A Machine Learning-Based Error Mitigation Approach for Reliable Software Development on IBM's Quantum Computers

Asmar Muqeet
Simula Research Laboratory and
University of Oslo
Oslo, Norway
asmar@simula.no

Shaukat Ali
Simula Research Laboratory and
Oslo Metropolitan University
Oslo, Norway
shaukat@simula.no

Tao Yue
Simula Research Laboratory
Oslo, Norway
taoyue@gmail.com

Paolo Arcaini
National Institute of Informatics
Tokyo, Japan
arcaini@nii.ac.jp

ABSTRACT

Quantum computers have the potential to outperform classical computers for some complex computational problems. However, current quantum computers (e.g., from IBM and Google) have inherent noise that results in errors in the outputs of quantum software executing on the quantum computers, affecting the reliability of quantum software development. The industry is increasingly interested in machine learning (ML)-based error mitigation techniques, given their scalability and practicality. However, existing ML-based techniques have limitations, such as only targeting specific noise types or specific quantum circuits. This paper proposes a practical ML-based approach, called Q-LEAR, with a novel feature set, to mitigate noise errors in quantum software outputs. We evaluated Q-LEAR on eight quantum computers and their corresponding noisy simulators, all from IBM, and compared Q-LEAR with a state-of-the-art ML-based approach taken as baseline. Results show that, compared to the baseline, Q-LEAR achieved a 25% average improvement in error mitigation on both real quantum computers and simulators. We also discuss the implications and practicality of Q-LEAR, which, we believe, is valuable for practitioners.

CCS CONCEPTS

• **Software and its engineering**; • **Computing methodologies** → **Feature selection**; • **Computer systems organization** → **Quantum computing**;

KEYWORDS

Software Engineering, Error Mitigation, Quantum Computing, Machine learning, Quantum noise

ACM Reference Format:
Asmar Muqeet, Shaukat Ali, Tao Yue, and Paolo Arcaini. 2024. A Machine Learning-Based Error Mitigation Approach for Reliable Software Development on IBM's Quantum Computers. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering (FSE Companion '24), July 15–19, 2024, Porto de Galinhas, Brazil.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3663529.3663830

1 INTRODUCTION

Quantum Computing (QC) holds immense promise for tackling complex computational problems beyond the capabilities of classical computers [29]. However, the practical realization of this potential faces challenges, with quantum noise being a prominent obstacle. *Quantum noise*, stemming from imperfections and environmental interactions, significantly impacts the accuracy of computations performed by quantum computers [43]. Consequently, the accuracy of software[1] running on a noisy quantum computer is compromised, even when correctly implemented, thereby limiting QC's practical applications and advantage over classical computers.

Developing quantum software while addressing noise poses various challenges, including uncertainty about whether the quantum software is producing an incorrect output or if the output is flawed due to noise in the quantum computer. Recognizing the noise issue, industry leaders in QC, such as IBM, have identified *quantum error correction* (i.e., error correction during circuit executions) and *quantum error mitigation* (i.e., error correction post-circuit execution) as pivotal building blocks in their roadmap to facilitate the development of practical QC software [14, 27]. This paper focuses on quantum error mitigation, i.e., applying automated error mitigation techniques after software execution on a quantum computer to eliminate the noise effects from the software outputs. Such noise elimination serves as a valuable tool for quantum software engineers, which facilitates software development and testing with outputs that have undergone a noise-cleansing process. Doing so, thereby, increases software engineers' assurance of the correctness of quantum software under real-world quantum computing conditions–inherent noise in quantum computers.

In practice, several error mitigation techniques have been incorporated into industrial frameworks such as IBM's Qiskit [11]

FSE Companion '24, July 15–19, 2024, Porto de Galinhas, Brazil

ACM ISBN 979-8-4007-0658-5/24/07
https://doi.org/10.1145/3663529.3663830

[1] Quantum software is currently being built as quantum circuits, i.e., a sequence of quantum gate operations applied to quantum bits (qubits).

to correct output errors in quantum circuits. Notable methods include Probabilistic Error Cancellation (PEC) [51] and Zero-Noise Extrapolation (ZNE) [33]. While these techniques show promise in mitigating output errors, they often require a comprehensive understanding of specific noise characteristics for each circuit. For instance, to use PEC, it is needed to identify the predominant type of noise error and create mathematical models for each type of noise that can impact a given circuit. However, this process incurs an exponential cost in terms of circuit sampling (i.e., the number of repeated executions of a circuit required to build a noise model), rendering it impractical in terms of scalability for current quantum computers [18]. On the other hand, ZNE is accurate only for specific circuits where noise lacks temporal correlation, a condition not met by the majority of current quantum algorithms [46].

In recent years, there has been a shift among industry practitioners towards machine learning (ML)-based error mitigation for practical, reliable, and scalable solutions [34], with state-of-art being QRAFT [39] which leverages an ensemble-based ML algorithm for quantum error mitigation in the presence of noise. However, a critical limitation of the current methods, including QRAFT, is the absence of a reliable feature set that can accurately quantify the noise magnitude of a quantum circuit. Consequently, the ML models of these methods could exacerbate errors (instead of removing them) by making inaccurate adjustments based on wrong noise estimates, as evidenced by the results of our empirical study.

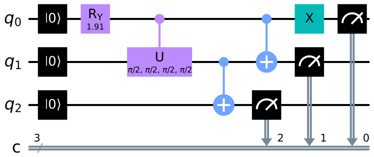
This paper presents an ML-based error mitigation approach (Q-LEAR) to address the limitations of current ML-based error mitigation methods. Q-LEAR proposes a set of novel features, including the Depth-cut Program Error (*Dpe*), which cuts a quantum circuit at specific circuit depths and leverages quantum operations' reversibility feature to quantify noise magnitude. With *Dpe*, we estimate noise magnitude more accurately when compared with the state of the art. Q-LEAR enables ML models to effectively learn and mitigate quantum circuit output error caused by noise. We empirically evaluate the effectiveness of Q-LEAR with various ML models on real quantum circuits and across eight IBM's quantum computers and their corresponding noisy simulators. Results show that ML models trained with Q-LEAR perform significantly better than QRAFT on IBM's quantum computers and simulators. Notably, Q-LEAR demonstrates an average improvement of 25% in error mitigation compared to QRAFT when executed on eight IBM's quantum computers and simulators. The results emphasize that Q-LEAR's ML model trained with its feature set has the potential to substantially improve the reliability of quantum software development, especially on IBM's quantum computers. Our work is industry-relevant because we used real data from IBM's quantum computers, and employed real quantum computers and simulators. Additionally, we discuss the implications and practicality of Q-LEAR to provide valuable insights for practitioners.

2 BACKGROUND

Qubit. Quantum Computing (QC) uses *quantum bits*, i.e., *qubits*, as its fundamental information units. A qubit can exist in a superposition of states $|0\rangle$ and $|1\rangle$, with associated amplitudes for each, and the state of one qubit immediately influences the state of another one when they are entangled. The amplitude is a complex number comprising both magnitude and phase in its polar representation. In the Dirac notation [17], a qubit is denoted as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β represent the amplitudes associated with states $|0\rangle$ and $|1\rangle$, respectively. The superposition of a qubit upon measurement collapses to a single basis state where the probability of observing a qubit in either state $|0\rangle$ or $|1\rangle$ is determined by the squared magnitude of α and β, with the sum of all squared magnitudes being 1: $|\alpha|^2 + |\beta|^2 = 1$.

```
 1. #initialize the empty circuit
 2. qc = QuantumCircuit()
 3. # create 3 qubits
 4. q0 = QuantumRegister(1,'q_0')
 5. q1 = QuantumRegister(1,'q_1')
 6. q2 = QuantumRegister(1,'q_2')
 7. # create 3 classic registers
 8. c0 = ClassicalRegister(1, 'c_0')
 9. c1 = ClassicalRegister(1, 'c_1')
10. c2 = ClassicalRegister(1, 'c_2')
11. # Apply a Ry gate on qubit_1
12. qc.ry(1.91,q0)
13. # Entangle qubit_1 with 2, 2 with 3
14. qc.cu(pi/2, pi/2, pi/2, q0,q1)
15. qc.cx(q1,q2)
16. qc.cx(q0,q1)
17. qc.x(q0)
18. # measure the qubits for readout
19. qc.measure(q0,c0)
20. qc.measure(q1,c1)
21. qc.measure(q2,c2)
```

(a) Python code in Qiskit



(b) Quantum Circuit

Figure 1: W-state quantum program

Quantum Gate and Circuit. Gate-based quantum computers manipulate a qubit with a quantum gate, which is a unitary operator that changes the qubit's state based on a unitary matrix [19]. For example, a *Hadamard* gate puts a single qubit in superposition. Currently, quantum programs are represented in the circuit model [5]; a quantum program is a sequence of quantum gates acting on a set of qubits. Each gate operation is a single-time step in the unitary evolution of a quantum system [5]. **Circuit depth** is an essential indicator of circuit complexity, representing the longest sequence of gate operations in a circuit. Fig. 1a shows the Python code for the program that creates a three-qubit entangled state known as W-state [28] and Fig. 1b shows its corresponding quantum circuit. In Fig. 1a, lines 1-6 create an empty quantum circuit with three qubits q_0, q_1, q_2 initialized to state $|0\rangle$. Then, lines 7-10 create three classical registers c_0, c_1, c_2 to hold the measurement results of the qubits. In line 12, the R_y gate operation is performed on the circuit to rotate the Y axis of q_0's Bloch sphere by 1.91 degrees. Next, in line 14 a conditional *Hadamard* gate is applied on q_1 to put it in superposition. In lines 15 and 16, two conditional *NOT* (*CNOT*) gates are then applied to entangle all three qubits. After applying another *NOT* gate (also called *X* gate) in line 17, we obtain a W state ($|W\rangle = \frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle)$). Lines 19-21 apply the measurement operation on all three qubits so that we can get as output one of the three states ($|001\rangle, |010\rangle, |100\rangle$) with equal probability on the classical registers c_0, c_1, and c_2.

Transpilation. Each quantum computer has its own native gate set and qubit connection topology, implying that each qubit can only interact with another qubit if a physical connection exists between them. Hence, some gate operations cannot be performed due to limited physical connections. *Transpilation* transforms a quantum logical circuit to a transpiled circuit only containing hardware-defined gate operations and additional swap operations for solving the limited physical connections of the hardware [5].

Quantum Noise. *Noise* arises from various sources. First, environmental factors, e.g., magnetic fields and radiation can impact computations [10, 43]. When qubits interact with their surroundings, these interactions can cause disturbances and information loss in quantum states, called *decoherence* [3]. Second, even when qubits are isolated from the environment, unwanted interactions can occur among them, resulting in *crosstalk noise* [7, 41, 45]. Third, *imprecise calibrations* of quantum gates, which are necessary to optimize gate parameters and reduce errors while improving fidelity, can introduce noise [9]. Small calibration errors may cause minor changes in qubit phases, amplitudes, etc., resulting in undesired states after a series of gate operations [6].

Noise in computing systems is not a unique concept to quantum computing; it also exists in the classical world, notably in domains like the Internet of Things (IoT) and cyber-physical systems [48, 55]. This raises the question of whether classical noise filtering or error correction techniques can be directly applied to quantum computing (QC). While some principles from classical methods, such as error correction codes derived from information theory, can find applications in QC [12, 30], it is crucial to acknowledge that quantum noise possesses unique characteristics. Quantum noise exhibits phenomena such as entanglement, superposition, and quantum interference, which differentiate it from classical noise. These quantum characteristics make quantum noise significantly more complex. In contrast, classical noise can be described using classical probability theory and arises from random fluctuations, electronic interference, thermal effects, etc. [50]. Classical noise sources often exhibit behaviors where noise events are independent and adhere to probability distributions like Gaussian or Poisson [50]. However, when it comes to quantum noise, understanding the underlying distribution becomes extremely challenging. This difficulty arises primarily due to the restrictions in quantum computing inherent to quantum mechanics, such as the no-cloning theorem and state collapse [35]. These quantum principles limit the ability to clone quantum states and introduce uncertainties in the measurement process, making it difficult to model quantum noise with the same level of predictability as classical noise.

3 RELATED WORK AND THEIR LIMITATIONS

Existing quantum noise error mitigation methods can be classified into three categories.

Probabilistic Error Cancellation (PEC). Since being introduced in [51] for Markovian noise (errors at a time point independent of what occurred in the past), PEC has been extended to handle non-Markovian noise by [26]. It uses the quasi-probability decomposition of the inverse noise process, resulting in a linear combination of noisy circuits. Various PEC methods have recently been proposed, including [21, 52, 54], for various noise errors. *Limitation:* However, PEC-based methods require complete knowledge about noise characteristics specific to a circuit, including identifying the dominant noise error type and establishing mathematical models for each noise channel in the circuit. Hence, applying PEC to different quantum circuits executing on different quantum computers becomes extremely challenging.

Zero-Noise Extrapolation (ZNE). ZNE gathers execution data of quantum circuits at different error rates and extrapolates to the zero noise limit [33]. Various studies have extended ZNE with different extrapolation methods [8, 23, 33, 51]. *Limitation:* However, ZNE-based methods assume that noise is uncorrelated with time, which has been invalidated by recent studies [46]. In the presence of time-correlated noise, scaling quantum circuits for different error rates without altering their spectral distribution becomes difficult [46].

Learning-based methods. Recent research focuses on using ML for noise error mitigation to deal with some limitations of methods such as PEC and ZNE. Examples include Clifford data regression (CDR) [13], Learning-based PEC (L-PEC) [47], ML-QEM [34], and QRAFT [39]. CDR employs near-Clifford quantum circuits as training data to develop a regression model that mitigates noise; however, it works only for quantum circuits with Clifford gates. L-PEC does not rely on prior knowledge of noise channels as other PEC methods do. It generates multiple variants of the target quantum circuit by replacing non-Clifford gates with gates that are easier to simulate classically. The execution of these variants is then used as training data for a probabilistic noise mitigation model. However, L-PEC assumes that single-qubit Clifford gates are noise-free, which is not always true for real-world quantum computers [36]. ML-QEM reduces noise in quantum circuits by leveraging variations in circuit properties like native gate counts and angle values. It utilizes the noise parameters of a specific quantum computer and native gate counts of the target circuit to mitigate noise errors. However, a significant limitation of ML-QEM is that its effectiveness is limited to accurately remove noise only from circuits similar to those in the training data.

QRAFT leverages the reversibility of quantum circuits as a pseudo oracle for training a noise mitigation regression model. ***Limitation:*** Although QRAFT avoids assumptions about noise and quantum gate types, it doubles the circuit depth for feature calculation, so increasing the decoherence likelihood and making error mitigation difficult [38]. To this end, we introduce a novel Depth-cut Program Error (*Dpe*) feature in Q-LEAR that uses the same pseudo oracle as QRAFT, but divides a quantum circuit into smaller subcircuits to avoid doubling the original circuit depth (see section 5.2). Further, QRAFT is more prone to cross-talk noise in feature calculations, as it uses all qubits in the reversed quantum circuit, leading to additional cross-talk noise compared to the original circuit. In contrast, we only measure a subset of qubits for our *Dpe* feature that are measured in the original circuit.

4 PRELIMINARIES

We describe the necessary terms to ease understanding of the rest of the paper.

Output State. It is one of the possible states observed after the quantum circuit execution. For instance, in the 3-qubit W-state quantum circuit (Fig. 1), there are 2^3 (i.e., 8) possible output states

Table 1: Ideal and noisy outputs of the W-state circuit (Fig. 1) after 1024 executions on IBM's Quito (5-qubit quantum computer). Column *Output States with Probabilities* shows Output States with associated probabilities.

	Output States with Probabilities							
Circuit Output	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
Ideal	0	0.33	0.33	0	0.33	0	0	0
Noisy	0.002	0.034	0.29	0.007	0.29	0.011	0.009	0.004

(see Table 1). Each output state has a probability determined by the circuit's logic and operations.

Circuit Output. It refers to all output states and their respective probabilities observed after circuit execution. For example, the ideal output of W-state (row 1 of Fig. 1) comprises 8 possible output states and their probabilities.

Output Error. Quantum noise manifests as errors in the quantum circuit's output. For example, the *Noisy* row of Table 1 shows the noisy outputs of the W-state circuit (see Fig. 1), where we observe wrong output states and probabilities compared to *Ideal.* In literature, to quantify the amount of noise in the circuit output, Hellinger distance (HLD) has been used to calculate the similarity between ideal and observed circuit outputs [15, 16, 24]:

$$h(P,Q) = \frac{1}{\sqrt{2}}||\sqrt{P} - \sqrt{Q}||_2 \tag{1}$$

where P and Q are the true and observed noisy probability distributions of a circuit's outputs. Hellinger distance (HLD) is between 0 and 1, where 0 means that two outputs are identical and 1 is the opposite. For example, the HLD between the *Ideal* and *Noisy* outputs in Table 1 is 0.2, i.e., a 20% difference in the output.

Output State Error.

$$err_s = |P_s - Q_s| \tag{2}$$

where P_s is the ideal probability, and Q_s is the observed noisy probability of a state. For instance, the output state error for $|001\rangle$ (Table 1) is $|0.33 - 0.034| = 0.296$.

5 APPROACH

We propose a **L**earning-based **E**rror mitigation **A**pproach with a **R**obust feature set (Quantum-LEAR or Q-LEAR for short), to reduce the effect of noise from program output. Q-LEAR has circuit- and output-level features. Fig. 2 shows the process (having three steps) for calculating the Q-LEAR features for a given quantum circuit and a target quantum computer. First, we transpile a quantum circuit (see Sect. 2). Second, we divide the transpiled circuit into subcircuits for *Dpe* feature calculation (detailed in Sect. 5.2). Third, we execute the transpiled circuit and all subcircuits on the quantum computer or simulator and calculate Q-LEAR's feature set for each output state derived from the circuit and its output.

5.1 Circuit-level Features

Circuit-level features are: 1) the total number of qubits, also called **circuit width** (*Cw*), 2) the **circuit depth** (*Cd*), 3) the number of **single gate operations** (Gc_{1q}), and 4) the number of **two-qubit gate operations** (Gc_{2q}). We chose *Cw* since its higher value leads to a higher probability of cross-talk noise, hence inducing errors in the circuit output [45]. *Cd* also directly relates to the effect of noise on the circuit output [38] as decoherence increases with the increased depth. Quantum gates also play a crucial role in characterizing noise effect since different numbers of single and two-qubit gates are impacted by noise in distinct ways [22]. Note that after transpilation, the circuit-level features, e.g., depth, can significantly differ from the logical circuit. Thus, we calculated features on the transpiled circuits.

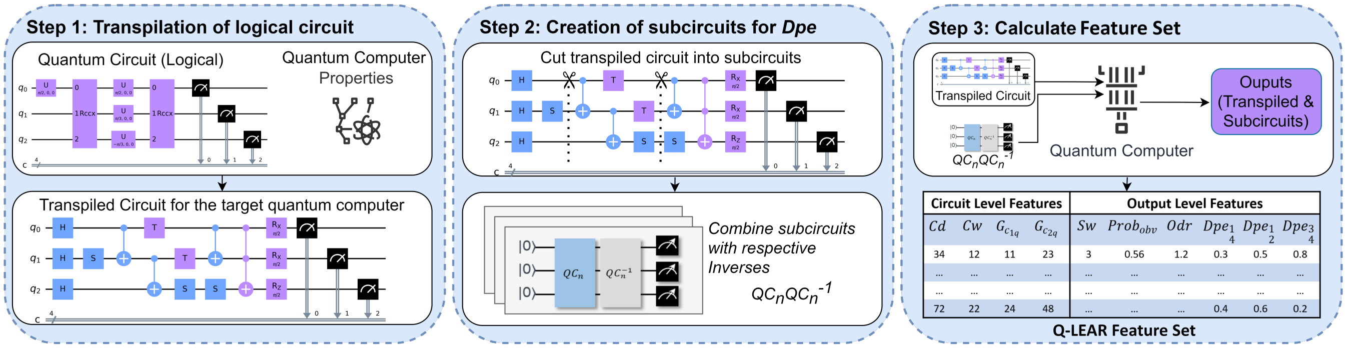
In our feature set related to gate operations in a quantum circuit, we specifically focus on generic features that can be calculated for gate-based quantum computers. This approach involves excluding non-generic gate features like counts of U1, U2, and U3 gates, which were used in QRAFT. In transpiled circuits, U1, U2, and U3 gates are substituted with varying numbers of other single and two-qubit gates. The choice of replacement gates depends on the logic of the quantum algorithm and the gate set supported by the target hardware (i.e., the one and two-qubit gates physically available in the quantum hardware). While different quantum computers may have distinct gate sets, the commonality is that presently only one and two-qubit gates are physically supported. To ensure compatibility across various quantum computers and quantum circuits, we have chosen to categorize gates based on the number of qubits they act on in the transpiled circuit. However, in the current evaluation of Q-LEAR, we focus on quantum circuits transpiled using IBM's supported gate set, which includes CX as two qubits and ID, RZ, SX, and X as one qubit quantum gates. Although theoretically, Q-LEAR could accommodate any quantum circuit, practical limitations arise as Q-LEAR is currently confined to circuits utilizing gate sets supported by IBM's existing quantum hardware.

5.2 Output-level Features

From a quantum circuit output, we use the following output-level features. **Observed Probability** ($Prob_{obv}$) is the probability associated with an output state observed after execution on the quantum computer or simulator. **Odds Ratio** (*Odr*) quantifies the strength of association between each output state of two consecutive quantum circuit executions, defined as $Odr = \frac{odds_r}{odds_{r+1}}$, where $odds_r$ and $odds_{r+1}$ are the odds of a specific output state in two consecutive executions [25]. An event's odd is calculated as $odds_r = \frac{P_s}{1-P_s}$, where P_s is the probability of the output state s in one circuit execution. Compared to probabilities of output states, the odds ratio offers a slight advantage under noise due to its scale-invariance property, i.e., remaining unchanged even when all probabilities are multiplied by a constant noise factor [25]. This property makes the odds ratio less susceptible to constant noise factors [25].

State Weight (*Stw*) is the number of qubits in state $|1\rangle$. It was first studied by [39] as a feature for mitigating noise error; the study shows that states with lower weights have higher noise errors as higher-weight states (with more qubits in $|1\rangle$) have a higher chance of relaxing to states with lower weights due to noise. Relaxed qubits end up accumulating errors in lower-weight states.

Depth-cut Program Error (*Dpe***)** measures the noise magnitude affecting the circuit's output. QRAFT [39] uses the inverse of a quantum circuit to quantify noise impact on the circuit output.


Step 1: Transpilation of logical circuit
Quantum Circuit (Logical)
Quantum Computer Properties
Transpiled Circuit for the target quantum computer
Step 2: Creation of subcircuits for Dpe
Cut transpiled circuit into subcircuits
Combine subcircuits with respective Inverses
$QC_nQC_n^{-1}$
Step 3: Calculate Feature Set
Transpiled Circuit
$QC_nQC_n^{-1}$
Quantum Computer
Ouputs (Transpiled & Subcircuits)
Q-LEAR Feature Set

Circuit Level Features				Output Level Features					
Cd	Cw	$G_{c_{1q}}$	$G_{c_{2q}}$	Sw	$Prob_{obv}$	Odr	$Dpe_{\frac{1}{4}}$	$Dpe_{\frac{1}{2}}$	$Dpe_{\frac{3}{4}}$
34	12	11	23	3	0.56	1.2	0.3	0.5	0.8
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
72	22	24	48	...	...	...	0.4	0.6	0.2

Figure 2: Overview of the process of calculating Q-LEAR's feature set for a given quantum circuit and quantum computer.

However, this doubles the circuit depth, increasing decoherence noise and weakening correlation between noise effects on inverted and original circuits. To overcome this limitation, we introduce the *Dpe* method. It is inspired from QRAFT [39] regarding the pseudo oracle and CutQC [49] to cut the original circuit into smaller subcircuits based on the circuit depth with a specific interval, then calculating the output error with Eq. 1 for each subcircuit. With the increased circuit depth, the effect of noise also increases [38]. Thus, employing multiple depth cuts and computing the output error allows quantifying the noise effect up to a certain depth and provides insights into the temporal evolution of noise's impact on the circuit output.

However, cutting a quantum circuit at various depths has two limitations. First, such cuts may violate the intended logic of a quantum program [49], leading to undesirable outcomes. Second, accurate calculation of the circuit error requires knowledge of the ideal output of the circuit at different depths, which is impractical on real quantum computers. To address these limitations, we leverage the reversible property of a quantum circuit. When a quantum operation transforms a quantum state $|\Psi\rangle$ into a new state $|\Psi'\rangle$ = $U|\Psi\rangle$, the conjugate transpose of U, denoted as $U^\dagger$, acts as the reverse operation that undoes the effect of U [1]. Consequently, when $U^\dagger$ is applied to $|\Psi'\rangle$, the original state $|\Psi\rangle$ is restored:

$$U^\dagger|\Psi'\rangle = U^\dagger(U|\Psi\rangle) = (U^\dagger U)|\Psi\rangle = I|\Psi\rangle = |\Psi\rangle \tag{3}$$

Here, I is the identity matrix. With the reversible property, we cut a quantum circuit at a smaller depth and append the inverse of the subcircuit to ensure the quantum logic remains valid. Using the inverse subcircuit, we also get to know the ideal outcome of the quantum circuit, which should always be $|0^{\times n}\rangle$ (i.e., all measured qubits should be $|0\rangle$) with a probability of 1 and all other states have a probability of 0.

To calculate *Dpe* for any quantum circuit, we define depth cut points at the beginning to 1/4th of circuit depth, 1/4th to 1/2th of circuit depth, and 1/2th to 3/4th of circuit depth. We omit the concluding section from 3/4th to the end due to its primary composition of measurement gate operations. Additionally, the inverse of the measurement gate results in no operation on any qubit, which contributes no practical value. We use these ranges to ensure that the total depth of each depth-cut subcircuit, when joined with its inverse, remains less than the total depth of the original full circuit. This condition is essential to prevent correlation weakening due to additional noise from decoherence. Moreover, using these regular interval ranges guarantees that we always calculate a specific number of *Dpe* features (in this case, three, i.e., $Dpe_{1/4}$, $Dpe_{1/2}$, $Dpe_{3/4}$ for any circuit), making *Dpe* comparable among different circuits. For example, let's consider a quantum circuit Q with five qubits and a circuit depth of 136. To calculate *Dpe* for this circuit, we divide Q into three subcircuits: Q_1, Q_2, and Q_3. Q_1 represents the subcircuit from depth 0 to 34 (1/4th of 136), Q_2 is the subcircuit from depth 35 to 68 (1/2th of 136), and Q_3 corresponds to the subcircuit from depth 69 to 102 (3/4th of 136). The *Dpe* for subcircuits can be calculated using Hellinger distance from Eq. 1 as $Dpe_n = h(|0^{\times q}\rangle, |Q_nQ_n^\dagger\rangle)$, where q stands for the number of measured qubits in the original circuit, $|Q_nQ_n^\dagger\rangle$ represents the observed output of the subcircuit appended with its inverse, and n stands for the n^{th} subcircuit.

6 EVALUATION AND ANALYSIS

6.1 Research Question

We assess the effectiveness of Q-LEAR in training ML models to diminish noise errors from quantum circuit outputs by answering the following research questions (RQs):

RQ1 What is the relationship between Q-LEAR's feature set and error in circuit output due to noise?

RQ2 How effective is Q-LEAR in training ML models for error mitigation, compared with state-of-the-art QRAFT?

RQ3 Do all features play an important role in mitigating errors from circuit output?

6.2 Experiment Design

6.2.1 *Benchmarks.*

We employ the MQT benchmark [42], which has a diverse set of quantum circuits tailored for various quantum computers. The circuits in MQT are categorized into two groups: those designed for educational purposes (*learning-level*) and those addressing real-world problems (*application-level*). We selected circuits from both categories that can be executed on all of IBM's quantum computers. In total, we obtained 56 learning-level and six application-level circuits. For application-level circuits, an additional selection criterion was that they must solve real optimization problems. This led to the inclusion of the following circuits in our evaluation. (i) *Ground State* (*GS*): Finds ground state of hydrogen molecules; (ii) *Pricing Call* (*PC*): Estimates the fair price of a single European call option using iterative amplitude estimation; (iii) *Pricing Put* (*PP*): Estimates the fair price of a single European put option using iterative amplitude estimation; (iv) *Quantum Approximate Optimization Algorithm* (*QAOA*): Solves a Max-Cut problem instance; (v) *Vehicle Routing* (*RT*): Solves a vehicle routing problem instance; and (vi) *Traveling Salesman Problem* (*TSP*): Solves a Traveling Salesman Problem instance.

MQT benchmark provides the final optimized quantum circuits for all selected problems. For the circuit execution, we used eight industrial IBM quantum computers accessible publicly through the IBM Quantum Cloud platform: *Lagos*, *Nairobi*, *Perth*, *Belem*, *Jakarta*, *Lima*, *Manila*, and *Quito*.

6.2.2 Machine Learning Models. Error mitigation in a quantum circuit's output is a regression problem. Thus, we selected the most common ML models [44] for regression, i.e., Linear-Regression (LR), Lasso-Regression (Lasso), Ridge-Regression (Ridge), Elastic-Regression (Elastic), Support-Vector Regression (SVR), K-Nearest Neighbour Regression (KNNR), Ensemble-of-Trees-regression (EDT), Light Gradient Boosting Machine (LGBM), Extreme Gradient Boosting (Xgb), and Multilayer-Perceptron (MLP).

6.2.3 Training and Testing. To generate training data for ML models, we used IBM's quantum simulator in Qiskit Runtime [11], with noise models from IBM corresponding to the selected quantum computers. Due to the long waiting queues for getting access to IBM's quantum computers, generating training data on real quantum machines is infeasible. The noise models provided by IBM are constructed using calibration data, approximating the noise experienced in real quantum computers. These models enable classical simulators to produce results with simulated noise.

For supervised learning, we used an ideal simulator without a noise model to obtain the ground truth for each observed state. However, using an ideal simulator has its limitations, as classical simulators cannot fully simulate complex quantum algorithms. Thus, we selected learning-level circuits that can be simulated on classical computers to generate training data. We hypothesized that the selected features would be generalizable to more complex circuits on real quantum computers. To verify this hypothesis, we used application-level circuits to generate test data by executing them on real quantum computers instead of simulators.

For the training data, we executed the selected 56 learning-level circuits on noise models of all eight selected quantum computers, resulting in approximately 10k quantum states. We executed the six application-level circuits for the testing data on all eight real quantum computers, yielding 1060 quantum states. For hyperparameter tunning, we used Optuna [2], which uses Bayesian optimization. We opted for the Bayesian optimization to have a fair comparison with QRAFT since QRAFT is also trained with Bayesian optimization for hyperparameter tuning. For each trial in Bayesian optimization, the fitness of an ML model was calculated as an average of five-fold cross-validation. For all ML models, we used Mean Square Error as the loss function.

6.2.4 Metrics. For RQ1, we use Pearson correlation [31] to quantify the relationship between Q-LEAR's feature set and the errors in the quantum circuit output caused by noise. For circuit-level features, e.g., depth and width, previous studies (e.g., [38]) have already demonstrated a positive correlation with quantum noise. Thus, we do not study their correlation. Regarding output-level features, we use the metric *Output State Error* (Eq. 2) to assess the impact of noise on a specific output state of a quantum circuit, and the *Output Error* (Eq. 1) metric to evaluate the overall effect of noise on the circuit's output. We employed Pearson correlation to characterize the relationship between output-level features and the metrics *Output State Error* and *Output Error*. Specifically, for *Stw* and *Odr*, we computed the correlation with *Output State Error*, as these features are derived for each output state of a circuit. Conversely, for *Dpe* features, the correlation was determined with *Output Error*, given that these features are not calculated for a particular output state but rather for the entire circuit output.

For RQ2, we initially assess the quality of trained ML models using common regression metrics found in the literature [40]. These metrics include the Pearson correlation coefficient (*R*), coefficient of determination (R^2), root mean square error (*RMSE*), root mean square relative error (*RMSRE*), mean absolute percentage error (*MAPE*), and mean square error (*TestLoss*). For a comparative analysis with QRAFT, we first identify the best-performing ML model based on the aforementioned regression quality metrics. Subsequently, we compare Q-LEAR with QRAFT using *Output Error* (see Eq. 1) as a metric. *Output Error* represents the Hellinger distance (HLD) between two probability distributions. QRAFT's evaluation mainly used two metrics: (i) *State Error:* Measured as Mean Square Error (MSE), which in this paper is defined as *TestLoss* and used along with other regression metrics. (ii) *Program Error:* Initially, in QRAFT, program error was calculated using Total Variance Distance (TVD). However, we opted for HLD for several reasons. HLD is more suitable because it is widely used for comparison with noise [15, 16, 24]. Also, HLD considers both the difference in probability values and the overall shape of the distribution. This is crucial, as a quantum program may yield low probability outcomes, but as long as the distribution shape aligns with the expected ideal shape, the result is considered correct. In quantum circuits, specific probabilities obtained in an ideal setting may require more shots (i.e., the number of repeated circuit executions to obtain a probability distribution as an output) when subjected to noise. However, if the distribution shape matches the ideal shape under noise, additional shots are unnecessary. TVD does not account for such scenarios, making HLD a more appropriate choice.

For RQ3, we employed the Leave-one-covariate-out (LOCO) method to determine the feature importance for each feature in the proposed feature set. LOCO is a comprehensive method for feature importance assessment [32], involving the exclusion of one feature at a time, retraining the model, and evaluating its performance. In our case, LOCO was utilized to ascertain the impact of a specific feature on *Output Error*. To mitigate random bias, we conducted the LOCO process 10 times. For statistical analysis, we employed the Mann-Whitney U [20] statistical test and Vargha Delaney $\hat{A}_{12}$ [53] effect size measure. The statistical tests were conducted on 10 observations of *Output Error* for a given circuit-computer pair without a specific feature, comparing them with 10 observations when utilizing the full feature set. This setup allows us to assess the importance of each feature for each test circuit-computer pair, providing valuable insights into the impact of individual features.

Table 2: RQ1 - Results of Pearson correlation analyses for output-level features with *Output State Error* and *Output Error*. Columns *Stw* and *Odr* denote *State Weight* and *Odds Ratio*; $Dpe_{1/4}$, $Dpe_{1/2}$, and $Dpe_{3/4}$ denote *Dpe* at various cuts.

Circuits	*Stw*	*Odr*	$Dpe_{1/4}$	$Dpe_{1/2}$	$Dpe_{3/4}$
GS	0.003	-0.27	0.40	0.64	0.51
PC	-0.44	-0.16	0.49	0.94	0.95
PP	-0.43	-0.19	0.63	0.88	0.89
QAOA	-0.10	-0.19	0.58	0.57	0.52
RT	0.61	-0.22	0.94	0.91	0.92
TSP	-0.11	-0.20	0.70	0.59	0.62

6.3 Results and Analyses

6.3.1 RQ1 – Relation of Q-LEAR's feature set with circuit error.

Effective features must have some relationship with the target value (in our case the errors in quantum circuit output), allowing ML models to leverage this relationship for predicting the target value. Hence, RQ1 studies such relationship between errors in quantum circuit output and our proposed features. Regarding circuit-level features: circuit depth, width, and counts of single and two-qubit gates, their relationship with noise is well-explored in the literature (e.g., [38]), which has shown a positive correlation with quantum noise. Therefore, we do not study their correlation in this research question.

For output-level features, instead, we executed the selected application-level circuits on the selected quantum computers and simulators [11] and obtained both outputs affected by noise and ideal outputs. To measure the noise effect on a specific output state of a quantum circuit, as well as on overall circuit output, we used the metrics introduced in Section 6.2.4. Results are shown in Table 2. These correlations explain how the selected features can be used for error mitigation and how they impact the accuracy of quantum circuit outputs. Column *Stw* in Table 2 shows that for *PP* and *PC*, states with lower weights tend to exhibit higher *Output State Error* due to an overall negative correlation between *Stw* and *Output State Error*. However, their correlation varies across circuits. For instance, for *RT*, a positive correlation was observed. For *GS* and *QAOA*, the correlation is relatively weak. These findings are consistent with QRAFT [39], which also employs *Stw*. QRAFT also highlights that the circuits with a higher number of output states, having larger weights, tend to experience more errors in output states with lower weights. This is because states with higher weights have more qubits in the excited state, making them more prone to relaxation into states with lower weights [39].

For *Odr* and *Output State Error* (see Table 2), for all circuits, negative correlations were observed, showing that reducing the odds of observing a specific output state leads to an increase in *Output State Error*. This suggests that states with lower odds are more susceptible to noise effect, or they might be noise-induced states (e.g., see Table 1's row *Noisy*). Note that the correlation magnitude varies across the circuits, as expected. From columns $Dpe_{1/4}$, $Dpe_{1/2}$, and $Dpe_{3/4}$, we observe that each *Dpe* feature exhibits a moderate to strong positive correlation with *Output Error* for all circuits implying that *Dpe* can be used to quantify the noise magnitude affecting a specific circuit output.

Table 3: RQ2 – Performance of ML models on test data for six most commonly used regression performance metrics – across all eight real quantum computers. Each model with the best performance for a specific metric is in bold.

Model	*R*	R^2	*RMSE*	*RMSRE*	*MAPE*	*TestLoss*
MLP	0.787	**0.605**	**0.071**	0.213	5.479	**0.005**
LGBM	0.792	0.567	0.074	0.227	5.655	**0.005**
XGB	0.761	0.554	0.075	0.210	5.179	0.006
EDT	**0.804**	0.590	0.072	0.212	5.189	**0.005**
LR	0.751	0.542	0.076	0.201	5.068	0.006
Ridge	0.751	0.542	0.076	0.201	5.068	0.006
Lasso	0.754	0.551	0.075	0.200	**5.039**	0.006
Elastic	0.754	0.551	0.075	0.200	**5.039**	0.006
SVR	0.637	0.120	0.105	**0.187**	5.058	0.011
KNNR	0.710	0.485	0.080	0.227	6.272	0.006

> **RQ1:** Overall, the Pearson correlation analysis revealed significant correlations for Q-LEAR's feature set with errors in quantum circuit output. This suggests that the feature set has the potential to be utilized for mitigating the noise-induced errors in the circuit output.

6.3.2 RQ2 – Comparison with QRAFT.

In this RQ, we train machine learning models using Q-LEAR's feature set and compare them with the state-of-the-art QRAFT. To identify the best-performing regression model to compare with QRAFT, we used metrics *R*, R^2, *RMSE*, *RMSRE*, *MAPE*, and *TestLoss* introduced in Sect. 6.2.4.

Table 3 presents the results of these metrics for all quantum computers. All models exhibit comparable performance across all metrics, indicating that the regression models learned similarly from the selected feature set. Higher *R* and R^2 values indicate better results, while lower values are preferred for other metrics. Table 3 shows that MLP slightly outperforms the others as it is the best in three metrics; thus, we use it for comparison against QRAFT.

Table 4 presents a comparison between Q-LEAR and QRAFT. The column ***Obv*** shows the *Output Error* without any mitigation, the column ***M*** represents the *Output Error* after applying Q-LEAR's mitigation, and the column ***Q*** shows the *Output Error* after QRAFT's mitigation. The columns ***%M*** and ***%Q*** demonstrate the percentage improvement from ***Obv***. Additionally, the column ***%B*** shows the percentage improvement that Q-LEAR achieved compared to QRAFT in column ***Q***. In Table 4, positive percentage improvements are highlighted in green, while negative improvements (indicating performance below the compared values) are shown in red.

Table 4a presents the average comparison of Q-LEAR's MLP and QRAFT in terms of *Output Error* (Eq. 1) for the six application-level quantum circuits and all the selected quantum computers. The table shows a large difference in noise magnitude between the simulators and real computers. For example, for *GS*, on average, the observed *Output Error* (***Obv***) on the simulators is 0.35, but on the real computers it's 0.55. This difference is a big error margin indicating the need for training data from real computers, which is unfortunately limited due to restricted access. Regarding simulators, the Q-LEAR's MLP model demonstrates a substantial improvement in *Output Error* (column *%M*) compared to QRAFT (column *%Q*) across all circuits. Though the magnitude of the error

Table 4: RQ2 – Comparison of MLP of Q-LEAR (M) with that of QRAFT (Q), in terms of *Output Error*, on IBM's quantum computers and simulators. Column *Obv* shows the averaged values of observed *Output Error* without any prediction from Q-LEAR's MLP or QRAFT. Columns *%M* and *%Q* show the percentage improvement (given by $\frac{v2-v1}{|v1|} * 100$) in error mitigation that Q-LEAR's MLP and QRAFT achieved over *Obv*. *%B* shows the percentage improvement that Q-LEAR's MLP achieved over baseline QRAFT.

(a) Circuit-level (across all selected quantum computers)

Circuit	Simulators						Real Computers					
	M	*Q*	*Obv*	*%M*	*%Q*	*%B*	*M*	*Q*	*Obv*	*%M*	*%Q*	*%B*
GS	0.20	0.36	0.35	43.0	-3.0	44.4	0.45	0.60	0.55	18.0	-9.0	25.0
PC	0.32	0.34	0.54	41.0	37.0	6.0	0.60	0.62	0.72	17.0	14.0	3.2
PP	0.34	0.35	0.55	38.0	36.0	2.8	0.59	0.63	0.72	18.0	12.0	6.3
QAOA	0.26	0.29	0.44	41.0	34.0	10.3	0.57	0.48	0.64	11.0	25.0	-18.7
RT	0.05	0.12	0.23	78.0	48.0	58.3	0.10	0.72	0.28	64.0	-157.0	86.1
TSP	0.24	0.34	0.33	27.0	-3.0	29.4	0.36	0.66	0.43	16.0	-53.0	45.4
Average				**44.6**	24.8	**25.2**				**24.0**	-28.0	**25.0**

(b) Computer-level (across all application-level quantum circuits)

Computer	Simulators						Real Computers					
	M	*Q*	*Obv*	*%M*	*%Q*	*%B*	*M*	*Q*	*Obv*	*%M*	*%Q*	*%B*
Lagos	0.16	0.29	0.33	52.0	12.0	45.0	0.44	0.60	0.54	19.0	-11.0	27.0
Nairobi	0.24	0.34	0.41	41.0	17.0	29.4	0.46	0.61	0.56	18.0	-9.0	24.5
Perth	0.21	0.33	0.40	48.0	18.0	36.3	0.47	0.64	0.56	16.0	-14.0	26.5
Belem	0.43	0.29	0.56	23.0	48.0	-48.2	0.65	0.61	0.69	6.0	12.0	-6.5
Jakarta	0.16	0.26	0.34	53.0	24.0	38.4	0.38	0.61	0.52	27.0	-17.0	38.0
Lima	0.23	0.34	0.41	44.0	17.0	32.3	0.44	0.63	0.55	20.0	-15.0	30.1
Manila	0.25	0.26	0.41	39.0	37.0	3.8	0.35	0.63	0.50	30.0	-26.0	44.4
Quito	0.20	0.28	0.39	49.0	28.0	28.5	0.36	0.62	0.52	31.0	-19.0	41.9
Average				**38.1**	25.1	**21.0**				**21.0**	-12.3	**28.2**

mitigation is smaller for real computers, the Q-LEAR's MLP model still outperforms QRAFT, with a total average improvement of 24%. In contrast, QRAFT tends to overestimate the noise magnitude, especially in real computer data (highlighted in red). An explanation is that QRAFT, during feature calculation, doubles the circuit depth to quantify noise, thereby increasing decoherence and cross-talk probabilities. This is, however, not the case for the Q-LEAR's MLP model, as it employs *Dpe*, which does not need to go beyond the circuit depth. In contrast to the baseline QRAFT (column %B), Q-LEAR's MLP model demonstrated an average improvement of 25% for both simulators and real quantum computers. This indicates that the feature set of Q-LEAR was effective in enhancing performance, achieving up to a 25% improvement over the baseline QRAFT.

Table 4b shows the comparison of the Q-LEAR's MLP model and QRAFT regarding *Output Error* (Eq. 1) for each selected quantum computer across all application-level quantum circuits to study error mitigation capabilities for each specific quantum computer. The table shows that, on average, MLP achieved 38.1% error mitigation compared to QRAFT's 25.1% on the simulators and 21% as compared to QRAFT's −12.3% on real computers. QRAFT, in general, seems to overestimate noise in real computers with the only exception of *Belem* computer where QRAFT is better than Q-LEAR's MLP. For the rest, Q-LEAR's MLP performs better, showing that Q-LEAR has the potential to capture noise patterns for most computers. *Belem* being an exception can be caused by two reasons. Either the noise pattern in *Belem* significantly differs from other computers, or the training data is insufficient to generalize across all quantum computers. We will investigate this in the future.

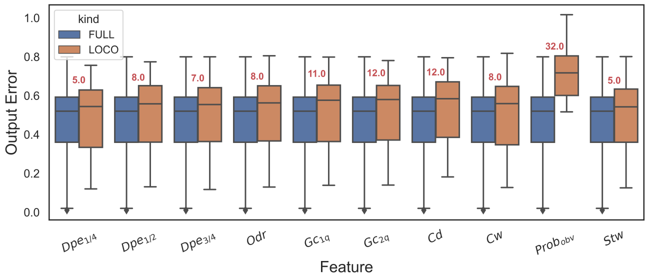


Figure 3: RQ3 – LOCO feature importance boxplots for the MLP model on test data for real computers and application-level circuits. The text (red) shows the %difference between the medians of boxplots with and without a specific feature.

> **RQ2:** Compared to state-of-the-art, Q-LEAR is effective in reducing *Output Error* on simulators and real quantum computers. Training ML models with Q-LEAR on simulator data even outperforms the state-of-the-art method trained on real quantum computers highlighting Q-LEAR's potential in capturing noise and motivating its application in training data obtained from real quantum computers.

6.3.3 *RQ3 – Feature Importance.*

We used the Leave-one-covariate-out (LOCO) feature importance technique introduced in Section 6.2.4 to comprehensively assess each feature's impact on the model's performance. To minimize the effect of training variability, we retrained the MLP model with a full feature set ten times. We also executed the LOCO process for each feature ten times. Next, we calculated an average *Output Error* for each real computer and application-level circuit pair, resulting in 48 observations for the full feature set and 48 observations for the LOCO process for each feature. Fig. 3 shows that excluding $Prob_{obv}$ caused a median increase of 32% *Output Error* as compared to the full Q-LEAR (the blue boxplot). All other features have a median increase of over 5% *Output Error*. This shows that all features are important for the MLP model performance. Moreover, we also see slight increases in the variances for the LOCO boxplots as compared to those of the full feature set. This suggests that the full Q-LEAR helps to achieve more trustworthy predictions when compared with the models trained after dropping any of the features.

We also conducted the Mann-Whitney U [20] statistical test and Vargha Delaney $\hat{A}_{12}$ [53] effect size measure. For statistical analysis, we used 10 observations of *Output Error* for a circuit-computer pair without a feature and compared them with 10 observations with the full feature set. Due to space constraints, we provide a summary of findings; however, full results are provided with the code [4]. In summary, out of 48 total circuit-computer pairs, for $Dpe_{1/4}$ and $Dpe_{1/2}$, there are 71% pairs, for which removing these features results in a statistically significantly worse *Output Error* than keeping them. For $Dpe_{3/4}$, the percentage is 62%, for *Odr* is 58%, for Gc_{1q} is 75%, for Gc_{2q} is 69%, for *Cd* is 60%, for *Cw* is 75%, for $Prob_{obv}$ is 88%, and for *Stw* is 62%. For all the statistically significant pairs, the $\hat{A}_{12}$ effect size was *large* according to the classification of $\hat{A}_{12}$ of Vargha et al. [53]. When looking at all computer-circuit pairs (i.e., a total of 48), we observed that for 48%, all features were important. For the remaining pairs, we observed that only a subset of features were important. For instance, for *QAOA* in the *Lagos* computer, all features’ p-values are less than 0.05, with large effect sizes indicating that all features played an important role. On the other hand, for *QAOA* on the *Perth* computer, only $Dpe_{1/2}$, Gc_{1q}, *Cw*, and $Prob_{obv}$ have p-values less than 0.05 with large effect sizes. This insight can be used to recommend specific features for specific computers and circuits, which is an interesting future direction.

> **RQ3:** The $Prob_{obv}$ feature holds the highest significance and all features contribute significantly to error mitigation.

7 THREATS TO VALIDITY

Construct Validity pertains to how accurately a measurement assesses the intended theoretical concept. One such threat is associated with the metrics employed to evaluate the effectiveness of Q-LEAR. In this work, we used widely accepted regression metrics [40], including *R*, R^2, *RMSE*, *RMSRE*, *MAPE*, and *TestLoss* introduced in Section 6.2.4 to assess the performance of the machine learning models.

Another concern relates to the choice of metric for comparison with QRAFT. We opted for the Hellinger Distance (HLD) metric to calculate the output error, in contrast to the Total Variance Distance (TVD) used by QRAFT due to: 1) HLD is widely used for evaluating performance in the presence of noise [15, 16, 24]; 2) Unlike TVD, HLD considers the difference in both probability values and the overall shape of the distribution, which is crucial as a quantum program may produce outcomes with low probabilities, yet the result is deemed correct as long as the distribution shape aligns with the expected ideal shape.

Internal Validity concerns the extent to which experiments can establish a causal relationship between independent and dependent variables. One such threat is about hyperparameters of the ML models. To this end, we used Bayesian optimization for hyperparameter tuning and implemented five-fold cross-validation. This helped mitigate dataset selection bias and ensured a more robust and unbiased evaluation of the models. Another internal validity threat relates to the choice of the depth-cut interval used for calculating the *Dpe* feature. We opted for 1/4th intervals as they provided a fine segmentation that suited the specific characteristics of the quantum programs under investigation. In many quantum circuits, there are distinctive segments such as state preparation, computational steps, and measurements. The selected intervals closely aligned with these segment divisions in our experiment. For example, in most programs we examined, the last 1/4th of their circuits predominantly consisted of measurement operations. However, we acknowledge that this choice may not be universally optimal. Determining the most suitable intervals requires a separate experiment, and this aspect is part of our future research plans.

Conclusion Validity focuses on the statistical significance of the results derived from an experiment. Using an ML model introduces inherent randomness, meaning that the presented results can exhibit variability. To address this concern, in RQ1, we implemented five-fold cross-validation to minimize the randomness during the ML model training. Regarding RQ2, we assessed ten distinct runs of the complete LOCO experiment to illustrate the median change in performance. The multiple runs of the experiment help dealing with the randomness introduced by ML, providing a more robust and reliable analysis of the outcomes.

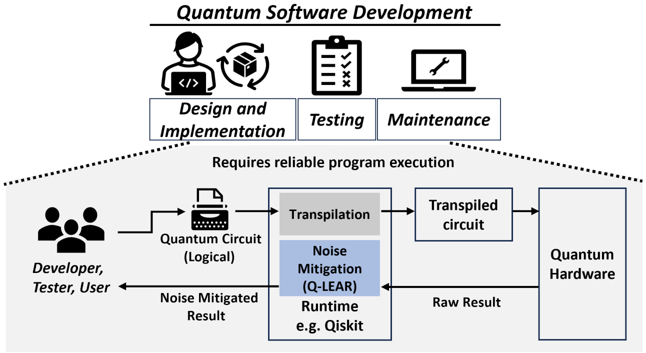


Figure 4: Quantum Program Execution Flow using IBM’s Qiskit Framework integrated with Q-LEAR.

External Validity concerns the applicability of our method to other datasets and domains. One challenge in this regard is the number of qubits used in quantum circuits, as the impact of noise significantly increases with a higher number of qubits. Due to limited access to real quantum computers, we used circuits with only 5 to 7 qubits. Note that using simulators with a higher number of qubits does not ensure optimal performance on real quantum computers, as the noise approximation in simulators may differ substantially from that in actual quantum hardware, as evidenced by the *Obv* column in Table 4. The limited number of available circuits for training and testing restricts the generalizability of our findings. To address this limitation, we used the widely adopted benchmark MQT [42] to select circuits that represent commonly used quantum circuits. Our approach involved using simpler circuits for training and more complex, real-world problem-solving circuits for testing, to maintain practical relevance in the noisy QC era.

8 DISCUSSION

Practical Implications. We provide a practical solution to support reliable quantum software development across various stages, including design and implementation, testing, and maintenance. During implementation, testing, and maintenance, having a dependable quantum program execution workflow is critical for verifying the developed coding solution and assessing the quantum software’s correctness. While quantum software development can initially be done using ideal quantum computer simulators that are noise-free, these simulators become impractical as the complexity of quantum programs increases exponentially with the growing number of qubits [29]. For practical quantum programs, developers often have to rely on real quantum computers for program execution. However, this introduces a new level of uncertainty in the output of quantum programs due to the effects of noise.

Fig. 4 shows the integration of Q-LEAR with IBM’s Qiskit framework. By mitigating the noise effect from the noisy output of quantum computer after each program execution, Q-LEAR enhances the reliability of outputs obtained from quantum program execution.

In our experiments, we demonstrated that Q-LEAR can be applied to IBM's quantum computers as a post-processing approach. In the quantum software development process, when executing quantum software on a quantum computer, Q-LEAR empowers quantum software developers to effortlessly mitigate noise from the original outputs produced by quantum software. This capability allows developers to analyze and verify whether the quantum software aligns with their intended behavior, ultimately enhancing the precision and reliability of quantum software development.

Integration with Quantum Software Stack. Q-LEAR functions as a post-processing module designed to mitigate the effects of noise in the outputs of a quantum program. The process of quantum program execution varies depending on the quantum platform and the framework used for quantum software development. In a typical process, the logical quantum circuit (in our case, written in Qiskit) is first transpiled for the target quantum computer. The execution of the transpiled circuit is then handled by the runtime service provider, such as IBM's Qiskit, and the raw results are obtained from the target quantum hardware. Q-LEAR is designed as a separate module that can be integrated with runtime service providers like Qiskit to consume the raw output from quantum hardware and perform noise mitigation. The feature set in Q-LEAR comprises generic features that can be calculated from the transpiled version of the quantum circuit and the raw results from the quantum hardware. This design allows Q-LEAR to seamlessly integrate with widely used quantum software development frameworks such as Qiskit, as it does not require underlying API calls or logic access from the black-box runtime services and can seamlessly be integrated into the quantum software execution process.

Insights on Quantum Noise. In our experiments, the significant disparity between the classically modeled quantum noise, i.e., noisy simulators, and the noise in real quantum computers is evident, as indicated by Table 4 column (***Obv***). Noisy simulators currently serve as a weak approximation of real quantum noise. It is well-established in machine learning that data quality is a crucial component. While training machine learning models for noise mitigation using simulators can yield progress, it has limitations. Our experiments demonstrated that Q-LEAR outperforms the state-of-the-art even when trained using data from simulators. However, to achieve substantial improvements in noise mitigation on real quantum computers, there is a pressing need for training data collected from real systems. Presently, wide access to real quantum computers is severely restricted, making it impractical to gather sufficient data for more effective machine learning model training. To support learning-based methods for noise mitigation, there is a need for enhanced infrastructure to support quantum technology. This infrastructure development is a crucial component of the quantum software development roadmap outlined by key companies like IBM [27].

Generalizability to Other Quantum Computers. In this work, our experimentation focused on the industrial case study of IBM quantum computers. However, the applicability of Q-LEAR is not limited to IBM quantum computers alone. Q-LEAR is designed to work with the majority of gate-based quantum computers, including those from IBM, Google, and Rigetti. Q-LEAR operates with the transpiled version of a quantum program, and the transpilation process is managed by the vendor's own runtime services such as IBM's Qiskit, Google's Cirq, or Rigetti's Forrest. This design ensures that Q-LEAR can work with quantum circuits across multiple quantum computers. Additionally, the feature set employed by Q-LEAR is generic and can be computed for all gate-based quantum computers. Currently, the primary differences between quantum computers from different vendors lie in the basis gate set supported and the topology of physical connections. These disparities influence changes in the transpilation process, producing different transpiled quantum circuits for the same logical circuit on different quantum computers. However, Q-LEAR's feature set distinguishes between gates based on the number of qubits they act on and calculate features from the transpiled circuits, making it applicable to all current gate-based quantum computers.

Moreover, Q-LEAR is a post-processing module that operates independently of specific quantum computing runtimes, allowing it to process raw outputs from any gate-based quantum computing platform like Qiskit, Cirq, or Forrest. This allows Q-LEAR to integrate with quantum computers beyond IBM's ecosystem. The ML model trained by Q-LEAR for IBM computers may not be directly applicable to other quantum computers like Google's Sycamore or Rigetti's Aspen due to variations in quantum noise influenced by the physical and environmental characteristics of each system [37]. Retraining the ML model for each quantum computer vendor is necessary, but it's a one-time unavoidable cost due to the inherent nature of quantum noise

9 CONCLUSION AND FUTURE WORK

To use machine learning (ML) for quantum error mitigation in the current noisy quantum computers, we introduce Q-LEAR that utilizes a reliable feature set for training machine learning models. These features are derivable from a quantum circuit and its corresponding output, allowing ML algorithms to mitigate errors in quantum circuit outputs. We evaluated Q-LEAR with six application-level quantum circuits on IBM quantum computers and their corresponding simulators. Our results, in general, show an average improvement of 25% compared to state-of-the-art on industrial-grade quantum computers and simulators. Our feature importance experiment results show that in general for error mitigation, all features are important. However, for some circuit-computer pairs, the significance of each feature varies. In the future, we will experiment with diverse circuits and investigate the relationship between noise and individual features, particularly how this relationship is influenced by diverse quantum operations across various quantum circuits.

ACKNOWLEDGMENTS

The work is supported by the Qu-Test project (Project #299827) funded by the Research Council of Norway. S. Ali is also supported by Oslo Metropolitan University's Quantum Hub. Paolo Arcaini is supported by Engineerable AI Techniques for Practical Applications of High-Quality Machine Learning-based Systems Project (Grant Number JPMJMI20B8), JST-Mirai.

REFERENCES

[1] Nabila Abdessaied and Rolf Drechsler. 2016. *Background.* Springer International Publishing, Cham, 9–43. https://doi.org/10.1007/978-3-319-31937-7_2

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2623–2631. https://doi.org/10.1145/3292500.3330701

[3] R Alicki. 2004. Decoherence and the Appearance of a Classical World in Quantum Theory. *Journal of Physics A: Mathematical and General* 37, 5 (feb 2004), 1948. https://doi.org/10.1088/0305-4470/37/5/B02

[4] Asmar. 2024. *AsmarMuqeet/QLEAR: Public Release.* https://doi.org/10.5281/zenodo.11181417

[5] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. 1995. Elementary gates for quantum computation. *Phys. Rev. A* 52 (Nov 1995), 3457–3467. Issue 5. https://doi.org/10.1103/PhysRevA.52.3457

[6] Jeff P. Barnes, Colin J. Trout, Dennis Lucarelli, and B. D. Clader. 2017. Quantum error-correction failure distributions: Comparison of coherent and stochastic error models. *Physical Review A* 95, 6 (jun 2017). https://doi.org/10.1103/physreva.95.062338

[7] Teresa Brecht, Wolfgang Pfaff, Chen Wang, Yiwen Chu, Luigi Frunzio, Michel H Devoret, and Robert J Schoelkopf. 2016. Multilayer microwave integrated quantum circuits for scalable quantum computing. *npj Quantum Information* 2, 1 (2016), 1–4.

[8] Zhenyu Cai. 2021. Multi-exponential error extrapolation and combining error mitigation techniques for NISQ applications. *npj Quantum Information* 7, 1 (2021), 80.

[9] Pascal Cerfontaine, René Otten, and Hendrik Bluhm. 2020. Self-Consistent Calibration of Quantum-Gate Sets. *Physical Review Applied* 13, 4 (apr 2020). https://doi.org/10.1103/physrevapplied.13.044071

[10] I. L. Chuang, R. Laflamme, P. W. Shor, and W. H. Zurek. 1995. Quantum computers, factoring, and decoherence. *Science* 270, 5242 (1995), 1633–1635. https://doi.org/10.1126/science.270.5242.1633 arXiv:9503007 [quant-ph]

[11] Andrew Cross. 2018. The IBM Q experience and QISKit open-source quantum computing software. In *APS March meeting abstracts*, Vol. 2018. L58–003.

[12] Diogo Cruz, Francisco A Monteiro, and Bruno C Coutinho. 2023. Quantum error correction via noise guessing decoding. *IEEE Access* 11 (2023), 119446–119461.

[13] Piotr Czarnik, Andrew Arrasmith, Patrick J. Coles, and Lukasz Cincio. 2021. Error mitigation with Clifford quantum-circuit data. *Quantum* 5 (Nov. 2021), 592. https://doi.org/10.22331/q-2021-11-26-592

[14] Antonio D. Córcoles, Abhinav Kandala, Ali Javadi-Abhari, Douglas T. McClure, Andrew W. Cross, Kristan Temme, Paul D. Nation, Matthias Steffen, and Jay M. Gambetta. 2020. Challenges and Opportunities of Near-Term Quantum Computing Systems. *Proc. IEEE* 108, 8 (2020), 1338–1352. https://doi.org/10.1109/JPROC.2019.2954005

[15] Samudra Dasgupta and Travis S. Humble. 2022. Assessing the stability of noisy quantum computation. In *Quantum Communications and Quantum Imaging XX*, Keith S. Deacon and Ronald E. Meyers (Eds.), Vol. 12238. International Society for Optics and Photonics, SPIE, 1223809. https://doi.org/10.1117/12.2631809

[16] Samudra Dasgupta and Travis S. Humble. 2022. Characterizing the Reproducibility of Noisy Quantum Circuits. *Entropy* 24, 2 (2022). https://doi.org/10.3390/e24020244

[17] Paul Adrien Maurice Dirac. 1939. A new notation for quantum mechanics. In *Mathematical Proceedings of the Cambridge Philosophical Society*, Vol. 35. Cambridge University Press, 416–418.

[18] Deep Dive. [n. d.]. With fault tolerance the ultimate goal, error mitigation is the path that gets quantum computing to usefulness. ([n. d.]). https://research.ibm.com/blog/gammabar-for-quantum-advantage?trk=public_post_comment-text

[19] David P DiVincenzo. 1998. Quantum gates and circuits. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 454, 1969 (1998), 261–276.

[20] Yadolah Dodge. 2008. *Mann–Whitney Test.* Springer New York, New York, NY, 327–329. https://doi.org/10.1007/978-0-387-32833-1_243

[21] Samuele Ferracin, Akel Hashim, Jean-Loup Ville, Ravi Naik, Arnaud Carignan-Dugas, Hammam Qassim, Alexis Morvan, David I. Santiago, Irfan Siddiqi, and Joel J. Wallman. 2022. Efficiently improving the performance of noisy quantum computers. arXiv:2201.10672 [quant-ph]

[22] Marc Ganzhorn, G Salis, DJ Egger, A Fuhrer, Matthias Mergenthaler, Clemens Müller, Peter Müller, Stephan Paredes, M Pechal, Max Werninghaus, et al. 2020. Benchmarking the noise sensitivity of different parametric two-qubit gates in a single superconducting quantum computing platform. *Physical Review Research* 2, 3 (2020), 033447.

[23] Tudor Giurgica-Tiron, Yousef Hindy, Ryan LaRose, Andrea Mari, and William J Zeng. 2020. Digital zero noise extrapolation for quantum error mitigation. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 306–316.

[24] Robin Harper, Steven T. Flammia, and Joel J. Wallman. 2020. Efficient learning of quantum noise. *Nature Physics* 16, 12 (aug 2020), 1184–1188. https://doi.org/10.1038/s41567-020-0992-8

[25] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression.* Vol. 398. John Wiley & Sons.

[26] Mingxia Huo and Ying Li. 2021. Self-consistent tomography of temporally correlated errors. *Communications in Theoretical Physics* 73, 7 (may 2021), 075101. https://doi.org/10.1088/1572-9494/abf72f

[27] IBM. 2024. Quantum Roadmap. https://www.ibm.com/roadmaps/quantum/.

[28] IBM Qiskit. 2023. W state quantum circuit. https://quantum-computing.ibm.com/composer/docs/iqx/guide/entanglement#w-state.

[29] Gregg Jaeger. 2007. Classical and quantum computing. *Quantum Information: An Overview* (2007), 203–217.

[30] Akshaya Jayashankar and Prabha Mandayam. 2023. Quantum error correction: Noise-adapted techniques and applications. *Journal of the Indian Institute of Science* 103, 2 (2023), 497–512.

[31] Wilhelm Kirch (Ed.). 2008. *Pearson's Correlation Coefficient.* Springer Netherlands, Dordrecht, 1090–1091. https://doi.org/10.1007/978-1-4020-5614-7_2569

[32] Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. 2018. Distribution-free predictive inference for regression. *J. Amer. Statist. Assoc.* 113, 523 (2018), 1094–1111.

[33] Ying Li and Simon C. Benjamin. 2017. Efficient Variational Quantum Simulator Incorporating Active Error Minimization. *Phys. Rev. X* 7 (Jun 2017), 021050. Issue 2. https://doi.org/10.1103/PhysRevX.7.021050

[34] Haoran Liao, Derek S. Wang, Iskandar Sitdikov, Ciro Salcedo, Alireza Seif, and Zlatko K. Minev. 2023. Machine Learning for Practical Quantum Error Mitigation. arXiv:2309.17368 [quant-ph]

[35] Shunlong Luo. 2010. From quantum no-cloning to wave-packet collapse. *Physics Letters A* 374, 11-12 (2010), 1350–1353.

[36] Thomas J Maldonado, Johannes Flick, Stefan Krastanov, and Alexey Galda. 2022. Error rate reduction of single-qubit gates via noise-aware decomposition into native gates. *Scientific Reports* 12, 1 (2022), 6379.

[37] Stefano Martina, Stefano Gherardini, Lorenzo Buffoni, and Filippo Caruso. 2022. Noise fingerprints in quantum computers: Machine learning software tools. *Software Impacts* 12 (2022), 100260. https://doi.org/10.1016/j.simpa.2022.100260

[38] Zhonghao Pan, Yang Feng, Zhiyuan Li, Yunxin Liu, and Yuanchun Li. 2023. Understanding the Impact of Quantum Noise on Quantum Programs. In *2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 426–437. https://doi.org/10.1109/SANER56733.2023.00047

[39] Tirthak Patel and Devesh Tiwari. 2021. Qraft: Reverse Your Quantum Circuit and Know the Correct Program Output. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Virtual, USA) *(ASPLOS '21)*. Association for Computing Machinery, New York, NY, USA, 443–455. https://doi.org/10.1145/3445814.3446743

[40] Vagelis Plevris, German Solorzano, Nikolaos P Bakas, and Mohamed El Amine Ben Seghier. 2022. Investigation of performance metrics in regression analysis and machine learning-based prediction models. In *8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2022)*. European Community on Computational Methods in Applied Sciences.

[41] J. S. Pratt and J. H. Eberly. 2001. Qubit cross talk and entanglement decay. *Phys. Rev. B* 64 (Oct 2001), 195314. Issue 19. https://doi.org/10.1103/PhysRevB.64.195314

[42] Nils Quetschlich, Lukas Burgholzer, and Robert Wille. 2023. MQT Bench: Benchmarking Software and Design Automation Tools for Quantum Computing. *Quantum* (2023). MQT Bench is available at https://www.cda.cit.tum.de/mqtbench/.

[43] Salonik Resch and Ulya R. Karpuzcu. 2021. Benchmarking Quantum Computers and the Impact of Quantum Noise. *ACM Comput. Surv.* 54, 7, Article 142 (jul 2021), 35 pages. https://doi.org/10.1145/3464420

[44] Iqbal H Sarker. 2021. Machine learning: Algorithms, real-world applications and research directions. *SN computer science* 2, 3 (2021), 160.

[45] Mohan Sarovar, Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, and Robin Blume-Kohout. 2020. Detecting crosstalk errors in quantum information processors. *Quantum* 4 (sep 2020), 321. https://doi.org/10.22331/q-2020-09-11-321

[46] Kevin Schultz, Ryan LaRose, Andrea Mari, Gregory Quiroz, Nathan Shammah, B. David Clader, and William J. Zeng. 2022. Impact of time-correlated noise on zero-noise extrapolation. *Phys. Rev. A* 106 (Nov 2022), 052406. Issue 5. https://doi.org/10.1103/PhysRevA.106.052406

[47] Armands Strikis, Dayue Qin, Yanzhu Chen, Simon C. Benjamin, and Ying Li. 2021. Learning-Based Quantum Error Mitigation. *PRX Quantum* 2 (Nov 2021), 040330. Issue 4. https://doi.org/10.1103/PRXQuantum.2.040330

[48] Sen Tan, Josep M Guerrero, Peilin Xie, Renke Han, and Juan C Vasquez. 2020. Brief survey on attack detection methods for cyber-physical systems. *IEEE Systems Journal* 14, 4 (2020), 5329–5339.

[49] Wei Tang, Teague Tomesh, Martin Suchara, Jeffrey Larson, and Margaret Martonosi. 2021. CutQC: using small Quantum computers for large Quantum circuit evaluations. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Virtual,

USA) *(ASPLOS '21)*. Association for Computing Machinery, New York, NY, USA, 473–486. https://doi.org/10.1145/3445814.3446758

[50] C Tannous and J Langlois. 2015. Classical noise, quantum noise and secure communication. *European Journal of Physics* 37, 1 (2015), 013001.

[51] Kristan Temme, Sergey Bravyi, and Jay M. Gambetta. 2017. Error Mitigation for Short-Depth Quantum Circuits. *Phys. Rev. Lett.* 119 (Nov 2017), 180509. Issue 18. https://doi.org/10.1103/PhysRevLett.119.180509

[52] Ewout Van Den Berg, Zlatko K Minev, Abhinav Kandala, and Kristan Temme. 2023. Probabilistic error cancellation with sparse Pauli–Lindblad models on noisy quantum processors. *Nature Physics* 19, 8 (2023), 1116–1121. https://doi.org/10.1038/s41567-023-02042-2

[53] Andrá's Vargha and Harold D. Delaney. 2000. A Critique and Improvement of the "CL" Common Language Effect Size Statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics* 25, 2 (2000), 101–132. http://www.jstor.org/stable/1165329

[54] Joel J. Wallman and Joseph Emerson. 2016. Noise tailoring for scalable quantum computation via randomized compiling. *Phys. Rev. A* 94 (Nov 2016), 052325. Issue 5. https://doi.org/10.1103/PhysRevA.94.052325

[55] Yulei Wu. 2020. Robust learning-enabled intelligence for the internet of things: A survey from the perspectives of noisy data and adversarial examples. *IEEE Internet of Things Journal* 8, 12 (2020), 9568–9579.

Received 2024-02-08; accepted 2024-04-18