Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence (IJCAI-25)
Survey Track

Tensor Network: from the Perspective of AI4Science and Science4AI

Junchi Yan[12*], **Yehui Tang**[1], **Xinyu Ye**[1], **Hao Xiong**[13]
Xiaoqiu Zhong[1], **Yuhan Wang**[1] and **Yuan Qi**[23]
[1]Shanghai Jiao Tong University
[2]Shanghai Innovation Institute
[3]Fudan University
{yanjunchi, yehuitang, xinyu_ye, taxuexh, 21-zxq, wang.yuhan}@sjtu.edu.cn, qiyuan@fudan.edu.cn

Abstract

Tensor network has been a promising numerical tool for computational problems across science and AI. For their emerging and fast development especially in the intersection between AI and science, this paper tries to present a compact review, regarding both their applications and its own recent technical development including open-source tools. Specifically, we make the observations that tensor network plays a functional role in matrix compression and representation, information fusion, as well as quantum-inspired algorithms, which can be generally regarded as Science4AI in our survey. On the other hand, there is an emerging line of research in tensor network in AI4Science especially like learning quantum many-body physics by using e.g. neural network quantum state. Importantly, we unify tensorization methodologies across classical and modern architectures, and particularly show how tensorization bridges low-order parameter spaces to high-dimensional representations without exponential parameter growth, and further point out their potential use in scientific computing. We conclude the paper with outlook for future trends.

1 Motivation and Contribution

Tensor networks (TNs) have become a powerful computational tool for efficiently representing and processing high-dimensional data structures, playing a crucial role in addressing the growing complexity of problems both in artificial intelligence (AI) [Wall and D'Aguanno, 2021; Liang *et al.*, 2024; Su *et al.*, 2024] and scientific research [Bachmayr *et al.*, 2016; Schütt *et al.*, 2017; Orús, 2019]. TN is essentially a network of tensors, which are high-dimensional arrays, connected by contractions along shared indices. Each tensor can be visualized as a node, and the indices or legs represent the dimensions along which tensors are connected or contracted. The power of TNs lies in their ability to decompose and represent large data structures with potentially exponentially fewer parameters compared to what would be required without such structured decomposition [Bridgeman and Chubb, 2017].

*Correspondence author.

The inception of TNs is rooted in the realm of quantum physics, where they were originally developed as tools to solve the many-body quantum systems problem [Orús, 2019; Yuan *et al.*, 2021]. Their ability to factorize high-dim tensors into more manageable lower-dim components allows them to become indispensable in science especially physics. This factorization inherently resolves the curse of dimensionality, converting an exponential set of variables into a polynomial scale. Consequently, TNs have transcended their traditional boundaries and have become instrumental in general AI, from data science [Oseledets, 2011; Lu *et al.*, 2021] to machine learning [Stoudenmire and Schwab, 2016; Xiong *et al.*, 2024], and more recently in the design and optimization of AI models [Lu *et al.*, 2021; Xiong *et al.*, 2025].

In AI, TNs enable the efficient handling of high-order correlations in data, a task where traditional linear methods may fall short. They have been successfully integrated into the training processes of neural networks [Kossaifi *et al.*, 2020; Panahi *et al.*, 2020], providing substantial reductions in parameter sizes through tensor decompositions—such as the Canonical Polyadic, Tucker, and Tensor Train decompositions—while preserving model expressiveness. This integration is not only computationally efficient but also enhances model scalability, as cited in recent studies that highlight reductions in operational complexities in deep learning architectures while maintaining or even enhancing accuracy [Kossaifi *et al.*, 2020; Wang *et al.*, 2023].

The utilization of TNs in scientific contexts is equally transformative. In fields like material science and quantum computing, they facilitate precise simulations of molecular interactions and quantum states, processes that are otherwise computationally prohibitive [Simeon and De Fabritiis, 2024]. For example, TNs allow for the simulation of quantum circuits through matrix product states and operators, enriching scientific insights with computational models that mirror physical behaviors [Phan *et al.*, 2018].

The adoption of TNs in both AI and scientific research underscores a critical paradigm shift: the recognition that high-dimensional data and complex correlations can be systematically captured through structured and physics-inspired decompositions. However, this interdisciplinary subject has also exposed gaps in understanding how TNs fundamentally unify principles from science and AI. A key motivation of this survey lies in revealing a picture that empowers researchers to

Notations	Descriptions
$\|\cdot\rangle$	The Dirac notation of a pure quantum state.
ρ	The density matrix / operator.
$\mathbf{I}$	Identity matrix.
$\otimes$	Tensor product.
$\times_n$	Mode-n product.
$\boldsymbol{i}$	Imaginary unit.
$\|a, b\rangle$	The abbreviation of $\|a\rangle \otimes \|b\rangle$.
$\mathbb{R}$	Field of real numbers.
$\mathbb{C}$	Field of complex numbers.

Table 1: Summary of notations used in the paper.

leverage TNs not merely as algorithmic tools but as conceptual bridges between disciplines. **The paper highlights:**

1) **AI4Science:** TNs synergize with deep learning to model high-dimensional quantum systems, enabling precise simulation of entanglement dynamics and material properties beyond classical computational limits.
2) **Science4AI:** TN-inspired tensor decompositions compress high-dimensional AI models, with the potential of reducing model's size while preserving expressivity and enhancing efficiency through multilinear algebraic priors.

2 Preliminaries on Tensor Network

Normal letters represent scalars (e.g., a). Bold lowercase letters represent vectors ($\mathbf{a}$). Bold uppercase letters denote matrices ($\mathbf{A}$). Calligraphic uppercase letters are used for tensors ($\mathcal{T}$), which generalize matrices to higher dimensions. The description of the notations are given in Tab. 1.

2.1 Tensor Operations and Notations

An element of a high-order tensor can be accessed by multiple indices. E.g., an element T_{ijk} of a 3-D tensor $\mathcal{T} \in \mathbb{R}^{m\times n\times p}$ can be accessed by the indices $i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$ and $k \in \{1, 2, \ldots, p\}$.

Tensor Product: It generalizes the outer product of vectors. For two tensors $\mathbf{A} \in \mathbb{R}^{I_1\times I_2}$ and $\mathbf{B} \in \mathbb{R}^{J_1\times J_2}$, the outer product produces a higher-order tensor $\mathcal{C} \in \mathbb{R}^{I_1\times I_2\times J_1\times J_2}$ with elements:

$$\mathcal{C}_{i_1,i_2,j_1,j_2} = \mathbf{A}_{i_1,i_2}\mathbf{B}_{j_1,j_2}$$

Tensor product is essential in TNs, combining tensors into higher-dimensional structures.

Mode-n Multiplication: It is a generalization of matrix multiplication. Given a tensor $\mathcal{T} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ and a matrix $\mathbf{A} \in \mathbb{R}^{J_n\times I_n}$, mode-$n$ multiplication is defined as:

$$(\mathcal{T} \times_n \mathbf{A})_{i_1,\ldots,i_{n-1},j_n,i_{n+1},\ldots,i_N} = \sum_{i_n} \mathcal{T}_{i_1,\ldots,i_N}\mathbf{A}_{j_n,i_n}.$$

This is crucial in tensor decompositions like Tucker decomposition, enabling the manipulation of high-dimensional data.

2.2 Tensor Decomposition

Tensor decomposition techniques factorize a high-order tensor into a product of lower-order tensors, similar to singular value decomposition (SVD). These decompositions help approximate tensors with lower-rank components, reducing computational complexity.

Canonical Polyadic (CP) Decomposition: It expresses a tensor as a sum of rank-1 tensors. For a 3-D tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$, the CP decomposition is:

$$\mathcal{T} \approx \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{c}_r,$$

where $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$, and R is the rank of the decomposition, i.e., the number of components in the low-rank approximation.

Tucker Decomposition: Ass a more flexible tensor factorization method, it decomposes a tensor into a core tensor and factor matrices. For 3D $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$, its decomposition is:

$$\mathcal{T} \approx \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C},$$

where $\mathcal{G} \in \mathbb{R}^{R_1\times R_2\times R_3}$ is the core tensor. $\mathbf{A} \in \mathbb{R}^{I\times R_1}$, $\mathbf{B} \in \mathbb{R}^{J\times R_2}$, and $\mathbf{C} \in \mathbb{R}^{K\times R_3}$ factor matrices.

Tensor Train (TT) Decomposition: The Tensor Train decomposition approximates high-dimensional tensors by a chain of low-rank tensors. For a 3D tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$, the TT decomposition is expressed as:

$$\mathcal{T} \approx \sum_{r_1,r_2} \mathbf{A}_{r_1} \times_1 \mathbf{B}_{r_1,r_2} \times_2 \mathbf{C}_{r_2},$$

where $\mathbf{A}_{r_1} \in \mathbb{R}^{I\times r_1}$, $\mathbf{B}_{r_1,r_2} \in \mathbb{R}^{r_1\times J\times r_2}$, and $\mathbf{C}_{r_2} \in \mathbb{R}^{r_2\times K}$ are the factor tensors.

3 Tensor Network for Science & Engineering

TNs have been an effective tool for science, especially in quantum mechanics related areas, e.g., chemistry, quantum many-body physics, quantum field theory etc. In this section, we primarily focus on the applications of tensor networks in the efficient simulation of quantum many-body systems and the calculation of the ground state. We in general take an AI4Science perspective in this section.

3.1 MPS and PEPS in Quantum Systems

TNs have become essential for simulating quantum many-body systems as they efficiently represent complex quantum states by decomposing them into smaller, manageable tensors. These networks are especially powerful when dealing with larger many-body systems, where traditional methods like exact diagonalization or full wave function representations are computationally prohibitive. Matrix Product States (MPS, Fig. 1a) and Projected Entangled Pair States (PEPS, Fig. 1c) are most widely used among the various types of TNs, as they provide powerful tools for simulating quantum systems, particularly in low-dimensional settings.

MPS is a subclass of TNs that represent quantum states on a one-dimensional lattice. The quantum state is represented as a product of matrices associated with each lattice site, with their contraction yielding the wave function. MPS is particularly useful for one-dimensional systems or systems with weak entanglement, where the state can be efficiently approximated using relatively low bond dimensions. Specifically, an MPS representation of a quantum state $|\psi\rangle$

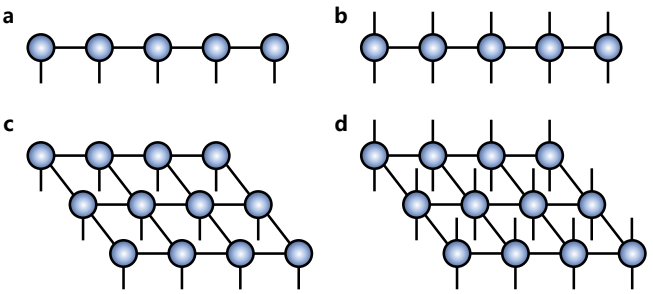

Figure 1: Schematic diagram of several Tensor Networks commonly used in science: (a) Matrix Product States (MPS); (b) Matrix Product Operators (MPO); (c) Projected Entangled Pair States (PEPS); (d) Projected Entangled Pair Operators (PEPO).

on a chain of L sites for open boundary conditions takes the form [Schollwöck, 2011]:

$$|\psi\rangle = \sum_{\sigma_1,\ldots,\sigma_L} \mathcal{A}^{\sigma_1}\mathcal{A}^{\sigma_2}\ldots\mathcal{A}^{\sigma_L}|\sigma_1,\ldots,\sigma_L\rangle. \quad (1)$$

Here, $\mathcal{A}^{\sigma_i}$ are tensors of dimension $D \times D \times d$, with d the dimension of the local Hilbert space and D the bond dimension governing the amount of entanglement captured. MPS have proven highly efficient for simulating one-dimensional systems, as they allow for a compact representation of quantum states with polynomial scaling in L and D.

PEPS, on the other hand, is a more general class of TNs suited for two-dimensional systems. PEPS represents quantum states on a two-dimensional lattice and extends the MPS by using tensors that have additional indices corresponding to the two-dimensional grid. These tensors are connected in a way that mimics the structure of quantum entanglement across a 2D lattice [Orús, 2014]. A PEPS representation of a quantum state on a lattice of $L \times L$ sites takes the following form:

$$|\psi\rangle = \sum_{\{\sigma_i\}} \sum_{a_{i,j}} \prod_{i,j} \mathcal{A}^{\sigma_{i,j}}_{a_{i,j}} |\sigma_{i,j}\rangle. \quad (2)$$

Each tensor $\mathcal{A}^{\sigma_{i,j}}_{a_{i,j}}$ at position (i,j) is a $d \times D \times D \times D$ tensor (where D is the bond dimension), and the network structure allows for an efficient representation of the wavefunction. PEPS are particularly well-suited for representing strongly correlated systems, such as those occurring in higher-dimensional lattice systems, and for capturing complex patterns of entanglement.

While MPS is highly efficient for one-dim systems, PEPS provides a natural extension for higher-dim systems, where the entanglement is generally more complex and cannot be simply captured by MPS representation. Both MPS and PEPS are part of the broader TN framework, which allows for the efficient representation and manipulation of quantum states. Besides, neural network-based models have been investigated recently in representing and simulating quantum systems and demonstrate superior performance in tasks such as tomography [Quek *et al.*, 2021], properties prediction [Tang *et al.*, 2024; Tang *et al.*, 2025] and quantum sensing [Cimini *et al.*, 2021]. Furthermore, hybrid models that integrate the physical priors of TNs with the expressive power of neural networks have been proposed [Chen *et al.*, 2023], offering a promising avenue for simulation of complex quantum systems.

3.2 TNs for Ground State Calculations

One of the primary applications of MPS and PEPS is the calculation of the ground state of quantum many-body systems. The general approach to finding the ground state using MPS or PEPS involves expressing the Hamiltonian in the corresponding TN formalism (MPO for MPS and PEPO for PEPS), and then minimizing the energy expectation value through variational optimization.

For MPS, the ground state $|\psi\rangle$ is approximated by iteratively optimizing the tensors A^{σ_i} at each lattice site. The ground state energy is minimized by adjusting the tensors to lower the expectation value $E = \frac{\langle\psi|\widehat{H}|\psi\rangle}{\langle\psi|\psi\rangle}$. The primary technique used in this optimization is the Density Matrix Renormalization Group (DMRG) method, which has proven to be highly efficient for one-dimensional systems [Schollwöck, 2011]. For example, for the antiferromagnetic Heisenberg chain, the system Hamiltonian $\widehat{H}$ is:

$$\widehat{H} = \sum_{i=1}^{L-1} \frac{J}{2}\widehat{S}_i^+\widehat{S}_{i+1}^- + \frac{J}{2}\widehat{S}_i^-\widehat{S}_{i+1}^+ + J^z\widehat{S}_i^z\widehat{S}_{i+1}^z - h\sum_i \widehat{S}_i^z. \quad (3)$$

The Hamiltonian can also be represented by sums of tensor products of operators, $\widehat{H} = J^z\widehat{S}_1^z \otimes \widehat{S}_2^z \otimes \widehat{I} \otimes \widehat{I} \otimes \widehat{I} \cdots + \widehat{I} \otimes J^z\widehat{S}_2^z \otimes \widehat{S}_3^z \otimes \widehat{I} \otimes \widehat{I} \cdots + \cdots$, which directly corresponds to Matrix Product Operators (MPO, Fig. 1b).

DMRG optimizes the MPS by adjusting the tensors at each site through alternating sweeps over the system. At each site, a local eigenvalue problem of the tensor is solved, and the MPS is updated using Singular Value Decomposition (SVD) to maintain the desired bond dimension. This iterative optimization continues until the energy converges to its minimum. DMRG is particularly efficient for one-dimensional systems and can handle large systems with hundreds of sites. However, its applicability is relatively limited for systems with long-range interactions or in higher dimensions, where the entanglement grows more rapidly and MPS may not efficiently capture the full complexity of the quantum state [Markov and Shi, 2008].

For PEPS, the Hamiltonian is expressed as a Projected Entangled Pair Operator (PEPO, Fig. 1d), which is a generalization of the MPO for two-dimensional systems. The PEPO facilitates the calculation of the energy expectation value $\langle\psi|\widehat{H}|\psi\rangle$ through the contraction of the PEPO and PEPS tensors [Orús, 2014]. Similar to MPS, the PEPS tensors are optimized iteratively to minimize the energy. The optimization process involves adjusting the tensors at each lattice site while keeping the others fixed, alternating between different sweeps over the lattice to ensure convergence to the ground state.

4 Tensor Network for Machine Learning

TNs have become a powerful tool in machine learning, offering improvements in both efficiency and expressiveness. This chapter focuses on two main applications: compressing neural network weights and parameter-efficient fine-tuning, and

using tensor networks for expressive data representation that captures local and global correlations in tasks like network embedding and multimodal fusion. In this section, we try to take a Science4AI perspective.

4.1 TN for Weighted Matrix Compressing

In neural networks, the trainable parameters are typically represented as matrices or higher-order tensors with considerable size, which introduce significant computational and resource challenges. Moreover, as the model becomes more complex, there is a higher risk of overfitting, which can negatively impact the model's ability to generalize to unseen data.

Recent research has shown that generalization can be improved by restricting or refining the degrees of freedom within these high-dimensional tensors. Shallow tensor network methods, such as MPS [Perez-Garcia *et al.*, 2007] and MPO [Pirvu *et al.*, 2010], offer an elegant solution by representing these large tensors more compactly, achieving remarkable compression rates without sacrificing performance. The underlying principle is that a dense weight matrix is redundant for accurate linear transformations in neural networks due to the predominantly short-range correlations in data. For instance, in a fully connected layer, the rank of the weight matrix is inherently constrained due to short-range correlations or entanglements among input pixels [Denil *et al.*, 2013]. Consequently, a lower-rank matrix can efficiently substitute the original without loss of predictive performance.

Many works have studied the efficiency of tensorizing the components of neural networks to bridge low-order parameter spaces to high-dimensional representations. For example, the MPO representation can significantly reduce the number of variational parameters, as its parameter count scales linearly with system size. [Gao *et al.*, 2020] proposed to use MPO to represent linear transformation matrices in deep neural networks, and their results show MPO cannot only improve the efficiency in training and reduce the memory space, but also slightly improve the test accuracy using much fewer number of parameters than in the original networks. Specifically, they first reshape the weight matrix $\mathbf{W}$ into a 2n-indexed tensor $\mathbf{W}_{j_1 \ldots j_n, i_1 \ldots i_n}$. The MPO representation of $\mathbf{W}$ is obtained by factorizing it into a product of n local tensors $\mathrm{Tr}\left(w^{(1)}[j_1, i_1] w^{(2)}[j_2, i_2] \cdots w^{(n)}[j_n, i_n]\right)$, where $w^{(k)}[j_k, i_k]$ is a $D_{k-1} \times D_k$ matrix. The tensor elements of $w^(k)$, instead of the elements of $\mathbf{W}$, are the variational parameters in the training procedure of deep neural networks. The number of parameters increases with the increase of the bond dimension D_k, which serves as a tunable parameter that controls the expressive power.

[Qing *et al.*, 2023] proposed automatically differentiable tensor network (ADTN), which encodes the variational parameters into the contraction of tensor networks that contains exponentially fewer parameters, and utilizes automatic differentiation technique to reach the optimal accuracy after compression. The experiments showed that they can compress two linear layers in VGG-16 with approximately 10^7 parameters to two ADTN's with just 424 parameters, where the testing accuracy on CIFAR-10 is improved from 90.17% to 91.74%. [Xie *et al.*, 2024] leveraged the emerging tensor ring (TR) factorization to compress the neural network and proposed an approach based on prime factorization that simultaneously identifies the optimal tensor reshaping and TR decomposition. To identify the optimal execution order, they construct a novel tree structure to schedule the execution of core tensors, minimizing computational complexity.

Pre-trained large language models (LLMs) have revolutionized natural language processing (NLP) across various tasks. However, as model sizes continue to grow, full fine-tuning becomes impractical due to its high computational and memory costs. To address this challenge, parameter-efficient fine-tuning (PEFT) techniques have been developed, allowing model adaptation by updating only a small subset of parameters. Low-Rank Adaptation (LoRA) [Hu *et al.*, 2021] is a classic PEFT methods, which fine-tunes LLMs by introducing low-rank matrices into the pre-trained model's weight updates. Recently, some research has extended LoRA using tensor-based techniques [Bershatsky *et al.*, 2024] to reduce the number of trainable parameters within the low-rank framework. Considering that low-rank approximations may sometimes result in a performance gap compared to full fine-tuning, especially for complex tasks, QuanTA [Chen *et al.*, 2024] introduced a novel efficient high-rank fine-tuning inspired by quantum circuit structures, surpassing the limitations of LoRA. However, the hyperparameters in QuanTA, such as the number of tensors applying on the same axes, have not been optimized. Therefore, choosing an optimal set of tensors could further enhance their performance.

The requirement for precise tensor-rank selection and effective decomposition strategy poses computational and algorithmic hurdles, often leading to NP-hard problem. The permutation of tensor entries and the selection of optimal tensor-rank configurations can impact the efficacy drastically, necessitating efficient heuristic approaches [Li *et al.*, 2022]. Despite these challenges, recent works in TN structure search algorithms, e.g. Alternating Local Enumeration (TnALE) [Li *et al.*, 2023], offer promising pathways to avoid combinatorial explosion and prove exponentially convergence speedup compared to previous study [Li *et al.*, 2022].

4.2 TN for Expressive Representation

TNs also provide a mathematically grounded framework for efficient representation learning, inspired by the tensor product of n quantum states, which spans a 2^n-dimensional Hilbert space. By strategically composing low-dimensional sub-tensors, these models achieve exponential parameter compression while preserving high-dimensional semantic relationships. This dual capability to capture both local interactions and global correlations makes them highly expressive.

In natural language processing, the word2ket [Panahi *et al.*, 2020] method decomposes word embeddings into rank-r order-n entangled tensors using the formulation:

$$\mathbf{v} = \sum_{k=1}^{r} \bigotimes_{j=1}^{n} \mathbf{v}_{jk} \quad \text{with } \mathbf{v}_{jk} \in \mathbb{R}^q. \tag{4}$$

It reduces the storage complexity of traditional $d \times p$ embedding matrices from $O(dp)$ to $O(rnq \log p/q)$. It has shown significant space savings in text summarization tasks.

Extending node2ket [Xiong *et al.*, 2024] to the setting of network embedding, the framework constructs node representations as product states across orthogonal subspaces:

$$\mathbf{x}_i = \bigotimes_{c=1}^{C} \mathbf{u}_{ic} \quad \text{where } \|\mathbf{u}_{ic}\|_2 = 1, \tag{5}$$

enabling multi-scale relationship modeling through hierarchical inner products:

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \prod_{c=1}^{C} \langle \mathbf{u}_{ic}, \mathbf{u}_{jc} \rangle, \tag{6}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operation. This architecture has been successfully applied to large-scale networks, achieving high accuracy in tasks such as network reconstruction and node classification.

Emerging extensions integrate TNs with deep architectures, notably through quantum-inspired dynamic rank adaptation [Hua *et al.*, 2022] and hybrid tensor-transformer designs [Ma *et al.*, 2019], enabling TNs as a unifying paradigm for high-dimensional representation learning.

TNs with their multilinear structure and efficient data representation, can capture complex relationships within data while reducing computational complexity. Compared to traditional DNNs, TNs show promise in tasks like multimodal data fusion [Baltrusaitis *et al.*, 2019]. Integrating TNs could improve the performance of deep learning models and address some of their limitations.

TNs have proven to be a powerful tool for efficiently representing and fusing high-dimensional data. Zadeh et al. [Zadeh *et al.*, 2017] introduced TNs with deep information fusion layers, named Tensor Fusion Layers (TFLs), for multimodal sentiment analysis. TFLs were designed to capture both intra-modality and inter-modality dynamics, effectively aggregating multimodal interactions, which was crucial for tasks like multimodal sentiment analysis involving text, visual, and acoustic data.

TFL begins by embedding the three vectors $\mathbf{z}_t$, $\mathbf{z}_v$, and $\mathbf{z}_a$ for each modality, rather than use raw data types. It then concatenates a scalar 1 with each of these embedded vectors:

$$\mathbf{z}'_t = [\mathbf{z}_t^\top, 1], \quad \mathbf{z}'_v = [\mathbf{z}_v^\top, 1], \quad \mathbf{z}'_a = [\mathbf{z}_a^\top, 1].$$

Next, it computes the outer product of these concatenated vectors to form a feature tensor $\mathbf{Z}$:

$$\mathbf{Z} = \mathbf{z}'_t \otimes \mathbf{z}'_v \otimes \mathbf{z}'_a.$$

This tensor is subsequently processed by a two-layer fully connected neural network to produce a prediction. The TFL captures both unimodal and multimodal interactions, making it more expressive than simple concatenation-based fusion, which only models unimodal interactions.

However, while TFLs offer strong fusion capabilities, their computational complexity increases exponentially as the number of modalities increases, which can lead to inefficient training. To address this, Low-Rank Multimodal Fusion (LMF) [Liu *et al.*, 2018] was introduced to reduce the computational burden and overfitting risk. LMF employs a special BTT (block term decomposition) layer, reducing the complexity of the TFL from $O\left(\prod_{m=1}^{M} d_m\right)$ to $O\left(d_h \sum_{m=1}^{M} d_m\right)$, where d_m represents the dimensions of each modality, and d_h is the hidden dimension.

Though TFL and LMF show improved fusion, they restrict the order of interactions, which leads to a loss of higher-order interaction information. To tackle this issue, the Polynomial Tensor Product (PTP) [Hou *et al.*, 2019] block is proposed. PTP merges all feature vectors into a long feature vector:

$$\mathbf{z}_{12\cdots M} = [\mathbf{z}_1^\top; \mathbf{z}_2^\top; \cdots; \mathbf{z}_M^\top].$$

It then creates a polynomial feature tensor of degree P as:

$$\mathbf{Z}^P = \mathbf{z}_{12\cdots M} \otimes \mathbf{z}_{12\cdots M} \otimes \cdots.$$

This tensor is processed by a tensor layer, such as a CP (canonical polyadic) layer, to model high-order interactions:

$$h = \text{ReLU}\left(\text{FFN}(\mathbf{W}_1 \mathbf{z}_{12\cdots M}; \cdots; \mathbf{W}_P \mathbf{z}_{12\cdots M}; \Lambda)\right),$$

where $\mathbf{W}_i$ are weight matrices and Λ is a learnable diagonal matrix. The PTP block, similar to deep polynomial neural networks, captures all nonlinear and high-order interactions.

In addition to multimodal data fusion, TNs can also capture high-order relationships within each modality. For example, in image-sentence matching tasks, TNs help overcome the limitations of traditional methods, which typically match global features in a shared space, neglecting intra-modality dynamics. The Cross-Modal Hybrid Feature Fusion (CMHF) framework [Xu *et al.*, 2021] directly learns image-sentence similarity by fusing multimodal features with both inter- and intra-modality relations. By using flexible attention mechanisms, TNs capture fine-grained interactions within each modality, improving the overall similarity learning process. Therefore, even outside of multimodal tasks, TNs are still highly effective in general machine learning. In these cases, TNs can be applied in a similar manner to decompose and reassemble feature vectors at each layer, capturing higher-order interactions between the data.

5 Tensor Network Development and Tools

Computational techniques and tools essential for efficiently handling TNs in modern applications. This section discusses efficient algorithms for tensor operations, parallel and distributed computing with GPUs, and key software for TNs. Additionally, we highlight advancements in compression and resource management, focusing on adaptive techniques for optimizing efficiency, memory usage, and accuracy.

5.1 Efficient Algorithms for Tensor Operations

Tensor contraction and decomposition are helpful for mitigating computational overhead while maintaining data integrity in high-dimensional space. The challenge lies in optimizing the contraction sequence to minimize cost. Strategies like contraction search heuristics and dynamic programming have proven effective, but further optimization for large-scale settings is needed [Taylor, 2024]. The choice of contraction path significantly impacts performance, with recent advances focusing on hybrid strategies combining heuristic and deterministic methods to improve efficiency [Li *et al.*, 2022].

Decomposition techniques like CP, Tucker, and Tensor Train decompositions simplify complex tensor structures,

boosting efficiency. These methods offer different trade-offs between accuracy and computational cost. CP is simple and interpretable but suffers from high complexity, particularly for higher-order tensors [Phan *et al.*, 2018]. Tucker provides more flexibility by allowing multi-dimensional cores, at the cost of increased computational expense. TT handles large-scale tensors by approximating them through low-rank tensor contractions [Kossaifi *et al.*, 2020].

Emerging methods, such as Hierarchical Tucker (HT) decomposition, use hierarchical partitioning to decompose tensors into smaller, more manageable components, aiding in both data compression and feature preservation [Cichocki *et al.*, 2017]. By recursively breaking down high-dimensional data into lower-dimensional sub-tensors, HT efficiently captures multi-level correlations within the data. This hierarchical structure significantly alleviates the curse of dimensionality, which typically leads to exponential growth in parameter space for high-dimensional datasets. As a result, HT becomes particularly advantageous for scalable applications where computational efficiency is critical.

Current research emphasizes integrating TNs with machine learning frameworks, where tensor operations optimize model sizes and computational loads [Yang *et al.*, 2023]. This intersection opens new avenues for applying tensorization, such as tuning neural network architectures to enhance performance and reduce overhead [Wang *et al.*, 2023].

As the field progresses, it is crucial to develop algorithms that adaptively manage resources and leverage hardware advancements while prioritizing efficiency in complex tensor operations. Future research should explore innovative methods like quantum-inspired computations and adaptive learning strategies [Rieser *et al.*, 2023]. These efforts will help overcome computational barriers and broaden tensor network applications across AI and computational science.

5.2 Parallel & Distributed Computing Techniques

Central to parallel and distributed computing is effective partitioning of computations across multiple processors while minimizing communication overhead. Embedding tensor contractions in GPUs has accelerated computations by optimizing kernel operations and reducing data transfer overhead between CPU and GPU memory [Schutski *et al.*, 2020], enabling manipulating larger tensors than would be possible sequentially. To further enhance scalability, distributed tensor computations allocate workloads across multiple processors. Frameworks like MPI enable inter-node communication, allowing tensor decomposition into smaller, concurrently processable chunks [Cichocki, 2014b]. This approach is advantageous for large-scale simulations and data analytics.

An emerging trend is dynamic load balancing and adaptive resource allocation, which ensures even workload distribution, minimizing idle times and optimizing resource usage. These strategies are crucial as TN applications expand into more heterogeneous and dynamic environments [Cichocki *et al.*, 2017]. Additionally, cloud-based platforms increasingly support scalable resource allocation, allowing for elastic scaling to meet tensor computation demands, particularly in real-time data processing [Cichocki, 2014b].

As data size and dimensionality increases, maintaining consistency and synchronization across nodes can escalate overhead, reducing parallelism benefits. Designing algorithms to map tensor operations efficiently onto hardware, especially with irregular or sparse tensors, remains a significant challenge [Kanakagiri and Solomonik, 2024].

Innovative strategies include heuristic and probabilistic graphical models to optimize contraction paths without exhaustive searches [Schutski *et al.*, 2020], reducing complexity and enhancing performance.

Looking ahead, integrating machine learning to predict optimal parallel execution configurations can further improve efficiency and scalability. Such models could dynamically adjust task partitioning and resource allocation based on computation patterns and data structures. Quantum computing principles may also offer novel solutions to scaling bottlenecks [Robeva and Seigal, 2019].

5.3 Software Tools and Libraries

Key tools include deep learning libraries such as TensorFlow and PyTorch, which have integrated TNs into their ecosystems. These platforms allow for the seamless expansion of traditional neural network architectures into tensor-inspired frameworks. TensorFlow's adaptable computational graph and PyTorch's dynamic computation graph support real-time execution, offering flexibility for researchers exploring TN algorithms in machine learning frameworks. These platforms are useful for developing high-dimensional models [Cichocki, 2014a].

Specialized tensor software like TensorLy offers powerful abstractions tailored for tensor operations. It supports a wide range of decompositions, including CPD and Tucker, and allows customization of decomposition processes to suit specific data properties, making it valuable for scientific applications [Rabanser *et al.*, 2017].

ITensor focuses on quantum simulations and condensed matter physics, specializing in high-precision quantum TN computations. It treats tensor manipulations as accessible objects, in contrast to traditional linear algebra approaches, and uses a graphical representation of tensors to handle complex quantum states and operations [Cichocki *et al.*, 2017].

TensorNetwork and SciNet blend NumPy's simplicity with TN contraction and optimization capabilities. Enhanced by efficient numerical schemes, these libraries support large tensor data and benefit from GPU acceleration for practical feasibility in scientific computing [Kolda and Hong, 2020].

Graphical tools like TensorTrace and GuiTeNet facilitate tensor visualization, aiding researchers in building, testing, and visualizing tensor models, thus reducing cognitive load and enhancing model interpretability [Cheng *et al.*, 2020].

Emerging trends emphasize improving computational efficiency, resource management, and scalability. Integrating tensor compression and parallel computation techniques addresses the curse of dimensionality [Yin *et al.*, 2021]. These libraries also support distributed computing and adaptive resource allocation for sustainable AI models that minimize energy consumption and promote efficient tensor handling.

Despite advancements, challenges remain, particularly in ensuring numerical stability when working with high-dimensional tensors. Hybrid tensor architectures and on-the-

fly compression strategies continue to be key areas for research, aiming to develop more robust tools for various scientific and industrial applications [Wu *et al.*, 2020].

5.4 TN Compression and Resource Management

TN compression uses techniques like CP, Tucker, and TT decompositions to reduce the storage and computational demands of high-dimensional data. These methods transform high-dimensional tensors into lower-dimensional components, offering substantial space savings [Cichocki *et al.*, 2014; Rabanser *et al.*, 2017].

TT decomposition is particularly notable for its linear storage costs and ability to preserve key properties of the decomposed components [Cichocki *et al.*, 2017]. Its hierarchical structure and the Tensor Ring variant allow for flexible rank adjustments that optimize storage while maintaining data fidelity. These efficiencies are evident in quantum computing and machine learning, where complex data are handled effectively [Biamonte *et al.*, 2017; Tang and Yan, 2022]. A critical aspect of compression is adaptive rank selection, where the ranks of decomposed components adjust dynamically to data complexity. This adaptability ensures efficient storage without compromising accuracy, facilitating the use of TNs on memory-limited hardware and improving performance on GPUs and high-performance platforms [Charlier *et al.*, 2021].

Emerging trends highlight sparse tensor computing to optimize storage and performance. Sparse tensor can reduce storage needs without sacrificing computational accuracy. Compiler support systems like MLIR assist in generating optimized sparse tensor operations, easing developers' workload [Bik *et al.*, 2022]. These innovations underscore the benefits of tensor compression and resource management.

However, balancing compression with precision remain challenging. Excessive compression can compromise essential data features, affecting model accuracy. Further exploration of methods to maintain precision while employing aggressive compression is essential [Ma *et al.*, 2024]. Advancing adaptive ranking techniques and sparse tensor algorithms are critical for progress. Future research should focus on refining cost-based tensor program optimizers, allowing users to balance storage, computation speed, and resource usage. Studies could lead to new TN applications, especially in energy-efficient AI, where resource management is vital.

5.5 Evaluation of Computational Trade-Offs

A key trade-off in TNs is between computational precision and operational efficiency. High-precision tensor operations, necessary in quantum simulations and scientific computations, demand significant resources and can lead to long execution times. For example, the precision needed in quantum many-body simulations increases the complexity of tensor contractions and decompositions, requiring a careful balance [Ferris and Poulin, 2013].

Efficient use of computational resources is another key aspect of TN implementations. Memory footprint and compute time are integral to evaluating performance across different scales. Recent studies propose methods to improve resource allocation efficiency, reducing overhead without significant accuracy loss [Yin *et al.*, 2021].

Effective resource management integrates memory-efficient algorithms and hardware acceleration. The application of GPUs and distributed frameworks helps mitigate computational constraints, leveraging parallel processing [Wu *et al.*, 2024]. Benchmarking plays a vital role in evaluating TN algorithms. Standardized workflows and performance metrics are used to assess computational trade-offs and optimize contraction paths and network configurations [Ramírez *et al.*, 2024].

Performance analysis tools like ITensor and TensorNetwork enable systematic comparisons of TN algorithms under various conditions [Fishman *et al.*, 2022]. Despite progress, challenges remain, such as optimizing multi-way arrays and scaling across diverse platforms. Continuous research into novel decompositions, contraction strategies, and hardware advancements is needed to enhance the adaptability and effectiveness of TNs [Li *et al.*, 2022].

6 Conclusion and Outlook

In this paper, we have explored the role of TNs in both AI and scientific research. We have demonstrated how TNs, initially developed for quantum many-body problems, have evolved to serve as powerful tools for addressing complex, high-dimensional data in AI applications, ranging from deep learning to machine learning model optimization. In scientific computing, TNs facilitate efficient simulations of quantum systems, material properties, and molecular interactions, thus overcoming the limitations of traditional methods. Through their ability to decompose high-dimensional data into manageable components, TNs enable both significant computational savings and enhanced model expressivity.

Looking ahead, the potential for TNs to bridge AI and scientific disciplines is immense. In the context of AI4Science, TNs can be integrated with deep learning models to offer new insights into quantum systems and complex physical phenomena, with applications ranging from quantum chemistry to material science. On the other hand, Science4AI applications are likely to see a significant expansion, with TN-inspired tensor decompositions offering ways to compress large-scale AI models, reducing their size while maintaining or improving performance. Furthermore, the combination of TNs with hybrid quantum-classical machine learning models may open new frontiers in both AI and quantum computing.

As these interdisciplinary developments progress, there is an increasing need for unified frameworks and open-source tools that facilitate the seamless application of TNs across domains. This paper provides a foundation for future research and offers an optimistic outlook for the further evolution of tensor networks in solving some of the most challenging problems in science and AI.

Acknowledgments

This work was in part supported by National Natural Science Foundation of China (92370201, 62222607, 623B1009).

References

[Bachmayr *et al.*, 2016] Markus Bachmayr, Reinhold Schneider, and André Uschmajew. Tensor networks and hierarchical ten-

sors for the solution of high-dimensional partial differential equations. *Foundations of Computational Mathematics*, 16:1423–1472, 2016.

[Baltrusaitis *et al.*, 2019] Tadas Baltrusaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *TPAMI*, page 423–443, February 2019.

[Bershatsky *et al.*, 2024] Daniel Bershatsky, Daria Cherniuk, Talgat Daulbaev, Aleksandr Mikhalev, and Ivan Oseledets. Lotr: Low tensor rank weight adaptation. *arXiv:2402.01376*, 2024.

[Biamonte *et al.*, 2017] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 2017.

[Bik *et al.*, 2022] Aart J. C. Bik, Penporn Koanantakool, T. Shpeisman, Nicolas Vasilache, Bixia Zheng, and Fredrik Kjolstad. Compiler support for sparse tensor computations in mlir. *TACO*, 19:1 – 25, 2022.

[Bridgeman and Chubb, 2017] Jacob C Bridgeman and Christopher T Chubb. Hand-waving and interpretive dance: an introductory course on tensor networks. *Journal of physics A: Mathematical and theoretical*, 50(22):223001, 2017.

[Charlier *et al.*, 2021] Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin, and Ghislain Durif. Kernel operations on the gpu, with autodiff, without memory overflows. *JMLR*, 22(74):1–6, 2021.

[Chen *et al.*, 2023] Zhuo Chen, Laker Newhouse, Eddie Chen, Di Luo, and Marin Soljacic. Antn: Bridging autoregressive neural networks and tensor networks for quantum many-body simulation. *Advances in neural information processing systems*, 36:450–476, 2023.

[Chen *et al.*, 2024] Zhuo Chen, Rumen Dangovski, Charlotte Loh, Owen Dugan, Di Luo, and Marin Soljačić. Quanta: Efficient high-rank fine-tuning of llms with quantum-informed tensor adaptation. *arXiv:2406.00132*, 2024.

[Cheng *et al.*, 2020] Lei Cheng, Zhongtao Chen, Qingjiang Shi, Yik-Chung Wu, and S. Theodoridis. Towards flexible sparsity-aware modeling: Automatic tensor rank learning using the generalized hyperbolic prior. *IEEE Transactions on Signal Processing*, 70:1834–1849, 2020.

[Cichocki *et al.*, 2014] A. Cichocki, D. Mandic, L. D. Lathauwer, Guoxu Zhou, Qibin Zhao, C. Caiafa, and A. Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32:145–163, 2014.

[Cichocki *et al.*, 2017] A. Cichocki, A. Phan, Qibin Zhao, Namgil Lee, I. Oseledets, Masashi Sugiyama, and D. Mandic. Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives. *Found. Trends Mach. Learn.*, 9:431–673, 2017.

[Cichocki, 2014a] A. Cichocki. Era of big data processing: A new approach via tensor networks and tensor decompositions. *arXiv:1403.2048*, 2014.

[Cichocki, 2014b] A. Cichocki. Tensor networks for big data analytics and large-scale optimization problems. *arXiv:1407.3124*, 2014.

[Cimini *et al.*, 2021] Valeria Cimini, Emanuele Polino, Mauro Valeri, Ilaria Gianani, Nicolò Spagnolo, Giacomo Corrielli, Andrea Crespi, Roberto Osellame, Marco Barbieri, and Fabio Sciarrino. Calibration of multiparameter sensors via machine learning at the single-photon level. *Physical Review Applied*, 2021.

[Denil *et al.*, 2013] Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando De Freitas. Predicting parameters in deep learning. *NeurIPS*, 26, 2013.

[Ferris and Poulin, 2013] A. Ferris and D. Poulin. Tensor networks and quantum error correction. *Physical review letters*, 113 3:030501, 2013.

[Fishman *et al.*, 2022] Matthew Fishman, Steven White, and Edwin Miles Stoudenmire. The itensor software library for tensor network calculations. *SciPost Physics Codebases*, page 004, 2022.

[Gao *et al.*, 2020] Ze-Feng Gao, Song Cheng, Rong-Qiang He, Zhi-Yuan Xie, Hui-Hai Zhao, Zhong-Yi Lu, and Tao Xiang. Compressing deep neural networks by matrix product operators. *Physical Review Research*, 2(2):023300, 2020.

[Hou *et al.*, 2019] Ming Hou, Jiajia Tang, Jianhai Zhang, Wanzeng Kong, and Qibin Zhao. Deep multimodal multilinear fusion with high-order polynomial pooling. In *NeurIPS*, 2019.

[Hu *et al.*, 2021] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv:2106.09685*, 2021.

[Hua *et al.*, 2022] Chenqing Hua, Guillaume Rabusseau, and Jian Tang. High-order pooling for graph neural networks with tensor decomposition. *NeurIPS*, 35:6021–6033, 2022.

[Kanakagiri and Solomonik, 2024] Raghavendra Kanakagiri and Edgar Solomonik. Minimum cost loop nests for contraction of a sparse tensor with a tensor network. In *SPAA*, pages 169–181, 2024.

[Kolda and Hong, 2020] Tamara G Kolda and David Hong. Stochastic gradients for large-scale tensor decomposition. *SIAM Journal on Mathematics of Data Science*, 2(4):1066–1095, 2020.

[Kossaifi *et al.*, 2020] Jean Kossaifi, Zachary C Lipton, Arinbjorn Kolbeinsson, Aran Khanna, Tommaso Furlanello, and Anima Anandkumar. Tensor regression networks. *JMLR*, 21(123):1–21, 2020.

[Li *et al.*, 2022] C. Li, Junhua Zeng, Zerui Tao, and Qianchuan Zhao. Permutation search of tensor network structures via local sampling. *ArXiv*, abs/2206.06597, 2022.

[Li *et al.*, 2023] C. Li, Junhua Zeng, Chunmei Li, C. Caiafa, and Qianchuan Zhao. Alternating local enumeration (tnale): Solving tensor network structure search with fewer evaluations. *ArXiv*, abs/2304.12875, 2023.

[Liang *et al.*, 2024] Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Tensor attention training: Provably efficient learning of higher-order transformers. *arXiv:2405.16411*, 2024.

[Liu *et al.*, 2018] Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. Efficient low-rank multimodal fusion with modality-specific factors. In *ACL*, 2018.

[Lu *et al.*, 2021] Sirui Lu, M'arton Kan'asz-Nagy, I. Kukuljan, and J. Cirac. Tensor networks and efficient descriptions of classical data. *arXiv:2103.06872*, 2021.

[Ma *et al.*, 2019] Xindian Ma, Peng Zhang, Shuai Zhang, Nan Duan, Yuexian Hou, Ming Zhou, and Dawei Song. A tensorized transformer for language modeling. *NeurIPS*, 32, 2019.

[Ma *et al.*, 2024] Linjian Ma, Matthew Fishman, Edwin Miles Stoudenmire, and Edgar Solomonik. Approximate contraction of arbitrary tensor networks with a flexible and efficient density matrix algorithm. *Quantum*, 8:1580, 2024.

[Markov and Shi, 2008] Igor L Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *SIAM Journal on Computing*, 38(3):963–981, 2008.

[Orús, 2014] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of physics*, 349:117–158, 2014.

[Orús, 2019] Román Orús. Tensor networks for complex quantum systems. *Nature Reviews Physics*, 1(9):538–550, 2019.

[Oseledets, 2011] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[Panahi *et al.*, 2020] Aliakbar Panahi, Seyran Saeedi, and Tom Arodz. word2ket: Space-efficient word embeddings inspired by quantum entanglement. In *ICLR*, 2020.

[Perez-Garcia *et al.*, 2007] D Perez-Garcia, F Verstraete, MM Wolf, and JI Cirac. Matrix product state representations. *Quantum Information and Computation*, 7(5-6):401–430, 2007.

[Phan *et al.*, 2018] A. Phan, A. Cichocki, I. Oseledets, G. G. Calvi, S. Ahmadi-Asl, and D. Mandic. Tensor networks for latent variable analysis: Higher order canonical polyadic decomposition. *TNNLS*, 31:2174–2188, 2018.

[Pirvu *et al.*, 2010] Bogdan Pirvu, Valentin Murg, J Ignacio Cirac, and Frank Verstraete. Matrix product operator representations. *New Journal of Physics*, 12(2):025012, 2010.

[Qing *et al.*, 2023] Yong Qing, Ke Li, Peng-Fei Zhou, and Shi-Ju Ran. Compressing neural network by tensor network with exponentially fewer variational parameters. *arXiv:2305.06058*, 2023.

[Quek *et al.*, 2021] Yihui Quek, Stanislav Fort, and Hui Khoon Ng. Adaptive quantum state tomography with neural networks. *npj Quantum Information*, 2021.

[Rabanser *et al.*, 2017] Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. Introduction to tensor decompositions and their applications in machine learning. *arXiv:1711.10781*, 2017.

[Ramírez *et al.*, 2024] Sergio Sánchez Ramírez, Jofre Vallès-Muns, and Artur Garcia-Saez. Einexprs: Contraction paths of tensor networks as symbolic expressions. *arXiv:2403.18030*, 2024.

[Rieser *et al.*, 2023] Hans-Martin Rieser, Frank Köster, and A. Raulf. Tensor networks for quantum machine learning. *Proceedings of the Royal Society A*, 479, 2023.

[Robeva and Seigal, 2019] Elina Robeva and Anna Seigal. Duality of graphical models and tensor networks. *Information and Inference: A Journal of the IMA*, 8(2):273–288, 2019.

[Schollwöck, 2011] Ulrich Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of physics*, 326(1):96–192, 2011.

[Schutski *et al.*, 2020] Roman Schutski, Taras Khakhulin, Ivan Oseledets, and Dmitry Kolmakov. Simple heuristics for efficient parallel tensor contraction and quantum circuit simulation. *Physical Review A*, 102(6):062614, 2020.

[Schütt *et al.*, 2017] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8(1):13890, 2017.

[Simeon and De Fabritiis, 2024] Guillem Simeon and Gianni De Fabritiis. Tensornet: Cartesian tensor representations for efficient learning of molecular potentials. *NeurIPS*, 36, 2024.

[Stoudenmire and Schwab, 2016] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. *NeurIPS*, 29, 2016.

[Su *et al.*, 2024] Zhan Su, Fengran Mo, Prayag Tiwari, Benyou Wang, Jian-Yun Nie, and Jakob Grue Simonsen. Mixture of experts using tensor products. *arXiv:2405.16671*, 2024.

[Tang and Yan, 2022] Yehui Tang and Junchi Yan. Graphqntk: Quantum neural tangent kernel for graph data. *NeurIPS*, 2022.

[Tang *et al.*, 2024] Yehui Tang, Nianzu Yang, Mabiao Long, and Junchi Yan. Ssl4q: semi-supervised learning of quantum data with application to quantum state classification. In *ICML*, 2024.

[Tang *et al.*, 2025] Yehui Tang, Mabiao Long, and Junchi Yan. Quadim: A conditional diffusion model for quantum state property estimation. In *ICLR*, 2025.

[Taylor, 2024] Jordan K. Taylor. An introduction to graphical tensor notation for mechanistic interpretability. *arXiv:2402.01790*, 2024.

[Wall and D'Aguanno, 2021] Michael L Wall and Giuseppe D'Aguanno. Tree-tensor-network classifiers for machine learning: From quantum inspired to quantum assisted. *Physical Review A*, 104(4):042408, 2021.

[Wang *et al.*, 2023] Maolin Wang, Yu Pan, Zenglin Xu, Xiangli Yang, Guangxi Li, and Andrzej Cichocki. Tensor networks meet neural networks: A survey and future perspectives. *arXiv:2302.09019*, 2023.

[Wu *et al.*, 2020] Bijiao Wu, Dingheng Wang, Guangshe Zhao, Lei Deng, and Guoqi Li. Hybrid tensor decomposition in neural network compression. *Neural networks : the official journal of the International Neural Network Society*, 132:309–320, 2020.

[Wu *et al.*, 2024] Kai-Hsin Wu, Chang-Teng Lin, Ke Hsu, Hao-Ti Hung, Manuel Schneider, Chia-Min Chung, Ying-Jer Kao, and Pochung Chen. The cytnx library for tensor networks. *arXiv:2401.01921*, 2024.

[Xie *et al.*, 2024] Kun Xie, Can Liu, Xin Wang, Xiaocan Li, Gaogang Xie, Jigang Wen, and Kenli Li. Neural network compression based on tensor ring decomposition. *TNNLS*, 2024.

[Xiong *et al.*, 2024] Hao Xiong, Yehui Tang, Yunlin He, Wei Tan, and Junchi Yan. Node2ket: Efficient high-dimensional network embedding in quantum hilbert space. In *ICLR*, 2024.

[Xiong *et al.*, 2025] Hao Xiong, Yebin Yang, Huaijin Wu, Xiaoqiu Zhong, Yehui Tang, Zhuo Xia, Xiaoxing Wang, and Junchi Yan. In-place transparent high-order product for transformers. In *SIGKDD*, 2025.

[Xu *et al.*, 2021] Xing Xu, Yifan Wang, Yixuan He, Yang Yang, Alan Hanjalic, and Heng Tao Shen. Cross-modal hybrid feature fusion for image-sentence matching. *ACM Trans. Multimedia Comput. Commun. Appl.*, 17(4), November 2021.

[Yang *et al.*, 2023] Jingxiang Yang, Liang Xiao, Yong-Qiang Zhao, and J. Chan. Unsupervised deep tensor network for hyperspectral–multispectral image fusion. *TNNLS*, 35:13017–13031, 2023.

[Yin *et al.*, 2021] Miao Yin, Yang Sui, Siyu Liao, and Bo Yuan. Towards efficient tensor decomposition-based dnn model compression with optimization framework. *CVPR*, pages 10669–10678, 2021.

[Yuan *et al.*, 2021] Xiao Yuan, Jinzhao Sun, Junyu Liu, Qi Zhao, and You Zhou. Quantum simulation with hybrid tensor networks. *Physical Review Letters*, 127(4):040501, 2021.

[Zadeh *et al.*, 2017] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Tensor fusion network for multimodal sentiment analysis. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *EMNLP*, 2017.