

Quantum toric code decoding method based on syndrome-preliminary error fusion module and ResNet architecture

Nai-Hua Ji(纪乃华), Ping-Li Song(宋平俐), Wei Wang(王伟), Hui-Qian Sun(孙汇倩), and Hong-Yang Ma(马鸿洋)
Citation: Chin. Phys. B, 2026, 35 (**6**): 060303. DOI: 10.1088/1674-1056/ae1818
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Planar: A software for exact decoding quantum error correction codes with planar structure

Dongyang Feng(冯东阳), Hanyan Cao(曹涵彦), and Pan Zhang(张潘)
Chin. Phys. B, 2025, 34 (**5**): 050311. DOI: 10.1088/1674-1056/adcb26

A graph neural network and multi-task learning-based decoding algorithm for enhancing *XZZX* code stability in biased noise

Bo Xiao(肖博), Zai-Xu Fan(范在旭), Hui-Qian Sun(孙汇倩), Hong-Yang Ma(马鸿洋), and Xing-Kui Fan(范兴奎)
Chin. Phys. B, 2025, 34 (**5**): 050306. DOI: 10.1088/1674-1056/adbadb

Global receptive field transformer decoder method on quantum surface code data and syndrome error correction

Ao-Qing Li(李熬庆), Ce-Wen Tian(田策文), Xiao-Xuan Xu(徐晓璇), Hong-Yang Ma(马鸿洋), and Jun-Qing Liang(梁俊卿)
Chin. Phys. B, 2025, 34 (**3**): 030306. DOI: 10.1088/1674-1056/adab63

Quantum decoder design for subsystem surface code based on multi-head graph attention and edge weighting

Nai-Hua Ji(纪乃华), Hui-Qian Sun(孙汇倩), Bo Xiao(肖博), Ping-Li Song(宋平俐), and Hong-Yang Ma(马鸿洋)
Chin. Phys. B, 2025, 34 (**2**): 020309. DOI: 10.1088/1674-1056/ada1c7

Decoding topological XYZ^2 codes with reinforcement learning based on attention mechanisms

Qing-Hui Chen(陈庆辉), Yu-Xin Ji(姬宇欣), Ke-Han Wang(王柯涵), Hong-Yang Ma(马鸿洋), and Nai-Hua Ji(纪乃华)
Chin. Phys. B, 2024, 33 (**6**): 060314. DOI: 10.1088/1674-1056/ad342b

---

Quantum toric code decoding method based on syndrome-preliminary error fusion module and ResNet architecture

Nai-Hua Ji(纪乃华)[1], Ping-Li Song(宋平俐)[1], Wei Wang(王伟)[1], Hui-Qian Sun(孙汇倩)[1], and Hong-Yang Ma(马鸿洋)[2,†]

[1] *School of Information and Control Engineering, Qingdao University of Technology, Qingdao* 266033, *China*
[2] *School of Sciences, Qingdao University of Technology, Qingdao* 266033, *China*

(Received 15 May 2025; revised manuscript received 25 October 2025; accepted manuscript online 28 October 2025)

Quantum error correction technology is based on the principle of redundant encoding, encoding logical quantum information into multiple physical qubits to provide important support for the stable operation of quantum computers. To address the issues of low decoding accuracy and limited feature extraction in quantum error correction, this paper proposes a toric code decoder based on a syndrome-preliminary error fusion module (SPEFM) and a ResNet architecture. This decoder takes full advantage of the correlations between X and Z errors. In the SPEFM, the syndrome and preliminary error predictions are deeply fused, while a unidirectional Swin transformer architecture is incorporated to extract global error features from the syndrome data, significantly improving both decoding accuracy and computational efficiency. In addition, this paper further extracts local error features from the fused features using the deep residual structure of ResNet, enhancing the decoder's ability to capture quantum error patterns. Experimental results show that the decoder is applicable to different code distances ($d = 4, 6, 8, 10$) under the depolarizing noise model. Its bit error rate is lower than that of the minimum weight perfect matching (MWPM) algorithm, and its logical error rate is lower than both the MWPM algorithm and the ResNet18 decoder. Furthermore, the decoding threshold is increased to 0.163, representing a 3.82% improvement over the MWPM algorithm threshold of 0.157.

Keywords: quantum error correction, toric code, ResNet, Swin transformer

PACS: 03.67.Pp, 03.67.–a

DOI: 10.1088/1674-1056/ae1818 **CSTR:** 32038.14.CPB.ae1818

1. Introduction

Quantum computing (QC) leverages the principles of quantum mechanics to solve complex problems that are intractable for classical computation.[1–3] However, its practical application is severely hindered by errors. Quantum systems are highly susceptible to environmental disturbances, leading to quantum decoherence,[4–6] which poses significant challenges to the reliability and stability of quantum computing. To address this issue, quantum error correction (QEC) has emerged as a critical technology to ensure the feasibility of quantum computing.[7–12] The core of QEC lies in quantum error correction codes (QECC), which protect the integrity of quantum information by encoding it into multiple physical qubits and leveraging redundant information to detect and correct errors induced by noise.[13–16] The threshold theorem states that when the noise strength is below a certain threshold, increasing the code distance can significantly suppress the logical error rate.[17,18] Among various QECC schemes, topological QECC[19–22] has attracted considerable attention due to its high fault-tolerant threshold and low resource overhead. In particular, the toric code,[23–27] a prototypical example of topological QECC, can be implemented on a two-dimensional (2D) lattice using local stabilizer measurements. This structure not only facilitates physical realization but also exhibits strong error resilience, making it one of the most promising error correction schemes for quantum computing.

The optimal decoding of a QECC is an NP-hard problem.[28–30] Various decoders have been proposed to achieve near-optimal decoding accuracy. Conventional decoding algorithms for QECCs include the MWPM algorithm,[31] the union-find algorithm (UF),[32] the reorganized group algorithm (RG),[33,34] the Monte Carlo–Markov chain algorithm (MCMC),[35,36] and the maximum likelihood decoding algorithm (MLD).[37] These conventional decoding algorithms analyze syndrome information and use logical inference to identify and correct qubit errors, making them essential for building efficient QEC systems. In recent years, deep learning-based QEC decoders, including architectures such as multilayer perceptrons, neural networks, and transformers, have significantly improved decoding accuracy and computational efficiency by automatically learning quantum error features, providing more efficient error correction solutions for quantum computing. Wagner *et al*. demonstrated that incorporating

†Corresponding author. E-mail: mahongyang@qut.edu.cn

© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

the symmetries of toric codes into the training of feedforward neural network decoders significantly reduces data requirements and improves decoding performance.[38] Varsamopoulos *et al*. transformed surface code decoding into a classification problem and employed feedforward neural networks for decoding, achieving performance comparable to or better than traditional algorithms.[39] Chamberland and Ronagh proposed a deep neural network-based universal decoding framework, with their neural decoder outperforming traditional lookup table decoders in key metrics such as pseudo-threshold.[40] Ni designed a neural network decoder for large-distance 2D toric codes, and experiments showed that its performance is close to that of the MWPM algorithm.[41] Choukroun and Wolf proposed a transformer-based deep quantum error correction decoder that significantly outperforms conventional methods in error correction accuracy under various noise models and code lengths.[42] Cao *et al*. proposed the qecGPT framework utilizing generative transformers for quantum error correction decoding, with experimental results demonstrating superior accuracy over conventional algorithms while maintaining generalization capability.[43]

However, existing deep learning decoding methods still have certain limitations. Traditional single network architectures show limited improvements in decoding accuracy and computational efficiency, and conventional feature fusion strategies struggle to represent the complex correlations among quantum errors. To address these issues, this paper proposes a deep learning decoding method based on SPEFM and ResNet. In the SPEFM, a multi-scale feature fusion structure is employed to enhance the model's ability to analyze quantum error features, while a unidirectional Swin transformer is used to optimize the processing efficiency of syndrome data. In addition, the integration of residual structures strengthens the transmission of multi-level features, thereby enabling efficient modeling of the correlated characteristics of quantum errors.

2. Background

2.1. Toric code

The principle of QEC is based on the stabilizer formalism,[44–46] which is a quantum encoding method constructed using the commutation properties of the Pauli group. Stabilizer codes are a type of QECC defined in terms of the stabilizer formalism.[47,48] By specifying a set of independent and mutually commutative stabilizer generators, such codes define a code space that encodes logical qubits into multiple physical qubits, enabling detection and correction of quantum errors on up to $\lfloor \frac{d-1}{2} \rfloor$ physical qubits (where d is the code distance). The code space C is the set of all quantum states $|\psi\rangle$ satisfying the stabilizer conditions:

$$C = \left\{ |\psi\rangle \in \mathscr{H}_2^{\otimes n} \mid S|\psi\rangle = |\psi\rangle, \forall S \in \mathscr{S} \right\}, \tag{1}$$

where $\mathscr{H}_2^{\otimes n}$ is the Hilbert space for n qubits, $\mathscr{S}$ is the stabilizer group formed by a set of stabilizer generators, and each of its elements S consists of tensor products of Pauli operators.

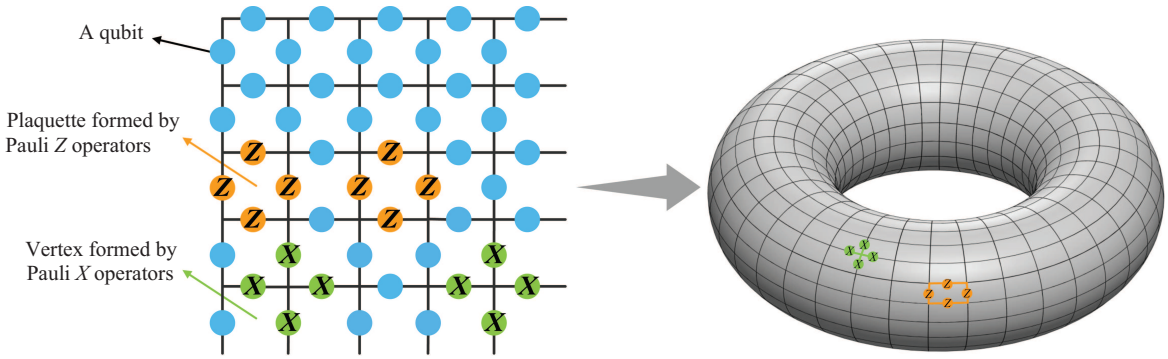


Fig. 1. A planar representation of a toric code with code distance $d = 5$ and its corresponding toric torus. The blue circles represent physical qubits, the orange set of 4 qubits with Pauli Z operators forms a plaquette operator, and the green set of 4 qubits with Pauli X operators forms a vertex operator.

The subspace formed by quantum states serves as the encoded space for logical qubits. To ensure the encoded quantum information remains protected during logical operations, the logical operators must preserve the code space. These logical operators constitute a subgroup C of the Pauli group, characterized by commuting with all elements of the stabilizer group $\mathscr{S}$ while not belonging to $\mathscr{S}$ themselves. This property is formally defined as follows:

$$L := \{P \in \mathscr{P}_n \mid PS = SP\} \setminus \mathscr{S}, \mathscr{P}_n = \{I, X, Y, Z\}^{\otimes n}, \tag{2}$$

where $\setminus$ denotes the set difference, $\mathscr{P}_n$ denotes the Pauli group, I denotes the identity mapping, X denotes the quantum bit flip, Z denotes the phase flip, Y denotes the combined flip, and $\{\cdots\}^{\otimes n}$ denotes all possible n-fold tensor products.

The toric code is an earlier proposed stabilizer code with good periodicity and efficient local operability. The left part of Fig. 1 illustrates the planar representation of a toric code with code distance $d = 5$. Physical qubits are positioned on the edges of a two-dimensional lattice with periodic boundary conditions (represented by blue circles in Fig. 1), form-

ing a topological toroidal structure (toric torus on the right side of Fig. 1), where the rightmost edges connect to the leftmost edges and the topmost edges connect to the bottommost edges. On a two-dimensional lattice, the toric code comprises $2d^2$ physical qubits and $2d^2$ stabilizer generators.[49] These stabilizer generators consist of vertex operators and plaquette operators. The vertex operators are constructed as the tensor product[50,51] of Pauli X operators acting on the four adjacent qubits at each lattice vertex, while the plaquette operators are formed by the tensor product of Pauli Z operators applied to the four edge-adjacent qubits. The definitions of these operators are as

$$X_L = \prod_{j\in \mathrm{star}(s)} \sigma_j^x, \quad Z_p = \prod_{j\in \mathrm{boundary}(s)} \sigma_j^z, \tag{3}$$

where X_L denotes the vertex operators, Z_p represent the plaquette operators, σ^x and σ^z correspond to the Pauli operators, and j signifies the edge.

2.2. Swin transformer

The transformer architecture has demonstrated remarkable success in natural language processing.[52] However, when directly applied to computer vision tasks, traditional vision transformers[53,54] face challenges such as high computational resource demands for high-resolution images and difficulty in capturing pixel-level details. To address these limitations, Liu *et al.* proposed an improved hierarchical vision transformer architecture — Swin transformer.[55] The core innovation of the Swin transformer primarily resides in its shifted window mechanism, which alternately employs regular and shifted window partitioning between adjacent transformer layers. The regular windows partition the image into non-overlapping local regions, confining self-attention computation to achieve linear complexity, while the shifted windows establish cross-window connections through window boundary offsets to enhance the model's representational capacity. Experiments demonstrate that this design significantly reduces computational complexity while improving multiscale feature extraction, achieving excellent performance on multiple mainstream vision benchmarks.

Errors in quantum error correction often exhibit nonlocal correlations, making them difficult to capture with traditional local analysis methods. The Swin transformer's shifted window mechanism effectively identifies and corrects these global error patterns while maintaining computational efficiency and improving correction accuracy.

3. Toric code decoding method

3.1. Error correction process

The decoding process of QECC draws inspiration from classical error correction codes, particularly through the implementation of stabilizer codes to detect and correct quantum errors. The essence of stabilizer codes lies in utilizing stabilizer generators to define the code space of quantum states and detecting errors by measuring the eigenvalues of these generators. Specifically, stabilizer generators can be decomposed into two independent parity check matrices, designed to identify phase flip errors (Z errors) and bit flip errors (X errors). The stabilizer generators define a block parity check matrix structured as

$$H = \begin{pmatrix} H_Z & 0 \\ 0 & H_X \end{pmatrix},$$

where H_Z and H_X represent phase flip and bit flip parity checks. When an error occurs on a qubit, the corresponding error operator anticommutes with a subset of stabilizer generators, producing a measurable syndrome computed as

$$s = He, \tag{4}$$

where e denotes the qubit errors, typically represented as a binary vector. As illustrated in Fig. 2, in the toric code, a bit flip error (X error) on a qubit generates non-zero syndromes at adjacent plaquettes, while a phase flip error (Z error) produces non-zero syndromes at adjacent vertices. When identical errors occur at the same location, their corresponding syndromes cancel each other out.

The decoder F_θ proposed in this paper is designed to predict qubit errors based on the toric code simulations implemented by Krastanov and Jiang.[56] It computes the syndrome s via the parity check matrix H and predicts the errors using the decoder

$$F_\theta\left([s, f_\omega(s)]\right) = \hat{e}, \tag{5}$$

where θ denotes learnable parameters, $f_\omega(s)$ represents the parameterized feature transformation neural network proposed in this paper with parameters ω, $\hat{e}$ signifies the predicted errors, and $[\cdot,\cdot]$ indicates the data fusion operation. The decoder aims to accurately predict errors on qubits by learning the mapping relationship between syndromes and error patterns. After obtaining a correct approximate estimate, due to the self-inverse nature of the quantum operations, applying the same operation as the initial error can restore the quantum state to its original form, thereby achieving error correction.

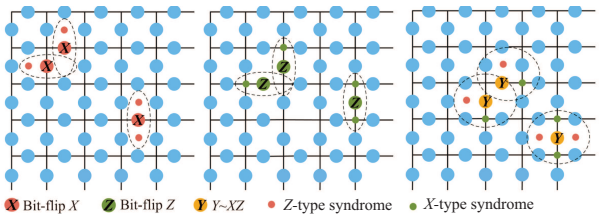


Fig. 2. Quantum errors and corresponding syndromes on a toric code with code distance $d = 5$. Bit flip (X) errors generate pairs of red syndromes at adjacent plaquettes, phase flip (Z) errors produce pairs of green syndromes at adjacent vertices, and $Y\sim XZ$ errors simultaneously induce both red and green syndrome pairs at neighboring plaquettes and vertices, respectively. When syndromes of the same type generated by identical errors occur at overlapping locations, they mutually cancel out.

3.2. Designing unidirectional Swin transformer module

In the field of QEC, error propagation typically exhibits complex spatial correlations, making purely local analysis inadequate for capturing cross-qubit error patterns. Given the extreme sensitivity of qubits to environmental noise, quantum computing systems must continuously perform error correction to maintain quantum information integrity. Insufficient correction speed allows noise to persistently interfere with qubits during the correction process, leading to error accumulation. Such accumulated errors may eventually exceed the correction capability of the QECC, resulting in correction failure. To address these challenges by capturing global characteristics of quantum errors while solving the issues of prolonged correction time with larger code distances, this paper introduces the window partitioning and shifting mechanism of the Swin transformer into QEC. We propose a unidirectional Swin transformer (USWT), as illustrated in Fig. 3, which achieves linear growth in both computational overhead and memory requirements.

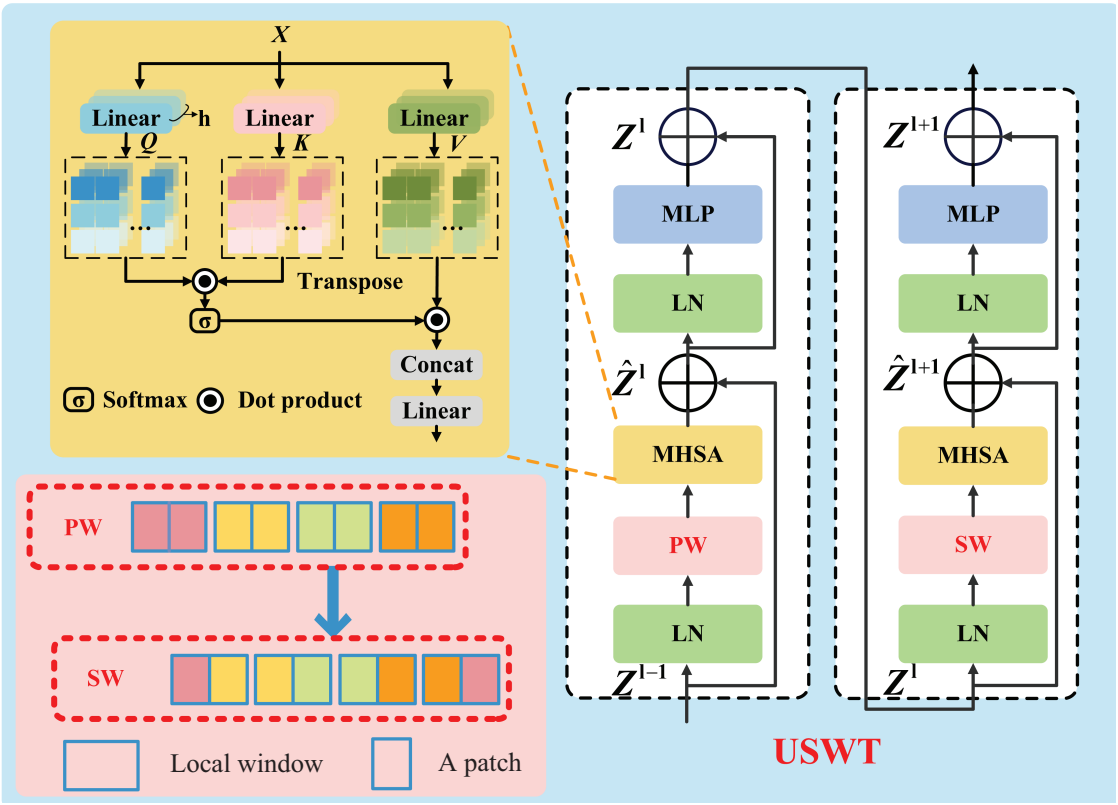


Fig. 3. USWT architecture and the computation process of the multi-head self-attention mechanism. Architecture diagram of the USWT, including the multi-head self-attention computation flow and the partitioned window or shifted window mechanisms.

This structure consists of two USWT modules. Within each module, the input data first undergoes layer normalization, then passes through either the partitioned window layer or the shifted window layer, followed by multi-head self-attention computation.[57] Specifically, the input data X is linearly projected into three distinct vectors: query (Q), key (K), and value (V). Each self-attention layer contains h heads, enabling the model to extract features from h-independent subspaces. The computation for each attention head is formulated as

$$\text{Attention}(Q,K,V) = \text{SoftMax}\left(\frac{QK^{\text{T}}}{\sqrt{d_k}}\right)V, \tag{6}$$

where d_k denotes the dimensionality of each subspace. During computation, the dot product of Q and K is calculated, scaled by $\sqrt{d_k}$, and then normalized via the softmax function to generate attention weights, which are applied to V for weighted aggregation. Each attention head operates independently, and the outputs of all heads are concatenated and linearly transformed to produce the final result:

$$\begin{aligned}&\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,\ldots,\text{head}_{\text{h}})W^O,\\ &\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),\end{aligned} \tag{7}$$

where W_i^O, W_i^Q, W_i^K, and W_i^V represent linear transformation matrices. The multi-head self-attention layer employs residual connections to propagate outputs to subsequent layers. After a second layer normalization, the features are fed into a multi-layer perceptron layer, with residual connections to complete the output of the current module. Feature fusion between the two modules is achieved through residual connections, and the calculation processes are as follows:

$$\hat{Z}^l = \text{MHSA}\left(\text{PW}\left(\text{LN}\left(Z^{l-1}\right)\right)\right) + Z^{l-1},$$

$$
\begin{aligned}
Z^l &= \mathrm{MLP}\left(\mathrm{LN}\left(\hat{Z}^l\right)\right) + \hat{Z}^l, \\
\hat{Z}^{l+1} &= \mathrm{MHSA}\left(\mathrm{SW}\left(\mathrm{LN}\left(Z^l\right)\right)\right) + Z^l, \\
Z^{l+1} &= \mathrm{MLP}\left(\mathrm{LN}\left(\hat{Z}^{l+1}\right)\right) + \hat{Z}^{l+1},
\end{aligned} \tag{8}
$$

where PW represents the window partitioned layer, SW represents the shifted window layer, MHSA represents the multi-head self-attention mechanism, LN denotes layer normalization, and MLP denotes the feedforward neural network.

The following details the partitioned window and shifted window mechanisms (with the architectural diagram presented in Fig. 3). In the first USWT module, the input data are partitioned into multiple local windows of size $W \times d$, based on its length (where W denotes the window size and d represents the data dimensionality). For datasets with varying code distances, the number of windows N_{um} is fixed to 2, with the window size $W = (n+m)/N_{\mathrm{um}}$, where n is the length of preliminary error predictions and m is the length of syndrome data. Each window contains two adjacent quantum state data patches, and multi-head self-attention computations are performed within these local windows to capture localized quantum error patterns. However, this partitioning isolates windows from cross-window interactions, limiting the model's ability to model global contextual dependencies. To address this limitation, the second USWT module introduces a shifted window partitioning strategy, where all windows are shifted rightward by half the window size. Specifically, the first half of the initial window is concatenated with the latter half of the final window to form new window partitions. This shifting operation not only establishes cross-window connectivity but also maintains the computational efficiency of non-overlapping windows. Subsequently, multi-head self-attention computations are executed within each new local window, enabling the model to progressively integrate global contextual information and effectively model long-range error correlations.

3.3. Designing syndrome-preliminary error fusion module

In conventional decoder designs, input data typically comprises only quantum syndrome information. To make full use of the characteristic information provided by the syndrome, this paper integrates syndrome information with preliminary error predictions through deep fusion. Current deep learning approaches commonly use simple channel concatenation for data fusion, where information is combined via concatenation along specific dimensions. However, this method exhibits significant limitations. First, concatenated data lacks interactive mechanisms to model intrinsic relationships between components, resulting in insufficient information integration. Second, concatenation fails to project features into higher-dimensional spaces for enhanced representational capacity and does not support multi-level feature fusion, restricting integration to single-layer feature linkage. These limitations lead to suboptimal performance in complex tasks, particularly in scenarios requiring the full utilization of complementary information across multi-scale data, where concatenation-based fusion often underperforms practical demands.

To address these limitations, this paper proposes the syndrome-preliminary error fusion module (SPEFM), which enhances inter-data interactivity and information-sharing capabilities by incorporating global context guidance and an adaptive feature refinement mechanism, thereby improving model performance. As illustrated in Fig. 4, within this module, the syndrome is first processed by a feature transformation neural network to generate preliminary error estimates. These preliminary error estimates provide the decoder with reliable initial predictions, accelerating its convergence process and effectively reducing misidentification probability. Furthermore, to augment the model's representational capacity, the fused data of syndromes and preliminary error estimates are mapped into a two-dimensional space as follows:

$$
H = W_H \odot \mathrm{concat}(s, f_\omega(s)), \tag{9}
$$

where $\odot$ denotes the Hadamard product, $W_H \in \mathbb{R}^{(m+n)\times d}$ represents learnable parameters, and $f_\omega(s)$ denotes a shallow neural network parameterized by ω. Subsequently, the concatenated features are fed into the USWT for global contextual modeling, which prioritizes the most salient features while suppressing less critical ones. After processing by the USWT, the features undergo a split operation and are dynamically fused with the original data. To illustrate the fusion process, consider a pair of processed feature components as an example. This procedure employs dual weight parameters, both initialized to 1, which are normalized via a sum-based normalization algorithm. This algorithm preserves weight proportions through linear computations, mitigating distribution shift, as formalized below:

$$
w_i' = \frac{w_i}{\sum_{j=1}^{n} w_j} (i \in [1,2], n = 2), \tag{10}
$$

where w_i represents the original weights, w_i' denotes the normalized weights, and $\sum_{j=1}^{n} w_j$ signifies the sum of all weights. Subsequently, the two processed data features are individually multiplied by the normalized weights, and dynamic feature fusion is implemented through weighted summation. Finally, residual connections are employed to integrate the original features with the enhanced features via weighted summation, enabling multi-level information complementarity, followed by concatenation of the results.

Experimental validation (Subsection 4.4) demonstrates that the proposed SPEFM captures complex interdependencies between syndromes and preliminary error predictions more precisely than traditional concatenation methods, thereby

significantly enhancing decoding accuracy. Furthermore, SPEFM offers novel insights for optimizing conventional fusion approaches by proposing innovative solutions for effectively integrating multi-scale data, which provides a valuable reference for research in related fields.

3.4. Constructing SPEFM-ResNet18 decoder

This paper proposes a novel SPEFM-ResNet18 decoder architecture, as illustrated in Fig. 4. First, syndrome data are processed by the SPEFM, where a feature transformation neural network generates preliminary error predictions. This network is optimized by minimizing the following objective function:

$$\mathscr{L}_1 = -\frac{1}{N}\sum_{i=1}^{N}\left[e_i \log(f_\omega(s)_i) + (1-e_i)\log(1-f_\omega(s)_i)\right], \tag{11}$$

where e_i represents the true error values, and N denotes the total number of samples. Through optimization of this objective function, the network effectively captures the primary mapping relationship between syndrome data and quantum errors. Second, the preliminary error predictions are fused with syndromes and fed into subsequent components of the SPEFM to enhance decoder robustness via synergistic analysis of multi-scale data. To further extract deeper error patterns and generate refined error predictions, a ResNet18[58] module is employed after the SPEFM. This module employs stacked residual blocks, each containing two convolutional layers and ReLU activation functions. The residual connections mitigate gradient vanishing in deep networks. Through this progressive feature extraction strategy, the network sequentially captures both local and global error patterns. Finally, a global average pooling layer compresses the fused features into a fixed-length feature vector, which is then fed into a fully connected layer to output precise error predictions.

The proposed decoding method employs cross-entropy to measure the divergence between predicted and true error distributions, with the loss function defined as

$$\mathscr{L}_2 = -\frac{1}{N}\sum_{i=1}^{N}\left[e_i \log\left(F_\theta(s)_i\right) + (1-e_i)\log\left(1-F_\theta(s)_i\right)\right], \tag{12}$$

where F_θ denotes the SPEFM-ResNet18 decoder. Since the physical error rate in QEC reflects only individual qubit errors, while the logical error rate better quantifies the reliability of encoded and corrected qubits in practical computations, this paper adapts the method proposed by Choukroun and Wolf[42] to transform the loss function $\mathscr{L}_2$ into the following paradigm via the composite differentiable function $\Lambda(\mathbb{L},x)$ and logical matrix $\mathbb{L}$:

$$\mathscr{L}_3 = -\frac{1}{N}\sum_{i=1}^{N}\begin{bmatrix}(\mathbb{L}e)_i \log\left([\Lambda(\mathbb{L},\mathrm{bin}(F_\theta(s))]_i\right) \\ +(1-(\mathbb{L}e)_i)\log\left(1-[\Lambda(\mathbb{L},\mathrm{bin}(F_\theta(s))]_i\right)\end{bmatrix}, \tag{13}$$

where bin indicates numerical binarization. The overall model loss function is formulated as a triply cascaded hybrid loss:

$$\mathscr{L} = \lambda_1\mathscr{L}_1 + \lambda_2\mathscr{L}_2 + \lambda_3\mathscr{L}_3, \tag{14}$$

where λ_1, λ_2, and λ_3 are weighting coefficients for the loss components. All three loss components L_1, L_2, L_3, and the overall triply cascaded hybrid loss adopt the design paradigm from Ref. [42].

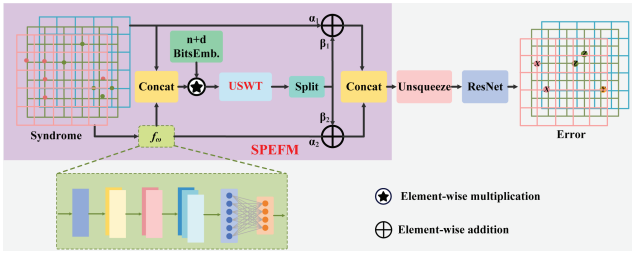


Fig. 4. The schematic diagram of the overall decoder architecture. Syndromes undergo feature fusion with preliminary error predictions through the SPEFM structure. These fused features are then fed into the ResNet18 network to predict refined error values.

4. Experimental results

4.1. Experimental setup and model performance

This experiment is conducted on the toric code dataset under code distances of $d = 4, 6, 8, 10$ for decoder training. Input data are encoded as one-hot vectors with lengths ranging from 200 to 400, and physical error rates are randomly sampled across 9 discrete noise levels within the interval 0.05 to 0.2. The USWT adopts a model dimension of 128 and 16 attention heads. For deeper feature extraction, ResNet18 is implemented, comprising four sequential residual block groups–each group contains 2 residual blocks, and each block integrates 2 convolutional neural networks. The feature transformation neural network includes three convolutional layers and one fully connected layer, using the GELU non-linear activation function. The hybrid loss function weights are set to $\lambda_1 = 0.5, \lambda_2 = 0.5$ and $\lambda_3 = 1.0$. The initial learning rate is 5×10^{-4}, modulated by a cosine decay scheduler to 5×10^{-7} by the final epoch. The SPEFM-ResNet18 decoder employs the Adam optimizer[59] for training, with a batch size of 128 samples. Training spans 200 epochs, each comprising 5×10^3 mini-batches. Testing involves 10^6 samples with a batch size of 512, evaluated across 18 noise levels randomly generated within the physical error rate range 0.05 to 0.2. All experiments run on an NVIDIA 3090 GPU with 24 GB GPU memory.

To validate the performance advantages of the proposed decoder, this paper conducts comparative experiments using the MWPM algorithm and the ResNet18 decoder. The MWPM algorithm, as a conventional decoder for topological codes, employs the optimized implementation proposed by Higgott *et al.*,[60] with its decoding computations executed on the CPU. Although minor GPU memory usage (approximately

335 MiB across all code distances) is observed, this allocation stems from system background processes and is unrelated to the algorithm's intrinsic computations. The ResNet18 decoder adopts the following architecture: a feature transformation neural network first generates preliminary error predictions, which are then fused with syndrome data via concatenation and dimensionality expansion operations before being fed into the ResNet18 network. This paper adopts the depolarizing noise model as the standard error benchmark. This noise model describes stochastic perturbations arising from isotropic interactions between qubits and their environment, with its dynamic evolution formally characterized through quantum channels. Specifically, the evolution of a single qubit's density matrix under depolarizing noise is expressed as

$$\rho \to (1-p)\rho + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z, \qquad (15)$$

where the total error probability $p \in [0,1]$ is defined as the sum of the probabilities for X, Y, or Z errors, subject to isotropic distribution ($p_x = p_y = p_z = p/3$). Correspondingly, the qubit retains its original state ρ with probability $(1-p)$, remaining error-free.

Computational efficiency and resource consumption are important metrics for evaluating decoder performance. Based on the aforementioned experimental setup, we present the computational efficiency and resource consumption results of the proposed SPEFM-ResNet18 decoder at code distances $d = 4, 6, 8, 10$. As shown in Table 1, rows 1 to 4 correspond to the average decoding time, peak memory consumption, and training time under different code distances. When the code distance expands stepwise from $d = 4$ to $d = 10$ with a step size of 2, the average decoding time remains between 16.36 μs and 17.62 μs. The gradual increase in decoding time is primarily due to the unified design of the model's fundamental computational paradigm, which enables the GPU to efficiently parallelize computational tasks. During the decoder testing phase, the peak memory consumption increases from 2890 MiB at $d = 4$ to 19070 MiB at $d = 10$. This significant growth is primarily due to the substantial increase in the number of input physical qubits as the code distance expands. Concurrently, Table 1 demonstrates that as the code distance increases discretely from $d = 4$ to $d = 10$, the training time per epoch escalates from 196.12 s to 1,286.79 s. This substantial growth in training duration is primarily driven by the surge in physical qubits iteratively processed during training as the code distance expands.

Table 1. Computational efficiency and resource consumption of SPEFM-ResNet18 decoder.

Code distance	Avg. decoding time (μs)	Peak memory consumption (MiB)	Training time (s/epoch)
$d = 4$	16.36	2890	196.12
$d = 6$	16.52	6960	393.09
$d = 8$	16.73	12540	738.21
$d = 10$	17.62	19070	1286.79

4.2. Loss function analysis

Figure 5 presents the convergence curves of the hybrid loss function for the SPEFM-ResNet18 decoder under different code distances. Experimental results demonstrate that the loss functions for all code distances exhibit a characteristic convergence pattern of rapid initial decline followed by gradual stabilization, with no overfitting observed throughout the training process, thereby validating the effectiveness of the training strategy. Specifically, during the initial training phase (1–50 epochs), the hybrid loss values decrease sharply, indicating the model's rapid learning of the mapping relationships between syndromes and preliminary error predictions. In the intermediate training stage (50–150 epochs), the loss function transitions to smoother optimization. After exceeding 150 training epochs, the loss functions for all code distances reach steady-state convergence but remain at relatively high values. This primarily results from the design of our triple weighted hybrid loss function, whose weight sum equals 2, leading to higher total loss values compared to single loss functions. Further observation reveals that the converged loss values for shorter code distances consistently remain higher than those for longer code distances. This discrepancy stems from distinct error modeling characteristics. The relatively simpler error correlation patterns in shorter code distances al-

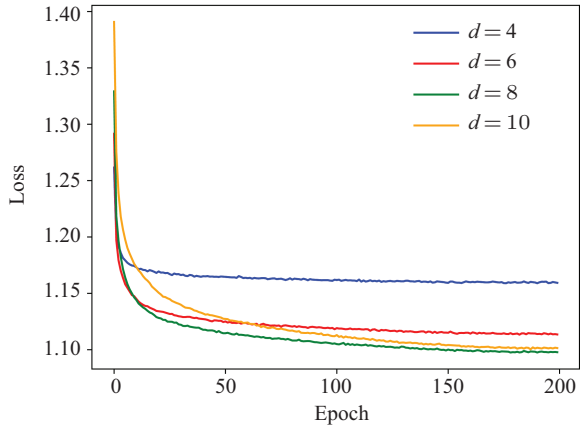


Fig. 5. Hybrid loss function curves for the SPEFM-ResNet18 decoder under code distances $d = 4, 6, 8, 10$. The horizontal axis represents training epochs, and the vertical axis denotes the hybrid loss function. Loss functions for different code distances are color-coded as blue, red, green, and orange.

low the model to reach local optima earlier, yet their limited encoding redundancy constrains the ability to adapt to complex errors. In contrast, the intricate error propagation paths in longer code distances compel the model to develop stronger error modeling capabilities through deep feature learning.

4.3. Bit error rate analysis

A lower bit error rate (BER) demonstrates that the decoder can detect errors more precisely.[42] Figure 6 shows comparative curves of the BER versus the physical error rate for the MWPM algorithm and the SPEFM-ResNet18 decoder under different code distances. The BER calculation formula is defined as follows:

$$\mathrm{BER} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(\hat{e}_i \neq e_i), \tag{16}$$

where n denotes the total number of physical qubits, $\hat{e}_i$ and e_i represent the predicted and true error on the i-th physical qubit, respectively, and $\mathbb{I}$ is the indicator function, which equals 1 if the condition is satisfied and 0 otherwise. Experimental data demonstrate that for the same decoder at a given physical error rate, the BER decreases as the code distance increases. Furthermore, under identical code distances and physical error rates, our proposed decoder achieves lower BER values than the MWPM algorithm. This advantage directly reflects enhanced error correction capability at the physical layer.

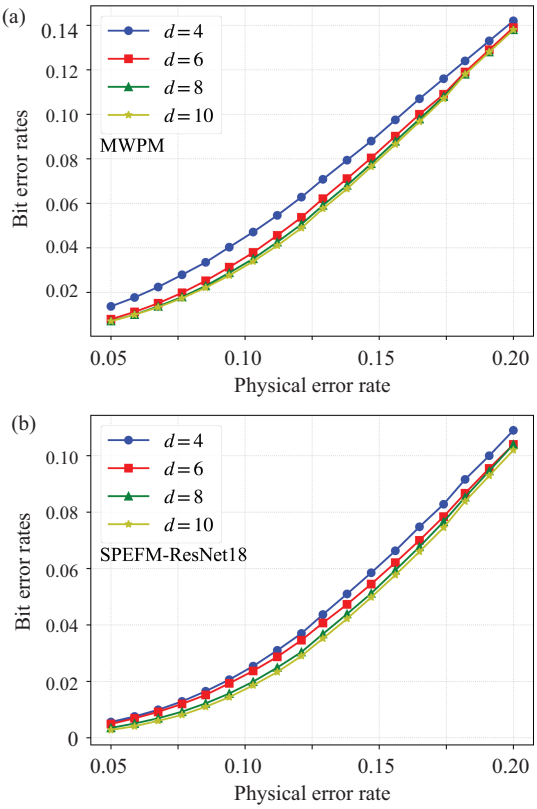


Fig. 6. (a) BER characteristic curves based on MWPM decoder at code distances $d = 4, 6, 8, 10$. (b) BER characteristic curves based on the SPEFM-ResNet18 decoder at code distances $d = 4, 6, 8, 10$. The horizontal axis represents the physical error rate and the vertical axis represents the bit error rate. Different code distances are color-coded as blue, red, green, and yellow.

4.4. Logical error rate analysis

The logical error rate (LER) is an important metric for evaluating the performance of quantum error correction codes, reflecting the reliability of qubits in actual computation after error correction. Figure 7 shows the LER curves of the toric code under the MWPM decoder, the SPEFM-ResNet18 decoder, and the ResNet18 decoder. The formula for calculating LER is as follows:

$$\mathrm{LER} = \frac{1}{B}\sum_{j=1}^{B}\mathbb{I}\left[\exists i : (\mathbb{L}\hat{e}_j \bmod 2)_i \neq (\mathbb{L}e_j \bmod 2)_i\right], \tag{17}$$

where B denotes the total number of samples, and i is the index of all logical qubits. The threshold refers to a critical physical error rate and serves as an important indicator of decoder performance. When the physical error rate is below this threshold, topological codes can effectively suppress errors, and the probability of errors decreases significantly with increasing code distance. We tested the three decoders under different physical error rates. The threshold of the MWPM decoder is approximately 0.157 (Fig. 7(a)), while the threshold of the SPEFM-ResNet18 decoder is approximately 0.163 (Fig. 7(b)). Compared with MWPM, the threshold of the SPEFM-ResNet18 decoder is increased by about 3.82%. When using the ResNet18 decoder alone, due to the limitations of the model's local receptive field, the LER curve does not exhibit a clear threshold intersection point (Fig. 7(c)), indicating its difficulty in effectively modeling the long-range topological correlations of quantum error correction codes.

The experimental results further show that under the same physical error rate and code distance, the LER of the ResNet18 decoder is consistently lower than that of the traditional MWPM algorithm. This indicates that the ResNet18 decoder significantly outperforms the traditional MWPM algorithm in error correction performance by modeling the complex correlations of quantum features. Moreover, the SPEFM-ResNet18 decoder achieves further optimization of LER compared to the ResNet18 decoder. The integration of the SPEFM module with ResNet18 enhances the global error correction feature extraction capability, validating the critical role of decoder architecture innovation in improving quantum decoding performance. In addition, an analysis of the LER curves at different code distances under the same decoding scheme reveals that when the physical error rate is below the threshold, increasing the code distance effectively reduces the LER. This occurs because in low error rate environments, increased code distance provides additional redundant qubits that supply more error correction information. This enhances the decoder's ability to identify and correct errors, thereby reducing the LER. However, when the physical error rate exceeds the threshold, higher code distance leads to an increase in LER instead. This trend is likely due to larger code distances introducing higher

encoding complexity, which introduces additional errors and degrades decoder performance.

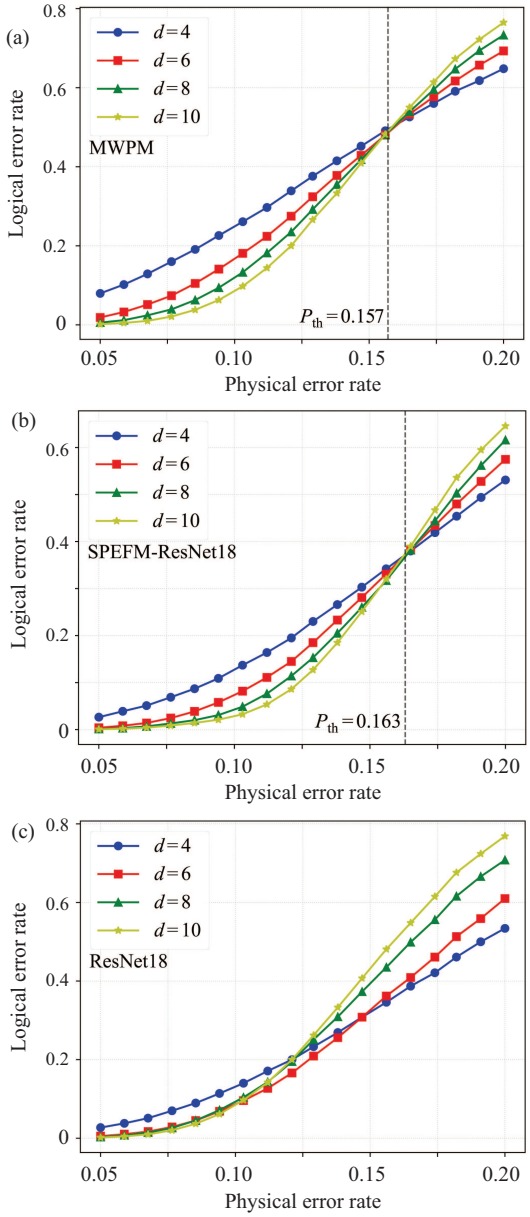


Fig. 7. (a) LER characteristic curves for the MWPM decoder under code distances $d = 4, 6, 8$, and 10. (b) LER characteristic curves for the SPEFM-ResNet18 decoder under code distances $d = 4, 6, 8$, and 10. (c) LER characteristic curves for the ResNet18 decoder under code distances $d = 4, 6, 8$, and 10. The horizontal axis represents the physical error rate, and the vertical axis denotes the LER. Different code distances are color-coded as blue, red, green, and yellow.

4.5. Ablation experiments analysis

To verify the role of BER optimization in enhancing the final decoding performance and to justify the selection of loss function weights $\lambda_1 = 0.5, \lambda_2 = 0.5$, $\lambda_3 = 1.0$, we conducted ablation experiments at code distance $d = 6$ by adjusting the weights of the three loss functions in formula (14). The results are shown in Fig. 8. First, the blue curve corresponds to simultaneous optimization of loss functions $\mathscr{L}_1$, $\mathscr{L}_2$, and $\mathscr{L}_3$ with weights set to $\lambda_1 = 0.5, \lambda_2 = 0.5$, and $\lambda_3 = 1.0$ respectively. Under this configuration, the LER versus physical error rate curve is the lowest, demonstrating optimal decoding performance. Second, the red curve represents LER variation without BER optimization ($\lambda_2 = 0.0$). This curve shows that LER is significantly higher across all physical error rates compared to BER-optimized results, indicating that BER optimization plays a crucial role in achieving low LER and validating the decoder's superior performance. Furthermore, we examined the decoder's performance when optimizing only BER, with results shown as the green curve. These results reveal that the decoder maintains basic decoding capability through physical layer optimization alone, even without logical layer supervision, indirectly demonstrating the importance of physical layer error correction. Finally, setting weight $\lambda_1 = 0.0$ for loss function $\mathscr{L}_1$ and weights $\lambda_2 = 0.5, \lambda_3 = 1.0$ for $\mathscr{L}_2$ and $\mathscr{L}_3$ yields the orange curve. At identical physical error rates, its LER values are higher than those of the blue curve, indicating that preliminary error predictions from the feature transformation neural network effectively enhance the decoder's LER performance.

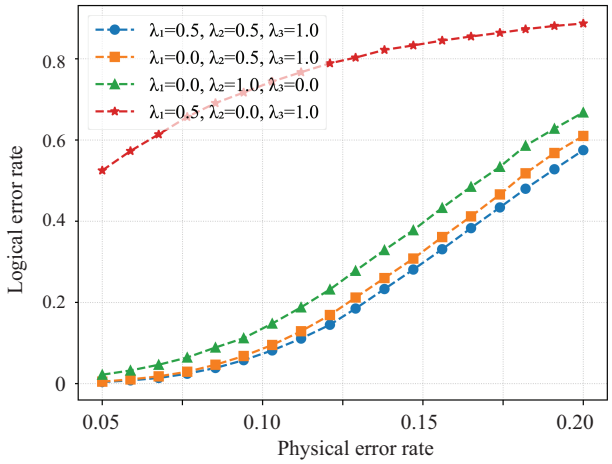


Fig. 8. LER performances of the SPEFM-ResNet18 decoder at code distance $d = 6$ under different loss weight conditions.

5. Conclusion

This paper proposes an innovative decoding scheme for quantum error correction of toric codes, achieving significant performance improvements through the integration of the SPEFM and ResNet18 modules. Specifically, it proposes a multi-scale feature fusion architecture to enhance feature extraction, introduces a sliding window interaction mechanism to improve computational efficiency, and integrates residual structures to enhance decoding accuracy. Experimental results show that the proposed scheme outperforms the MWPM algorithm in terms of BER and surpasses both the MWPM and ResNet18 algorithms in terms of LER. Notably, the decoding threshold is increased to 0.163, representing a 3.82% improvement compared to the 0.157 threshold of the MWPM algorithm. Our future work will build upon the SPEFM-ResNet18 decoder to conduct an in-depth investigation of QEC, with the

aim of further enhancing the decoding thresholds and accuracy for QECC such as toric codes, surface codes, and color codes, while maintaining decoding efficiency. Furthermore, effective extrapolation of decoder performance to quantum hardware and accurate prediction of the maximum decodable distance are critically significant for evaluating the practical application value of quantum error correction schemes. We will simultaneously optimize decoder performance and closely track advancements in quantum hardware, thereby enabling precise quantitative assessment of the maximum decodable distance in physical quantum systems.

Acknowledgments

Project supported by the Joint Fund of the Natural Science Foundation of Shandong Province, China (Grant Nos. ZR2022LLZ012 and ZR2021LLZ001) and the Key Research and Development Program of Shandong Province, China (Grant No. 2023CXGC010901).

References

[1] Krinner S, Lacroix N, Remm A, Di Paolo A, Genois E, Leroux C, Hellings C, Lazar S, Swiadek F, Herrmann J, Norris G J, Andersen C K, Müller M and Blais A 2022 *Nature* **605** 669
[2] Xin T, Wang B X, Li K R, Kong X Y, Wei S J, Wang T, Ruan D and Long G L 2018 *Chin. Phys. B* **27** 020308
[3] Gill S S, Kumar A, Singh H, Singh M, Kaur K, Usman M and Buyya R 2022 *Software: Practice and Experience* **52** 66
[4] Zurek W H 2003 *Rev. Mod. Phys.* **75** 715
[5] Schlosshauer M 2019 *Physics Reports* **831** 1
[6] Schlosshauer M 2005 *Rev. Mod. Phys.* **76** 1267
[7] Sang S, Zou Y and Hsieh T H 2024 *Phys. Rev. X* **14** 031044
[8] Livingston W P, Blok M S, Flurin E, Dressel J, Jordan A N and Siddiqi I 2022 *Nat Commun* **13** 2307
[9] Heußen S, Locher D F and Müller M 2024 *PRX Quantum* **5** 010333
[10] Xu Q, Seif A, Yan H, Mannucci N, Sane B O, Van Meter R, Cleland A N and Jiang L 2022 *Phys. Rev. Lett* **129** 240502
[11] Wang H W, Xue Y J, Qu Y J, Mu X Y and Ma H Y 2022 *npj Quantum Inf* **8** 134
[12] Ji Y X, Chen Q H, Wang R, Ji N H and Ma H Y 2024 *Quantum Inf Process* **23** 255
[13] Qin D, Xu X and Li Y 2022 *Chin. Phys. B* **31** 090306
[14] Ji N H, Sun H Q, Xiao B, Song P L and Ma H Y 2025 *Chin. Phys. B* **34** 020309
[15] Li A Q, Tian C W, Xu X X, Ma H Y and Liang J Q 2025 *Chin. Phys. B* **34** 030306
[16] Li A Q, Li F, Gan Q D and Ma H Y 2023 *Applied Sciences* **13** 9689
[17] Piveteau C, Sutter D, Bravyi S, Gambetta J M and Temme K 2021 *Phys. Rev. Lett.* **127** 200505
[18] Li F, Li A Q, Gan Q D and Ma H Y 2024 *Chin. Phys. B* **33** 040307
[19] Qu Y J, Chen Z , Wang W J and Ma H Y 2023 *Chin. Phys. B* **32** 100307
[20] Shinde U U and Bandaru R 2024 *Sci Rep* **14** 14289
[21] Chen Q H, Ji Y X, Wang K H, Ma H Y and Ji N H 2024 *Chin. Phys. B* **33** 060314
[22] Qi L, Yan Y, Xing Y, Zhao X D, Liu S, Cui W X, Han X, Zhang S and Wang H F 2021 *Phys. Rev. Res.* **3** 023037
[23] Lin Y, Huang S and Brown K R 2024 *Phys. Rev. A* **109** 052438
[24] Fitzek D, Eliasson M, Kockum A F and Granath M 2020 *Phys. Rev. Research* **2** 023230
[25] Andreasson P, Johansson J, Liljestrand S and Granath M 2019 *Quantum* **3** 183
[26] Torlai G and Melko R G. 2017 *Phys. Rev. Lett.* **119** 030501
[27] Domingo Colomer L, Skotiniotis M and Muñoz-Tapia R 2020 *Physics Letters A* **384** 126353
[28] Hsieh M H and Le Gall F 2011 *Phys. Rev. A* **83** 052331
[29] Ji N H, Chen Z, Qu Y J, Bao R Y, Yang X and Wang S M 2023 *Front. Phys.* **11** 1164567
[30] Iyer P and Poulin D 2015 *Front. Phys.* **61** 5209
[31] Cook W and Rohe A 1999 *INFORMS Journal on Computing* **11** 138
[32] Huang S, Newman M and Brown K R 2020 *Phys. Rev. A* **102** 012419
[33] Duclos-Cianci G and Poulin D 2010 *Phys. Rev. Lett.* **104** 050504
[34] Duclos-Cianci G and Poulin D 2013 arXiv:1304.6100 [quant-ph]
[35] Hutter A, Wootton J R and Loss D 2014 *Phys. Rev. A* **89** 022326
[36] Wootton J R and Loss D 2012 *Phys. Rev. Lett.* **109** 160503
[37] Bravyi S, Suchara M and Vargo A 2014 *Phys. Rev. A* **90** 032326
[38] Wagner T, Kampermann H and Bruß D 2020 *Phys. Rev. A* **102** 042411
[39] Varsamopoulos S, Criger B and Bertels K 2017 *Quantum Sci. Technol.* **3** 015004
[40] Chamberland C and Ronagh P 2018 *Quantum Sci. Technol.* **3** 044002
[41] Ni X 2020 *Quantum* **4** 310
[42] Choukroun Y and Wolf L 2024 *AAAI* February 20–27, 2024, Vancouver, Canada, p. 64
[43] Cao H, Pan F, Wang Y and Zhang P 2023 arXiv: 2307.09025 [quant-ph]
[44] Dauphinais G, Kribs D W and Vasmer M 2024 *Quantum* **8** 1261
[45] Li K, Wan Y, Hung L Y, Lan T, Long G, Lu D, Zeng B and Laflamme R 2017 *Phys. Rev. Lett.* **118** 080502
[46] Poulin D 2005 *Phys. Rev. Lett.* **95** 230504
[47] Terhal B M 2015 *Rev. Mod. Phys.* **87** 307
[48] Devitt S J, Munro W J and Nemoto K 2013 *Rep. Prog. Phys.* **76** 076001
[49] Wang H W, Xue Y J, Ma Y L, Hua N and Ma H Y 2022 *Chin. Phys. B* **31** 010303
[50] Dennis E, Kitaev A, Landahl A and Preskill J 2002 *Journal of Mathematical Physics* **43** 4452
[51] Chen G, Zhang W H, Yin P, Li C F and Guo G C 2021 *Fundamental Research* **1** 27
[52] Singh S and Mahmood A 2021 *IEEE Access* **9** 68675
[53] Han K, Wang Y, Chen H, Chen X, Guo J, Liu Z, Tang Y, Xiao A, Xu C, Xu Y, Yang Z, Zhang Y and Tao D 2022 *IEEE Trans. Pattern Anal. Mach. Intell.* **45** 87
[54] Khan S, Naseer M, Hayat M, Zamir S W, Khan F S and Shah M 2022 *ACM Comput. Surv.* **54** 1
[55] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S and Guo B 2021 *ICCV, October* 11-17, 2021, p.10012
[56] Krastanov S and Jiang L 2017 *Sci Rep* **7** 11003
[57] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L and Polosukhin I 2017 arXiv:1706.03762 [cs.CL]
[58] He K, Zhang X, Ren S and Sun J 2016 *2016 CVPR*, June 26–July 1, 2016, Las Vegas, USA, p. 770
[59] Kingma D P and Ba J 2014 arXiv: 1412.6980 [cs.LG]
[60] Higgott O 2022 *ACM Transactions on Quantum Computing* **3** 1