Addressing Temporal Variations in Qubit Quality Metrics for Parameterized Quantum Circuits

Mahabubul Alam  
Department of Electrical Engineering  
Pennsylvania State University  
University Park, PA  
mxa890@psu.edu

Abdullah Ash-Saki  
Department of Electrical Engineering  
Pennsylvania State University  
University Park, PA  
axsi1251@psu.edu

Swaroop Ghosh  
Department of Electrical Engineering  
Pennsylvania State University  
University Park, PA  
szg212@psu.edu

Abstract—The public access to noisy intermediate-scale quantum (NISQ) computers facilitated by IBM, Rigetti, D-Wave, etc., has propelled the development of quantum applications that may offer quantum supremacy in the future large-scale quantum computers. Parameterized quantum circuits (PQC) have emerged as a major driver for the development of quantum routines that potentially improve the circuit’s resilience to the noise. PQC’s have been applied in both generative (e.g. generative adversarial network) and discriminative (e.g. quantum classifier) tasks in the field of quantum machine learning. PQC’s have been also considered to realize high fidelity quantum gates with the available imperfect native gates of a target quantum hardware. Parameters of a PQC are determined through an iterative training process for a target noisy quantum hardware. However, temporal variations in qubit quality metrics affect the performance of a PQC. Therefore, the circuit that is trained without considering temporal variations exhibits poor fidelity over time. In this paper, we present training methodologies for PQC in a completely classical environment that can improve the fidelity of the trained PQC on a target NISQ hardware by as much as 21.91%.

Index Terms—Quantum Computing, Parameterized Quantum Circuits, Fidelity, Decoherence, Noise.

I. INTRODUCTION

Quantum computing has observed a shift from being a purely academic exploration to a realistic industrial technology in recent years. However, the qubits have small coherence time (i.e., the quantum states are short-lived), the gate operations are imperfect, and the overall computation is extremely error-prone. Moreover, the near-term quantum devices offer a limited number of qubits without the costly feature of error correction. Due to these limitations, it is impossible to implement and test the target quantum algorithms (e.g., Shor’s factorization, grover’s search, etc.) on a classical computer. Therefore, the circuit that is trained without considering temporal variations exhibits poor fidelity over time. In this paper, we present training methodologies for PQC in a completely classical environment that can improve the fidelity of the trained PQC on a target NISQ hardware by as much as 21.91%.

Motivation: The trained PQC is supposed to be noise resilient as the training is generally performed with the noisy hardware in the loop approach to address the impact of noise as shown in Figure 4(a) [10]–[12]. However, the quantum computers operate under extremely controlled environment (i.e. operating temperature is in millikelvin range [13]) and the qubit performance metrics that define the qubit quality (e.g. T1 relaxation time, T2 dephasing time, single-qubit gate error, multi-qubit gate error, readout error, etc.) experience significant fluctuations over time. Generally, the quantum computers (e.g., IBMQX4 and IBMQX2 from IBM) are periodically calibrated through randomized benchmarking [14] and the updated qubit quality metrics are reported for the users to validate their quantum experiments on any target hardware. The variations in the performance metrics of the qubits in IBMQX4 quantum computer is shown in Figure 2. The data has been collected over a 43 days period. The significant variations in the qubit quality metrics indicate that variational circuits that are trained at any particular time using the hardware in the loop training methodology may not show the desired behavior all the time.

The temporal variability at the output of a quantum circuit is expected for any arbitrary quantum circuit. As a motivational example, we have executed the workload shown in Figure 1(b) on 5-qubit IBMQ4 quantum computer (the coupling graph of the device is shown in Figure 1(a)) on 5 different occasions.

Fig. 1. (a) Coupling graph of IBMQX4 (Tenerife) hardware from IBM; (b) Random quantum workload; (c) outcome at different points in time.

2-Qubit CNOT gates allowed between: 1-2, 2-1, 3-2, 4-2
Single-qubit Parameterized Gates (θ, φ, Z) are supported for all the qubits.

Terminal state of the qubits (after executing the workload) should be another basis state (10).
The qubits are prepared in the basis state $|00\rangle$. Ideally, at the end of the execution period, the qubits will be in another basis state $|10\rangle$. A projective measurement on the target hardware is expected to generate a measurement of ‘10’ most of the time. However, due to temporal variations of the qubit quality metrics, we have received significantly different outcomes at different points of time as shown in Figure 1(c). The y-axis shows the fidelity of the measurements (which is the % of the correct output for 1024 samples at a time). For circuits such as circuit-centric binary quantum classifiers based on PQC (discussed in Section II), the final outcome is decided after analyzing the measurement distributions in a classical computer which can be completely wrong due to the temporal variations of the qubit quality metrics.

**Contributions:** In this paper, we, (a) demonstrate training methodologies of PQC and address their respective pros and cons; (b) present a fully classical heuristic training methodology for PQC to address the temporal variations in qubit quality metrics; (c) used PQC based circuit-centric quantum classifiers to demonstrate our solutions and verified their effectiveness on real quantum hardware from IBM.

## II. BINARY QUANTUM CLASSIFIERS

### A. Quantum Computing Preliminaries

1) **Qubit and State Vector:** Qubit is the building block of quantum computers. Qubit state is expressed with a ket ($|\psi\rangle$) notation which is known as *state vector*. A single qubit state $|\psi\rangle$ is described as $|\psi\rangle = a|0\rangle + b|1\rangle$. Here, $|0\rangle$ and $|1\rangle$ are known as basis states represented by $|0\rangle$ and $|1\rangle$, and $a$ and $b$ are complex numbers s.t. $|a|^2 + |b|^2 = 1$.

2) **Density Matrix:** An alternate approach of representing qubit state is the *density matrix* ($\rho$) formalism which is expressed as $\rho = \sum p_i |\psi_i\rangle \langle \psi_i |$ where $p_i$ is the probability of pure state $|\psi_i\rangle$. This representation is beneficial since qubit states may end up in a mixed state due to noise that need to be expressed using density matrix.

3) **Quantum Gates:** Quantum gates are the operations that modulate the state of qubits. Mathematically, quantum gates are represented by $2^n \times 2^n$ unitary matrices ($n =$ number of qubits). When multiple gates work on different qubits, the overall unitary matrix can be calculated using tensor product ($\otimes$). For example, in Fig. 3(a) two U3 (native gate of IBMQX4) gates are working on qubit-1 and 0. Therefore, the overall gate matrix will be $U = U3 \otimes U3$.

4) **Expectation Value:** Expectation value is the average of the eigenvalues, weighted by the probabilities that the state is measured to be in the corresponding eigenstate. In quantum computers, measurement of a qubit is performed in the so-called Z-basis or computational basis $|0\rangle$ and $|1\rangle$. These are the eigenvectors (eigenstates) of Pauli-Z ($\sigma_z$) operator with eigenvalues $+1$ and $-1$ respectively. For quantum computing, a positive (negative) expectation value means that the measurements will yield more $|0\rangle$ ($|1\rangle$) than $|1\rangle$ ($|0\rangle$), if a qubit prepared in identical setup is measured many times. The measurements will always yield $|0\rangle$ ($|1\rangle$) if the expectation value is exactly $+1$ ($-1$). If the expectation value is 0, it means the qubit state is in a perfect superposition of both $|0\rangle$ and $|1\rangle$.

For more clarity, suppose the state of a qubit after a quantum computation routine is $|\psi\rangle = 0.8|0\rangle + 0.6|1\rangle$ (note the higher amplitude of $|0\rangle$). The expectation value of Pauli-Z operator in this state $|\psi\rangle$ is $\langle \psi | \sigma_z | \psi \rangle = 0.28$, a positive expectation value which validates the above discussion. Figure 3(a) shows the variations in the expectation value of a target qubit with respect to a gate parameter ($\theta$).

### B. Classifier Basics

Binary classification is the task of classifying any input data into one of two possible groups. In supervised machine learning, this classification problem is solved by training a mathematical model ($f(x, \theta)$) with a properly labeled input data-set $\{(x_1,y_1), (x_2,y_2), ..., (x_M,y_M)\}$ where $x_i$ is the feature vector (can be multi-dimensional) of the $i^{th}$ input data and $y_i$ is the associated label. The mathematical model predicts the class of any input data based on its features ($x$) and the parameters ($\theta$) of the model. The parameters ($\theta$) are updated iteratively until the model predictions are satisfactory over the input data-set.

In [6], a binary classification on quantum computers is proposed for classical data where a PQC serves as the mathematical model. A state-preparation routine is required to encode the classical data and feed it to the PQC. The output is captured from a target qubit. During the training phase of the PQC, the parameters are updated iteratively based on the given input data-set so that the probability of getting 1 through a measurement of the target qubit for one class is maximized (and 0 for the other class).

### C. State Preparation

A state preparation circuit (which is applied to the qubits at ground state) is used to convert any classical input data to a quantum format so that quantum gates can be applied on the data and/or quantum speed-up can be exploited. The structure of this circuit depends on the chosen encoding scheme. A multitude of quantum encoding scheme of classical
data have been proposed [15]. Two of the most promising encoding schemes - basis encoding and amplitude encoding - are discussed in the following Section.

Basis Encoding: In this scheme, binary 0 (1) is encoded as computational basis state $|0\rangle$ ($|1\rangle$). For instance, a classical data $x = 9$ (binary 1001) can be represented by 4-qubits (say, $Q_3Q_2Q_1Q_0$) where $Q_3$ and $Q_0$ ($Q_2$ and $Q_1$) are prepared in qubit state $|1\rangle$ ($|0\rangle$). The effect of the state-preparation routine can be written as $U_\phi : x \in \{0, 1\}^n \rightarrow |\psi_x\rangle$.

Here, $U_\phi$ is the unitary transformation that prepares the desired quantum state representative of classical data. For IBM quantum computers, all qubits start from a $|0\rangle$ state. Therefore, quantum NOT gate (Pauli-X, $\sigma_x$) has to be applied on $Q_3$ and $Q_0$ whereas Identity gates are applied on $Q_2$ and $Q_1$ to prepare $x = 9$ state. Thus, for this case $U_\phi = \sigma_x \otimes I \otimes I \otimes \sigma_x$. Although, the scheme results in a trivial quantum state-preparation circuit (that only requires NOT and Identity gates) which is fairly easy to implement on existing quantum hardware, the required number of qubits may grow linearly with the number of input features (e.g., two 4-bit classical features will require 8 physical qubits).

Amplitude Encoding: In this scheme, normalized input vectors $x = (x_1, x_2, \ldots, x_N)^T \in \mathbb{R}$ of dimension $N = 2^n$ are associated with the amplitudes of a n qubit state $|\psi_x\rangle (U_\phi : x \in \mathbb{R} \rightarrow |\psi_x\rangle = \sum_{\{x_1, x_2, \ldots, x_N\}} \phi_i |i\rangle)$. Example: The state $|\psi\rangle$ of a 2-qubit quantum system, due to superposition, is a linear combination of all possible computational basis state i.e. $\psi$ can be written as $a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ such that $\sqrt{a^2 + b^2 + c^2 + d^2} = 1$. Suppose, we have a classical input vector $x = \{1, 2, 3, 4\}$. After normalizing the input vector, we get $x_{norm} = \{0.183, 0.365, 0.547, 0.730\}$. The amplitude encoding scheme will encode this normalized classical input vector entries as the amplitude of the computational basis states of the whole quantum system such that $\psi$ becomes $0.183|00\rangle + 0.365|01\rangle + 0.547|10\rangle + 0.730|11\rangle$.

In this scheme, the number of qubits grows only logarithmically with the dimension of the classical input vectors (e.g. for the above example, only $\log_2(4) = 2$ qubits are required to encode 4 classical values). Furthermore, multiple inputs in superposition state can be processed simultaneously leading to potential speed-up in computation. Mathematically, quantum algorithms that are only polynomial in the number $n$ of qubits can perform computations on the $2^n$ amplitudes leading to a poly-logarithmic processing time. However, the encoding scheme results in a non-trivial state-preparation circuit which can be unsuitable for existing quantum hardware.

D. Model Circuit

The model circuit is a parameterized unitary transformation $U_\theta$ (where $\theta$ is a set of trainable variables) that acts as the mathematical model for the classification task. The model circuit transforms encoded state $|\psi_x\rangle$ to another state, say, $\psi' (|\psi'\rangle = U_\theta |\psi_x\rangle)$. Generally, the model circuit has a layered architecture. Each layer can have identical or dissimilar constructs. A single layer consists of a parametric and an entanglement sub-layer as shown in Figure 3(b). The parametric sub-layer consists of the parametric single qubit gates ($PG(\theta)$ in Figure 3(b)). These parameters $(\theta)$ are updated during training in an iterative fashion. The entanglement sub-layer consists of multi-qubit gates ($MQ$ gates shown in Figure 3(b)) which create a dependency between the target qubit and all other qubits in the circuit. The state preparation and model circuit is executed, the state of the target qubit is measured, and these execution and measurement operations are repeated multiple times. The measured distribution is analyzed in a classical computer to determine the class of a single input during inferencing. The selection of gates for the model circuit depends on the available native gates of the target NISQ hardware. Tree-like structures (TTN) have been proposed for the entanglement sub-layer as shown in Figure 3(c) [16]. MERA’s are similar to TTN’s, but make use of additional unitary transformations to effectively capture a broader range of quantum correlations as shown in Figure 3(c) [17]. Higher-depth circuits are more susceptible to decoherence induced errors which is the prominent source of error for qubits with a short lifetime. Therefore, the entanglement sub-layer structure should be chosen based on the available native gates and coupling graph with a goal to minimize the depth of the circuit.

III. Training of PQC

A. Existing Approaches

i) Train the $PQC$ in a hardware-in-the-loop fashion. Hereafter, we term this approach as $app01$. In this approach, the $PQC$ is executed on a real quantum computer. For a certain input, the output is measured and then the measured output is post-processed in a classical computer. Statistical techniques such as, Kullback-Leibler (KL) divergence method is used to calculate the disparity between the target distribution and the measured distribution (hence the cost) to update the parameters with any classical optimization techniques such as stochastic gradient descent or particle swarm optimization etc. [10]–[12].
Then the PQC is executed again with updated parameters and process iterates until measured output matches target output up to a certain threshold. While it may seem to be an ideal approach, the technique is plagued with certain impediments. First, qubits quality changes over time (Fig. 2) which means that a trained PQC on a certain day may not show optimal behavior over time due to qubit specification drift. Second, the quantum computers are expected to operate in a client-server fashion. Iterative training scheme may get prohibitively lengthy. Moreover, unlike classical bit states, intermediate quantum-mechanical states cannot be saved in a memory for lengthy. Moreover, unlike classical bit states, intermediate quantum-mechanical states cannot be saved in a memory for lengthy. Moreover, unlike classical bit states, intermediate quantum-mechanical states cannot be saved in a memory for lengthy.

ii) Simulation based training of the PQC where a model quantum computer is simulated (we name it app02). The simulation results in the expectation value of the result qubit which is then compared with target expectation value to calculate the cost. Now, we can define the following cost-function to iteratively update the parameters of the PQC (Fig. 4(b)) to solve the binary classification problem (described for the hybrid approach) [6]:

\[ J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (y_i - \text{expectation}(\text{PQC}(x_i, \theta); Q_T))^2 \]  

where \( m \) is the batch-size, \( y_i \) is the label of the \( i^{th} \) data in the batch (data are labeled as -1 and +1 for class A and class B respectively), \( x_i \) is the \( i^{th} \) input, and 'expectation(\text{PQC}(x_i,\theta); Q_T)' is the expectation value of the target qubit (\( Q_T \)) for the \( i^{th} \) input and current values of the \( \theta \). The target is to minimize the cost. Gradient descent technique is applied to achieve the optimization goal where the partial derivatives of the cost function (Equation 1) with respect to the circuit parameters are calculated using numerical differentiation [18].

In this approach, the client need not wait for the server (quantum hardware) to train and get the parameters of PQC.
B. Validation Setup

1) Data Source: In order to validate the effectiveness of our proposed training methodology, we have picked 4-bit parity classification problem (which can be also thought of as a high-fidelity 4-qubit parity gate realization problem using PQC [9]) with 16 known inputs/outputs combinations with two output classes (even and odd parity).

2) Evaluation Method: Although parameterized quantum circuits can minimize the effects of noise, it cannot suppress it altogether. Therefore, the expectation values cannot be optimized to exactly -1/+1 values for all the inputs during the PQC training period which indicates that a measurement is not guaranteed to result in the desired class output (0/1) for a certain input. Thus, the same circuit is executed multiple times (known as shots in IBMQX) and the target qubit is measured in each trial to get a distribution or ratio of 1s to 0s in the output. For binary classification, a large ratio e.g., >1 (<1) indicates the input belongs to the class represented by logic ‘1’ (‘0’). Example: A trained parity classifier is executed 1024 times on 4-qubits (Q2 Q3 Q1 Q0) of IBMQX4 with input state Q2 Q3 Q1 Q0 = 0100 (note the input has odd number of 1s i.e. odd-parity) with Q0 being the result/target qubit. The execution resulted in a distribution of ‘0000’ 762 times and ‘0001’ 262 times. The ratio of 1s to 0s of the target qubit is 0.34 (< 1) which indicates class belongs to logic ‘0’ or odd parity (alternately, correct output ‘0’/incorrect output ‘1’ = 2.9 > 1). In an ideal noise-less quantum computer, this ratio of 1s to 0s would have been 0. However, a class decision cannot be taken with confidence when the ratio is close to 1. In a series of measurements, the goal is to get a high ratio value between the correct and the incorrect outputs from a noisy device. The ratio between the correct and incorrect outputs is also a representation of the fidelity of the circuit.

V. RESULTS AND DISCUSSIONS

The 4-bit binary inputs for the parity classification is encoded to four qubits using the basis encoding scheme (Section II). Parametric U3(θ, φ, λ) gates of IBMQX4 have been used as the parametric gates of the model circuits. The ‘0’ outcomes (odd parity), and ‘1’ outcomes (even parity) have been labeled as +1 and -1 respectively for training based on Equation 1 using stochastic gradient descent (SGD) technique.

We have performed the parameter optimization of the model circuits using three different strategies, app01, app02 and app03 described in Section III and tested the performance of the optimized circuits in both real quantum computer, IBMQX4 and in simulation. TTNP2L and TTN2NL stand for the cost of two-layer TTN architectures during training for target ideal(pure) and noisy hardware respectively. However, it is to be noted that we adopted a simulated hardware-in-the-loop approach to mimic app01 due to interrupted and limited access to the real quantum computer. In this simulated approach, we substitute the NISQ computer with our noisy quantum computer simulation framework (described in Section IV) and use the error specifications of a respective day to optimize the PQC parameters. Note that we have used only 100 iterations of SGD for all the circuits over all the training approaches.

Fig. 7 reports the performance (ratio of correct to incorrect outputs) of a binary classifier circuit with 1-layer TTN topology (Fig. 6) on IBMQX4. The PQC generated from the app01 approach performed best in terms of the ratio of the correct and incorrect outcome over 1024 repeated measurements on the given day (the day on which the parameters were optimized) as evident from Figure 7 (TTN_Noisy). The average of the ratios (TTN_Noisy) was found 4.92. However, when the same circuit is executed on a different day (TTN_NoisyDD in Figure 7), it shows random behavior with substantially degraded performance in some cases (average of the ratios: 4.02). This trend validates one of our argument against app01 stated in Section III i.e. parameters optimized at one time may not be optimal at a different time.

The optimized circuit for the app02 approach performed poorly (average of the ratios: 3.45) over the entire input dataset(TTN_Pure in Figure 7). From the figure, it is evident that the circuit optimized with app03 consistently gives better performance than TTN_Pure and TTN_NoisyDD corroborating our claim in Section III. The ratio of the correct and incorrect outcome for all possible inputs are significantly higher (average of the ratios: 4.31) for the app03 approach. It is to be noted TTN_Pure, TTN_NoisyDD and TTN_NoisyAvg data are collected on the same day from IBMQX4.

We further substantiate our claim through simulation with the real hardware being substituted with our NISQ computer simulator in Section IV. Two topologies of the parity classifier model circuits (TTN and ALT) have been chosen as shown in Figure 6(b)&(c) both of which satisfies the coupling graph of the IBMQX4 hardware shown in Figure 1(a). For each topology, both single-layer and double-layer flavor is simulated i.e. a total of 4 test circuits are simulated. The circuits are optimized with app02 (TTNP1L, TTNP2L, ALTP1L, and ALTP2L) and app03 (TTNN1L, TTNN2L, ALTN1L, and ALTN2L) to show the superiority of the proposed approach app03.

Figure 8 shows the aggregate actual cost over the entire input data-set for the trained PQC’s (app02 - TTNP1L, TTNP2L, ALTP1L, ALTP2L and app03 - TTNN1L, TTNN2L, ALTN1L, ALTN2L) for a set of qubit quality metrics data (error specification) of IBMQX4 collected over a 43 days period. The cost here can be interpreted as a measure of the difference between the ideal (expected) result and the result with noise. The lesser the cost the closer the result is to expected. The actual cost for the app03 is consistently smaller than the app02. For instance, the average cost over the entire
input data-set for the 43 days period for TTNP1L (app02) is 0.042 which is 23.53% larger than the average cost (0.034) for TTNN1L(app03).

Thus, both real computer experiments and simulations support our proposed PQC parameter optimization technique considering the noise in real NISQ devices. We later executed the optimized PQC’s for app02 (TTN1L,TTN2L) and app03 (TTN1L,TTN2L) on IBMQX4 for 100 different times with randomly chosen inputs (1024 shots per time) and the cumulative probability (CP) distribution of the ratio’s of the correct/incorrect outputs (r) are shown in Figure 9. The higher ratio values for any given cumulative probability for app03 (e.g. TTN1L \( r = 5.24 \) for \( CP = 0.6 \)) than app02 (e.g. TTN1L \( r = 4.36 \) for \( CP = 0.6 \)) in Figure 9 substantiate our previous claim that the app03 optimized PQC’s would outperform app02 on a target quantum hardware. The average value of \( r \) for all the app03 PQC’s were found to be 21.91% higher than app02 for the TTN parity classifiers.

VI. CONCLUSIONS

We present the shortcomings of current training approaches for parameterized quantum circuits (PQC) and proposed a fully classical training methodology for target NISQ hardware to address the impact of temporal variations in qubit quality metrics. We present a simulation framework to model the circuit behavior on a target noisy quantum hardware. We validate our proposed solutions through comprehensive simulations and experiments on a real quantum device (IBM-QX4) of a quantum classifier built with PQC. The proposed methodology can improve the performance of any PQC based quantum application on a target NISQ hardware.

Acknowledgements: This work is supported by SRC (2847.001), NSF (CNS-1814710, CNS-1722557, CCF-1718474, DGE-1723687 and DGE-1821766) and DARPA Young Faculty Award (D15AP00089).

REFERENCES

[1] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” arXiv preprint arXiv:1411.4028, 2014.
[2] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” Nature, vol. 549, no. 7671, p. 242, 2017.
[3] J. Romero, J. P. Olson, and A. Aspuru-Guzik, “Quantum autoencoders for efficient compression of quantum data,” Quantum Science and Technology, vol. 2, no. 4, p. 045001, 2017.
[4] P.-L. Dallaire-Demers and N. Killoran, “Quantum generative adversarial networks,” Physical Review A, vol. 98, no. 1, p. 012324, 2018.
[5] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, “The expressive power of parameterized quantum circuits,” arXiv preprint arXiv:1810.11922, 2018.
[6] M. Schuld, A. Bochvar, K. Svore, and N. Wiebe, “Circuit-centric quantum classifiers,” arXiv preprint arXiv:1804.00633, 2018.
[7] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, “Continuous-variable quantum neural networks,” arXiv preprint arXiv:1806.08872, 2018.
[8] J. Romero and A. Aspuru-Guzik, “Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions,” arXiv preprint arXiv:1901.00848, 2019.
[9] K. Heya, Y. Suzuki, Y. Namakura, and K. Fujii, “Variational quantum gate optimization,” arXiv preprint arXiv:1810.12745, 2018.
[10] D. Zhu, N. Linke, M. Benedetti, K. Landsman, N. Nguyen, C. Alderete, A. Perdomo-Ortiz, N. Korda, A. Garfoot, C. Brecque et al., “Training of quantum circuits on a hybrid quantum computer,” arXiv preprint arXiv:1812.08862, 2018.
[11] V. Leyton-Ortega, A. Perdomo-Ortiz, and O. Perdomo, “Robust implementation of generative modeling with parameterized quantum circuits,” arXiv preprint arXiv:1901.08047, 2019.
[12] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, and A. Perdomo-Ortiz, “A generative modeling approach for benchmarking and training shallow quantum circuits,” npj Quantum Information, vol. 5, no. 1, p. 45, 2019.
[13] J. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Blok, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong et al., “Unsupervised machine learning on a hybrid quantum computer,” arXiv preprint arXiv:1712.05771, 2017.
[14] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, “Randomized benchmarking of quantum gates,” Physical Review A, vol. 77, no. 1, p. 012307, 2008.
[15] M. Schuld and N. Killoran, “Quantum machine learning in feature hilbert spaces,” Physical review letters, vol. 122, no. 4, p. 040504, 2019.
[16] W. J. Huggins, P. Patil, B. Mitchell, K. B. Whaley, and M. Stoudenmire, “Towards quantum machine learning with tensor networks,” Quantum Science and Technology, 2018.
[17] L. Cincio, J. Dziarmaga, and M. M. Rams, “Multiscale entanglement renormalization ansatz in two dimensions: Quantum ising model,” Physical review letters, vol. 100, no. 24, p. 240603, 2008.
[18] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, “Evaluating analytic gradients on quantum hardware,” Physical Review A, vol. 99, no. 3, p. 032331, 2019.
[19] G. Barbato, E. Barini, G. Genta, and R. Levi, “Features and performance of some outlier detection methods,” Journal of Applied Statistics, vol. 38, no. 10, pp. 2133–2149, 2011.
[20] A. Ash-Saki, M. Alani, and S. Ghosh, “Qure: Qubit re-allocation in noisy intermediate-scale quantum computers,” in Proceedings of the 56th Annual Design Automation Conference 2019. ACM, 2019, p. 141.
[21] J. R. Johansson, P. D. Nation, and F. Nori, “Qutip 2: A python framework for the dynamics of open quantum systems,” Computer Physics Communications, vol. 184, no. 4, pp. 1234–1240, 2013.
[22] I. B. M. Corporation. Ibm q experience. [Online]. Available: https://quantumexperience.ng.bluemix.net/qx/editor