal{U}_{S_{set1}}$ 和右子节点 $\mathcal{U}^{S_{set2}}$相关联的集合。$\mathcal{U}^{S_{set1}} \in \mathbb{R}^{R_1 \times I_{\min(S_{set1})} \times \cdots \times I_{\max(S_{set1})}}$ 也可以递归地分解为其左子节点 $\mathcal{U}^{D_{set1}}$ 和右子节点 $\mathcal{U}^{D_{set1}}$。前三个步骤如下

$$\mathcal{U}^{S_{set}} \approx \mathcal{G}^{s} \times_1^2 \mathcal{U}^{S_{set1}} \times_1^2 \mathcal{U}^{S_{set2}}, \tag{12}$$

$$\mathcal{U}^{S_{set1}} \approx \mathcal{G}^{s1} \times_1^2 \mathcal{U}^{D_{set1}} \times_1^2 \mathcal{U}^{D_{set2}}, \tag{13}$$

$$\mathcal{U}^{S_{set2}} \approx \mathcal{G}^{s2} \times_1^2 \mathcal{U}^{D_{set3}} \times_1^2 \mathcal{U}^{D_{set4}}, \tag{14}$$

其中 $\mathcal{G}^{s} \in \mathbb{R}^{R_1 \times R_2}$、$\mathcal{G}^{s1} \in \mathbb{R}^{R_1 \times R_3 \times R_4}$ 和 $\mathcal{G}^{s2} \in \mathbb{R}^{R_2 \times R_5 \times R_6}$。该过程可以递归执行，以获得树状结构。HT分解的张量网络图如图 2 （f）所示。

2.3.7 PEPS分解

还可以考虑具有不同拓扑结构和高维连接的张量网络结构。PEPS分解 [8],[24], [162],也称为张量网格分解 [163],是一种推广张量列车的高维张量网络。PEPS分解提供了一种能够捕获更多高维信息的自然结构。PEPS核心可以表示为 $\mathcal{G}^{(m,n)} \in \mathbb{R}^{I_{mn} \times R_{l_{mn}} \times R_{r_{mn}} \times R_{u_{mn}} \times R_{d_{mn}}}$。数学公式 [48]为

$$\mathcal{X}_{i_1,i_2,\ldots,i_{MN}} = \sum_{h^{(R)},h^{(C)}} \sum_{m,n} \mathcal{G}^{(m,n)}_{i_{mn};h^{(R)}_{l_{mn}},h^{(R)}_{r_{mn}},h^{(C)}_{u_{mn}},h^{(C)}_{d_{mn}}}. \tag{15}$$

指标定义如下

$$\begin{cases} l_{mn} = (n-2)M + m, \\ r_{mn} = (n-1)M + m, \\ u_{mn} = (m-2)N + n, \\ d_{mn} = (m-1)N + n, \\ R_i^{(R)} = 1, \quad \text{while} \quad i < 0 \quad or \quad i > M(N-1), \\ R_i^{(C)} = 1, \quad \text{while} \quad i < 0 \quad or \quad i > N(M-1), \end{cases} \tag{16}$$

其中 M 和 N 分别是张量核心的行数和列数，而 $h_i^{(R)}$ 和 $h_j^{(C)}$ 分别是行方向和列方向上的秩。PEPS分解的张量网络图如图 2 （g）所示。PEPS分解具有随分离距离增加而呈多项式衰减的相关性。相比之下，MPS分解具有指数相关性衰减。这表明，PEPS分解具有更强的表示能力 [8] ，因为它增强了不同张量模态之间的相互作用。

3 使用张量神经网络进行网络压缩

深度神经网络具有极高的空间和时间复杂度，因为深度堆叠的层包含大规模矩阵乘法。因此，深度神经网络通常需要数天时间进行训练，同时在推理过程中占用大量内存。此外，已有研究证明深度神经网络中存在大量权重冗余 [164], ，这表明在保持性能的同时压缩深度神经网络是可行的。受此启发，人们开发了广泛的压缩技术，包括剪枝 [165], 、量化 [166], 、蒸馏 [167] 和低秩分解 [52] 。其中，将张量网络应用于深度神经网络以构建张量神经网络是一个不错的选择，因为张量神经网络能够以少得多的参数近似原始权重 [91]。在这一方向上，研究人员已经完成了许多研究，尤其是通过各种张量分解格式重构卷积层和全连接层 [30], [38], [42], [52]。借助紧凑的架构，这些张量神经网络能够以更少的冗余实现更好的性能。本节介绍五种常见的张量神经网络，即第 3.1节的张量卷积神经网络、第 3.2节的张量循环神经网络、第 3.3节的张量化Transformer、第 3.4节的张量图神经网络，以及第 3.5节的张量化受限玻尔兹曼机。

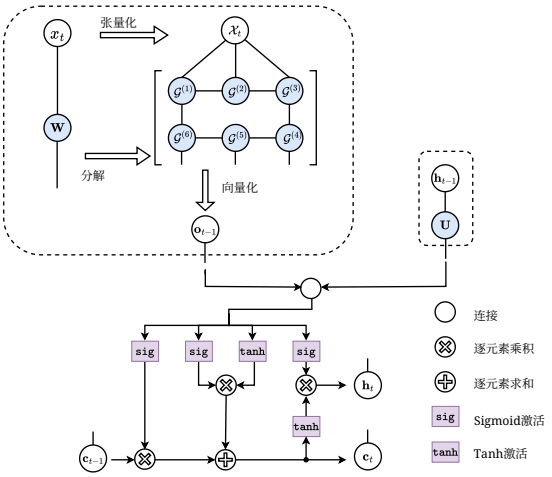


图4：TR长短期记忆网络。该模型通过用张量环分解替换输入到隐藏层变换权重，有效减少了长短期记忆网络模型的参数。

3.1 张量卷积神经网络

卷积神经网络近年来取得了巨大成功。然而，卷积神经网络庞大的规模会导致权重冗余和多余计算，从而影响其性能和效率。张量分解方法可以有效解决这一问题。通常，将采用张量格式表示的卷积神经网络称为张量卷积神经网络。在介绍张量卷积神经网络之前，我们先形式化定义一个原始卷积神经网络，如图 3（a）所示：

$$\mathcal{Y} = \mathcal{X} \circledast \mathcal{C} + \mathbf{b}, \tag{17}$$

其中，$\mathcal{C} \in \mathbb{R}^{K \times K \times I \times O}$ 表示卷积权重，$\mathcal{X} \in \mathbb{R}^{I \times H \times W}$ 表示输入，$\mathcal{Y} \in \mathbb{R}^{O \times H' \times W'}$ 表示输出，$\mathbf{b} \in \mathbb{R}^{O}$ 表示偏置，$\circledast$ 表示卷积算子。K 表示卷积核窗口的大小，I 表示输入通道，H 和 W 分别表示 $\mathcal{X}$的高度和宽度，O表示输出通道，H' 和 W' 分别表示 $\mathcal{Y}$的高度和宽度。张量卷积神经网络主要关注对通道 I和 O 进行分解。具体而言，首先将权重 $\mathcal{C}$重塑为 $\tilde{\mathcal{C}} \in \mathbb{R}^{K \times K \times I_1 \times I_2 \times \ldots I_M \times O_1 \times O_2 \times \ldots O_N}$，其中 $\prod_{k=1}^{M} I_k = I$和 $\prod_{k=1}^{N} J_k = J$。然后，通过对重塑后的卷积核 $\tilde{\mathcal{C}}$进行张量化，可以得到张量卷积神经网络。

为加速卷积神经网络的训练和推理过程，CP- 卷积神经网络 [30], [31], [32], [33], [34] 通过将卷积权重分解为CP格式构建，如图 3（d）所示。CP-CNN仅包含向量作为子组件，因此结构极其紧凑，并具有最高的压缩比。与CP-CNN一样，还可以通过将张量格式应用于卷积权重来实现其他张量卷积神经网络（如图 2中的示例所示）。Tucker分解是一种广泛使用的张量格式，通常应用于卷积神经网络以形成Tucker-CNN [35], [36]。不同于简单的Tucker格式，BTT-CNN具有一个超边 R_c，可表示Tucker分解的求和。其他BTT-CNN [42] 也已被提出。与Tucker卷积神经网络相比，BTT-CNN更为强大，通常能取得更好的结果 [42]。高度紧凑的TT格式也已被引入卷积神经网络，以实现TT-CNN [37], [39], [40]。与张量列车相比，张量环格式通常更加紧凑， [41],而且TR-CNN [41]比TT-CNN更为强大。有趣的是，为了解决张量层中的退化问题，研究人员提出了稳定的CPD-EPC [33] 分解方法，并针对CP卷积层和混合Tucker2-CP卷积层进行了最小敏感性设计。

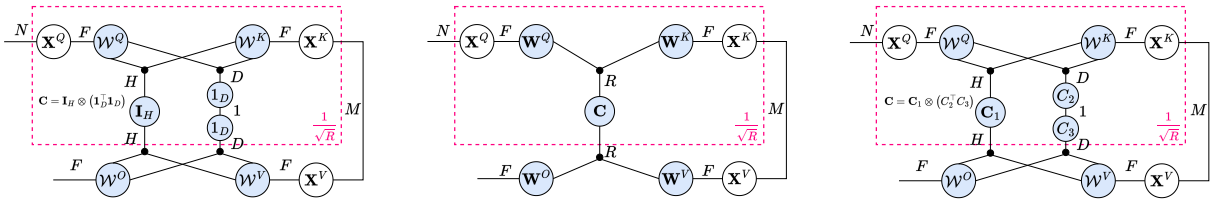
还有一些张量化卷积神经网络分解的不仅仅是卷积核心。张量化网络（T-Net） [46] 将整个网络视为单层架构，再对其进行分解。因此，T-Net以更轻量的结构取得了更好的结果。CP高阶卷积（CP-HOConv） [49] 利用CP格式处理高阶数据任务，例如时空情感估计。针对多任务，Yang等人 [47] 分别采用张量列车格式和Tucker格式，提出了张量列车多任务（TTMT）和Tucker多任务（TMT）模型，以缓解硬共享架构中的负迁移问题，并减少软共享架构中的参数量。此外，研究者还提出了一种用于多任务的类PEPS拼接张量网络层 [48]。与因硬共享架构而面临负迁移问题的TTMT和TMT模型不同，PEPS结构仅包含一个软共享层，因此能够取得更好的性能。

3.2 张量循环神经网络

RNN（如原始RNN和LSTM）在序列数据上取得了良好的性能。然而，在处理高维输入数据（例如视频和文本数据）时，RNN中的输入到隐藏状态变换和隐藏状态到隐藏状态变换会导致较高的内存占用率和计算成本。为了解决这一问题，在实践中，低秩张量分解能够有效压缩变换过程。首先，我们将RNN表示为

$$\mathbf{h}^{(t+1)} = \phi(\mathbf{W}\mathbf{x}^{(t)} + \mathbf{U}\mathbf{h}^{(t)} + \mathbf{b}), \tag{18}$$

其中，$\mathbf{h}^{(t)} \in \mathbb{R}^{O}$ 和 $\mathbf{x}^{(t)} \in \mathbb{R}^{I}$ 分别表示时间 t 的隐藏状态和输入特征，$\mathbf{W} \in \mathbb{R}^{O \times I}$ 是输入到隐藏状态的矩阵，$\mathbf{U} \in \mathbb{R}^{O \times O}$ 表示隐藏状态到隐藏状态矩阵，$\mathbf{b} \in \mathbb{R}^{O}$ 是偏置。$\phi(\cdot)$ 表示一系列构成 RNN变体的运算，包括原始RNN和 LSTM [168]。



(a) 多头自注意力（MHSA）。 (b) 可调头自注意力（THSA）。 (c) 的设计空间。 $\mathbf{C}$

图 5：自注意力模块的张量图 [59]。(a) 可以用张量图表示经典的多头自注意力（MHSA）机制。通过设置 $\mathbf{C} = \mathbf{I}_H \otimes (\mathbf{1}_D^\top \mathbf{1}_D)$，MHSA 可视为可调头自注意力（THSA）的一种特例。(b) Tuformer 的 THSA 是一种更通用的自注意力形式，它通过可训练矩阵 $\mathbf{C}$实现。(c) THSA 的设计空间表示为$\mathbf{C} = \mathbf{C}_1 \otimes (C_2^\top C_3)$，这是 MHSA 的直接推广形式。

公式（18）也可以改写为连接形式，这种形式广泛应用于张量分解：

$$\mathbf{h}^{(t+1)} = \phi([\mathbf{W}, \mathbf{U}][\mathbf{x}^{(t)}, \mathbf{h}^{(t)}] + \mathbf{b}), \tag{19}$$

其中，$[\mathbf{W}、\mathbf{U}] \in \mathbb{R}^{O \times (I+O)}$ 和 $[\mathbf{x}^{(t)}、\mathbf{h}^{(t)}] \in \mathbb{R}^{(I+O)}$ 分别表示 $\mathbf{W}$、$\mathbf{U}$ 和$\mathbf{x}^{(t)}$、$\mathbf{h}^{(t)}$的连接。如图4所示，循环神经网络通常有两种分解方式：(a) 仅对$\mathbf{W}$进行张量化，因为它通常是循环神经网络中最大的组成部分；(b) 对 $[\mathbf{W}、\mathbf{U}]$ 进行张量化，以实现极致压缩。注意，由于$\mathbf{U}$ 通常小于$\mathbf{W}$，因此没有研究仅分解$\mathbf{U}$的工作。实现张量循环神经网络（TRNN）的过程与实现张量卷积神经网络（TCNN）相同，即将权重重塑为高阶形式，并用张量格式替换。

最直接、简单的压缩方法是仅对规模庞大的输入到隐藏状态矩阵 $\mathbf{W}$进行分解。CP-RNN 和 Tucker-RNN [35]分别采用 CP 分解和 Tucker 格式直接构建。凭借极其紧凑的低秩结构，与其他张量格式相比，CP-RNN 始终能够实现最小的规模。TT-RNN [50] 将 TT 格式应用于 RNN，以获得较高的参数压缩率。然而，TT-RNN 具有线性结构，且两端较小，这限制了基于 TT 的模型的表示能力和灵活性。为释放线性架构的潜力，研究者提出了张量环，将两端连接起来，形成环形格式 [22]。TR-An RNN [52] 通过引入张量环构建而成，从而实现更加紧凑的网络。BTT-RNN [42], [51] 基于广义的张量分解方法 BTT 分解 [169]构建而成。BTT-RNN 能够自动学习参数间相关性，从而隐式地剪除冗余的密集连接，同时取得更好的性能。

此外，研究人员还利用TD压缩RNN的两个变换层，并且有些研究甚至开发了同时适用于RNNs和CNNs的分解方法。TT-GRU [55] 和HT-RNN [53] 分解 $[\mathbf{W}、\mathbf{U}]$，以获得更高的压缩比。具体而言，TT-GRU [55]采用TT进行分解，而HT-RNN [53] 采用HT分解。不同于以往对隐藏矩阵进行分解的研究，Conv-TT-LSTM [57] 利用TT的思想来表示卷积运算。如图4所示，通过类似TT的卷积，Conv-TT-LSTM能够以更少的参数替代卷积长短期记忆网络，同时在动作基准测试中取得良好结果。为同时适配CNNs和RNNs，研究人员采用了一种结合HT和TT分解的混合TD方法（称为HT-TT）[45]，以压缩CNN和RNN $[\mathbf{W}、\mathbf{U}]$ 矩阵。此外，张量收缩层（TC-Layer） [44]被设计用于替代全连接层，因此可以用作CNN的最后一层以及RNNs中的隐藏层。有趣的是，TC-Layer是基于张量列车的层的一个特例，其秩被设为1。

3.3 张量化Transformer

Transformer [130], [170] 因擅长处理序列数据而广为人知。与卷积神经网络和循环神经网络相比，Transformer可以堆叠成大规模模型，从而实现显著的性能提升 [131]。然而，与经典深度神经网络类似，Transformer仍然存在冗余，可以进一步缩小规模并提高效率 [62]。因此，可以利用TD这一灵活的压缩工具来减少Transformer中的参数数量 [58], [59], [61]。

经典Transformer主要由自注意力（SA）和前馈网络（FFNs）组成。SA使用参数$\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V, \mathbf{W}^O$处理给定的查询矩阵$\mathbf{Q}$、键矩阵 $\mathbf{K}$ 和值矩阵$\mathbf{V}$ 。更一般地，SA被分为 n 个注意力头：$\mathbf{W}_i^Q\}^n, \{\mathbf{W}_i^K\}^n, \{\mathbf{W}_i^V\}^n, \{\mathbf{W}_i^O\}^n$。每个注意力头可计算为

$$\mathrm{Att}_i(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{W}_i^Q \mathbf{W}_i^{K^T} \mathbf{K}^T}{\sqrt{d}}\right) \mathbf{V}\mathbf{W}_i^V \mathbf{W}_i^{O^T}. \tag{20}$$

然后，$\mathrm{SA}((\mathbf{Q}, \mathbf{K}, \mathbf{V})) = \sum_{i=1}^n \mathrm{Att}_i(\mathbf{Q}, \mathbf{K}, \mathbf{V})$。另一个重要组件FFN的形式为

$$\mathrm{FFN}(\mathbf{X}) = \mathrm{ReLU}(\mathbf{X}\mathbf{W}^{in} + \mathbf{b}^{in})\mathbf{W}^{out} + \mathbf{b}^{out}, \tag{21}$$

其中$\mathbf{X}$ 是输入，$\mathbf{b}^{in}$ 和$\mathbf{b}^{out}$ 是偏置，$\mathbf{W}^{in}$ 和$\mathbf{W}^{out}$ 是权重。显然，Transformer 的参数数量主要取决于其线性变换矩阵，即$\mathbf{W}^Q$、$\mathbf{W}^K$、$\mathbf{W}^V$、$\mathbf{W}^O$、$\mathbf{W}^{in}$ 和$\mathbf{W}^{out}$。

因此，大多数压缩研究都集中于消除这些矩阵的参数。例如，有研究提出了矩阵乘积算子结构，将 Transformer 中的每个矩阵 [59],分解为核心张量（包含核心信息）和较小的辅助张量。随后进一步采用微调策略，在冻结核心张量权重以保留原始矩阵主要信息的同时，继续训练辅助张量，从而提升性能。此外，鉴于低秩矩阵乘积算子结构可能导致性能严重下降，研究者提出了 Hypoformer [63] ，其基于混合张量列车分解；该方法将稠密矩阵部分与低秩矩阵乘积算子部分拼接起来。Hypoformer 在减少压缩和加速基础 Transformer所需的操作数量与参数数量的同时，保留了满秩特性。此外，通过将所有矩阵拼接成一个更大的张量，Tucker-Bert [62]对拼接后的

张量进行Tucker分解，从而大幅减少参数数量，在实现极致压缩的同时保持相当出色的结果。与压缩原始注意力操作相比，多路多模态Transformer（MMT）[60]探索了一种新颖的广义张量注意力操作，用于对多模态数据集中的模态感知的多路相关性进行建模。有趣的是，Tuformer [61] 将多头自注意力推广为Tucker形式，因此具有更强的表达能力并取得了更好的结果，如图5所示。最近，通过使用因果胶囊和控制潜变量交互的Tucker格式张量Transformer，张量因果学习 [64] 也得以付诸实践。

3.4 TGNNs

GNNs 在一系列应用和领域中取得了突破性的性能表现 [171]。经典 GNN 层由一个用于聚合邻居节点信息的聚合函数和一个用于更新当前节点信息的更新函数组成。例如，GNN 第 k 层中节点 v 的处理步骤可以表述为

$$\begin{aligned}\mathbf{a}_v^{(k)} &\leftarrow \text{Aggregate}_{(k)}\left(\left\{\mathbf{h}_u^{(k-1)}, \forall u \in \mathcal{N}(v)\right\}\right),\\ \mathbf{h}_v^{(k)} &\leftarrow \text{Update}_{(k)}\left(\mathbf{h}_v^{(k-1)}, \mathbf{a}_v^{(k)}\right),\end{aligned} \tag{22}$$

其中，$\mathbf{a}_v^{(k)}$ 是聚合后的嵌入向量，$\mathbf{h}_v^{(k-1)}$ 是节点嵌入向量，$\mathcal{N}(v)$ 是邻居节点集合。更新函数通常选择简单的单层感知器，而聚合函数则通常选择简单的求和或最大化操作。由于忽略了节点之间的高阶非线性信息，经典 GNNs 的模型表达能力较低 [65]。鉴于 TGNNs 在表达能力与计算效率之间取得了良好平衡，使用 TGNNs 进行图数据处理十分有益。

为了高效地参数化置换不变多线性映射，以建模无向图结构中邻居之间的交互，TGNN [65] 采用对称CP层作为其节点聚合函数。已有研究表明，TGNN具有表示任意置换不变多线性多项式的强大能力，包括求和函数和均值池化函数。与无向图处理相比，TGNNs更适合高阶图结构，例如知识图谱或多视图图。传统的关系图卷积网络忽略了知识图谱中的三线性交互关系，而是以加法方式组合实体所拥有的信息。TGCN [66]通过采用低秩Tucker层作为聚合函数提出，以提高多线性建模的效率并降低计算空间需求。RTGNN [68],采用Tucker格式结构，在共同特征空间中提取图结构特征，被提出用于捕获多视图图学习任务中的潜在高阶相关信息。TGNNs同样适用于动态时空图处理场景中的高阶相关性建模。例如，DSTGNN [67] 分别应用可学习的TTG和STG模块来发现动态时间关系和空间关系。随后，DSTGNN通过PEPS层探索STG模块与TTG模块之间的动态纠缠相关性，从而减少DSTGNN参数的数量。

3.5 张量化受限玻尔兹曼机

RBMs [126] 是生成式随机神经网络，能够从输入集学习概率分布。标准RBM由一个可见因子 $\mathbf{v} \in \mathbb{R}^M$ 和一个隐藏因子 $\mathbf{h} \in \mathbb{R}^N$ 组成，并为联合向量 $\{\mathbf{v},\mathbf{u}\}$ 赋予如下能量函数：

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{v}^T\mathbf{w}\mathbf{h} - \mathbf{v}^T\mathbf{b} - \mathbf{c}^T\mathbf{h}, \tag{23}$$

其中，$\mathbf{b} \in \mathbb{R}^M$ 和 $\mathbf{c} \in \mathbb{R}^N$ 分别是可见层和隐藏层的偏置，$\mathbf{W} \in \mathbb{R}^{M\times N}$ 是映射权重矩阵。联合向量 $\{\mathbf{v},\mathbf{u}\}$ 的概率分布可以定义为

$$P(v,h) = \frac{1}{Z}e^{-E(\mathbf{v},\mathbf{h})}, \tag{24}$$

其中，Z 是配分函数。随后，可以通过可学习分布形式定义某些损失函数，以优化参数。RBM参数似乎主要基于映射权重矩阵。

因此，大多数压缩研究都集中于减少权重矩阵变量的数量。例如，Tv-RBM [71] 探索了将RBM用于高阶输入。在该模型中，权重矩阵被转换为CP层结构，其中每个可见层表示为一个张量，而每个隐藏层仍表示为一个向量。在另一种高阶RBM，即Mv-RBM [72],中，其可见层和隐藏层都表示为矩阵，其权重则表示为TC层 [44]。MPO-RBM [172] 和TT-RBM [69]使用TT层表示权重矩阵，从而大幅压缩所需参数的数量。此外，TR-RBM [70] 对其RBM执行张量环分解，其中TR-RBM的可见层和隐藏层都被推广为张量。

备注。 紧凑型张量神经网络已展现出在保持模型性能的同时实现极高压缩率的潜力。然而，与其压缩率相比，其计算加速比并不显著，这主要是由收缩运算造成的。因此，有必要进一步研究如何改进所采用的收缩策略，因为未经优化的收缩策略可能导致运行时内存消耗不理想。

4 通过张量神经网络进行信息融合

在现实世界的数据分析中，所收集的数据可能来自多个来源；例如，视频数据中可以包含视觉、声音和文本来源 [145], [173]。以视觉问答任务为例，关键在于有效建模文本与图像信息这两种模态之间的交互。在处理此类数据时，以相同形式考虑各种来源是不可行的。因此，理想的做法是通过多个入口混合这些信息，以特殊的构建结构处理多模态来源。具有此类入口的方法称为信息融合方法。特征级融合 [174] 和决策级融合 [175] 是早期阶段常用的方法。然而，这些方法属于简单的线性方法，无法有效建模模态内动态。为解决这一问题，张量神经网络被用于融合任务，基于其天然的多线性性质对模态内动态进行建模。此外，张量神经网络能够处理高阶数据，这是一项得到广泛应用的能力。总之，张量网络为张量运算提供了有效的框架，因此，对所遇到的信息融合模块（如注意力模块和向量拼接模块）进行表达和泛化是自然且有意义的

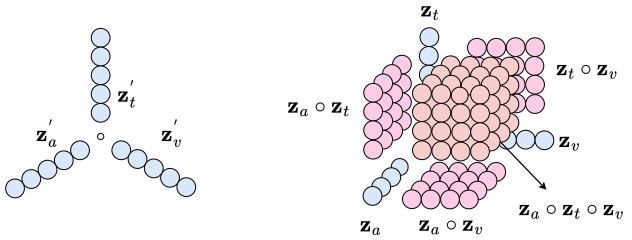


图6：公式（26）中张量融合过程的示意图。与张量网络图不同，每个圆圈对应一个值。

在深度学习中通过张量网络 [176]。因此，许多研究采用张量神经网络来捕获数据或参数之间的高阶交互。在本节中，我们介绍用于信息融合的两大类张量神经网络结构：第4.1节中的张量融合层，以及第 4.2 节中的多模态池化。

4.1 基于张量融合层的方法

多模态情感分析是一项包含三种交流模态的任务，即文本模态、视觉模态和声学模态 [73]。针对多模态情感分析，Zadeh等人 [73] 提出了带有名为张量融合层（TFL）的深度信息融合层的新型TNNs。该模型能够轻松学习模态内动态和模态间动态，并且能够聚合多模态交互，从而高效地融合这三种交流模态。具体而言，TFL首先接收由嵌入网络得到的嵌入特征向量$\mathbf{z_t}$、$\mathbf{z_v}$ 和$\mathbf{z_a}$ ，而不是原始的三种数据类型。然后，TFL将一个标量1 分别与每个嵌入特征向量进行拼接：

$$\mathbf{z}_t^{'} = \begin{bmatrix} \mathbf{z}_t \\ 1 \end{bmatrix} \mathbf{z}_v^{'} = \begin{bmatrix} \mathbf{z}_v \\ 1 \end{bmatrix} \mathbf{z}_a^{'} = \begin{bmatrix} \mathbf{z}_a \\ 1 \end{bmatrix}. \tag{25}$$

然后，如图6所示，TFL通过计算三个拼接向量之间的外积，得到一个特征张量 $\mathcal{Z}$：

$$\mathcal{Z} = \mathbf{z}_t^{'} \circ \mathbf{z}_v^{'} \circ \mathbf{z}_a^{'} = \begin{bmatrix} \mathbf{z}_t \\ 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{z}_v \\ 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{z}_a \\ 1 \end{bmatrix}. \tag{26}$$

最后，TFL通过两层全连接神经网络处理特征张量 $\mathcal{Z}$ 以获得预测结果$\mathbf{y}$ 。与仅考虑单模态交互的直接基于连接的融合相比 [73], ，TFL得益于同时捕获单模态交互和多模态交互。

尽管取得了成功，但随着模态数量的增加，TFL的计算复杂度和参数数量呈指数增长。例如，在多模态情感分析案例中 [73]，特征张量 $\mathcal{Z} \in \mathbb{R}^{129\times 33\times 33}$和隐藏向量$\mathbf{h} \in \mathbb{R}^{128}$ 可能导致需要优化17,981,568个参数。为解决参数过多的问题，低秩多模态融合（LMF） [75]采用特殊的BTT层，以克服TFL带来的巨大计算成本和过拟合风险。对于包含 n 个模态的一般情况，可以处理特征张量 $\mathcal{Z} = \circ_{m=1}^{M} z_m^{'}$。隐藏向量$\mathbf{h}$ 可计算如下：

$$\mathbf{h} = ReLU\left(\boldsymbol{he}(\mathbf{W}_1 z_1^{'}, \mathbf{W}_2 z_2^{'}, \cdots \mathbf{W}_M z_M^{'}, I) + \mathbf{b}\right),$$

其中$\mathbf{W}_i \in \mathbb{R}^{d_i \times d_h}$ 是权重矩阵， $I \in \mathbb{R}^{d_h \times d_h}$ 是单位矩阵。LMF 将 TFL 的计算复杂度从 $O\left(\prod_{m=1}^{M} d_m\right)$ 降低到 $O\left(d_h \times \sum_{m=1}^{M} d_m\right)$。

尽管 LMF 和 TFL 比其他方法实现了更好的融合效果，但它们限制了交互的阶数，导致高阶交互缺乏信息。为解决这一问题，研究人员提出了一个 PTP [76] 模块。PTP 的完整流程和张量网络图如图 7和图 8所示。

PTP首先将所有特征向量 $\{\mathbf{z}_m\}_{m=1}^{M}$ 合并为一个长特征向量

$$\mathbf{z}_{12\cdots M}^{\top} = \left[1, \mathbf{z}_1^{\top}, \mathbf{z}_2^{\top}, \cdots, \mathbf{z}_M^{\top}\right]. \tag{27}$$

次数为 P 的多项式特征张量表示为

$$\mathcal{Z}^P = \mathbf{z}_{12\ldots M} \circ \mathbf{z}_{12\ldots M} \circ \cdots \circ \mathbf{z}_{12\cdots M}. \tag{28}$$

PTP [76] 随后采用张量层（例如CP层）来处理多项式特征张量 $\mathcal{Z}^P$。CP层表示为

$$\mathbf{h} = \boldsymbol{he}(\mathbf{W}_1 \mathbf{z}_{12\ldots M}, \cdots \mathbf{W}_P \mathbf{z}_{12\ldots M}, \mathbf{\Lambda}) \tag{29}$$

其中$\mathbf{W}_i \in \mathbb{R}^{d_i \times d_h}$ 是权重矩阵， $\mathbf{\Lambda} \in \mathbb{R}^{d_h \times d_h}$是可学习对角矩阵。PTP的结构也等价于深度多项式神经网络 [177]的结构。PTP对所有非线性高阶交互进行建模。对于多模态时间序列数据，一种方法是使用“窗口”来刻画局部相关性，并在多个层中堆叠PTP模块。这样的模型称为分层多项式融合网络（HPFN） [76]。HPFN可以递归地处理局部时间模态模式，从而实现更好的信息融合效果。

单层PTP模块的结构类似于浅层卷积算术电路（ConvAC）网络的结构 [83]（见第5.3节）。ConvAC与PTP之间唯一的区别在于，标准ConvAC网络处理量子位置特征，而PTP处理时间模态模式和多项式拼接多模态特征。HPFN几乎等价于更深的ConvAC网络，其强大的表达能力可能正是由二者之间的联系所体现的。深度多项式神经网络中的递归关系也已被发现并实现，因此可以通过分层神经网络高效计算多项式输入 [76]。Chrysos等人 [177] 也发现了类似的结果。

4.2 基于多模态池化的方法

另一组信息融合方法源自视觉问答任务 [145]。在视觉问答任务中，最重要的是对视觉表示与文本表示之间的双线性交互进行参数化。为了解决这一问题，该领域发现了一些张量融合方法。多模态紧凑双线性池化（MCB） [78] 是一种用于视觉问答任务的知名融合方法，可被视为一种特殊的基于Tucker分解的神经网络。MCB试图优化简单的双线性融合操作

$$\mathbf{z} = \mathbf{W}[\mathbf{v} \circ \mathbf{q}], \tag{30}$$

其中，$\mathbf{v}$ 和$\mathbf{q}$ 是具有不同模态的输入向量，而$\mathbf{W}$ 是可学习的权重矩阵。此外，MCB根据计数草图投影函数的性质，优化了外积操作的计算成本。

多模态低秩双线性池化（MLB） [79] 在数据融合步骤中采用CP层，可形式化表示如下：

$$\mathbf{z} = \mathbf{1}^T \left(\mathbf{W}_v \mathbf{v} \circ \mathbf{W}_q \mathbf{q}\right), \tag{31}$$

其中，$\mathbf{W}_q$ 和$\mathbf{W}_v$ 分别是输入$\mathbf{q}$ 和$\mathbf{v}$的预处理权重矩阵，而$\mathbf{1}$ 是一个所有值均为 1 的向量。

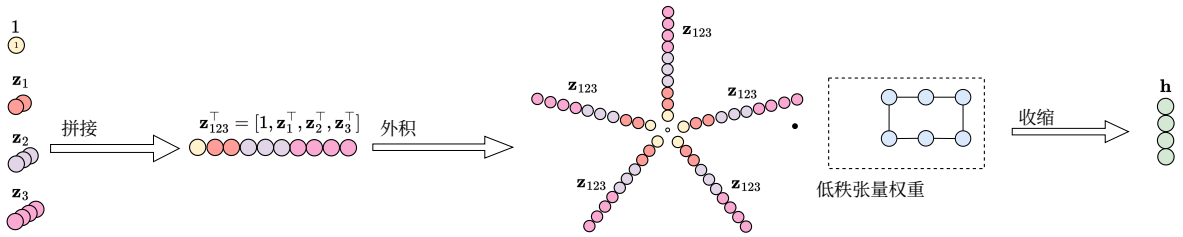


图7：多项式张量池化（PTP）的示意图 [76]。PTP首先将所有特征向量$\mathbf{z}_1,\mathbf{z}_2,\mathbf{z}_3$拼接成一个更长的特征向量$\mathbf{z}_{123}^\top = [1, \mathbf{z}_1^\top, \mathbf{z}_2^\top, \mathbf{z}_3^\top]$，然后通过对特征向量$\mathbf{z}_{123}$反复执行外积操作，得到多项式特征张量，最后采用张量层（例如TR层）将多项式特征张量合并为向量$\mathbf{h}$。

MLB方法的结构是 LMF 的一个特例（见第 4.1节）。当模态数量等于二时，MLB 融合方法也可以视为简单的乘积池化。

MUTAN [77] 是MCB和MLB的推广。MUTAN采用Tucker层来学习视觉特征与文本特征之间的双线性交互：

$$\begin{aligned}\mathbf{z} &= \left(\left(\mathcal{T}_c \times_1^1 \left(\mathbf{q}^\top \mathbf{W}_q\right)\right) \times_2^1 \left(\mathbf{v}^\top \mathbf{W}_v\right)\right) \times_3^1 \mathbf{W}_o, \\ \mathbf{z} &= \left(\mathcal{T}_c \times_1^1 \tilde{\mathbf{q}}\right) \times_2^1 \tilde{\mathbf{v}}, \end{aligned} \tag{32}$$

其中，$\tilde{\mathbf{q}} = \tanh\left(\mathbf{q}^\top \mathbf{W}_q\right)$ 和 $\tilde{\mathbf{v}} = \tanh\left(\mathbf{v}^\top \mathbf{W}_v\right)$, $\mathcal{T}_c$是融合权重张量，$\mathbf{W}_o$ 是输出处理权重矩阵。此外，MUTAN [77] 对融合权重张量 $\mathcal{T}_c$采用低秩分解，如下所示：

$$\mathcal{T}_c[:,:,k] = \sum_{r=1}^{R} \mathbf{m}_r^k \circ \mathbf{n}_r^{k\top}, \tag{33}$$

其中，$\mathbf{m}_r^k$ 和$\mathbf{n}_r^{k\top}$是权重向量，R 是秩的数量。MUTAN通过将交互张量分解为可解释的元素，在保持合理模型规模的同时，能够表示全面的双线性交互。

此外，紧凑三线性交互（CTI） [80] 被提出用于采用类注意力结构。该方法不将给定数据表示为单个向量，而是将每个模态表示为一个矩阵 $A \in \mathbb{R}^{n_1 \times d_a}$，其中 d_a 对应特征维度，n_1 表示状态数量。CTI同时在VQA任务中学习高层三线性联合表示，并克服了三线性交互学习中的计算复杂度和内存问题 [80]。

备注。在许多多模态任务中融合信息时，TNNs凭借天然的多线性和紧凑框架，能够取得令人期待的结果。然而，TNNs中使用的高阶外积操作可能导致计算复杂度意外增加，甚至造成数值特性不稳定。因此，考虑采用高效算法来降低内存消耗，并使用可行的初始化算法（例如 [92]）以获得良好的稳定性，这一点十分重要。

5 使用TNNs进行量子电路模拟

过去几年，量子计算理论的发展引起了广泛关注 [178], [179], [180]。与经典计算相比，量子系统在并行性方面具有优势

电子计算机 [181], [182], 从而能够实现时间复杂度更低的算法。例如，基于量子系统的 Shor算法 [183]在理论上比经典素数分解算法快一个指数级。量子电路是量子

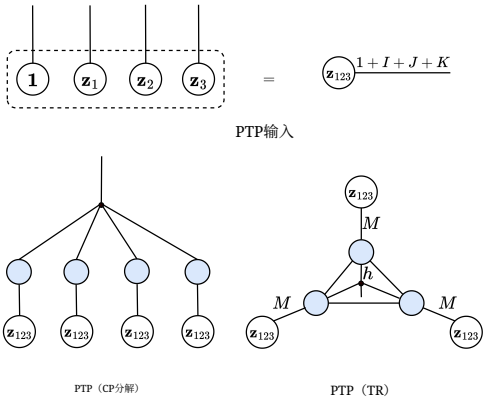


图8：PTP的张量网络图。在此策略中可以采用CP分解和TR结构。

系统的计算硬件实现，在理论上对应于张量网络和张量神经网络 [184],[185], [186], [187]。量子态是量子系统的数学实体，与满足一定约束的高阶张量一致 [8]。因此，张量神经网络可以作为经典计算机中的模拟器，用于建模真实的量子电路 [8],[148]。利用量子计算的超高并行性，一些特殊的张量神经网络可以在小型、近端量子设备上实现 [185]。基于张量神经网络的量子电路模拟主要关注张量网络作为经典神经网络与量子神经网络之间桥梁的作用，而不是更为一般的基于张量网络的量子电路模拟范式。如读者对通过张量网络进行一般电路模拟感兴趣，请参阅其他论文 [8], [26], [148],[188] 。在本节中，我们使用“经典数据”一词来表示经典电子计算机中的数据。我们将在第5.1节介绍通过张量网络将经典数据映射到量子态的方法，随后在第5.2节介绍针对所映射量子态的基本监督处理和无监督处理方法，最后在第5.3节介绍著名的量子张量神经网络模型，即 ConvAC。

5.1 经典数据的量子态嵌入

为了在量子系统中处理机器学习任务，应将输入数据转换为若干量子态的线性组合，并以这些量子态作为正交基：

$$\begin{aligned} |\psi\rangle &= \sum_{d_1 \ldots d_N = 1}^{M} \mathcal{A}_{d_1 \ldots d_N} |\psi_{d_1}\rangle \circ \cdots \circ |\psi_{d_N}\rangle, \\ s.t \quad & \sum_{d_1 \ldots d_N = 1}^{M} \mathcal{A}_{d_1 \ldots d_N}^2 = 1, \quad \mathcal{A}_{d_1 \ldots d_N} \geq 0, \end{aligned} \tag{34}$$

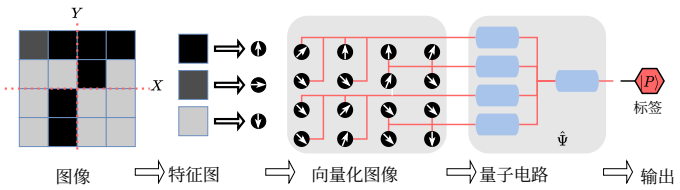


图 9：量子嵌入数据所采用的处理过程 [191]。量子电路可以通过张量神经网络在经典电子计算机上进行模拟，并且某些特殊的张量神经网络（例如 ConvAC）从理论上也可以在真实的量子电路上实现。

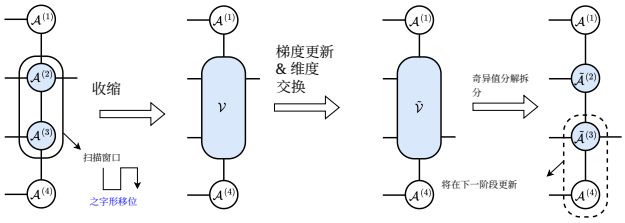


图10：Sweeping 方法的一个阶段 [81]。在每个阶段中，Sweeping 仅更新扫描窗口中的节点，该窗口沿之字形轨迹移动。

其中，$|\cdot\rangle$ 是复数值向量的狄拉克符号，[189], 和 $\circ$ 表示外积操作。张量$\mathcal{A}$ 是组合系数张量，通常通过低秩张量网络 [8] 来表示和分析。为了将经典数据嵌入量子态以适配量子系统，Stoudenmire 和 Schwab [81] 提出了一种量子态映射函数 $\phi^i(x_i)$，用于灰度图像中的第 i个像素 x_i，其形式为

$$\phi^i(x_i) = [\cos(\frac{\pi}{2}x_i), \sin(\frac{\pi}{2}x_i)]. \tag{35}$$

通过映射函数，可将像素值转换到 0.0 至 1.0 的范围内。此外，一幅完整的灰度图像 $\mathbf{x}$可以表示为各个像素所对应的映射量子态的外积：

$$\Phi^{1,2,\cdots N}(\mathbf{x}) = \phi^1(x_1) \circ \phi^2(x_2) \circ \cdots \phi^N(x_N), \tag{36}$$

其中 $\Phi^{1,\ 2,\ \ldots N}(\mathbf{x}) \in \mathbb{R}^{2\times 2\cdots\times 2}$。通过公式（36），可以将真实图像与真实量子系统建立关联。

对于自然语言文档，第 i 个词 $|x_i\rangle$ 也可以表示为正交量子态基的和$|\phi_{h_i}\rangle\,(h_i = 1,\ \ldots\ldots,\ M)$ [82], [83], [84], [190]，这些基对应于特定的语义含义 M:

$$\begin{aligned} |x_i\rangle &= \sum_{h_i=1}^{M} \alpha_{i,h_i} |\phi_{h_i}\rangle\,, \\ s.t \quad &\sum_{h_i=1}^{M} \alpha_{i,h_i}^2 = 1, \quad \alpha_{i,h_i} \geq 0, \end{aligned} \tag{37}$$

其中 $\alpha_{i,\ h_i}$ 是每个语义含义对应的组合系数。的约束条件为和。完成数据映射后，嵌入式量子数据可以在如图9所示的真实量子电路上由张量神经网络进行处理。张量神经网络的损失函数也可以通过量子电路的性质进行定义。这样的过程可以通过张量网络在经典电子计算机上进行模拟，并且理论上可以在真实量子系统上高效实现。

5.2 嵌入式量子数据处理

两类学习方法对于设计和优化可在真实量子电路上实现的张量神经网络（TNNs）十分重要且具有很大潜力。其中一类涉及应用密度矩阵重整化群（DMRG）算法 [8], [192] 来训练监督模型。另一类则采用Born机 [193] 的思想，通过无监督过程学习数据分布。我们将在下一部分介绍这些方法。

5.2.1 监督式张量网络模型

监督模型用于根据示例输入-输出对，对给定输入特征时标签（输出）的条件概率分布进行建模。以嵌入式量子数据作为输入，Stoudenmire和Schwab [81] 提出了监督式类MPS张量多线性模型，并采用类DMRG算法优化模型权重。在介绍具体实现之前，必须首先将其模型表述为优化一组按不同标签索引的函数的过程 ℓ:

$$f^{\ell}(\mathbf{x}) = \mathcal{W}^{\ell} \times_{1,2\cdots N}^{1,2\cdots N} \Phi(\mathbf{x}), \tag{38}$$

其中 $\Phi\,(\cdot)$ 是公式（36）中的特征映射函数，且 $\mathcal{W}^{\ell} \in \mathbb{R}^{\overbrace{2\times 2\cdots\times 2}^{N}}$ 是权重张量。然后，通过将 $\mathcal{W}^{\ell}$ 替换为矩阵乘积态张量网络，可以得到上述张量模型：

$$\mathcal{W}^{\ell}_{s_1 s_2 \cdots s_N} = \sum_{\{\alpha\}} \mathcal{A}^{(1)}_{s_1,\alpha_1} \cdots \mathcal{A}^{(m)}_{\alpha_{m-1},s_m,\alpha_m} \cdots \mathcal{A}^{(N)}_{\alpha_{N-1},s_N}, \tag{39}$$

其中 $\mathcal{A}^{(1)} \cdots \mathcal{A}^{(N)}$ 为核心张量。此外，作者提出了一种名为 Sweeping 的优化算法，该算法的灵感来自量子力学中的 [8] DMRG算法。该优化算法沿着一个 MPS 进行扫描，以优化二次代价函数：

$$c = \frac{1}{2} \sum_{n=1}^{N_T} \sum_{\ell} \left(f^{\ell}\left(\mathbf{x}_n\right) - \mathbf{y}_{n\ell} \right)^2, \tag{40}$$

其中 N_T 表示训练样本的数量，且 $\mathbf{y}_n$ 表示 $\mathbf{x}_n$的真实独热标签向量。优化过程通过随机梯度下降分阶段执行，以最小化该代价函数。图 10展示了其中一个阶段。在每个阶段，两个 MPS 张量 $\mathcal{A}^{(2)}$ 和 $\mathcal{A}^{(3)}$ 通过张量收缩合并为一个键张量 $\mathcal{V}$。随后，张量 $\mathcal{V}$ 通过梯度进行更新~。最后，$\mathcal{V}$ 使用 SVD算法分解回相互独立的张量。Sweeping 方法能够高效优化输入为量子嵌入数据的模型，也可用于训练 TNNs。此外，包括 PEPS结构 在内的其他张量网络 [194], 也可以采用类似 Sweeping 的方法进行处理。与这些针对量子嵌入数据的类 Sweeping 优化过程不同，Ivan 等人 [85] 直接使用随机梯度下降来优化所提出的广义张量网络。

5.2.2 无监督张量网络模型

无监督生成建模的目标是对给定数据的联合概率分布进行建模。生成对抗

网络（生成对抗网络） [195]和变分自编码器（VAE） [196]是用于处理经典数据分布的成功神经网络模型。嵌入式量子数据是训练和设计量子张量神经网络的关键，可通过张量网络生成概率分布。受量子力学中对量子态的概率解释启发，[197], 提出了一种基于矩阵乘积态的生成模型，称为 Born机 [193] 。Born机是一种基于能量的模型， [198] 源自量子力学。Born机的分布函数如下所示：

$$P(\mathbf{x}) = \frac{|\Psi(\mathbf{x})|^2}{Z}, \tag{41}$$

其中， $Z = \sum_x |\Psi(x)|^2$ 是归一化因子， $\Psi(\cdot)$是量子态嵌入函数；从量子力学的角度来看，$\mathbf{x}$ 的能量函数可表示为 $|\Psi(\mathbf{x})|^2$ 。$\Psi(\mathbf{x})$可以通过张量网络格式 [193]进行参数化。学习过程也可以通过类DMRG算法和规范变换策略 [12]来完成。式（41）中的分布定义对于设计量子张量神经网络的损失函数也很有用。此外，人们还提出了一系列Born机结构，包括树张量网络 [86]、、用于语言生成的均匀矩阵乘积态 [84]、、局部纯化态 [87]以及自回归矩阵乘积态 [88]。未来，预计通过特定的量子视角来表示和处理数据，将为在现实量子计算机上进一步实现张量神经网络铺平道路 [86]。

5.3 ConvAC网络

第5.2.1节中先前开发的量子数据处理模型（例如矩阵乘积态模型）以及第5.2.2节中的Born机，其表达能力都受到缺乏非线性的限制。经典非线性算子，例如激活函数（如修正线性单元（ReLU）函数）以及平均池化/最大池化，可以显著提升模型性能。然而，经典非线性无法直接在量子电路中实现。为解决这一问题，ConvAC网络 [89], [199] 被提出，用于采用可在量子系统中部署的乘积池化作为非线性算子，从而证明ConvAC可以转换为带有ReLU激活以及平均池化/最大池化的卷积网络。

ConvAC的整体结构可以用HT格式表示，并且已从理论上证明能够部署于现实量子系统中。ConvAC的张量图示例如图11所示，ConvAC的一个隐藏层采用CP格式。ConvAC还可以处理语言数据 [83] ，通过公式（37）将自然语言句子映射为量子态。ConvAC是一个里程碑，因为深度卷积网络及其非线性模块得以在量子电路上实现。它为将更多神经网络集成到量子系统中提供了启发。例如，Zhang等人 [90] 提出了张量空间语言模型（TSLM），已证明该模型是n元语言模型的推广。

备注。量子张量神经网络的实现是一个具有象征意义的里程碑，它通过张量网络在量子神经网络与经典神经网络之间搭建了桥梁。然而，关于模拟量子张量神经网络与现实物理系统之间的严格映射算法，仍有待进一步探索。此外，由于真正的高性能量子计算机距离开发完成尚有很长一段时间，在不久的将来验证量子张量神经网络性能的设想并不可行。尽管存在这些问题，关注量子神经网络的映射算法与性能验证仍然十分重要且有意义。

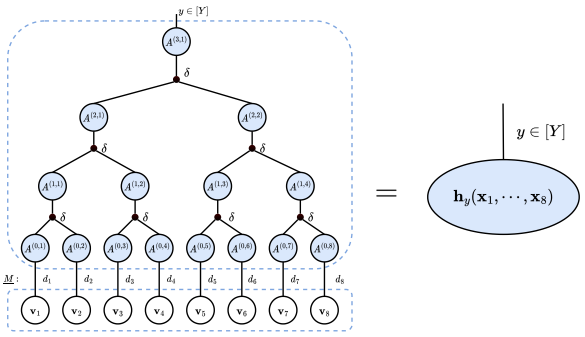


图11：ConvAC等价于类似HT的张量网络 [185]。$\mathbf{x}_1$，……，$\mathbf{x}_N \in \mathbb{R}^s$。$\mathbf{x}_1$ 对应于输入图像或特征图中的一个局部区域，而 $v^{(0,j)}$ 是$\mathbf{x}_j$的线性变换。单个模块的宽度为2。值得注意的是，单层等价于CP格式。

6 张量神经网络的训练策略

尽管前述张量神经网络能够在各种任务和机器上表现良好，但探索具有更高稳定性、更佳性能和更高效率的训练策略同样值得关注。本节将此类策略分为三组进行介绍；（1）用于稳定张量神经网络训练过程的策略见第6.1节，（2）用于选择和搜索张量神经网络秩的策略见第6.2节，以及（3）用于应用硬件加速的策略见第6.3节。

6.1 稳定训练方法

尽管取得了各种成功，张量神经网络仍然受到训练问题的困扰，这是由其多线性特征所导致的。与矩阵乘法等简单的线性运算相比，当模态数量线性增加时，张量收缩会产生具有指数级规模的数据流，即前向传播中的特征和反向传播中的梯度 [92]。一种解决方案是采用全精度 float64 格式来表示较大的权重，这在一定程度上可以缓解这些数值问题。然而，与较低精度格式（例如 float16）相比，全精度格式会导致更多的计算量和更高的时间消耗。尽管如此，低精度仍可能引发数值稳定性问题。为了解决这些问题，Panagakis 等人 [91] 提出了混合精度策略，以实现折中。这种动态精度策略能够有效减少内存占用，并提高训练稳定性。

解决训练问题的另一种可行方法在于制定合适的初始化方法。常用的自适应初始化方法包括 Xavier [202] 初始化和 Kaiming [203] 初始化，它们能够调节各层数据流的方差。然而，由于忽略了张量收缩中的交互作用，这两种初始化方法无法计算出 TNNs 的正确尺度。此外，不同张量格式彼此各异，因此难以针对多样化的张量层制定通用的拟合初始化方法。为解决这两个问题，Yu初始化 [92] 提出了一种基于 Xavier 的统一初始化范式，可对任意 TCNNs 进行自适应初始化。具体而言，Pan 等人从张量卷积超图 [30] 中提取出一个骨干图（BG），然后利用该 BG 将任意 TCNN 编码为邻接矩阵。最后，可通过邻接矩阵直接计算出 TCNN 的合适初始方差。图中展示了使用统一初始化的三种情况。12。尽管Yu初始化是

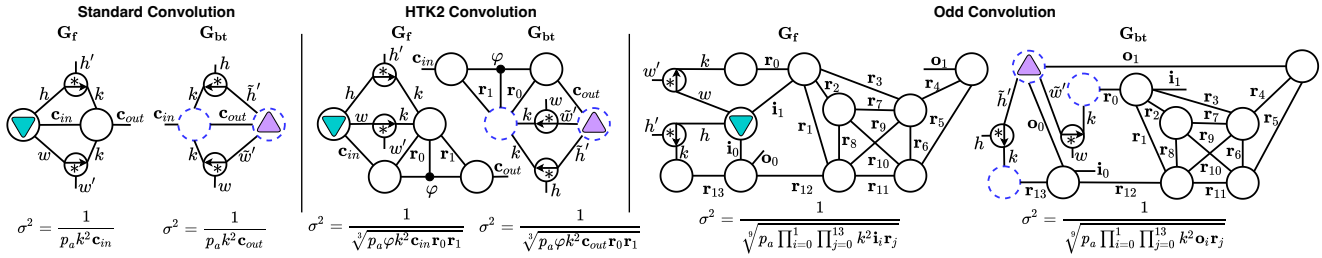


图12：统一TCNN初始化的三种情况 [92]。 σ^2 表示每个权重顶点的初始方差。$\mathbf{G}f$ 表示前向过程，$\mathbf{G}_b$ 表示反向过程。（i）标准卷积。对于相同的权重方差公式， [92] 中的方法在标准卷积上退化为Xavier/Kaiming初始化。（ii）超Tucker-2（HTK2）卷积。Tucker-2（TK2）是一种常见的TD，在ResNet中被用作瓶颈模块 [200]。HTK2通过将超边应用于TK2的权重顶点而形成。（iii）奇异卷积。奇异TD最初由 [201]提出。顶点之间的连接是不规则的，因此权重初始化成为一个复杂问题。这三种成功的初始化情况能够更好地展示统一初始化对多样化TCNN的潜在适应性。

为 TCNNs 提出的，但由于这些模型的基本层，即卷积层和线性层，均属于 TCNNs 的范畴，因此也可以广泛应用于其他 NNs，包括卷积神经网络、循环神经网络和 Transformer。

6.2 秩选择与搜索

先前的研究 [38], [42], [52] 主要致力于寻找高效的TN格式（例如TT和TR），以压缩神经网络，并凭借其天然紧凑结构实现了显著的效率提升。然而，尽管取得了这些显著成果，用于调整或选择TN合适秩的高效算法仍然缺乏，因为秩选择是一个NP难问题 [156]。因此，许多方法 [35], [41],[42], [50] 只能手动为所有秩设置数值，这会严重影响所得模型的训练过程。幸运的是，秩选择问题仍然可以通过启发式策略进行优化，例如贝叶斯优化 [96], [97], 强化学习（RL） [94]和进化算法（EAs） [93]。下面介绍一些用于TNN的秩选择方法。

深度神经网络利用神经架构搜索（NAS） [204] 来搜索最优的网络超参数，并取得了显著成功。由于秩可以被视为架构超参数，因此 NAS 适用于搜索具有更优秩设置的张量层。基于这一思路，渐进式搜索张量环网络（PSTRN） [93]采用结合进化算法（EA）的 NAS，为张量环网络（TRN）选择合适的秩。具体而言，PSTRN 采用一种启发式假设进行搜索：“当形状固定的 TRN 表现良好时，其部分或全部秩元素具有敏感性，并且每个秩元素都倾向于聚集在一个狭窄区域内，该区域称为感兴趣区域。”在感兴趣区域假设的指导下，与普通 EA 方法相比，PSTRN 以更高的概率达到最优点。PSTRN 由进化阶段和渐进阶段组成。在进化阶段，该方法在基准数据集上验证搜索空间中的秩，并选取得到最佳性能的秩。随后，在渐进阶段，PSTRN 在先前选定的秩附近采样新的秩，并将其插入新的搜索空间。经过若干轮后，启发式 EA 能够找到高性能解。凭借这种高效的设计，PSTRN 成功实现了优于手动设置的性能，表明其假设具有实用性。

除了神经架构搜索（NAS）之外，还有一些其他高效的秩选择方法。Zhao等人 [95] 通过对较大的秩值执行缩减过程，推断出一个CP秩，并采用变分贝叶斯优化过程。Hawkins和Zhang [97] 将这一CP方法 [95] 扩展到基于TT的张量神经网络，并采用Stein变分梯度下降方法。该方法结合了马尔可夫链蒙特卡罗（MCMC）方法的灵活性与变分贝叶斯推断的速度，从而构建了一种贝叶斯优化方法。在预训练网络中，Kim等人 [100] 和Gusak等人 [101]通过对权重张量展开进行贝叶斯矩阵分解（BMF） [205] 来推导近似秩。Konstantin等人 [96] 采用基于代理的贝叶斯优化方法，为神经网络压缩寻找最佳秩组合。与贝叶斯方法不同，Cheng等人 [94] 将秩搜索任务视为一个博弈过程，其搜索空间不规则，因此采用强化学习来为已训练的卷积神经网络寻找相对合适的秩。然而，该算法依赖于张量分解，这意味着其性能可能受到所选张量分解方法的影响。Yin等人 [99]利用交替方向乘子法（ADMM），逐步将原始权重转换为低秩表示（即张量列车）。Farnaz等人 [98] 提出了一种适用于张量环格式的自适应秩搜索框架，在该框架中，TR秩在每次迭代中逐步增加，而不是预先设定。

6.3 硬件加速

加速TNNs的训练和推理过程有助于降低资源消耗、调整实验，从而实现经济效益和绿色科研。一个直接且有效的方法是优化TNNs中张量运算的速度，以实现硬件加速。由于对TT格式张量神经网络进行推理不可避免地会产生大量冗余计算，因此提出了TIE方案 [102] ，通过将工作SRAM划分为多个组，并配合精心设计的数据选择机制，来加速TT层。Huang等人 [103]设计了一种具有更高输入输出带宽的并行计算方案，提高了张量收缩的速度。随后，他们提出了LTNN [103]，基于CMOS电阻式随机存取存储器，将TT格式张量神经网络映射到三维加速器上，并通过垂直输入输出连接显著提升带宽。因此，他们同时实现了TNNs的高吞吐量和低功耗。最近，Qu等人 [104]提出了一种空间二维处理元件（PE）数组架构，并构建了一个由片外动态随机存取存储器组成的硬件TT引擎。Kao等人 [105] 提出了一种高能效硬件加速器，通过结合沃尔什-哈达玛变换和离散余弦变换的混合方法，实现CP卷积加速。

许多与TNNs相关的通用张量运算加速方法也已得到发展。例如，Huang等人 [206] 观察到，张量矩阵化运算通常会消耗大量资源，因为其动态随机存取存储器访问建立在随机读取地址之上；因此，他们提出了一种采用顺序地址设计的张量存储方案，以改善动态随机存取存储器的访问性能。T2S-Tensor [207] 和Tensaurus [208]主要致力于为稠密和稀疏张量数据设计通用计算内核。Xie等人 [152] 和Liang等人 [209] 加速了用于获取最优张量收缩序列的搜索过程。Xie等人 [152] 解决了量子分析中双层张量网络收缩所涉及的巨大计算复杂度问题，并将该双层张量网络映射为相交的单层张量网络。Liang等人 [209]实现了多线程优化，以提高收缩的并行性。Fawzi等人 [210] 还阐述了利用强化学习构建高效通用张量运算的潜力。未来，有望开发出更多基于张量运算的通用硬件加速方案，以实现存储和时间消耗更低的TNNs。

注。评论分为三个部分。(1) 为了实现训练稳定性，可以借鉴恒等变换维护方面的思想，构建更加稳定的初始化方法。此外，还可以加入对抗样本，以增强网络鲁棒性。(2) 秩搜索对于进一步提升张量神经网络的性能十分重要。然而，由于这是一个NP难问题，秩搜索尚未得到充分探索。未来，可以通过梯度大小的引导以及进化算法来搜索合适的秩，从而搜索张量神经网络架构。(3) 最后，硬件方面的研究在加速和减少内存占用方面已经取得了一些成功。然而，这些方法几乎都是针对特定张量分解格式的特设设计，因此缺乏对其他张量神经网络结构的适用性。

7 张量神经网络工具箱

1973年，Pereyra 和 Scherer [211]，作为该领域的先驱，开发了用于基本张量运算的编程技术。近年来，随着现代计算机的发展，人们开发了更多基本张量运算工具箱，同时还提出了一系列功能强大的张量神经网络工具箱，用于张量神经网络的两大主要应用：网络压缩和量子电路模拟。本节根据设计目的，将张量神经网络工具箱分为三类：(1) 基本张量运算工具箱包含张量神经网络中的重要基础运算（例如张量收缩和置换）（第 7.1节）；(2) 网络压缩工具箱是基于其他基本运算工具的高层张量神经网络架构工具箱（第 7.2节）；(3) 量子电路模拟工具箱是用于量子电路模拟或量子机器学习过程的软件包，这些过程从量子视角使用张量网络（第 7.3节）。

7.1 基础张量操作工具箱

基本张量运算工具箱旨在实现一些特定的张量分解算法。许多基于不同编程语言和后端的基本张量工具箱已经为此目的而设计。例如，张量分解在线随机框架（OSTD） [110] 和 Tensor Toolbox [108]用于低秩分解，并采用 MATLAB 实现。至于基于 Python 的工具箱，基于 NumPy 的 TensorTools [212] 仅实现 CP分解，而 T3F [116] 专为在 TensorFlow [213]上进行 TT分解而设计。同样地，TensorD [111]基于 TensorFlow，支持 CP分解和 Tucker分解。Tntorch [113] 是一个基于 PyTorch 的张量建模库，支持 CP、Tucker 和 TT格式。TorchMPS [114], 、TT-Toolbox [112]和 Scikit-TT [118] 都是功能强大的、基于 Python 的专用 TT 求解器，能够高效实现 DMRG算法。Tensorly 是一个功能强大的通用张量分解库，支持多种分解格式以及各种 Python 后端，包括 CuPy、PyTorch、TensorFlow 和 MXNet [214]。TensorNetwork [117] 是一个功能强大的通用 TN 库，支持多种 Python 后端，包括 JAX、TensorFlow、PyTorch 和 NumPy。此外，也有一些基于 C++ 的工具箱可供使用。TenDeC ++ [215] 利用 C++ 中一种名为 PointerDeformer 的独特指针技术，支持对张量分解函数进行高效计算。ITensor [115] 是一个高效且灵活的 C++ 库，用于通用 TN 计算。

7.2 网络压缩工具箱

特定的张量神经网络工具箱用于辅助张量层的开发。尽管一些通用张量工具箱（如 Tensorly） [106] 在张量分解处理方面功能强大，并且可以在一定程度上利用其张量分解操作帮助初始化张量神经网络模块，但它们仍然缺乏用于直接构建张量神经网络的应用程序编程接口（API）支持。因此，有人开发了一个基于 Tensorly 的张量神经网络库（Tensorly-Torch），用于在任意 PyTorch 网络中构建张量层。Pan 等人还开发了一个功能强大的张量神经网络库，称为 TedNet [35]。TedNet 可以通过直接调用 API 快速构建张量神经网络层。此外，TedNet 支持仅用一行代码构建张量卷积神经网络和张量循环神经网络。

7.3 量子电路模拟工具箱

目前已经设计出了许多量子电路模拟工具箱。例如，一些张量列车工具箱（如 Scikit-TT 和 TorchMPS）可以在一定程度上部分模拟量子电路，尽管它们并非专门为量子电路模拟而设计。相比之下，通用张量网络工具箱（例如 TensorNetwork 和 ITensor）可以模拟任意量子电路。此外，通过优化张量收缩，TeD-Q [121], 这一面向量子机器学习的张量网络增强型开源软件框架，能够模拟大规模量子电路。此外，Yao [119], 这一用于设计量子算法的可扩展高效库，可以支持将量子电路转储为张量网络。尽管目前还没有量子张量神经网络的实际实现，但这些量子电路模拟工具可能对量子张量神经网络的模拟具有潜在价值。

备注。 尽管当前的工具箱取得了成功，但仍有一些方面需要改进。(1) 现有的基础张量运算工具箱基于高级软件框架构建，限制了其充分利用张量计算固有能力的能力。(2) 现有的面向张量神经网络的深度模型实现工具箱只能包含数量有限的预定义张量神经网络结构，无法允许用户自由设计结构。(3) 现有的量子模拟工具箱更侧重于使用张量网络模拟量子电路，而不便于通过张量神经网络处理嵌入式量子数据。

8 结论与未来展望

本综述探讨张量网络与神经网络之间的联系，总结了压缩神经网络参数的技术，

对数据不同维度之间的交互进行建模，并将量子神经网络与经典神经网络联系起来。如前所述，张量神经网络具有显著优势和巨大潜力，可应用于许多领域。未来，张量神经网络研究可受益于张量友好型硬件的优化与实现；我们相信，张量神经网络将在量子物理和量子计算机相关研究中产生重大影响。

基于硬件设计的加速。尽管许多 TNNs 在理论上具有较低的计算复杂度，但由于置换操作数量众多，实际的硬件部署通常难以达到这一目标 [35]，且缺乏足够的并行性 [103]。可以开发用于通用张量加速或 TNN 加速的高效硬件及匹配软件。如第 6.3节所述，现有的张量加速软件和硬件结构都针对特定的 TN 结构，或基于并行矩阵加速。要实现 TNNs，通用 TN 操作的加速是必要且紧迫的。

量子物理中的应用。在一些需要处理大规模张量的特定物理应用中，例如波函数模拟 [216],，可以研究专门设计的 TNNs，以高效解决涉及高阶交互的问题。受通用逼近定理启发，一些简单的 NNs 已被应用于波函数模拟任务，例如自由玻色子和费米子系统 [217]。然而，由于维数灾难，简单的 NNs 难以应用于超大规模波函数模拟任务，而 TNNs 凭借 TNs 的紧凑性，可以轻松处理大规模张量任务。

量子力学中的实现。现有的张量神经网络主要采用张量网络的数学形式，很少考虑这些张量网络所描述的量子系统的物理性质 [12], [91]。由于张量神经网络与量子电路结构密切相关，因此可以将量子力学中张量网络的概念和理论应用于张量神经网络，从而获得效率更高、效果更好的模型。例如，从纠缠熵理论的角度优化和解释张量神经网络 [218]，可能是构建可解释且高效的神经网络的一个有意义的方向。

MERA 的潜在用途。多尺度纠缠重整化 ansatz（MERA） [219], [220], [221] 是一类树状张量网络，能够以层次化方式表示，同时保持计算优势。MERA 已展现出捕捉量子力学中强关联基态诸多复杂物理性质的能力 [219]。基于这一成功，MERA 不仅有望更好地刻画张量神经网络的表达能力，还可能带来新的实际应用。

致谢

本论文部分得到了国家重点研发计划（编号：2018年AAA0100204）、深圳市科技创新委员会基础研究重点项目（编号：JCYJ20200109113403826）以及鹏城实验室重大重点项目（编号：PCL2021A06）的资助。

参考文献

[1] D. Cyganski 和 J. A. Orr，《张量理论在目标识别和方向确定中的应用》，*IEEE Trans. PAMI*，第 6 期，第 662–673 页，1985 年。[2] P. Koniusz、L. Wang 和 A. Cherian，《用于动作识别的张量表示》，*IEEE Trans. PAMI*，第 44 卷，第 2 期，第 648–665 页，2021年。

[3] J. Tang, X. Shu, G.-J. Qi, Z. Li, M. Wang, S. Yan, and R. Jain, “面向社会感知图像标签优化的三簇张量补全,” IEEE Trans. PAMI, vol. 39, no. 8, pp. 1662–1674, 2016年. [4] J. Tang, X. Shu, Z. Li, Y.-G. Jiang, and Q. Tian, “用于大规模图像重新标注的社会锚点单元图正则化张量补全,” IEEE Trans. PAMI, vol. 41, no. 8, pp. 2027–2034, 2019年. [5] I. Davidson, S. Gilpin, O. Carmichael, and P. Walker, “通过对功能性磁共振成像数据进行约束张量分析发现网络,” in ACM SIGKDD, 2013年. [6] E. Acar, Y. Levin-Schwartz, V. D. Calhoun, and T. Adali, “基于张量的脑电图与功能性磁共振成像融合：理解精神分裂症中的神经系统变化,” in ISCAS. IEEE, 2017 年 . [7] K. Keegan, T. Vishwanath, and Y. Xu, “应用于功能性磁共振成像数据的张量奇异值分解分类算法,” arXiv preprint arXiv:2111.00587, 2021年. [8] J. Biamonte and V. Bergholm, “张量网络简明指南,” arXiv preprint arXiv:1708.00006, 2017年. [9] T. Huckle, K. Waldherr, and T. Schulte-Herbrüggen, “量子张量网络中的计算,” 《线性代数及其应用》, vol. 438, no. 2, pp. 750–781, 2013年. [10] X. Chen, Z. He, and L. Sun, “用于时空交通数据插补的贝叶斯张量分解方法,” 《交通研究C辑》: 新兴技术, vol. 98, pp. 73–84, 2019年. [11] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, D. P. Mandic et al., “用于降维和大规模优化的张量网络：第1部分，低秩张量分解,” 机器学习基础与趋势, vol. 9, no. 4-5, pp. 249–429, 2016年. [12] A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. Oseledets, M. Sugiyama, D. P. Mandic et al., “用于降维和大规模优化的张量网络：第2部分，应用与未来展望,” 机器学习基础与趋势, vol. 9, no. 6, pp. 431–673, 2017年.[13] R. A. Harshman, “PARAFAC过程的基础：‘解释性’多模态因子分析的模型与条件,” 《加州大学洛杉矶分校语音学工作论文》, vol. 16, pp. 1–84, 1970年. [14] J. D. Carroll and J.-J. Chang, “通过“Eckart-Young”分解的N向推广分析多维尺度分析中的个体差异,” 《心理测量学》, vol. 35, no. 3, pp. 283–319, 1970年. [15] H. A. Kiers, “迈向多向分析中标准化的记号和术语,” 《化学计量学杂志》: 化学计量学学会期刊, vol. 14, no. 3, pp. 105–122, 2000年. [16] L. R. Tucker, “关于三模态因子分析的一些数学注记,” 《心理测量学》, vol. 31, no. 3, pp. 279–311, 1966年. [17] ——, “三向矩阵因子分析对变化测量的启示,” 测量变化的问题, vol. 15, 137, 1963年 122pp. – .[18] L. De Lathauwer, “高阶张量的块项分解——第I部分：分块矩阵引理,” 《SIAM矩阵分析与应用杂志》, vol. 30, no. 3, pp. 1022–1032, 2008年. [19] I. V. Oseledets, “张量列车分解,” 《SIAM科学计算杂志》, vol. 33, no. 5, pp. 2295–2317, 2011年. [20] A. Cichocki, “大数据处理时代：通过张量网络和张量分解实现的新方法,” arXiv preprint arXiv:1403 2048, 2014年 . .[21] F. Verstraete, V. Murg, and J. I. Cirac, “用于量子自旋系统的矩阵乘积态、投影纠缠对态和变分重整化群方法,” 《物理学进展》, vol. 57, 224, 2008年 2, 143no. pp. – .[22] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, “张量环分解,” arXiv preprint arXiv:1606.05535, 2016年. [23] D. Kressner and C. Tobler, “htucker——用于层次Tucker格式张量的MATLAB工具箱,” Mathicse, 洛桑联邦理工学院, 2012年.[24] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, “投影纠缠对态的计算复杂度,” 《物理评论快报》, vol. 98, no. 14, p. 140506, 2007年. [25] A. Milsted and G. Vidal, “多尺度纠缠重整化拟设的几何解释,” arXiv preprint arXiv:1812 00529, 2018年 . .[26] F. Pan and P. Zhang, “使用大批量张量网络方法模拟量子电路,” 《物理评论快报》, vol. 128, no. 3, p. 030501, 2022年.[27] Z. Zhang, G. I. Allen, H. Zhu, and D. Dunson, “张量网络分解：脑结构连接组与特征之间的关系,” 《神经影像》, vol. 197, pp. 330–343, 2019 年 . [28] A. Zare, A. Ozdemir, M. A. I wen, and S. Aviyente, “将主成分分析扩展至高阶数据结构：张量、张量分解和张量主成分分析简介,” IEEE会议录, vol. 106, no. 8, pp. 1341–1358, 2018年.

[29] J. Zhang, X. Li, P. Jing, J. Liu, and Y. Su, “用于子空间聚类的低秩正则化异构张量分解,” IEEE Signal Processing Letters, vol. 25, no. 3, pp. 333–337, 2017年. [30] K. Hayashi, T. Yamaguchi, Y. Sugawara, and S. Maeda, “探索用于卷积神经网络的尚未探索的张量网络分解,” i n NeurIPS, 2019年. [31] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempit- sky, “利用微调的CP分解加速卷积神经网络,” in ICLR, 2015 年 . [32] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, “利用卷积网络中的线性结构实现高效评估,” in NeurIPS, 2014年, pp. 1269–1277. [33] A.-H. Phan, K. Sobolev, K. Sozykin, D. Ermilov, J. Gusak, P. Tichavsk`y, V. Glukhov, I. Oseledets, and A. Cichocki, “用于卷积神经网络压缩的稳定低秩张量分解,” in ECCV. Springer, 2020年, pp. 522–539. [34] A. Nekooei and S. Safari, “基于量化张量分解的深度神经网络压缩及其在可重构硬件平台上的实现,” Neural Networks, vol. 150, pp. 350–363, 2022年. [35] Y. Pan, M. Wang, and Z. Xu, “Tednet：用于张量分解网络的PyTorch工具包,” Neurocomputing, vol. 469, pp. 234–238, 2022年. [36] Y. Liu and M. K. Ng, “通过带非线性响应的Tucker分解压缩深度神经网络,” Knowledge-Based Systems, 2022年.[37] T. Garipov, D. Podoprikhin, A. Novikov, and D. Vetrov, “终极张量化：同时压缩卷积层和全连接层,” arXiv preprint arXiv:1611.03214, 2016年. [38] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, “神经网络张量化,” in NeurIPS, 2015年. [39] J. Qi, C.-H. H. Yang, P.-Y. Chen, and J. Tejedor, “基于黎曼梯度下降的低秩张量列车深度神经网络及其在语音处理中的示例,” IEEE/ACM音频、语音与语言处理汇刊, vol. 31, pp. 633–642, 2023年. [40] D. Liu, L. T. Yang, P. Wang, R. Zhao, and Q. Z hang, “TT-TSVD：多模态张量列车分解及其在智能医疗卷积神经网络中的应用,” TOMM, vol. 18, 17, 2022年 1 1no. s, pp. – .[41] W. Wang, Y. Sun, B. Eriksson, W. Wang, and V. Aggarwal, “宽度压缩：张量环网络,” in CVPR, 2018 年 , pp. 9329–9338. [42] J. Ye, G. Li, D. Chen, H. Yang, S. Zhe, and Z. Xu, “块项张量神经网络.” Neural Networks：国际神经网络学会官方期刊, vol. 130, pp. 11–21, 2020年. [43] J. Kossaifi, A. Khanna, Z. Lipton, T. Furlanello, and A. Anandkumar, “用于简约深度网络的张量收缩层,” in CVPR研讨会, 2017年. [44] J. Kossaifi, Z. C. Lipton, A. Kolbeinsson, A. Khanna, T. Furlanello, and A. Anandkumar, “张量回归网络,” 《机器学习研究期刊》, vol. 21, no. 123, pp. 1–21, 2020年. [45] B. Wu, D. Wang, G. Zhao, L. Deng, and G. Li, “神经网络压缩中的混合张量分解,” Neural Networks, vol. 132, pp. 309–320, 2020年. [46] J. Kossaifi, A. Bulat, G. Tzimiropoulos, and M. Pantic, “T-Net：用单个高阶张量参数化全卷积网络,” in CVPR, 2019年. [47] Y. Yang and T. Hospedales, “深度多任务表示学习：一种张量分解方法,” in ICLR, 2017年. [48] M. Wang, Z. Su, X. Luo, Y. Pan, S. Zheng, and Z. Xu, “用于深度多任务学习的串联张量网络,” in ICONIP, 2020年. [49] J. Kossaifi, A. Toisoul, A. Bulat, Y. Panagakis, T. M. Hospedales, and M. Pantic, “因子化高阶卷积神经网络及其在时空情绪估计中的应用,” in CVPR, 2020年. [50] Y. Yang, D. Krompass, and V. Tresp, “用于视频分类的张量列车循环神经网络,” in ICML, 2017年. [51] J. Ye, L. Wang, G. Li, D. Chen, S. Zhe, X. Chu, and Z. Xu, “利用块项张量分解学习紧凑型循环神经网络,” in CVPR, 2018年. [52] Y. Pan, J. Xu, M. Wang, J. Ye, F. Wang, K. Bai, and Z. Xu, “利用张量环压缩循环神经网络以实现动作识别,” in AAAI, 2019年. [53] M. Yin, S. Liao, X. Liu, X. Wang, and B. Yuan, “利用完全分解的层次Tucker结构实现极紧凑的视频识别循环神经网络,” in CVPR, 2021年. [54] C. Jose, M. Cissé, and F. Fleuret, “Kronecker循环单元,” in ICML, 2018年.[55] A. Tjandra, S. Sakti, and S. Nakamura, “基于低秩张量表示的循环神经网络压缩,” IEICE Trans. Inf. Syst., vol. 103-D, no. 2, pp. 435–449, 2020年.

[56] D. W ang, B. Wu, G.-S. Zh ao, H. Ch en, L. Deng, T. Y an, an d G. Li , “用于压缩循环神经网络的快速乘法克罗内克 CP 分解,” IEEE 神经网络与学习系统汇刊, 卷 PP, 2021年.[57] J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar, “用于时空学习的卷积张量列车 LSTM,” 收录于 NeurIPS, 2020年.[58] X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song, “用于语言建模的张量化 Transformer,” NeurIPS, 2019年. [59] P. Liu, Z. Gao, W. X. Zhao, Z. Xie, Z. Lu, and J. Wen, “基于矩阵乘积算子，为预训练语言模型压缩实现轻量级微调,” 收录于 ACL/IJCNLP, 2021年. [60] J. Tang, K. Li, M. Hou, X. Jin, W. Kong, Y. Ding, and Q. Zhao, “MMT：用于多模态学习的多路多模态 Transformer,” 收录于 IJCAI, 2022年.[61] X. Liu, J. Su, and F. Huang, “ Tuformer：通过 Tucker 张量表示，以数据驱动的方式设计高表现力 Transformer,” 收录于 ICLR, 2022年. [62] B. Wang, Y. Ren, L. Shang, X. Jiang, and Q. Liu, “探索预训练语言模型的极端参数压缩,” 收录于 ICLR, 2022年. [63] L. Sunzhu, Z. Peng, G. Guobing, L. Xiuqing, W. Benyou, W. Junqiu, and J. Xin, “Hypoformer：面向边缘设备神经机器翻译的混合分解 Transformer,” EMNLP, 2022年.[64] M. A. O. Vasilescu, “因果深度学习：因果胶囊与张量 Transformer,” arXiv 预印本 arXiv:2301.00314, 2023年.[65] C. Hua, G. Rabusseau, and J. Tang, “结合张量分解的图神经网络高阶池化,” NeurIPS, 2022年.[66] P. Baghershahi, R. Hosseini, and H. Moradi, “通过张量分解实现图神经网络中的高效关系感知邻域聚合,” arXiv 预印本 arXiv:2212.05581, 2022年.[67] C. Jia, B. Wu, and X.-P. Zhang, “结合张量网络的动态时空图神经网络,” arXiv 预印本 arXiv:2003.08729, 2020年.[68] X. Zhao, Q. Dai, J. Wu, H. Peng, M. Liu, X. Bai, J. Tan, S. Wang, and P. Yu, “通过强化聚合实现的多视图张量图神经网络,” TKDE, 2022年.[69] F. Ju, Y. Sun, J. Gao, M. Antolovich, J. Dong, and B. Yin, “张量化受限玻尔兹曼机,” ACM Transactions on Knowledge Discovery from Data, 卷 13, 期 3, 页 1–16, 2019年. [70] M. Wang, C. Zhang, Y. Pan, J. Xu, and Z. Xu, “张量环受限玻尔兹曼机,” 收录于 IJCNN, 2019年.[71] T. D. Nguyen, T. Tran, D. Phung, and S. Venkatesh, “张量变量受限玻尔兹曼机,” 收录于 AAAI, 2015年.[72] G. Qi, Y. Sun, J. Gao, Y. Hu, and J. Li, “矩阵变量受限玻尔兹曼机,” 收录于 IJCNN, 2016年.[73] A. Zadeh, M. Chen, S. Poria, E. Cambria, and L.-P. Morency, “用于多模态情感分析的张量融合网络,” 收录于 EMNLP, 2017年. [74] S. Rendle, “因子分解机,” 收录于 ICDM, 2010年.[75] Z. Liu, Y. Shen, V. B. Lakshminarasimhan, P. P. Liang, A. B. Zadeh, and L.-P. Morency, “利用模态特定因子实现高效低秩多模态融合,” 收录于 ACL, 2018年. [76] M. Hou, J. Tang, J. Zhang, W. Kong, and Q. Zhao, “结合高阶多项式池化的深度多模态多线性融合,” NeurIPS, 2019年. [77] H. Ben-Younes, R. Cadene, M. Cord, and N. Thome, “MUTAN：用于视觉问答的多模态 Tucker 融合,” 收录于 ICCV, 2017年. [78] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, “用于视觉问答和视觉定位的多模态紧凑双线性池化,” 收录于 EMNLP, 2016年. [79] J.-H. Kim, K.-W. On, W. Lim, J. Kim, J.-W. Ha, and B.-T. Zhang, “用于低秩双线性池化的 Hadamard 乘积,” arXiv 预印本 arXiv:1610.04325, 2016年.[80] T. Do, T.-T. Do, H. Tran, E. Tjiputra, and Q. D. Tran, “用于视觉问答的紧凑三线性交互,” 收录于 I CCV, 2019年. [81] E. Stoudenmire and D. J. Schwab, “基于张量网络的监督学习,” NeurIPS, 2016年.[82] Q. Li, B. Wang, and M. Melucci, “CNM：一种用于匹配的可解释复值网络,” 收录于 NAACL, 2019年. [83] P. Zhang, Z. Su, L. Zhang, B. Wang, and D. Song, “受量子多体波函数启发的语言建模方法,” 收录于 CIKM, 2018 年 .[84] J. Miller, G. Rabusseau, and J. Terilla, “ 用于概率序列建模的张量网络,” 收录于 AISTATS, 2021年. [85] I. Glasser, N. Pancotti, and J. I. Cirac, “用于概率建模的张量网络分解的表达能力,” arXiv 预印本 arXiv:1806 50, 2018年 05964, . p. .[86] S. Cheng, L. Wang, T. Xiang, and P. Zhang, “用于生成建模的树张量网络,” Physical Review B, 卷 99, 155131, 2019年 15, no. p. .

[87] I. Glasser, R. Sweke, N. Pancotti, J. Eisert, and I. Cirac, “张量网络分解在概率建模中的表达能力,” NeurIPS, 2019年. [88] J. Liu, S. Li, J. Zhang, and P. Zhang, “用于无监督机器学习的张量网络,” Physical Review E, 卷 107, 期 1, 页 L012103, 2023年. [89] N. Cohen, O. Sharir, and A. S hashua, “论深度学习的表达能力：张量分析,” 收录于 COLT, 2016年. [90] L. Zhang, P. Zhang, X. Ma, S. Gu, Z. Su, and D. Song, “张量空间中的广义语言模型,” 收录于 AAAI, 2019年. [91] Y. Panagakis, J. Kossaifi, G. G. Chrysos, J. Oldfield, M. A. Nicolaou, A. Anandkumar, and S. Zafeiriou, “计算机视觉与深度学习中的张量方法,” Proc. IEEE, 卷 109, 期 5, 页 863–890, 2021年. [92] Y. Pan, Z. Su, A. Liu, J. Wang, N. Li, and Z. Xu, “张量卷积神经网络的统一权重初始化范式,” 收录于 ICML, 2022年. [93] N. Li, Y. Pan, Y. Chen, Z. Ding, D. Zhao, and Z. Xu, “通过渐进式搜索实现张量环网络的启发式秩选择,” Complex & Intelligent Systems, 页 1–15, 2021年. [94] Z. Cheng, B. Li, Y. Fan, and Y. Bao, “一种基于强化学习的张量环分解秩选择新方案，用于深度神经网络,” 收录于 ICASSP, 2020年. [95] Q. Zhao, L. Zhang, and A. Cichocki, “不完整张量的贝叶斯 CP 分解与自动秩确定,” IEEE Trans. PAMI, 卷 37, 期 9, 页 1751–1763, 2015年. [96] K. Sobolev, D. Ermilov, A.-H. Phan, and A. Cichocki, “PARS：通过低秩权重近似实现神经网络压缩的基于代理的自动秩选择,” Mathematics, 卷 10, 期 20, 页 3801, 2022年. [97] C. Hawkins and Z. Zhang, “具备自动秩选择功能的贝叶斯张量化神经网络,” Neurocomputing, 卷 453, 页 172–180, 2021年. [98] F. Sedighin, A. Cichocki, and A.-H. Phan, “张量环分解的自适应秩选择,” 《IEEE信号处理精选主题期刊》, 卷 15, 期 3, 页 454–463, 2021年. [99] M. Yin, Y. Sui, S. Liao, and B. Yuan, “借助优化框架实现高效的基于张量分解的DNN压缩,” 收录于 CVPR, 2021年. [100] Y. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, “深度卷积神经网络的压缩：实现快速、低功耗的移动应用,” 收录于 ICLR, 2016年. [101] J. Gusak, M. Kholyavchenko, E. Ponomarev, L. Markeeva, P. Blagoveschensky, A. Cichocki, and I. V. Oseledets, “神经网络的自动化多阶段压缩,” 收录于 ICCV Workshops, 2019年. [102] C. Deng, F. Sun, X. Qian, J. Lin, Z. Wang, and B. Yuan, “TIE：基于张量列车的深度神经网络节能推理引擎,” 收录于 ISCA, 2019年. [103] H. Huang, L. Ni, and H. Yu, “LTNN：面向逐层张量化神经网络、基于三维CMOS-RRAM的节能型机器学习加速器,” 收录于 SOCC, 2017年. [104] Z. Qu, L. Deng, B. Wang, H. Chen, J. Lin, L. Liang, G. Li, Z. Zhang, and Y. Xie, “通过张量列车分解实现硬件加速的高效数据处理,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 卷 41, 期 2, 页 372–385, 2022年. [105] C. Kao, Y. Hsieh, C. Chen, and C. Yang, “面向神经网络压缩的大规模张量分解硬件加速,” 收录于 MWSCAS, 2022年. [106] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic, “Tensorly：Python 中的张量学习,” 《机器学习研究期刊》, 卷 20, 6, 2019年 26 1 26pp. : – : .[107] A. H. Williams, T. H. Kim, F. Wang, S. Vyas, S. I. Ryu, K. V. Shenoy, M. Schnitzer, T. G. Kolda, and S. Ganguli, “通过张量成分分析，无监督发现跨多个时间尺度的解混低维神经动力学,” Neuron, 卷 98, 期 6, 页 1099–1115, 2018年. [108] T. G. Kolda and B. W. Bader, “Matlab张量工具箱,” 桑迪亚国家实验室 (SNL), 技术报告, 2006年. [109] J. Huang, L. Kong, X. Liu, W. Qu, and G. Chen, “一种用于张量分解的 C++ 库,” 收录于 IPCCC, 2019年. [110] A. Sobral, S. Javed, S. K. Jung, T. Bouwmans, and E. Zahzah, “用于多光谱视频序列背景减除的在线随机张量分解,” 收录于 ICCV Workshops, 2015年. [111] L. Hao, S. Liang, J. Ye, and Z. Xu, “TensorD：TensorFlow 中的张量分解库,” Neurocomputing, 卷 318, 页 196–200, 2018年. [112] I. Oseledets, S. Dolgov, V. Kazeev, D. Savostyanov, O. Lebedeva, P. Zhlobich, T. Mach, and L. Song, “TT-Toolbox,” 2016年. [113] R. Ballester-Ripoll, “tntorch——基于 PyTorch 的张量网络学习,” 2018年. [在线]。地址：https://github.com/rballester/tntorch [114] J. Miller, “TorchMPS,” 2019年. [在线]。地址：https://github.com/jemisjoky/torchmps

[115]用于张量网络计算的 TensorFlow 库，”arXiv 预印本 arXiv:2007.14822，2020年。[116] A. Novikov、P. Izmailov、V. Khrulkov、M. Figurnov 和 I. V. Oseledets，“TensorFlow 上的张量列车分解（T3F），”《机器学习研究期刊》，第 21 卷，第 30:1–30:7 页，2020年。[117] C. Roberts、A. Milsted、M. Ganahl、A. Zalcman、B. Fontaine、Y. Zou、J. Hidary、G. Vidal 和 S. Leichenauer，“TensorNetwork：用于物理学和机器学习的库，”arXiv 预印本 arXiv:1905 01330，2019，2019。[118] P. Gel、S. Klus、M. Scherer 和 F. Nske，“Scikit-TT 张量列工具箱，”2018年。 [在线]。可访问：https://github.com/PGelss/scikit$/_$tt [119] X.-Z. Luo、J.-G. Liu、P. Zhang 和 L. Wang，“Yao. jl：用于量子算法设计的可扩展高效框架，”Quantum，第 4 卷，第 341 页，2020年。[120] D. Kartsaklis、I. Fan、R. Yeung、A. Pearson、R. Lorenz、A. Toumi、G. de Felice、K. Meichanetzidis、S. Clark 和 B. Coecke，“lambeq：用于量子自然语言处理的高效高级 Python 库，”arXiv 预印本 arXiv:2110.04236，2021年。[121] J. E. Academy，“张量网络增强型分布式量子计算，”2022年。[在线]。可访问：https://github.com/JDEA-Quantum-Lab/TeD-Q [122] J. J. Hopfield，“具有涌现集体计算能力的神经网络与物理系统，”《美国国家科学院院刊》，第 79 卷，第 8 期，第 2554–2558 页，1982年。[123] D. E. Rumelhart、G. E. Hinton、R. J. Williams 等，“通过误差反向传播学习表示，”《认知建模》，第 5 卷，第 3 期，第 1 页，1988年。[124] J. Schmidhuber，“神经网络中的深度学习：综述，”《神经网络》，第 61 卷，第 85–117 页，2015年。[125] Y. LeCun、Y. Bengio 和 G. Hinton，“深度学习，”《自然》，第 521 卷，第 7553 期，第 436 页，2015年。[126] G. E. Hinton，“训练受限玻尔兹曼机的实用指南，”载于《神经网络：技巧与实践》。Springer，2012年，第 599–619 页。[127] G. Huang、Z. Liu、L. van der Maaten 和 K. Q. Weinberger，“密集连接卷积网络，”载于 CVPR，2017年。[128] P. Zhou、W. Shi、J. Tian、Z. Qi、B. Li、H. Hao 和 B. Xu，“用于关系分类的基于注意力的双向长短期记忆网络，”载于 ACL，2016年。[129] R. Dey 和 F. M. Salem，“门控循环单元（GRU）神经网络的门变体，”载于 MWSCAS，2017年。[130] A. Vaswani、N. Shazeer、N. Parmar、J. Uszkoreit、L. Jones、A. N. Gomez、Ł. Kaiser 和 I. Polosukhin，“注意力就是你所需要的一切，”载于 NeurIPS，2017年。[131] J. Devlin、M. Chang、K. Lee 和 K. Toutanova，“BERT：用于语言理解的深度双向 Transformer 预训练，”载于 NAACL- HLT，2019年。[132] A. Dosovitskiy、L. Beyer、A. Kolesnikov、D. Weissenborn、X. Zhai、T. Unterthiner、M. Dehghani、M. Minderer、G. Heigold、S. Gelly 等，“一幅图像胜过 16×16 个词：用于大规模图像识别的 Transformer，”载于 ICLR，2020年。[133] T. Wolf、L. Debut、V. Sanh、J. Chaumond、C. Delangue、A. Moi、P. Cistac、T. Rault、R. Louf、M. Funtowicz 等，“Huggingface 的 Transformer：最先进的自然语言处理，”arXiv 预印本 arXiv:1910.03771，2019年。[134] A. Krizhevsky、I. Sutskever 和 G. E. Hinton，“使用深度卷积神经网络进行 ImageNet 分类，”载于 NeurIPS，2012年。[135] K. Simonyan 和 A. Zisserman，“用于大规模图像识别的超深卷积网络，”arXiv 预印本 arXiv:1409.1556，2014年。[136] C. Szegedy、W. Liu、Y. Jia、P. Sermanet、S. Reed、D. Anguelov、D. Erhan、V. Vanhoucke 和 A. Rabinovich，“通过卷积走得更深，”载于 CVPR，2015年。[137] K. He、X. Zhang、S. Ren 和 J. Sun，“用于图像识别的深度残差学习，”载于 CVPR，2016年。[138] O. Russakovsky、J. Deng、H. Su、J. Krause、S. Satheesh、S. Ma、Z. Huang、A. Karpathy、A. Khosla、M. Bernstein 等，“ImageNet 大规模视觉识别挑战赛，”《国际计算机视觉杂志》，第 115 卷，第 3 期，第 211–252 页，2015年。[139] J. Jumper、R. Evans、A. Pritzel、T. Green、M. Figurnov、O. Ronneberger、K. Tunyasuvunakool、R. Bates、A. Žídek、A. Potapenko 等，“利用 AlphaFold 进行高精度蛋白质结构预测，”《自然》，第 596 卷，第 7873 期，第 583–589 页，2021年。[140] M. van Breugel、I. Rosa e Silva 和 A. Andreeva，“对中心体和中心粒蛋白及其复合物的 AlphaFold2 预测进行结构验证与评估，”《通讯生物学》，第 5 卷，第 1 期，第 1–10 页，2022年。

[141] OpenAI，“GPT-4 08774，2023 2303 技术报告，”arXiv 预印本 arXiv：。。[142] T. Mikolov、A. Deoras、D. Povey、L. Burget 和 J. ˇ Cernock`y，“大规模神经网络语言模型的训练策略，”载于 2011 IEEE 自动语音识别与理解研讨会。IEEE，2011年，第 196–201 页。[143] M. K. Leung、H. Y. Xiong、L. J. Lee 和 B. J. Frey，“组织调控剪接代码的深度学习，”《生物信息学》，第 30 卷，第 12 期，第 i121–i129 页，2014年。[144] M. Mondelli 和 A. Montanari，“关于双层神经网络学习与张量分解之间联系的研究，”载于第 22 届国际人工智能与统计学会议。PMLR，2019年，第 1051–1060 页。[145] S. Antol、A. Agrawal、J. Lu、M. Mitchell、D. Batra、C. L. Zitnick 和 D. Parikh，“VQA：视觉问答，”载于 ICCV，2015年。[146] A. Abbas、D. Sutter、C. Zoufal、A. Lucchi、A. Figalli 和 S. Woerner，“量子神经网络的力量，”《自然·计算科学》，第 1 卷，第 6 期，第 403–409 页，2021年。[147] S. C. Kak，“量子神经计算，”《成像与电子物理进展》，第 94 卷，第 259–313 页，1995年。[148] W. Huggins、P. Patil、B. Mitchell、K. B. Whaley 和 E. M. Stoudenmire，“迈向利用张量网络的量子机器学习，”《Quantum Science and Technology》，第 4 卷，第 2 期，第 024001 页，2019年。[149] Z. Xu、F. Yan 和 Y. Qi，“无限 Tucker分解：用于多路数据分析的非参数贝叶斯模型，”载于 ICML，2012年。[150] S. Zhe、Y. Qi、Y. Park、Z. Xu、I. Molloy 和 S. Chari，“ Dintucker：在大型多维数组上扩展高斯过程模型，”载于 AAAI，2016年。[151] R. Penrose，“负维张量的应用，”《组合数学及其应用》，第 1 卷，第 221–244 页，1971年。[152] Z. Xie、H. Liao、R. Huang、H. Xie、J. Chen、Z. Liu 和 T. Xiang，“张量网络态的优化收缩方案，”Physical Review B，第 96 卷，第 4 期，第 045128 页，2017年。[153] T. G. Kolda 和 B. W. Bader，“张量分解及其应用，”《SIAM评论》，第 51 卷，第 3 期，第 455–500 页，2009年。[154] N. D. Sidiropoulos、L. De Lathauwer、X. Fu、K. Huang、E. E. Papalexakis 和 C. Faloutsos，“用于信号处理和机器学习的张量分解，”IEEE Signal Processing Letters，第 65 卷，第 13 期，第 3551–3582 页，2017年。[155] U. Schollwöck，“矩阵乘积态算法：DMRG、TEBD 及其相关方法，”载于《强关联系统》。Springer，2013年，第 67–98 页。[156] C. J. Hillar 和 L.-H. Lim，“大多数张量问题都是 NP难，”《ACM杂志》，第 60 卷，第 6 期，第 1–39 页，2013年。[157] L. De Lathauwer、B. De Moor 和 J. Vandewalle，“多线性奇异值分解，”《SIAM矩阵分析与应用杂志》，第 21 卷，第 4 期，第 1253–1278 页，2000年。[158] U. Schollwöck，“矩阵乘积态时代的密度矩阵重整化群，”《物理学年鉴》，第 326 卷，192，2011年 1 月，第 96 页。。[159] B. Pirvu、V. Murg、J. I. Cirac 和 F. Verstraete，“矩阵乘积算符表示，”《新物理学杂志》，第 12 卷，第 2 期，第 025012 页，2010年。[160] A.-H. Phan、K. Sobolev、D. Ermilov、I. Vorona、N. Kozyrskiy、P. Tichavsky 和 A. Cichocki，“如何训练不稳定的环形张量网络，”arXiv 预印本 arXiv:2203.02617，2022年。[161] F. Sedighin 和 A. Cichocki，“利用多阶段张量环分解在嵌入空间中进行图像补全，”《人工智能前沿》，第 4 卷，第 687176 页，2021 年。[162] F. Sedighin 、 A. Cichocki 、 T. Yokota 和 Q. Shi，“利用张量列车在多路延迟嵌入空间中进行矩阵和张量补全及其在信号重构中的应用，”IEEE Signal Processing Letters，第 27 卷，第 810–814 页，2020年。[163] H. Huang、Y. Liu 和 C. Zhu，“用于图像补全的低秩张量网格，”arXiv 预印本 arXiv:1903.04735，2019年。[164] M. D. Zeiler 和 R. Fergus，“卷积网络的可视化与理解，”载于 ECCV，2014年。[165] P. Molchanov、A. Mallya、S. Tyree、I. Frosio 和 J. Kautz，“神经网络剪枝的重要性估计，”载于 CVPR，2019年。[166] J. Yang、X. Shen、J. Xing、X. Tian、H. Li、B. Deng、J. Huang 和 X. Hua，“量化网络，”载于 CVPR，2019年。[167] X. Jiao、Y. Yin、L. Shang、X. Jiang、X. Chen、L. Li、F. Wang 和 Q. Liu，“TinyBERT：为自然语言理解蒸馏 BERT，”载于 EMNLP，2020年。[168] S. Hochreiter 和 J. Schmidhuber，“长短期记忆，”《神经计算》，第 9 卷，第 8 期，第 1735–1780 页，1997年。[169] L. De Lathauwer，“高阶张量的分解：块项分解——第二部分：定义与唯一性，”《SIAM矩阵分析与应用杂志》，第 30 卷，第 3 期，第 1033–1066 页，2008年。

[170] C. Lubich、T. Rohwedder、R. Schneider 和 B. Vandereycken，《基于层次 Tucker 张量和张量列张量的动力学逼近》，《SIAM矩阵分析与应用杂志》，第 34 卷，第 2 期，第 470–494 页，2013年。[171] J. Zhou、G. Cui、S. Hu、Z. Zhang、C. Yang、Z. Liu、L. Wang、C. Li 和 M. Sun，《图神经网络：方法与应用综述》，AI Open，第 1 卷，第 57–81 页，2020年。[172] C. Chen、K. Batselier、C.-Y. Ko 和 N. Wong，《矩阵乘积算子受限玻尔兹曼机》，载于 IJCNN，2019年。[173] D. Lahat、T. Adali 和 C. Jutten，《多模态数据融合：方法、挑战与前景概述》，IEEE会议录，第 103 卷，第 9 期，第 1449–1477 页，2015年。[174] L. Morency、R. Mihalcea 和 P. Doshi，《迈向多模态情感分析：从网络中采集观点》，载于 ICMI，2011年。[175] H. Wang、A. Meghawat、L. Morency 和 E. P. Xing，《选择加性学习：提升多模态情感分析中的跨个体泛化能力》，CoRR，第 abs/1609.05244 号，2016年。[176] G. Zhou、Q. Zhao、Y. Zhang、T. Adalı、S. Xie 和 A. Cichocki，《从矩阵到高阶张量的关联成分分析：生物医学数据应用》，IEEE会议录，第 104 卷，331，2016年 2，第 310 页 –。[177] G. G. Chrysos、S. Moschoglou、G. Bouritsas、J. Deng、Y. Panagakis 和 S. Zafeiriou，《深度多项式神经网络》，IEEE Trans. PAMI，第 44 卷，第 8 期，第 4021–4034 页，2021年。[178] J. D. Hidary 和 J. D. Hidary，《量子计算：应用方法》。Springer，2021年，第 1 卷。[179] N. Zettili，《量子力学：概念与应用》，2003年。[180] H. Weimer、M. Müller、I. Lesanovsky、P. Zoller 和 H. P. Büchler，《里德伯量子模拟器》，Nature Physics，第 6 卷，388，2010年 5，第 382 页 –。[181] J. Eisert、M. Friesdorf 和 C. Gogolin，《非平衡量子多体系统》，Nature Physics，第 11 卷，第 2 期，第 124–130 页，2015年。[182] F. Valdez 和 P. Melin，《量子计算与深度学习算法及其应用综述》，Soft Computing，第 1–20 页，2022年。[183] P. W. Shor，《量子计算算法：离散对数与因式分解》，载于 FOCS，1994年。[184] A. Steane，《量子计算》，Reports on Progress in Physics，第 61 卷，第 2 期，第 117 页，1998年。[185] N. Cohen 和 A. Shashua，《作为广义张量分解的卷积整流网络》，载于 ICML，2016年。[186] D. Wu、R. Rossi、F. Vicentini 和 G. Carleo，《从张量网络量子态到张量递归神经网络》，arXiv 预印本 arXiv:2206.12363，2022年。[187] D. Guala、S. Zhang、E. Cruz、C. A. Riofrío、J. Klepsch 和 J. M. Arrazola，《利用张量网络量子电路进行图像分类的实用概述》，Scientific Reports，第 13 卷，第 1 期，第 4427 页，2023年。[188] Z. Chen、L. Newhouse、E. Chen、D. Luo 和 M. Solja ˇci´ c，《自回归神经张量网络：连接用于量子多体模拟的神经网络与张量网络》，arXiv 预印本 arXiv:2304 01996，2023年。[189] D. P. DiVincenzo，《量子计算》，Science，第 270 卷，第 5234 期，第 255–261 页，1995年。[190] G. Gan、P. Zhang、S. Li、X. Lu 和 B. Wang，《Morphte：将形态学注入张量化嵌入》，载于 NeurIPS，2022年。[191] M. Benedetti、E. Lloyd、S. Sack 和 M. Fiorentini，《作为机器学习模型的参数化量子电路》，Quantum Science and Technology，第 4 卷，第 4 期，第 043001 页，2019年。[192] C. Hubig、I. P. McCulloch、U. Schollwöck 和 F. A. Wolf，《带子空间扩展的严格单站点 DMRG算法》，Physical Review B，第 91 卷，第 15 期，第 155115 页，2015年。[193] Z.-Y. Han、J. Wang、H. Fan、L. Wang 和 P. Zhang，《使用矩阵乘积态进行无监督生成建模》，Physical Review X，第 8 卷，第 3 期，第 031012 页，2018年。[194] S. Cheng、L. Wang 和 P. Zhang，《使用投影纠缠对态进行监督学习》，Physical Review B，第 103 卷，125117，2021年 12，第 页。[195] I. Goodfellow、J. Pouget-Abadie、M. Mirza、B. Xu、D. Warde-Farley、S. Ozair、A. Courville 和 Y. Bengio，《生成对抗网络》，ACM通讯，第 63 卷，第 11 期，第 139–144 页，2020年。[196] D. P. Kingma 和 M. Welling，《自编码变分贝叶斯》，arXiv 预印本 arXiv:1312.6114，2013年。[197] M. Born，《碰撞过程的量子力学》，Zeitschrift für Physik，第 38 卷，第 11 期，第 803–827 页，1926年。[198] Y. LeCun、S. Chopra、R. Hadsell、M. Ranzato 和 F. Huang，《基于能量的学习教程》，Predicting Structured Data，第 1 卷，0，2006年 第 .

[199] Y. Levine、O. Sharir、N. Cohen 和 A. Shashua，《深度学习架构中的量子纠缠》，《物理评论快报》，第 122 卷，第 6 期，第 065301 页，2019年。[200] K. He、X. Zhang、S. Ren 和 J. Sun，《用于图像识别的深度残差学习》，载于 CVPR，2016年。[201] C. Li 和 Z. Sun，《张量网络分解的进化拓扑搜索》，载于 ICML，2020年。[202] X. Glorot 和 Y. Bengio，《理解深度前馈神经网络训练的难点》，载于 AISTATS，2010年。[203] K. He、X. Zhang、S. Ren 和 J. Sun，《深入研究整流器：在 t 分类任务上实现超越人类水平的性能》，载于 ICCV，2015年。[204] T. Elsken、J. H. Metzen 和 F. Hutter，《神经架构搜索：综述》，《机器学习研究期刊》，第 20 卷，第 1 期，第 1997–2017 页，2019年。[205] S. Nakajima、R. Tomioka、M. Sugiyama 和 S. D. Babacan，《通过变分贝叶斯主成分分析实现完美维数恢复》，载于 NeurIPS，2012年。[206] L. Huang、C. Deng、S. Ibrahim、X. Fu 和 B. Yuan，《随机低秩张量分解的超大规模集成电路硬件架构》，载于 ACSCC，2021年。[207] N. Srivastava、H. Rong、P. Barua、G. Feng、H. Cao、Z. Zhang、D. Albonesi、V. Sarkar、W. Chen、P. Petersen 等，《T2S-Tensor：高效生成用于稠密张量计算的高性能空间硬件》，载于 FCCM，2019年。[208] N. Srivastava、H. Jin、S. Smith、H. Rong、D. Albonesi 和 Z. Zhang，《Tensaurus：用于混合稀疏-稠密张量计算的通用加速器》，载于 HPCA，2020年。[209] L. Liang、J. Xu、L. Deng、M. Yan、X. Hu、Z. Zhang、G. Li 和 Y. Xie，《快速搜索张量网络的最优收缩序列》，《IEEE信号处理精选主题期刊》，第 15 卷，第 3 期，第 574–586 页，2021年。[210] A. Fawzi、M. Balog、A. Huang、T. Hubert、B. Romera-Paredes、M. Barekatain、A. Novikov、F. J. R Ruiz、J. Schrittwieser、G. Swirszcz 等，《利用强化学习发现更快的矩阵乘法算法》，《自然》，第 610 卷，第 7930 期，第 47–53 页，2022年。[211] V. Pereyra 和 G. Scherer，《张量积的高效计算机运算及其在多维逼近中的应用》，Mathematics of Computation，第 27 卷，第 123 期，第 595–605 页，1973年。[212] S. van der Walt、S. C. Colbert 和 G. Varoquaux，《NumPy 数组：高效数值计算的结构》，Comput. Sci. Eng.，第 13 卷，第 2 期，第 22–30 页，2011年。[213] M. Abadi、P. Barham 等，《Tensorflow：大规模机器学习系统》，载于 USENIX，K. Keeton 和 T. Roscoe 编，2016年。[214] T. Chen、M. Li 等，《Mxnet：面向异构分布式系统的灵活高效机器学习库》，arXiv 预印本 arXiv:1512.01274，2015年，2015年。[215] X. Liu，《TenDeC++：C 语言中的张量分解库》，2020年。[在线]。可通过以下地址获取：https://github.com/osmint/TenDeC [216] Z. Cai 和 J. Liu，《利用人工神经网络逼近量子多体波函数》，Physical Review B，第 97 卷，第 3 期，第 035116 页，2018年。[217] K. Choo、G. Carleo、N. Regnault 和 T. Neupert，《神经网络量子态中的对称性与多体激发》，《物理评论快报》，第 121 卷，第 16 期，第 167204 页，2018年。[218] H. He、Y. Zheng、B. A. Bernevig 和 N. Regnault，《利用稳定子码的张量网络态计算纠缠熵》，Physical Review B，第 97 卷，第 12 期，第 125102 页，2018年。[219] L. Cincio、J. Dziarmaga 和 M. M. Rams，《二维多尺度纠缠重整化拟设：量子伊辛模型》，《物理评论快报》，第 100 卷，第 24 期，第 240603 页，2008年。[220] K. Batselier、A. Cichocki 和 N. Wong，《Meracle：将张量列车逐层构造性地转换为 MERA》，Communications on Applied Mathematics and Computation，第 3 卷，第 257–279 页，2021年。[221] J. A. Reyes 和 E. M. Stoudenmire，《用于机器学习的多尺度张量网络架构》，《机器学习：科学与技术》，第 2 卷，第 3 期，第 035036 页，2021年。