QEM-Bench: Benchmarking Learning-based Quantum Error Mitigation and QEMFormer as a Multi-ranged Context Learning Baseline

Tianyi Bao [1 2] **Ruizhe Zhong** [1] **Xinyu Ye** [1] **Yehui Tang** [1] **Junchi Yan** [1 2]

Abstract

Quantum Error Mitigation (QEM) has emerged as a pivotal technique for enhancing the reliability of noisy quantum devices in the *Noisy Intermediate-Scale Quantum* (NISQ) era. Recently, machine learning (ML)-based QEM approaches have demonstrated strong generalization capabilities without sampling overheads compared to conventional methods. However, evaluating these techniques is often hindered by a lack of standardized datasets and inconsistent experimental settings across different studies. In this work, we present **QEM-Bench**, a comprehensive benchmark suite of *twenty-two* datasets covering diverse circuit types and noise profiles, which provides a unified platform for comparing and advancing ML-based QEM methods. We further propose a refined ML-based QEM pipeline **QEMFormer**, which leverages a feature encoder that preserves local, global, and topological information, along with a two-branch model that captures short-range and long-range dependencies within the circuit. Empirical evaluations on QEM-Bench illustrate the superior performance of QEMFormer over existing baselines, underscoring the potential of integrated ML-QEM strategies.

1. Introduction

Quantum computing promises to revolutionize fields such as cryptography (Gisin et al., 2002) and machine learning (Biamonte et al., 2017) by efficiently solving problems that are intractable on classical hardware. However, during the *Noisy Intermediate-Scale Quantum (NISQ)* era (Brooks, 2019), noise significantly impedes the practical realization of quantum systems, limiting their performance and reliability. *Quantum Error Mitigation (QEM)* techniques have therefore emerged to restore quantum advantages on noisy hardware by algorithmically suppressing noise-induced biases via post-processing (Temme et al., 2017; Li & Benjamin, 2017; Kandala et al., 2019; Huggins et al., 2021; Czarnik et al., 2021; Bravyi et al., 2022; Daley et al., 2022). While these strategies are crucial stepping stones toward achieving near-term quantum utility beyond classical supercomputers, they often suffer from drawbacks such as high sampling overheads (Cai et al., 2023) or large qubit overheads in methods like virtual distillation (VD) (Huggins et al., 2021). Moreover, some approaches rely heavily on prior knowledge of the noise models (Liao et al., 2025), limiting their generalization capabilities.

Machine learning (ML) has recently emerged as a promising solution to these limitations, leveraging neural networks for ideal outcome prediction without qubit or sampling overheads (Kim et al., 2020a; Liao et al., 2024). Nevertheless, current ML-based methods face several challenges. First, existing feature encoders either scale exponentially with the number of qubits (Kim et al., 2020a) or focus on coarse global information (e.g., counts of gates with rotation angles between 0 and $\frac{\pi}{2}$) (Liao et al., 2024). Second, prevailing architectures rarely capture multi-range dependencies. Both Kim et al. (2020a) and Liao et al. (2024) adopt vanilla MLP structures that fail to effectively encode circuit topology. Although Liao et al. (2024) additionally explores GNN architectures, this approach offers a less direct means of learning node attributes, resulting in reduced performance. Further, the ML-QEM field lacks standardized benchmarks: current studies frequently employ distinct and narrowly focused experimental settings. Such variability in datasets and evaluation protocols has led to a limited coverage of diverse noise models, insufficient examination of generalization capabilities, and difficulties in comparison among approaches.

To address these shortcomings, we unify a collection of *twenty-two* ML-QEM datasets, named **QEM-Bench**, that standardize key experimental factors and provide a comprehensive range of benchmarking scenarios. These datasets

[1]Sch. of Computer Science & Sch. of Artificial Intelligence, Shanghai Jiao Tong University [2]Shanghai Innovation Institute. Correspondence to: Junchi Yan <yanjunchi@sjtu.edu.cn>. This work is supported by NSFC (92370201, 62222607). Code and datasets are at: https://github.com/btyll/QEM-Bench-ICML.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

encompass diverse noise configurations (incoherent, coherent, fake-provider, and real IBM devices) and various circuit types (structured 1-D Ising models (Suzuki et al., 2013), QAOA (Zhou et al., 2020), and random unstructured circuits), with qubit counts ranging from small (4 qubits) to large (50 qubits). Furthermore, unlike Kim et al.; Liao et al., we propose an ML-QEM pipeline, named QEMFormer, that encodes quantum circuits as directed acyclic graphs (DAGs), preserving both gate-level information (node-based features), circuit attributes (global-level features) and graph topology. Simultaneously, in contrast to prior ML-based QEM methods, we develop a two-branch model that leverages both MLPs for capturing short-range gate-specific context and Graph Transformers for modeling long-range dependencies and topological structure, thereby offering a more robust approach to error mitigation. **Our contributions are:**

1) **Unifying and releasing datasets for QEM.** We introduce QEM-Bench, a collection of *twenty-two* ML-QEM datasets for standardized evaluation: (i) nine standard datasets for general-purpose testing, (ii) nine advanced datasets probing generalization capabilities, and (iii) four large-scale datasets with 50-qubit and 63-qubit circuits executed on real IBM quantum devices.

2) **The proposed QEMFormer approach.** We propose a pipeline that encodes quantum circuits as directed acyclic graphs (DAGs), preserving both gate-level, global-level, and topological information. Our two-branch model, QEMFormer, integrates MLP modules for short-range context and Graph Transformers for long-range dependencies.

3) **Benchmarking baselines across datasets.** We implement existing ML-based baselines and thoroughly compare them against QEMFormer on QEM-Bench. QEMFormer consistently outperforms other methods across the standard, advanced, and large-scale real-device datasets.

2. Related Works

2.1. Quantum Device Noise

Quantum device noise can be classified into two categories, *Markovian* and *non-Markovian*, based on whether the environment retains memory of system-environment interactions (Zhang et al., 2024). Detailed descriptions of the noise types considered in this study are provided in Appendix C.

A noise process is deemed Markovian if the system's state transformation $\rho \to \rho'$ depends solely on the current state ρ, independent of previous operations or temporal context. Markovian errors can be further subdivided into two main types. Incoherent (stochastic) errors include bit-flip errors, which flip a single qubit state between $|0\rangle$ and $|1\rangle$, phase-flip errors, which alter the relative phase without changing the probability amplitudes, and depolarizing errors, where a random Pauli operator is applied to each qubit (Nielsen & Chuang, 2000). Coherent errors arise from unintended or imperfect unitary rotations within quantum circuits. These errors are typically associated with slow noise processes (Huang et al., 2023; Beale et al., 2018). Additionally, other physical errors such as amplitude damping occur when energy is dissipated from the system into the environment, leading to state decay (Blume-Kohout et al., 2022). In contrast, *non-Markovian* errors exhibit memory effects, where the noise depends on the system's history. A key indicator of non-Markovian noise is the oscillation of a qubit's coherence and purity over time. The purity p of a qubit state, defined as $p = \mathrm{Tr}[\rho^2]$, can reveal such memory effects when it varies periodically (Agarwal et al., 2024).

2.2. Quantum Error Mitigation

Among non-ML-based QEM techniques, Zero-Noise Extrapolation (ZNE) is widely adopted. It estimates noise-free expectation values by intentionally increasing the noise levels and executing quantum circuits with measurements. While enjoying the capability of generalization, ZNE demands significant computational resources due to repeated circuit executions. Clifford Data Regression (CDR) is another prominent method (Czarnik et al., 2021). CDR approximates noise-free expectation values by replacing most non-Clifford gates in the target circuit with Clifford gates, which are efficiently classically simulable. This replacement facilitates the generation of training data, enabling a linear regression model to map noisy expectation values to their ideal counterparts. Although CDR leverages the efficient simulation of Clifford circuits, it requires a substantial number of training samples and may not generalize well to circuits with a high density of non-Clifford gates.

Machine Learning (ML)-based QEM approaches have recently gained attention. For instance, (Kim et al., 2020b) utilizes neural networks (NN) and concatenated neural networks to predict errors in the measurement outcomes of quantum states. However, their approach faces scalability issues as the input feature space, consisting of noisy measurement outcomes, grows exponentially with the number of qubits. Similarly, (Liao et al., 2024) evaluates several ML models, including linear regression, random forests, multi-layer perceptrons, and graph neural networks, aiming to minimize the sum of squared errors between mitigated and ideal expectation values. These models, however, often struggle to capture both short-range and long-range dependencies within quantum circuits, resulting in limited mitigation performance. Overall, while ML-based QEM methods offer promising improvements in efficiency and scalability, they are hindered by challenges in feature encoding and the ability to model complex circuit dependencies, necessitating further advancements to fully realize their potential.

3. Building Benchmarking Datasets for QEM

This section begins by revisiting the definitions and notations of learning-based QEM in Sec. 3.1. The construction of QEM-Bench, including circuit selection, noise configurations, and dataset statistics is presented in Sec. 3.2 - 3.4. Experimental settings are outlined in Sec. 3.5.

3.1. Problem Revisiting

We first provide a brief overview of the key concepts related to Quantum Error Mitigation (QEM). For details about quantum information and computing, please refer to the seminal textbook (Nielsen & Chuang, 2000).

3.1.1. Quantum State Vector

A quantum state is described by a state vector (or ket) in a complex Hilbert space. For a qubit, the state is: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. For an n-qubit system, the state vector resides in a 2^n-dimensional Hilbert space: $|\psi\rangle = \sum_{i=1}^{2^n} \alpha_i |i\rangle$, satisfying $\sum_{i=1}^{2^n} |\alpha_i|^2 = 1$. Quantum states evolve via quantum circuits (QC), which consist of applying quantum gates in sequence.

3.1.2. Expectation Value in Quantum Circuits

Consider a quantum circuit represented by the unitary U acting on an initial state $|\psi_0\rangle$. The **ideal expectation value** of an observable O is: $\langle O\rangle^{\text{ideal}} = \langle\psi_0|U^\dagger O U|\psi_0\rangle$. In practice, the measured value $\langle O\rangle^{\text{noisy}}$ deviates from the ideal due to factors such as decoherence, gate imperfections, and other noise sources (Cai et al., 2023). QEM aims to recover $\langle O\rangle^{\text{ideal}}$ from these noisy measurements $\langle O\rangle^{\text{noisy}}$.

Observables, Pauli Operators, and Hamiltonians. Observables are typically represented by Hermitian operators, with **Pauli operators** (e.g., $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{I}$, also named as Pauli matrices) serving as a fundamental basis for single-qubit measurements. For multi-qubit systems, operators are often expressed as tensor products of Pauli matrices, enabling the decomposition of more complex observables. A **Hamiltonian**, another crucial Hermitian operator, specifies the energy structure of a quantum system and governs its dynamics (Preskill, 2018). Many quantum applications, such as quantum chemistry and materials science simulations, involve estimating the expectation value of Hamiltonians or sums of Pauli operators (e.g., in variational quantum eigensolver (VQE) frameworks) (Peruzzo et al., 2014).

3.1.3. Learning-Based QEM Formulation

For machine learning-based approaches, the problem can be framed as a regression task. Specifically, we formulate the prediction of the ideal expectation value $\langle O\rangle^{\text{ideal}}$ as a graph-level regression task, although it is not strictly necessary to use graph structures when employing models such as MLPs.

Let $\mathcal{D} = \{(\mathcal{G}_k, \mathbf{X}_k, \mathbf{A}_k, y_k^{\text{noisy}}, y_k)\}_{k=1}^N$ denote a dataset of N quantum circuits, where each circuit C_k is represented as a directed acyclic graph (DAG) $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$. Nodes and edges in the graph are associated with feature vectors $\mathbf{X}_k \in \mathbb{R}^{|\mathcal{V}_k| \times d}$ and an adjacency matrix $\mathbf{A}_k \in \mathbb{R}^{|\mathcal{V}_k| \times |\mathcal{V}_k|}$, respectively. The noisy and ideal expectation values are denoted as $y_k^{\text{noisy}} = \langle O\rangle_k^{\text{noisy}}$ and $y_k = \langle O\rangle_k^{\text{ideal}}$.

The objective is to learn a graph transformer model f that maps each circuit's graph representation and its noisy measurement to a prediction $\hat{y}_k$ of the ideal expectation value:

$$\hat{y}_k = f(\mathcal{G}_k, \mathbf{X}_k, \mathbf{A}_k, y_k^{\text{noisy}}),$$

by minimizing the mean squared error (MSE) between predictions and true labels:

$$\min_f \frac{1}{N} \sum_{k=1}^{N} \big(y_k - \hat{y}_k\big)^2.$$

Without loss of generality, we focus on a single sample $\big(\mathcal{G}, \mathbf{X}, \mathbf{A}, y^{\text{noisy}}, y\big)$ for the remainder of the paper.

QEMFormer is divided into two major subsets: standard datasets and challenging datasets. As the QEM benchmark is constructed based on the quantum circuits, we first introduce the circuits we used for the benchmark.

3.2. Circuit Selection

We evaluate our approach across three representative classes of quantum circuits, each distinguished by unique structures and parameters: (i) Trotterized one-dimensional transverse-field Ising model (TFIM) circuits, (ii) randomly generated circuits, and (iii) QAOA circuits designed for the MaxCut problem. Each class is described in detail below.

Trotterized TFIM Circuits. Consider time-evolution circuits for the one-dim TFIM, governed by the Hamiltonian:

$$\hat{H} = -J \sum_j \hat{Z}_j \hat{Z}_{j+1} + h \sum_j \hat{X}_j = -J\, \hat{H}_{ZZ} + h\, \hat{H}_X, \quad (1)$$

where J is the nearest-neighbor exchange coupling and h is the transverse field strength (Suzuki et al., 2013). To simulate the time evolution over an interval t, one applies the unitary $U(t) = \exp\big(-i\, \hat{H}\, t\big)$. In practice, we use the first-order Trotter-Suzuki decomposition:

$$U(t) \approx \Big(e^{-i\,(-J\,\hat{H}_{ZZ})\,\Delta t}\, e^{-i\,(h\,\hat{H}_X)\,\Delta t}\Big)^{N_{\text{Trot}}}, \quad (2)$$

where $\Delta t = t/N_{\text{Trot}}$ and N_{Trot} is the number of Trotter steps. Each term in Eq. 2 is further decomposed into elementary gates (e.g., `CNOT` and single-qubit rotations). We randomly sample J, h, and t, varying N_{Trot} from 1 to 20 to produce circuits of different depths.

Random Unstructured Circuits. To evaluate the generality of models, we also construct *randomized* quantum circuits. Then, each gate in the circuit is selected uniformly from:

Table 1. Overview of datasets in QEM-Bench

Setting Type	Train/Validation		Test		Qubit Num
	# (Train/Val)	Key Parameters	#	Key Parameters	#
Trotter-Standard	800/100	Steps 1–20	300	Steps 1–20	10
Random-Standard	800/100	Size 10–150, $\theta \in [0, 2\pi]$	300	Size 10–150, $\theta \in [0, 2\pi]$	3-6
QAOA-Standard	800/100	Layers 12–18	300	Layers 12–18	6
Trotter-Step Zero-Shot	765/135	Steps 1–15, Pauli-Z	300	Steps 16–20, Pauli-Z	10
Random-Size Zero-Shot	658/117	$\leq$100 gates, Pauli-Z	425	100–150 gates, Pauli-Z	3-6
Unseen Pauli-Basis Obs	800/200	Steps 1–20, random Pauli-basis obs	200	Steps 1–20, New Pauli-basis obs	4
Kyiv Pre	400/78	Trotter Circs; extreme outlier filtered	113	Trotter Circs; extreme outlier filtered	50
Kyiv Raw	467/50	Trotter Circs	100	Trotter Circs	50
Brisbane Pre	500/61	Trotter Circs; extreme outlier filtered	148	Trotter Circs; extreme outlier filtered	63
Brisbane Raw	800/100	Trotter Circs	300	Trotter Circs	63

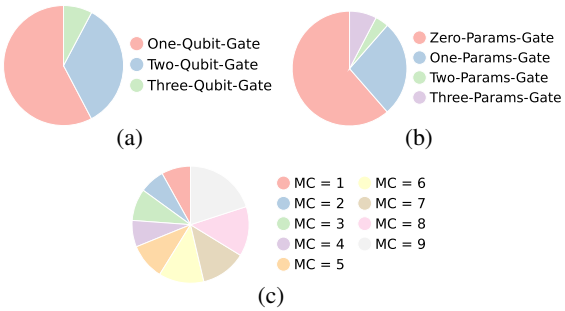

Figure 1. Distribution of gate types in the quantum circuit among datasets. (a) Proportions of one-, two-, and three-qubit gates in the random circuit dataset. (b) Distribution of zero-parameter, one-, two-, and three-parameter gates in the random circuit dataset. (c) Distribution of the maximum cut solutions for QAOA circuits in the QAOA dataset, grouped by solution value (MC = 1 to 9).

1) *Single-qubit gates:* {`id`, `u1`, `u2`, `u3`, `x`, `y`, `z`, `h`, `s`, `sdg`, `t`, `tdg`, `rx`, `ry`, `rz`}

2) *Two-qubit gates:* {`cx`, `cy`, `cz`, `ch`, `crz`, `cu1`, `cu3`, `swap`, `rzz`}

3) *Three-qubit gates:* {`ccx`, `cswap`}.

For parameterized gates (e.g. `u3`, `rx`, `ry`, `rz`), rotation angles are sampled from $[0, 2\pi]$. The qubit(s) upon which a gate acts are chosen uniformly among the available qubits.

For the random circuits, we set the max qubit number over teh This procedure yields a diverse ensemble of circuits with varying structures and depths.

QAOA Circuits for MaxCut. The Quantum Approximate Optimization Algorithm (QAOA) (Farhi et al., 2014; Harrigan & Sung, 2021) is designed to solve combinatorial optimization problems by encoding the objective function into a cost Hamiltonian. We focus on QAOAs for MaxCut in this paper, where one seeks to partition the vertices of a graph $G = (V, E)$ into two subsets to maximize the number of edges between them. Labeling each vertex $i \in V$ with a binary variable $z_i \in \{+1, -1\}$, the objective can be written as (Guerreschi & Matsuura, 2019):

$$\max_{\{z_i\}} \frac{1}{2} \sum_{(i,j) \in E} \big(1 - z_i\, z_j\big). \tag{3}$$

QAOA encodes this objective into the cost of Hamiltonian

$$H_C = \tfrac{1}{2} \sum_{(i,j) \in E} \big(\hat{I} - \hat{Z}_i\, \hat{Z}_j\big), \tag{4}$$

where $\hat{Z}_i$ is the Pauli-Z operator acting on qubit i. The QAOA circuit alternates between phase-separation unitaries $U_C(\gamma) = \exp\big(-i\,\gamma\, H_C\big)$ and mixing unitaries $U_B(\beta) = \exp\big(-i\,\beta\, H_B\big)$, where $H_B = \sum_{j=1}^{|V|} \hat{X}_j$, and $\{\gamma, \beta\}$ are variational parameters.

3.3. Circuit Set Statistics

We construct three distinct sets of quantum circuits. Statistics for each set of circuits are presented in Tab. 1. Each dataset comprises 1,200 unique circuits.

For the randomized circuits, the distributions of single-, two-, and three-qubit gates are illustrated in Fig. 1(a), and the distribution of gates requiring zero, one, two, or three rotation parameters is depicted in Fig. 1(b).

For QAOAs, we fix the number of qubits to 6, targeting the MaxCut: $N_q = 6$. Each circuit corresponds to a randomly generated 6-node graph $G = (V, E)$, where $|V| = 6$. The number of QAOA layers p for each circuit is randomly set from the integer range $[12, 18]$. The distribution of MaxCut values across generated graphs is given in Fig. 1(c).

3.4. Noise Configurations

We consider three primary noise configurations:

Fake Providers. To emulate the noise characteristics of real quantum devices, We utilize `FakeHanoiV2` and `FakeWashington`.

Incoherent Errors. Incoherent noise is simulated by applying Pauli-X and depolarizing errors to all gates, based on the Sycamore quantum device (Arute et al., 2019). The error rates are set as follows: single-qubit gate error at 0.16%, two-qubit gate error at 0.62%, and read-out error at 3.8%.

Coherent Errors. To further examine models' capabilities among types of noises, coherent noise is introduced by adding systematic over-rotation errors to two-qubit gates (CX, CY, CZ, and Swap) with an average over-rotation angle of 0.02π, in combination with the noise model derived from

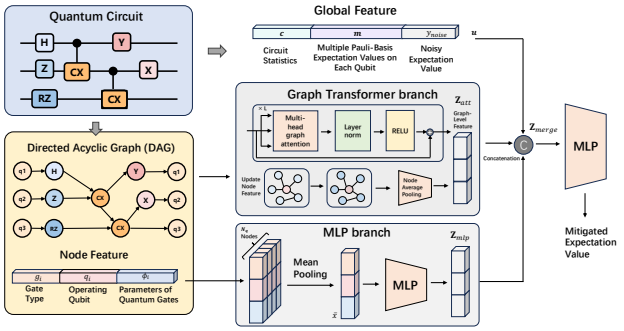

Figure 2. The structure of proposed QEMFormer.

the `FakeWashington` provider.

Real Quantum Devices. For experiments of large-scale circuits, we executed circuits on `IBM_Kyiv` and `IBM Brisbane`, two 127-qubit quantum computers provided by IBM.

3.5. Experimental Settings

We construct a total of **22 datasets** by combining three main setting types (*standard*, *advanced*, and *large-scale*) with three circuit families (*QAOA*, *random*, *Trotter*) and three noise configurations (*incoherent*, *all noise*, *provider*). Each dataset is partitioned into training, validation, and test sets.

3.5.1. Standard Settings

The **9 standard** datasets are derived from all possible combinations of (circuit family) $\times$ (noise configuration). Each standard dataset contains circuits with fixed depth/size and a predefined set of measurement operators (e.g., all Pauli-**Z**) that have been previously introduced. These datasets serve as a baseline evaluation for error mitigators under well-controlled conditions.

3.5.2. Advanced Settings

To examine generalization beyond the configurations seen in training, we introduce three advanced benchmarks, each paired with the same set of three noise configurations. This results in a total of **9 advanced** datasets:

Trotter-Step Zero-Shot: Trains on Trotter circuits with shallower depths, then tests on circuits with deeper (previously unseen) Trotter steps.

Random-Size Zero-Shot: Trains on random circuits up to a certain gate-count threshold, and tests on larger, more complex circuits outside the training distribution.

Unseen Pauli-Basis Observables: Trains on random n-qubit observables drawn from the set $\{\mathbf{I}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}\}^{\otimes n}$. During testing, each circuit is paired with an observable that was not included in the training set, thereby challenging the model to generalize to novel measurement operators.

3.5.3. Large-Scale Settings

We curate **4** datasets: two comprising 50-qubit Trotterized circuits run on `IBM_Kyiv` and two comprising 63-qubit Trotterized circuits run on `IBM_Brisbane`. In the *Raw* variants (*Kyiv Raw*, *Brisbane Raw*), noisy expectation values are taken directly from the measurement outcomes. In the corresponding *Pre* variants (*Kyiv Pre*, *Brisbane Pre*), we remove only the most extreme outliers that reflect severe noise corruption.

By incorporating standard, advanced, and large-scale datasets, QEM-Bench offers a comprehensive and multi-faceted evaluation of error mitigators. Standard datasets assess their general performance, advanced datasets examine their ability to extrapolate to new depths, sizes, and observables, and large-scale datasets evaluate their potential for practical application.

4. Proposed QEMFormer

To address the limitations of existing ML-based QEM methods, we introduce **QEMFormer** to enhance feature extraction and leverage the topology of quantum circuits. It encodes quantum circuits as directed acyclic graphs (DAGs), enabling the extraction of both local (gate-level) and global (circuit-level) features. Our architecture (presented in Fig. 2) employs a two-branch design: one branch utilizes multilayer perceptrons (MLPs) to capture short-range gate contexts, while the other leverages graph transformers with self-attention mechanisms to model long-range dependencies within the circuit. This integrated approach facilitates the effective interpretation of multi-ranged contextual information, leading to improved prediction of EVs.

4.1. Feature Encoder

Let $\mathcal{C}$ be a given quantum circuit operating on N_q qubits. In line with (Moflic et al., 2023; He et al., 2023), we construct a directed acyclic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from $\mathcal{C}$, where each node $v_i \in \mathcal{V}$ represents a quantum operation (or a special "start"/"end" symbol for each qubit), and edges $(v_i \to v_j) \in \mathcal{E}$ signify the directed flow of operations along qubits in $\mathcal{C}$.

4.1.1. Node-Level Features

Let $N = |\mathcal{V}|$ be the total number of nodes. We endow each node v_i with a feature vector $\mathbf{z}_i \in \mathbb{R}^d$, where

$$\mathbf{x}_i = \left[\mathbf{g}_i,\, \mathbf{q}_i,\, \boldsymbol{\phi}_i\right].$$

Here: $\mathbf{g}_i \in \{0,1\}^{N_{\text{type}}}$ is a one-hot vector representing the type of the i-th gate (e.g., $\mathrm{CNOT}, \mathrm{RX}, \mathrm{H}, \ldots$). The dimension N_{type} equals the total number of unique gate types in the dataset. $\mathbf{q}_i \in \{0,1\}^{N_q}$ is a multi-hot encoding that indicates which qubits are operated on by the gate associated with node v_i. $\boldsymbol{\phi}_i \in \mathbb{R}^{d_\phi}$ encodes all possible continuous parameters for parameterized gates (e.g., rotation angles). Specifically, if the dataset includes up to d_ϕ parameters, e.g.,

each gate might have zero/single/multiple rotation angles, then $\boldsymbol{\phi}_i$ reserves one entry for each rotation angle. In this way, $\mathbf{x}_i$ captures gate properties from diverse perspectives.

4.1.2. Global-Level Features

Besides local gate-level features, we also extract a global feature vector $\mathbf{u} \in \mathbb{R}^{d_u}$ to capture circuit-wide properties. This consists of two parts:

Circuit Statistics. We first extract the count of single-, two-, and three-qubit gates; we also gather the binned rotation angles including the number of gates whose rotation angle(s) lie in each of the intervals $[0, \frac{\pi}{2}), [\frac{\pi}{2}, \pi), [\pi, \frac{3\pi}{2}), [\frac{3\pi}{2}, 2\pi)$; and additionally we count the total number of gate operations within each circuit. We then normalize these circuit-related features to $[0, 1]$.

Multiple Pauli-Basis Expectation Values. For a quantum circuit with N_q qubits, we measure the expectation values of single-qubit Pauli operators $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ on each qubit, while applying the identity operator $\mathbf{I}$ to all other qubits. For each qubit $i \in \{1, \ldots, N_q\}$ and each Pauli operator $\mathbf{P} \in \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$, the measurement operator is defined as

$$\mathbf{P}_i = \mathbf{I}^{\otimes(i-1)} \otimes \mathbf{P} \otimes \mathbf{I}^{\otimes(N_q-i)}.$$

We then obtain the expectation values $\langle \mathbf{P}_i \rangle$ for all combinations of i and P, resulting in a total of $3N_q$ measurements. These expectation values provide a comprehensive characterization of the global noise affecting the quantum hardware from the qubit level. Formally, let

$$\mathbf{u} = \left[\mathbf{c},\, \mathbf{m},\, y^{\texttt{noisy}}\right] \in \mathbb{R}^{8+3N_q+1},$$

where $\mathbf{c} \in \mathbb{R}^8$ aggregates circuit-level statistics, including counts and binned angle distributions, $\mathbf{m} \in \mathbb{R}^{3N_q}$ contains the noisy expectation values of the $\{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$ observables measured across N_q qubits, and $y^{\texttt{noisy}} \in \mathbb{R}$ is the raw noisy expectation value of the target observable to predict.

Overall, each node v_i in the graph $\mathcal{G}$ is associated with a local feature vector $\mathbf{x}_i \in \mathbb{R}^d$, and the entire circuit $\mathcal{C}$ is denoted by a global feature vector $\mathbf{u} \in \mathbb{R}^{8+3N_q+1}$. This encoding scheme via graph effectively captures local gate interactions, circuit topology, and global context, offering a rich and scalable representation for QEM tasks. Also, it maintains an acceptable dimensionality compared to traditional encoders, ensuring computational efficiency.

4.2. Model Architecture

We adopt a two-branch design, to capture the short-range context, the long-range information, and the topological structure of quantum circuits. Denote by $\mathbf{X} \in \mathbb{R}^{N \times d_{in}}$ the initial node features for N nodes (gates) in the circuit graph, and by $\mathbf{u} \in \mathbb{R}^{d_u}$ the global (circuit-level) feature vector. Our model processes $\mathbf{X}$ in two parallel branches: an *MLP branch* and a *Graph Transformer branch*. The outputs of these branches are then concatenated together with $\mathbf{u}$, forming an aggregated representation that is passed through an additional 2-layer MLP to predict the expectation value.

(1) MLP Branch. In this branch, we first perform global mean-pooling on the node features to obtain a single vector:

$$\overline{\mathbf{x}} \;=\; \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \quad \in \mathbb{R}^{d_{in}}. \tag{5}$$

We then feed $\overline{\mathbf{x}}$ into a multi-layer perceptron (MLP) composed of K sequential layers with bath normalization (BN):

$$\begin{aligned} &\mathbf{x}^{(0)} = \overline{\mathbf{x}}, \qquad k = 1, \ldots, K. \\ &\mathbf{x}^{(k)} = \mathrm{Dropout}\Big(\mathrm{ReLU}\big(\mathrm{BN}\big(\mathbf{x}^{(k-1)}\mathbf{W}_{\mathrm{mlp}}^{(k)} + \mathbf{b}_{\mathrm{mlp}}^{(k)}\big)\big)\Big), \end{aligned} \tag{6}$$

At the final layer, we obtain the MLP branch output:

$$\mathbf{Z}_{\mathrm{mlp}} \;=\; \mathbf{x}^{(K)} \quad \in \mathbb{R}^{d_{mlp}}. \tag{7}$$

(2) Graph Transformer Branch. In parallel, we process the node features $\mathbf{X}$ via a Graph Transformer, which generalizes the self-attention mechanism (Vaswani et al., 2017; Devlin et al., 2019) to graph-structured data. For each Graph Transformer layer $l = 0, \ldots, L-1$, we compute multi-head attention as follows: for the c-th head ($c = 1, \ldots, C$),

$$\begin{aligned} &\mathbf{q}_{c,i}^{(l)} = W_{c,q}^{(l)}\,\mathbf{x}_i^{(l)} + \mathbf{b}_{c,q}^{(l)}, \quad \mathbf{k}_{c,j}^{(l)} = W_{c,k}^{(l)}\,\mathbf{x}_j^{(l)} + \mathbf{b}_{c,k}^{(l)}, \\ &\alpha_{c,ij}^{(l)} = \frac{\exp\Big(\frac{\langle \mathbf{q}_{c,i}^{(l)}, \mathbf{k}_{c,j}^{(l)} \rangle}{\sqrt{d}}\Big)}{\sum_{u \in \mathcal{N}(i)} \exp\Big(\frac{\langle \mathbf{q}_{c,i}^{(l)}, \mathbf{k}_{c,u}^{(l)} \rangle}{\sqrt{d}}\Big)}, \end{aligned} \tag{8}$$

where $\mathbf{x}_i^{(l)}$ is the feature of node i at layer l, and d is the head dimension. We then aggregate messages from neighbors:

$$\begin{aligned} \mathbf{v}_{c,j}^{(l)} &= W_{c,v}^{(l)}\,\mathbf{x}_j^{(l)} \;+\; \mathbf{b}_{c,v}^{(l)}, \\ \widehat{\mathbf{x}}_i^{(l+1)} &= \Big\|_{c=1}^{C} \Big[\sum_{j \in \mathcal{N}(i)} \alpha_{c,ij}^{(l)}\,\mathbf{v}_{c,j}^{(l)}\Big], \end{aligned} \tag{9}$$

where $\|_{c=1}^{C}[\cdot]$ denotes concatenation along the head dimension. To further stabilize training, we employ a gated residual connection with layer normalization (LN):

$$\begin{aligned} \mathbf{r}_i^{(l)} &= W_r^{(l)}\,\mathbf{x}_i^{(l)} + \mathbf{b}_r^{(l)}, \\ \boldsymbol{\beta}_i^{(l)} &= \sigma\Big(W_g^{(l)}\,\big[\widehat{\mathbf{x}}_i^{(l+1)};\, \mathbf{r}_i^{(l)};\, \widehat{\mathbf{x}}_i^{(l+1)} - \mathbf{r}_i^{(l)}\big]\Big), \\ \mathbf{x}_i^{(l+1)} &= \mathrm{ReLU}\Big(\mathrm{LN}\big((1 - \boldsymbol{\beta}_i^{(l)})\,\widehat{\mathbf{x}}_i^{(l+1)} \;+\; \boldsymbol{\beta}_i^{(l)}\,\mathbf{r}_i^{(l)}\big)\Big). \end{aligned} \tag{10}$$

After L layers, we obtain $\mathbf{X}^{(L)} = \{\mathbf{x}_1^{(L)}, \ldots, \mathbf{x}_N^{(L)}\}$. To produce a circuit-level representation for the final layer, we apply a global readout (e.g., mean-pooling):

$$\mathbf{Z}_{\mathrm{att}} \;=\; \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^{(L)} \quad \in \mathbb{R}^{d_{att}}. \tag{11}$$

Table 2. RMSE comparison over different settings and noise models (our method: QEMFormer). The best-performing method is highlighted in red, while the second-best is highlighted in blue. Lower RMSE values indicate better performance.

Method	Random Size Zero Shots			Trotter Step Zero Shots			Unseen Obs		
	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider
Noisy	0.179 ± 0.117	0.125 ± 0.081	0.131 ± 0.091	0.313 ± 0.101	0.498 ± 0.147	0.535 ± 0.170	0.134 ± 0.113	0.118 ± 0.097	0.134 ± 0.112
QEMFormer	0.090 ± 0.061	0.113 ± 0.078	0.069 ± 0.050	0.025 ± 0.015	0.151 ± 0.121	0.099 ± 0.051	0.098 ± 0.074	0.080 ± 0.057	0.096 ± 0.073
MLP	0.116 ± 0.073	0.119 ± 0.080	0.105 ± 0.074	0.049 ± 0.030	0.183 ± 0.148	0.117 ± 0.058	0.139 ± 0.110	0.146 ± 0.109	0.116 ± 0.093
GNN	0.200 ± 0.132	0.162 ± 0.105	0.074 ± 0.054	0.024 ± 0.017	0.179 ± 0.148	0.123 ± 0.062	0.102 ± 0.078	0.124 ± 0.099	0.104 ± 0.078
OLS	0.259 ± 0.151	0.164 ± 0.098	0.155 ± 0.093	0.026 ± 0.018	0.215 ± 0.141	0.098 ± 0.056	0.131 ± 0.087	0.090 ± 0.061	0.125 ± 0.089
ZNE	0.175 ± 0.114	0.157 ± 0.110	0.120 ± 0.090	0.255 ± 0.057	0.479 ± 0.159	0.279 ± 0.115	0.154 ± 0.133	0.111 ± 0.090	0.117 ± 0.100
CDR	0.080 ± 0.049	0.119 ± 0.082	0.072 ± 0.049	0.039 ± 0.032	0.293 ± 0.228	0.153 ± 0.128	0.119 ± 0.100	0.109 ± 0.092	0.123 ± 0.105
RF	0.113 ± 0.075	0.122 ± 0.082	0.086 ± 0.066	0.064 ± 0.044	0.206 ± 0.173	0.244 ± 0.100	0.131 ± 0.105	0.100 ± 0.076	0.110 ± 0.085
GTranQEM	0.103 ± 0.065	0.118 ± 0.080	0.073 ± 0.053	0.037 ± 0.023	0.183 ± 0.152	0.165 ± 0.095	0.100 ± 0.078	0.080 ± 0.061	0.113 ± 0.084

Table 3. RMSE for various circuit types and noise models. Multiple implementations of ZNE, CDR, and RF consistently performed weakly on QAOA circuits and are therefore excluded from the mitigation results to ensure a fair comparison.

Method	Random			QAOA			Trotter		
	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider
Noisy	0.192 ± 0.132	0.132 ± 0.091	0.129 ± 0.090	0.186 ± 0.106	0.174 ± 0.092	0.147 ± 0.072	0.278 ± 0.139	0.455 ± 0.217	0.414 ± 0.216
QEMFormer	0.087 ± 0.055	0.095 ± 0.064	0.054 ± 0.036	0.054 ± 0.035	0.061 ± 0.039	0.033 ± 0.021	0.024 ± 0.015	0.189 ± 0.145	0.062 ± 0.041
MLP	0.093 ± 0.061	0.139 ± 0.094	0.063 ± 0.046	0.056 ± 0.038	0.068 ± 0.043	0.047 ± 0.028	0.032 ± 0.023	0.200 ± 0.139	0.082 ± 0.055
GNN	0.111 ± 0.075	0.096 ± 0.064	0.080 ± 0.059	0.063 ± 0.042	0.070 ± 0.045	0.055 ± 0.037	0.037 ± 0.035	0.190 ± 0.156	0.080 ± 0.054
OLS	0.217 ± 0.132	0.153 ± 0.097	0.117 ± 0.076	0.100 ± 0.061	0.123 ± 0.078	0.066 ± 0.040	0.054 ± 0.039	0.200 ± 0.155	0.090 ± 0.066
ZNE	0.188 ± 0.136	0.146 ± 0.103	0.116 ± 0.088	-	-	-	0.247 ± 0.144	0.438 ± 0.220	0.213 ± 0.108
CDR	0.063 ± 0.043	0.101 ± 0.077	0.059 ± 0.044	-	-	-	0.175 ± 0.162	0.222 ± 0.182	0.405 ± 0.315
RF	0.127 ± 0.093	0.120 ± 0.085	0.111 ± 0.090	-	-	-	0.059 ± 0.035	0.185 ± 0.151	0.096 ± 0.066
GTranQEM	0.089 ± 0.060	0.103 ± 0.069	0.066 ± 0.046	0.054 ± 0.037	0.070 ± 0.044	0.045 ± 0.030	0.033 ± 0.023	0.198 ± 0.157	0.051 ± 0.033

(3) Feature Concatenation and Regressor. Let $\mathbf{Z}_{\text{mlp}} \in \mathbb{R}^{d_m}$ be the MLP branch output from Eq. 6, and $\mathbf{Z}_{\text{att}} \in \mathbb{R}^{d_h}$ be the final Graph Transformer output. We concatenate these two vectors with the global feature $\mathbf{u} \in \mathbb{R}^{d_u}$:

$$\mathbf{Z}_{\text{merged}} = \left[\mathbf{Z}_{\text{mlp}}, \mathbf{Z}_{\text{att}}, \mathbf{u}\right] \mathbf{W}_{\text{merge}} \in \mathbb{R}^{d_{\text{merge}}}, \quad (12)$$

where $\mathbf{W}_{\text{merge}} \in \mathbb{R}^{(d_{mlp}+d_{att}+d_u)\times d_{\text{merge}}}$ is trainable.

Finally, we feed $\mathbf{Z}_{\text{merged}}$ into an additional 2-layer MLPs (with batch normalization, dropout, nonlinear activation) to obtain the final prediction of the ideal expectation value:

$$\hat{y} = \text{MLP}_{\text{reg}}\left(\mathbf{Z}_{\text{merged}}\right), \quad (13)$$

where $\hat{y}$ denotes the error-mitigated expectation value. This two-branch design enables the model to simultaneously focus on gate features (via MLPs) and explore long-range patterns as well as circuit topology (via Graph Transformer), ultimately enhancing the predictive accuracy for QEM.

5. Benchmark Experiments

5.1. Setups and Evaluation Metrics

The quantum circuits in our experiments are simulated and executed on IBM's backend, and the 50-qubit circuits are executed on the IBM Kyiv device. See details for system and model configurations in Appendix A.

Table 4. Results of various QEM techniques on real-device datasets. Best entries are red, second-best entries are blue.

Method	Kyiv Pre			Kyiv Raw			Brisbane Pre			Brisbane Raw		
	MAE	RMSE	STD	MAE	RMSE	STD	MAE	RMSE	STD	MAE	RMSE	STD
Noisy	0.057	0.069	0.039	0.132	0.238	0.198	0.219	0.263	0.145	0.597	0.760	0.470
OLS	0.037	0.045	0.026	0.103	0.233	0.209	0.120	0.182	0.136	0.154	0.280	0.235
QEMFormer	0.018	0.026	0.020	0.098	0.223	0.208	0.098	0.164	0.131	0.123	0.272	0.242
MLP	0.020	0.029	0.020	0.151	0.226	0.167	0.163	0.203	0.122	0.144	0.277	0.236
GNN	0.035	0.045	0.029	0.114	0.237	0.208	0.118	0.179	0.135	0.248	0.340	0.233
ZNE	0.047	0.076	0.060	0.123	0.245	0.212	0.191	0.386	0.335	0.419	0.640	0.483
RF	0.030	0.037	0.022	0.100	0.237	0.215	0.073	0.174	0.158	0.165	0.297	0.247
GTranQEM	0.019	0.027	0.019	0.108	0.235	0.200	0.107	0.171	0.133	0.160	0.292	0.244

RMSE. The Root Mean Squared Error (RMSE) quantifies the square root of the average squared differences between the predicted values and the actual ground truth values. A lower RMSE indicates higher prediction accuracy, reflecting the model's effectiveness in minimizing prediction errors.

AE and MAE. The Absolute Error (AE) for each prediction measures the absolute difference between the predicted and actual values, i.e., $\hat{y}$ and y, and the Mean Absolute Error (MAE) is the average of these absolute errors among dataset. Formulas are provided in Appendix D.

5.2. Compared Methods

We evaluate several baseline methods for quantum error mitigation, encompassing machine learning approaches and classical statistical models. Below, we briefly describe each method. For ML-based methods, we implement:

Vanilla MLPs. We adopt the standard MLP architecture utilized in Kim et al. (2020a) and Liao et al. (2024), a two-layer MLP with ReLU activation and batch normalization. The second layer concatenates the noisy expectation value to predict the ideal one.

GNNs. Following Liao et al. (2024), we implement a GNN that begins with a linear projection of the input features and then applies multiple Transformer-based convolutional layers interleaved with adaptive pooling operations. The layer-wise representations are combined using a Jumping Knowledge mechanism to capture multi-scale information. Finally, the aggregated embedding is concatenated with the noisy EV and passed through fully connected layers to produce the ideal EV.

Random Forest (RF). An ensemble of 100 decision trees trained on bootstrap samples, each split selecting a random

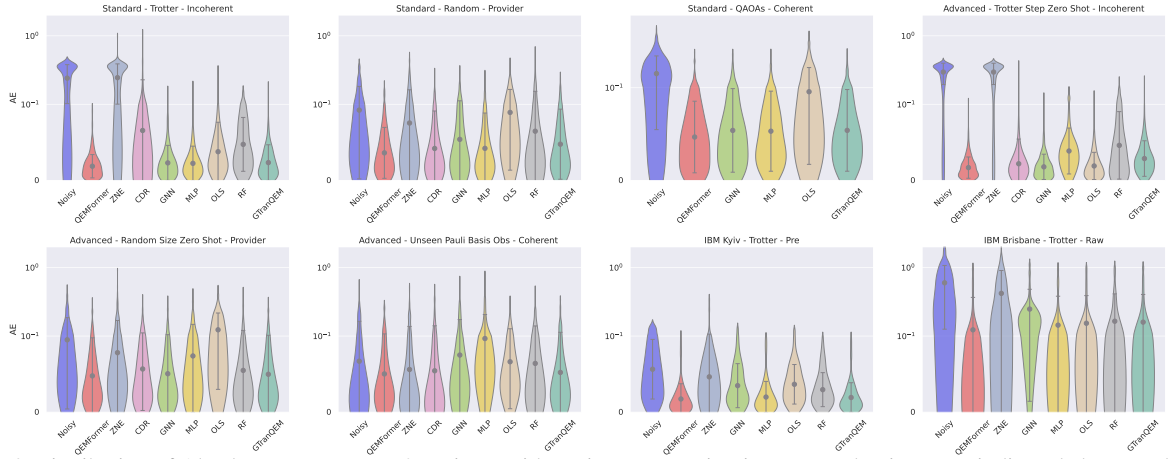

Figure 3. Distribution of Absolute Errors across 8 settings, with setting names, circuit types, and noise types indicated above each figure. Lower AE values (closer to 0) represent better performance.

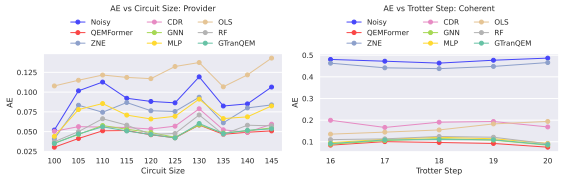

Figure 4. AE over circuit size for random circuits (left) and Trotter steps for Trotterized circuits (right) in the zero-shot task. The training uses circuits with sizes < 100 or Trotter steps < 15.

subset of features and using mean squared error reduction as the criterion (Liao et al., 2024). The final prediction is the average of all tree outputs, allowing for the modeling of complex, nonlinear dependencies.

GTranQEM. A quantum error mitigation framework that employs a non-message-passing graph transformer architecture, as introduced in (Bao et al., 2025).

For non-ML-based mitigators, we implement:

OLS. Ordinary Least Squares (OLS) regression model (Liao et al., 2024) is an approach assuming a linear relationship between the target variable and the input features. This model identifies optimal coefficients by minimizing the sum of squared residuals between observed and predicted values.

CDRs. Clifford Data Regression (CDR) (Czarnik et al., 2021) is trained on circuits modified with Clifford gate replacements and is evaluated on the original circuits. CDR leverages this data to learn corrections for error mitigation.

For reference, we also report results from Zero-Noise Extrapolation (ZNE) (Temme et al., 2017; Li & Benjamin, 2017), a cornerstone non-ML technique in QEM.

5.3. Experimental Results Analysis

On Standard Settings. Tab. 3 presents the RMSE across nine standard datasets. QEMFormer consistently achieves top or second-best performance across configurations. For instance, on random circuits with fake providers, QEMFormer attains the lowest RMSE of 0.054, compared to MLP's RMSE of 0.063 and GNN's RMSE of 0.080. Similarly, for Trotter circuits with incoherent noise, QEMFormer exhibits the best performance with an RMSE of 0.024.

On Advanced Settings. QEMFormer also shows strong performance in advanced settings, generalizing and extrapolating over circuit size, Trotter steps, and unseen Pauli-basis observables (see Tab. 2). For example, under the Trotter step zero-shot setting with incoherent noise, QEMFormer achieves an RMSE of 0.025, outperforming other methods. Furthermore, it consistently performs best under the coherent noise setting, which is more complex compared to incoherent and fake provider settings.

On Large-Scale Circuits. Tab. 4 reports results for 50-qubit circuits on IBM Kyiv and 63-qubit circuits on IBM Brisbane. QEMFormer attains the lowest errors on both platforms—achieving an MAE of 0.018 and RMSE of 0.026 on the Kyiv Pre, and an MAE of 0.123 and RMSE of 0.272 on the Brisbane Raw—thereby demonstrating its robustness to real-device noise, with or without outlier filtering.

We employ violin plots to illustrate the distribution of AE across test sets for comprehensive evaluation (see Fig. 3) and examine how AE varies with circuit size and Trotter steps in random and trotter zero-shot settings, respectively (see Fig. 4), demonstrating the stability of QEMFormer performance. Overall, these three categories of experiments demonstrate that our approach offers (a) superior or near-best accuracy across varied circuit types and noise conditions, (b) stable performance that generalizes to new operations or circuit sizes, and (c) strong real-hardware applicability on large-scale circuits.

6. Conclusion and Outlook

We have presented a comprehensive benchmark, QEM-Bench for quantum error mitigation, as well as a strong baseline QEMFormer. All resources will be released and the benchmark would be updated continuously.

Impact Statement

Quantum computing holds transformative potential for AI and beyond, yet its practical realization is hindered by quantum noise. This paper establishes a standardized benchmark for evaluating quantum error mitigation (QEM) techniques, addressing the need for fair and transparent comparisons. Additionally, we propose a novel QEM method, rigorously validated through extensive experiments. By facilitating open benchmarking, we believe that our work potentially fosters the development of hardware-agnostic error mitigation strategies, accelerating progress toward fault-tolerant quantum computation and advancing the broader quantum ecosystem.

References

Agarwal, A., Lindoy, L. P., Lall, D., Jamet, F., and Rungger, I. Modelling non-markovian noise in driven superconducting qubits. *Quantum Science and Technology*, 9(3): 035017, 2024.

Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J. C., Barends, R., Biswas, R., and Boixo, S. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, Oct 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1666-5.

Bao, T., Ye, X., Ruan, H., Liu, C., Wu, W., and Yan, J. Beyond circuit connections: A non-message passing graph transformer approach for quantum error mitigation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Beale, S. J., Wallman, J. J., Gutiérrez, M., Brown, K. R., and Laflamme, R. Coherence in quantum error-correcting codes. *arXiv preprint arXiv:1805.08802*, 2018.

Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., and Lloyd, S. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

Blume-Kohout, R., da Silva, M. P., Nielsen, E., Proctor, T., Rudinger, K., Sarovar, M., and Young, K. A taxonomy of small markovian errors. *PRX Quantum*, 3(2):020335, 2022.

Bravyi, S., Dial, O., Gambetta, J. M., Gil, D., and Nazario, Z. The future of quantum computing with superconducting qubits. *Journal of Applied Physics*, 132(16), October 2022. ISSN 1089-7550. doi: 10.1063/5.0082975.

Brooks, M. Beyond quantum supremacy: the hunt for useful quantum computers. *Nature*, 574(7776):19–21, 2019.

Cai, Z., Babbush, R., Benjamin, S. C., Endo, S., Huggins, W. J., Li, Y., McClean, J. R., and O'Brien, T. E. Quantum error mitigation, 2023.

Czarnik, P., Arrasmith, A., Coles, P. J., and Cincio, L. Error mitigation with clifford quantum-circuit data. *Quantum*, 5:592, 2021.

Daley, A. J., Bloch, I., Kokail, C., Flannigan, S., Pearson, N., Troyer, M., and Zoller, P. Practical quantum advantage in quantum simulation. *Nature*, 607:667 – 676, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Farhi, E., Goldstone, J., and Gutmann, S. A quantum approximate optimization algorithm, 2014.

Gisin, N., Ribordy, G., Tittel, W., and Zbinden, H. Quantum cryptography. *Rev. Mod. Phys.*, 74:145–195, Mar 2002. doi: 10.1103/RevModPhys.74.145.

Guerreschi, G. G. and Matsuura, A. Y. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific Reports*, 9(1), May 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-43176-9.

Harrigan, M. P. and Sung, K. J. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nature Physics*, 17 (3):332–336, February 2021. ISSN 1745-2481. doi: 10.1038/s41567-020-01105-y.

He, Z., Zhang, X., Chen, C., Huang, Z., Zhou, Y., and Situ, H. A gnn-based predictor for quantum architecture search. *Quantum Information Processing*, 22(2):128, Feb 2023. ISSN 1573-1332. doi: 10.1007/s11128-023-03881-x.

Huang, H.-L., Xu, X.-Y., Guo, C., Tian, G., Wei, S.-J., Sun, X., Bao, W.-S., and Long, G.-L. Near-term quantum computing techniques: Variational quantum algorithms, error mitigation, circuit compilation, benchmarking and classical simulation. *Science China Physics, Mechanics & Astronomy*, 66(5):250302, 2023.

Huggins, W. J., McArdle, S., O'Brien, T. E., Lee, J., Rubin, N. C., Boixo, S., Whaley, K. B., Babbush, R., and McClean, J. R. Virtual distillation for quantum error mitigation. *Physical Review X*, 11(4), November 2021. ISSN 2160-3308. doi: 10.1103/physrevx.11.041036.

Kandala, A., Temme, K., Córcoles, A. D., Mezzacapo, A., Chow, J. M., and Gambetta, J. M. Error mitigation extends the computational reach of a noisy quantum processor. *Nature*, 567(7749):491–495, Mar 2019. doi: 10.1038/s41586-019-1040-7.

Kim, C., Park, K. D., and Rhee, J.-K. Quantum error mitigation with artificial neural network. *IEEE Access*, 8: 188853–188860, 2020a. doi: 10.1109/ACCESS.2020.3031607.

Kim, C., Park, K. D., and Rhee, J.-K. Quantum error mitigation with artificial neural network. *IEEE Access*, 8: 188853–188860, 2020b.

Li, Y. and Benjamin, S. C. Efficient variational quantum simulator incorporating active error minimization. *Physical Review X*, 7(2), June 2017. ISSN 2160-3308. doi: 10.1103/physrevx.7.021050.

Liao, H., Wang, D. S., Sitdikov, I., Salcedo, C., Seif, A., and Minev, Z. K. Machine learning for practical quantum error mitigation. *Nature Machine Intelligence*, 6(12): 1478–1486, December 2024.

Liao, M., Zhu, Y., Chiribella, G., and Yang, Y. Noise-agnostic quantum error mitigation with data augmented neural models. *npj Quantum Information*, 11(1):8, January 2025.

Moflic, I., Garg, V., and Paler, A. Graph neural network autoencoders for efficient quantum circuit optimisation, 2023.

Nielsen, M. A. and Chuang, I. L. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P. J., Aspuru-Guzik, A., and O'Brien, J. L. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1), July 2014. ISSN 2041-1723. doi: 10.1038/ncomms5213.

Preskill, J. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, August 2018. ISSN 2521-327X. doi: 10.22331/q-2018-08-06-79.

Suzuki, S., Inoue, J.-i., and Chakrabarti, B. *Quantum Ising Phases and Transitions in Transverse Ising Models*, volume 862. Springer, 01 2013. ISBN 978-3-642-33038-4. doi: 10.1007/978-3-642-33039-1.

Temme, K., Bravyi, S., and Gambetta, J. M. Error mitigation for short-depth quantum circuits. *Physical Review Letters*, 119(18), November 2017. ISSN 1079-7114. doi: 10.1103/physrevlett.119.180509.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Zhang, H., Han, X., Zhang, G., Li, L., Cheng, L., Wang, J., Zhang, Y., Xia, Y., and Xia, C. Non-markovian noise mitigation in quantum teleportation: enhancing fidelity and entanglement. *Scientific Reports*, 14(1):23885, 2024.

Zhou, L., Wang, S.-T., Choi, S., Pichler, H., and Lukin, M. D. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Physical Review X*, 10(2), June 2020. ISSN 2160-3308. doi: 10.1103/physrevx.10.021067.

A. System and model configuration details

Our pipeline QEMFormer is implemented on a system running Ubuntu 20.04.4, with CUDA 12.2, PyTorch 1.13.0, and PyTorch Geometric 2.3.1. Most experiments are conducted on a server with 8 NVIDIA GeForce RTX 4090 GPUs with 24 GB CUDA memory, two AMD Ryzen Threadripper 3970X 32-Core Processors at 3.70 GHz and 128 GB RAM.

Additionally, detailed model hyper-parameters are presented in Tab. 7.

B. Circuit Structure Visualization

Visualization of examples of different types of quantum circuits is shown in Fig. 5.

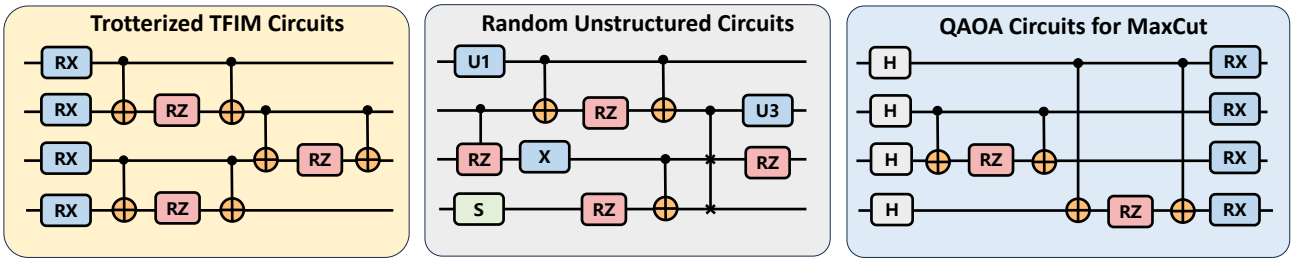

Figure 5. Examples of three types of quantum circuits.

C. Quantum Errors

Quantum error sources pose significant challenges to the advancement of quantum computing. These errors can be categorized as coherent or incoherent and as systematic or random, arising from factors such as imperfect qubit control, unwanted interactions, and measurement inaccuracies. In this work, we conduct quantum error mitigation (QEM) experiments targeting incoherent errors, coherent errors, and readout errors.

C.1. Incoherent Errors

Incoherent errors result from stochastic interactions between qubits and their environment, leading to non-unitary dynamics that cause decoherence and the loss of quantum information. These errors are typically modeled as random Pauli operators acting on the physical qubits under the assumption of a memoryless environment. We consider two primary types of incoherent errors: bit-flip errors and depolarizing errors.

Bit-Flip Errors. A bit-flip channel induces transitions between the computational basis states $|0\rangle$ and $|1\rangle$ with a probability of $1-p$. This channel is described by the Kraus operators:

$$E_0 = \sqrt{p}\, I = \sqrt{p} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{14}$$

$$E_1 = \sqrt{1-p}\, X = \sqrt{1-p} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{15}$$

where X is the Pauli-X operator. The action of the bit-flip channel on a quantum state ρ is given by:

$$\Phi_{\text{BF}}[\rho] = p\,\rho + (1-p)\, X\rho X. \tag{16}$$

Depolarizing Errors. The depolarizing channel represents a scenario where a single qubit is replaced by the maximally mixed state $\frac{I}{2}$ with probability p, while remaining unchanged with probability $1-p$. Mathematically, the depolarizing channel Φ_{DE} acts on a state ρ as:

$$\Phi_{\text{DE}}[\rho] = \frac{p}{2} I + (1-p)\,\rho. \tag{17}$$

Table 5. Mean Absolute Error (MAE) across different circuit types and noise models for the standard setting. Lower MAE values indicate better performance. Multiple implementations of ZNE and CDR consistently performed weakly on QAOA circuits and are therefore excluded from the mitigation results to ensure a fair comparison.

Method	Random			QAOA			Trotter		
	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider
Noisy	0.140	0.096	0.092	0.154	0.147	0.128	0.241	0.401	0.354
QEMFormer	0.067	0.071	0.036	0.039	0.047	0.026	0.019	0.102	0.047
MLP	0.073	0.103	0.042	0.041	0.053	0.037	0.023	0.145	0.061
GNN	0.082	0.072	0.054	0.047	0.054	0.040	0.023	0.110	0.059
OLS	0.172	0.118	0.089	0.080	0.096	0.052	0.038	0.126	0.061
ZNE	0.130	0.103	0.076	-	-	-	0.245	0.379	0.184
CDR	0.047	0.079	0.042	-	-	-	0.066	0.129	0.255
RF	0.086	0.085	0.065	-	-	-	0.048	0.108	0.069
GTranQEM	0.074	0.076	0.048	0.040	0.054	0.034	0.024	0.122	0.038

Table 6. MAE across different circuit types and noise models

Method	Random Size Zero Shot			Trotter Step Zero Shot			Unseen Obs		
	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider	Incoherent	Coherent	Provider
Noisy	0.135	0.095	0.095	0.296	0.476	0.507	0.072	0.067	0.074
QEMFormer	0.077	0.088	0.048	0.017	0.090	0.058	0.060	0.050	0.062
MLP	0.091	0.093	0.074	0.039	0.107	0.101	0.085	0.097	0.069
GNN	0.151	0.123	0.051	0.018	0.100	0.107	0.065	0.075	0.068
OLS	0.210	0.132	0.123	0.019	0.162	0.081	0.098	0.066	0.087
ZNE	0.132	0.112	0.078	0.296	0.451	0.254	0.078	0.064	0.072
CDR	0.087	0.088	0.057	0.022	0.184	0.083	0.072	0.061	0.073
RF	0.084	0.090	0.055	0.046	0.112	0.222	0.078	0.064	0.069
GTranQEM	0.080	0.095	0.050	0.029	0.101	0.135	0.070	0.052	0.076

Recognizing that for any ρ,

$$\frac{I}{2} = \frac{\rho + X\rho X + Y\rho Y + Z\rho Z}{4},$$

we can express the depolarizing channel as:

$$\Phi_{\text{DE}}[\rho] = \left(1 - \frac{3p}{4}\right)\rho + \frac{p}{4}\left(X\rho X + Y\rho Y + Z\rho Z\right). \tag{18}$$

This formulation reveals that the depolarizing channel comprises the operators $\left\{\sqrt{1-\frac{3p}{4}}\, I,\ \frac{\sqrt{p}}{2}\, X,\ \frac{\sqrt{p}}{2}\, Y,\ \frac{\sqrt{p}}{2}\, Z\right\}$.

C.2. Coherent Errors

Coherent errors stem from unintended or imperfect unitary operations within quantum circuits. These errors can be modeled by unitary operators of the form:

$$U(\theta) = e^{-\frac{i}{2}\theta\sigma}, \tag{19}$$

where $\theta = (\theta_1, \ldots, \theta_{4^{N_q}})$ quantifies the magnitude of the coherent error across the 4^{N_q} Pauli basis operators. Coherent errors transform pure quantum states into other pure states while maintaining quantum coherence due to their unitary nature. Despite preserving the purity of states, coherent errors can significantly undermine the reliability and accuracy of multi-qubit quantum computations.

D. Evaluation Metrics

In this section, we provide a comprehensive overview of the evaluation metrics employed in this manuscript. These metrics are essential for interpreting and assessing the performance of the proposed error mitigation techniques.

D.1. Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) is a widely used metric that measures the average magnitude of the prediction errors. It provides insight into the model's accuracy by penalizing larger discrepancies more heavily than smaller ones. Mathematically, RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

where:

- N is the total number of quantum circuits evaluated,
- $\hat{y}_i$ represents the predicted value for the i-th circuit,
- y_i denotes the actual ground truth value for the i-th circuit.

RMSE provides a single scalar value that summarizes the model's predictive performance, making it easier to compare different error mitigation strategies.

D.2. Absolute Error (AE)

Absolute Error (AE) quantifies the absolute difference between the predicted and actual values for each data point. Unlike RMSE, AE does not square the errors, thus treating all errors uniformly regardless of their direction or magnitude. It is defined as:

$$\text{AE}_i = |\hat{y}_i - y_i|$$

where:

- $\hat{y}_i$ is the predicted value for the i-th circuit,
- y_i is the actual ground truth value for the i-th circuit.

AE provides a straightforward measure of prediction accuracy for each quantum circuit, highlighting the exact deviation without emphasizing larger errors disproportionately.

D.3. Mean Absolute Error (MAE)

Mean Absolute Error (MAE) aggregates the absolute errors across all data points to provide an overall measure of prediction accuracy. It offers an interpretable average of the absolute discrepancies between predicted and actual values. MAE is mathematically expressed as:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\text{AE}_i = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$$

where:

- N is the total number of quantum circuits evaluated,
- AE_i is the absolute error for the i-th circuit.

MAE provides an easily interpretable metric that reflects the average prediction error across all evaluated circuits, facilitating the comparison of different error mitigation techniques in terms of their overall accuracy.

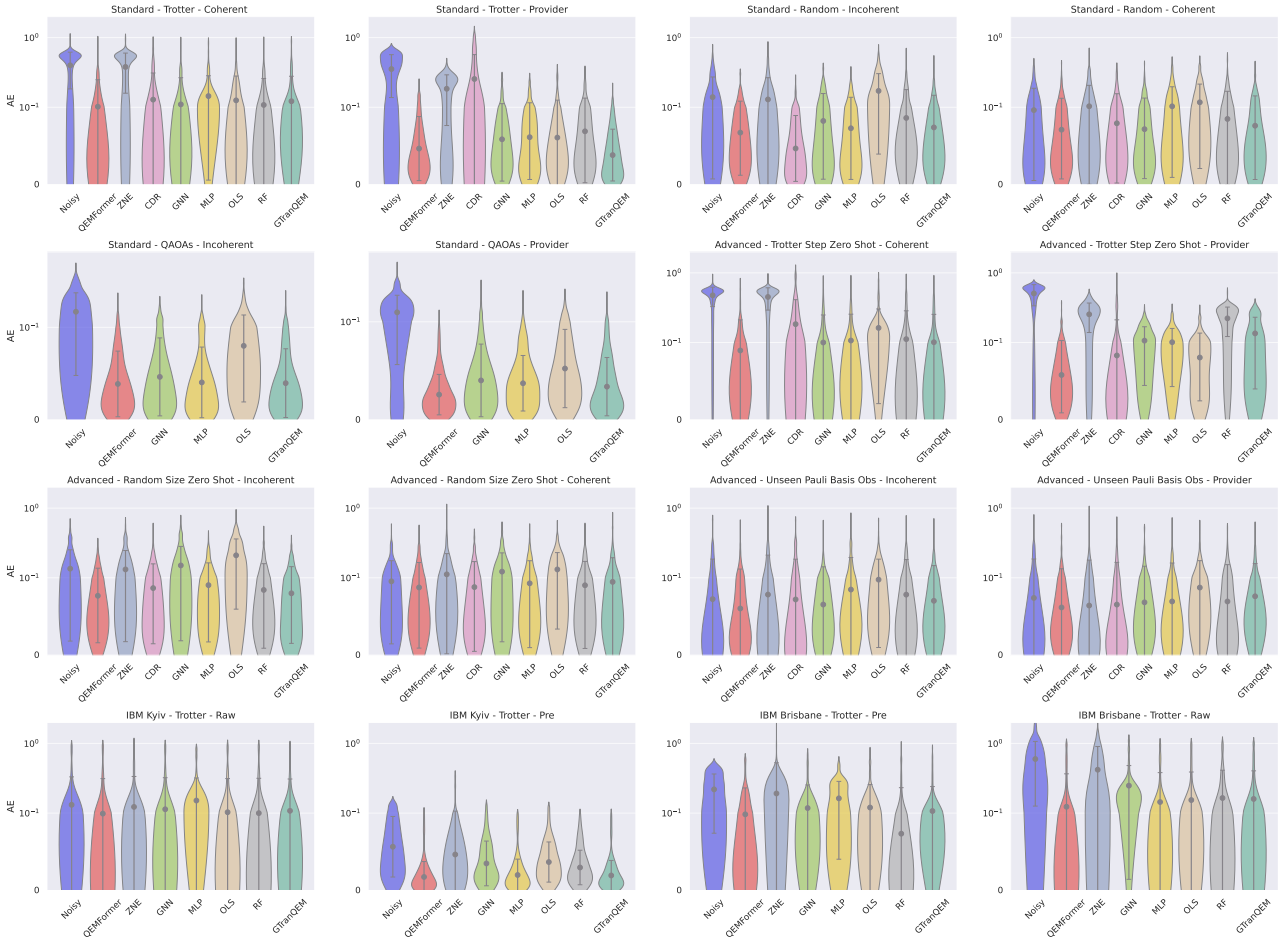

Figure 6. Violin plot depicting the distribution of AE. The setting names, circuit types, and noise configurations are annotated at the top of each figure.

D.4. Discussion of Metric Selection

The chosen metrics—RMSE, AE, and MAE—collectively offer a robust framework for evaluating the performance of error mitigation methods. RMSE emphasizes larger errors, making it suitable for identifying models that may have occasional significant deviations. AE provides a granular view of individual prediction errors, while MAE offers a balanced average that is less sensitive to outliers compared to RMSE. Together, these metrics ensure a comprehensive assessment of the error mitigators' effectiveness across various scenarios.

E. Additional Experimental Results

We present additional supplementary results alongside those in Section 5.3. Specifically, we summarize the Mean Absolute Error (MAE) for both the standard and advanced settings in Tabs 5 and 6, respectively. QEMFormer consistently outperforms other baselines, achieving the best or second-best results across both datasets. Furthermore, we include a violin plot illustrating the distribution of AE in Fig. 6, complementing the violin plots presented in Fig. 3 of the main text. Together, the tables and violin plots demonstrate the efficacy of QEMFormer compared to other baselines, highlighting its strong generalization capabilities and potential for handling large-scale circuits executed on real quantum devices.

Table 7. Model architecture configurations for different experimental settings.

Setting	MLP Layers	Graph Trans Conv Layers	Hidden Dim
Real Pre	3	1	128
Real Raw	4	2	128
QAOA-Coherent	3	1	128
QAOA-Incoherent	3	1	128
QAOA-Provider	4	1	128
Random-Coherent	3	1	128
Random-Provider	3	1	128
Random-Incoherent	3	1	256
Random-Size-Zero-Shot-Coherent	4	1	512
Random-Size-Zero-Shot-Incoherent	4	2	512
Random-Size-Zero-Shot-Provider	3	1	512
Trotter-Coherent	4	2	128
Trotter-Incoherent	4	1	64
Trotter-Provider	3	2	64
Trotter-Step-Zero-Shot-Coherent	3	1	128
Trotter-Step-Zero-Shot-Incoherent	4	1	64
Trotter-Step-Zero-Shot-Provider	4	2	128
Unseen-Obs-Incoherent	4	1	128
Unseen-Obs-Coherent	4	1	64
Unseen-Obs-Provider	4	1	128