QuantumNAT: Quantum Noise-Aware Training with Noise Injection, Quantization and Normalization

1Hanrui Wang, 2Jiaqi Gu, 3Yongshan Ding, 4Zirui Li, 5Frederic T. Chong, 2David Z. Pan, 1Song Han
1Massachusetts Institute of Technology, 2University of Taxes at Austin, 3Yale University, 4Shanghai Jiao Tong University, 5University of Chicago

https://qmlsys.mit.edu

ABSTRACT
Parameterized Quantum Circuits (PQC) are promising towards quantum advantage on near-term quantum hardware. However, due to the large quantum noises (errors), the performance of PQC models has a severe degradation on real quantum devices. Take Quantum Neural Network (QNN) as an example, the accuracy gap between noise-free simulation and noisy results on IBMQ-Yorktown for MNIST-4 classification is over 60%. Existing noise mitigation methods are general ones without leveraging unique characteristics of PQC; on the other hand, existing PQC work does not consider noise effect. To this end, we present QuantumNAT, a PQC-specific framework to perform noise-aware optimizations in both training and inference stages to improve robustness. We experimentally observe that the effect of quantum noise to PQC measurement outcome is a linear map from noise-free outcome with a scaling and a shift factor. Motivated by that, we propose post-measurement normalization to mitigate the feature distribution differences between noise-free and noisy scenarios. Furthermore, to improve the robustness against noise, we propose noise injection to the training process by inserting quantum error gates to PQC according to realistic noise models of quantum hardware. Finally, post-measurement quantization is introduced to quantize the measurement outcomes to discrete values, achieving the denoising effect. Extensive experiments on 8 classification tasks using 6 quantum devices demonstrate that QuantumNAT improves accuracy by up to 43%, and achieves over 94% 2-class, 80% 4-class, and 34% 10-class classification accuracy measured on real quantum computers. The code for construction and noise-aware training of PQC is available in the TorchQuantum library.

1 INTRODUCTION
Quantum Computing (QC) is a new computational paradigm that can be exponentially faster than classical counterparts in various domains. Parameterized Quantum Circuits (PQC) are circuits containing trainable weights and are promising to achieve quantum advantages in current devices. Among various PQCs, Quantum Neural Network (QNN) is a popular algorithm in which a network of parameterized quantum gates are constructed and trained to embed data and perform certain ML tasks on a quantum computer, similar to the training and inference of classical neural networks.

Currently we are in the Noisy Intermediate Scale Quantum (NISQ) stage, in which quantum operations suffer from a high error rate of $10^{-2}$ to $10^{-3}$, much higher than CPUs/GPUs ($10^{-6}$ FIT). The quantum errors unfortunately introduces detrimental influence on PQC accuracy. Figure 1 shows the single-qubit gate error rates and the measured accuracy of classification tasks with QNN on different hardware. Three key observations are: (1) Quantum error rates ($10^{-3}$) are much larger than classical CMOS devices’ error rates ($10^{-6}$ failure per $10^9$ device hours). (2) Accuracy on real hardware is significantly degraded (up to 64%) compared with noise-free simulation. (3) The same QNN on different hardware has distinct accuracy due to different gate error rates. IBMQ-Yorktown has a five times larger error rate than IBMQ-Santiago, and higher error causes lower accuracy.

Researchers have proposed noise mitigation techniques [23, 29] to reduce the noise impact. However, they are general methods without considering the unique characteristics of PQC, and can only be applied to PQC inference stage. On the other hand, existing PQC work [5, 11] does not consider the noise impact. This paper proposes a PQC-specific noise mitigation framework called QuantumNAT that optimizes PQC robustness in both training and inference stages, boosts the intrinsic robustness of PQC parameters, and improves accuracy on real quantum machines.

QuantumNAT comprises a three-stage pipeline. The first step, post-measurement normalization normalizes the measurement outcomes on each quantum bit (qubit) across data samples, thus removing the quantum error-induced distribution shift. Furthermore, we inject noise to the PQC training process by performing error gate insertion. The error gate types and probabilities are obtained from hardware-specific realistic quantum noise models provided by QC vendors.

In this paper, we are mainly using RNNs as benchmarks but the techniques can also be applied to other PQCs. The contributions of QuantumNAT are as follows:

- A systematic pipeline to mitigate noise impact: post-measurement normalization noise injection and post-measurement quantization.

---

Figure 1: Left: Current quantum hardware has much larger error rates ($10^{-3}$) than classical CPUs/GPUs. Right: Due to the errors, PQC (QNN) models suffer from severe accuracy drops. Different devices have various error magnitudes, leading to distinct accuracy. These motivate QuantumNAT, a hardware-specific noise-aware PQC training approach to improve robustness and accuracy.
QML and QNN. Quantum machine learning explores performing ML tasks on quantum devices. The path to quantum advantage on QML is typically provided by the quantum circuit’s ability to generate and estimate highly complex kernels, which would otherwise be intractable to compute with conventional computers. They have been shown to have potential speed-up over classical counterparts in various tasks, including metric learning, data analysis, and principal component analysis. Quantum Neural Networks is one type of QML models using variational quantum circuits with trainable parameters to accomplish feature encoding of input data and perform complex-valued linear transformations thereafter. Various theoretical formulations for QNN have been proposed such as quantum Boltzmann machine [1] and quantum classifier [5, 25, 26, 28], etc.

Quantum error mitigation. As the error forms the bottleneck of the quantum area. Researchers have developed various error mitigation techniques [29]. Extrapolation methods [23] perform multiple measurements of a quantum circuit under different error rates and then extrapolate the ideal measurement outcomes when there is no noise. [15] trains PQC using RL with noisy simulator. QuantumNAT is fundamentally different from existing methods: (i) Prior work focuses on low-level numerical correction in inference only; QuantumNAT embraces more optimization freedom in both training and inference. It improves the intrinsic robustness and statistical fidelity of PQC parameters. (ii) PQC has a good built-in error-tolerance which motivates QuantumNAT’s post-measurement quantization to reduce the numerical precision of intermediate results while preserving accuracy. (iii) QuantumNAT has a small overhead (<2%), while others introduce high measurements, circuit complexity cost, etc. We show that existing extrapolation method is orthogonal to QuantumNAT in Section 4.

Quantization and noise injection of classical NN. To improve NN efficiency, extensive work has been explored to trim down redundant bit representation in NN weights and activations [6, 27]. Though low-precision quantization limits the model capacity, it can improve the generalization and robustness [16]. An intuitive explanation is that quantization corrects errors by value clamping, thus avoiding cascaded error accumulation. Moreover, by sparsifying the parameter space, quantization reduces the NN complexity as a regularization mechanism that mitigates potential overfitting issues. Similarly, injecting noises into neural network training is demonstrated to help obtain a smoothed loss landscape for better generalization [19].

3 NOISE-AWARE PQC TRAINING

We use QNN as the benchmark PQC in this work. Figure 2 shows the QNN architecture. The inputs are classical data such as image pixels, and the outputs are classification results. The QNN consists of multiple blocks. Each has three components: encoder encodes the classical values to quantum states with rotation gates such as RY; trainable quantum layers contain parameterized gates that can be trained to perform certain ML tasks; measurement part measures each qubit and obtains a classical value. The measurement outcomes of one block are passed to the next block. For the MNIST-4 example in Figure 2, the first encoder takes the pixels of the down-sampled 4x4 image as rotation angles θ of 16 rotation gates. The measurement results of the last block are passed through a Softmax to output classification probabilities. QuantumNAT overview is in Figure 3.

3.1 Post-Measurement Normalization

Measurement outcome shift due to quantum noises. Before delving into the noise mitigation techniques, we first show analytically how quantum noises influence the QNN block output. The measurement outcomes of the QNN are sensitive to both the input parameters and any perturbations by some noisy quantum process. This section provides insights on such noisy transformations and discusses their impacts on QNN inference.

Theorem 3.1. (informal version). The measurement outcome \( y \) of a quantum neural network for the training input data \( x \) is transformed by the quantum noise that the system undergoes with a linear map \( f(y_x) = y_x + \beta_x \), where the translation \( \beta_x \) depends on the input \( x \) and quantum noises, while scaling factor \( \gamma \) is input independent.

We refer to Appendix Section A.2.2 for background and a complete proof. The main theoretical contribution of this theorem equips our proposed normalization methodology with robustness guarantees. Most importantly, we observe that the changes in measurement results can often be compensated by proper post-measurement normalization across input batches. For simplicity, we restrict our analysis on Z-basis single-qubit measurement outcome \( y \). Similar analytical results for multi-qubit general-basis measurement will follow if we apply the same analysis qubit by qubit. Theorem 3.1 is most powerful when applied on a small batch of input data \( x = \{x_1, \ldots, x_m\} \) where each \( x_i \) is a set of classical input values for the encoder of the QNN and \( m \) is the size of the batch. In an ideal noiseless scenario, the QNN model outputs measurement result \( y_i \) for each input \( x_i \). For a
noisy QNN, the measurement result undergoes a composition of two transformations: (1) a constant scaling by $γ$; (2) a input-specific shift by $β_i$, i.e., $f(y_i) = γy_i + β_i$. In the realistic noise regime, the scaling constant $γ ∈ [−1, 1]$. However, for small noises, $γ$ is close to 1, and $β_i$ is close to 0. Therefore, the distribution of noisy measurement results undergoes a constant scaling by $γ ≤ 1$ and a small shift by each $β_i$. In the small-batch regime when $β = {β_1, … , β_m}$ has small variance, the distribution is shifted by its mean $β = E[β]$. Thus $f(y_i) \approx γy_i + β$.

**Post-measurement normalization.** Based on the analysis above, we propose post-measurement normalization to offset the distribution scaling and shift. For each qubit, we collect its measurement results on a batch of input samples, compute their mean and std., then make the distribution of each qubit across the batch zero-centered and of unit variance. This is performed during both training and inference. During training, for a batch of measurement results: $y = (y_1, … , y_m)$, the normalized results are $\hat{y}_i = (y_i − E(y))/\sqrt{\text{Var}(y)}$. For noisy inference, we correct the error as $\hat{y}_i = (f(y_i) − E(f(y)))/\sqrt{\text{Var}(f(y))} = ((γy_i + β) − (γE[y] + β))/\sqrt{γ^2 \text{Var(y)}} = \hat{y}_i$.

Figure 4 compares the noise-free measurement result distribution of 4 qubits (blue) with their noisy counterparts (yellow) for MNIST-4. Qualitatively, we can clearly observe that the post-measurement normalization reduces the mismatch between two distributions. Quantitatively, we adopt signal-to-noise ratio, $\text{SNR} = \|A\|^2_2/\|A − \hat{A}\|^2_2$, the inverse of relative matrix distance (RMD), as the metric. The SNR on each qubit and each individual measurement outcome is clearly improved. Though similar, it is different from Batch Normalization [10] as the testing batch uses its own statistics instead of that from training, and there is no trainable affine parameter.

### 3.2 Quantum Noise Injection

Although the normalization above mitigates error impacts, we can still observe small discrepancies on each individual measurement outcome, which degrade the accuracy. Therefore, to make the QNN model robust to those errors, we propose noise injection to the training process.

**Quantum error gate insertion.** As introduced in Section 2, different quantum errors can be approximated by Pauli errors via Pauli Twirling. The effect of Pauli errors is the random insertion of Pauli $X$, $Y$, and $Z$ gates to the model with a probability distribution $E$. How to compute $E$ is out of the scope of this work. But fortunately, we can directly obtain it from the realistic device noise model provided by quantum hardware manufacturers such as IBMQ. The noise model specifies the probability $E$ for different gates on each qubit. For single-qubit gates, the error gates are inserted after the original gate. For two-qubit gates, error gates are inserted after the gate on one or both qubits. For example, the $X$ gate on qubit 1 on IBMQ-Yorktown device has $E = \{X: 0.00096, Y: 0.00096, Z: 0.00096, \text{None}: 0.99712\}$. When ‘None’ is sampled, we will not insert any gate. The same gate on different qubits or different hardware will have up 10× probability difference. As in Figure 5, during training, for each QNN gate, we sample error gates based on $E$ and insert it after the original gate. A new set of error gates is sampled for each training step. In reality, the QNN is compiled to the basis gate set of the quantum hardware (e.g., $X$, $\text{CNOT}$, $\text{RZ}$, $\text{CNOT}$, and $\text{ID}$) before performing gate insertion and training. We will also scale the probability distribution by a constant noise factor $T$ and scale the $X$, $Y$, $Z$ probability by $T$ during sampling. $T$ factor explores the trade-off between adequate noise injection and training stability. Typical $T$ values are in the range of $[0.5, 1.5]$. The gate insertion overhead is typically less than $2\%$.

**Readout noise injection.** Obtaining classical values from qubits is referred as readout/measurement, which is also error-prone. The realistic noise model provides the statistical readout error in the form of a $2 × 2$ matrix for each qubit. For example, the qubit 0 of IBMQ-Santiago has readout error matrix $[[0.984, 0.016], [0.022, 0.978]]$ which means the probability of measuring a $0$ as $0$ is $0.984$ and as $1$ is $0.016$. We emulate the readout error effect during training by changing the measurement outcome. For instance, originally $P(0) = 0.9, P(1) = 0.7$, the noise injected version will be $P'(0) = 0.3 \times 0.984 + 0.7 \times 0.022 = 0.31, P'(1) = 0.7 \times 0.978 + 0.3 \times 0.016 = 0.69$. 

![QuantumNAT Overview](image-url)

**Figure 3:** QuantumNAT Overview. (1) Post-measurement normalization matches the distribution of measurement results between noise-free simulation and real QC. (2) Based on realistic noise models, noise-injection inserts quantum error gates to the training process to increase the classification margin between classes. (3) Measurement outcomes are further quantized for denoising.

![SNR of each measurement outcome](image-url)

**Figure 4:** Post-measurement normalization reduces the distribution mismatch between noise-free simulation and noisy results, thus improving the Signal-to-Noise Ratio (SNR).
Compiled Quantum Circuits (Noise-free)

| RX | U3 | RX | U3 |
|----|----|----|----|

Sample Quantum Error Gates (Sample 1)

| RX | U3 | RX | RX | RX |
|----|----|----|----|----|

Sample Quantum Error Gates (Sample 2)

| RX | RX | RX | RX | RX |
|----|----|----|----|----|

Readout Error

**Figure 5:** Noise injection via error gate insertion. X, Y, Z are sampled Pauli error gates. 8 is the injected readout error. Probabilities for gate insertion are obtained from real device noise models.

**Figure 6:** Left: Error maps before and after post-measurement quantization. Most errors can be corrected. Right: 5-level quantization buckets with a quadratic penalty loss.

### Direct perturbation
Besides gate insertion, we also experimented with directly perturbing measurement outcomes or rotation angles as noise sources. For outcome perturbation, with benchmarking samples from the validation set, we obtain the error \( Err \) distribution between the noise-free and noisy measurement results and compute the mean \( \mu_{Err} \) and std \( \sigma_{Err} \). During training, we directly add noise with Gaussian distribution \( N(\mu_{Err}, \sigma_{Err}^2) \) to the normalized measurement outcomes. Similarly, for rotation angle perturbation, we add Gaussian noise to the angles of all rotation gates in QNN and make the effect of rotation angle Gaussian noise on measurement outcomes similar to real QC noise. We show in Section 4 that the gate insertion method is better than direct perturbations.

### 3.3 Post-Measurement Quantization
Finally, we propose post-measurement quantization on the normalized results to further denoise the measurement outcomes. We first clip the outcomes to \([p_{min}, p_{max}]\), where \( p \) are pre-defined thresholds, and then perform uniform quantization. The quantized values are later passed to the next block’s encoder. Figure 6 shows one real example from Fashion-4 on IBMQ-Santiago with five quantization levels and \( p_{min} = -2, p_{max} = 2 \). The left/middle matrices show the error maps between noise-free and noisy outcomes before/after quantization. Most errors can be corrected back to zero with few exceptions of being quantized to a wrong centroid. The MSE is reduced from 0.235 to 0.167, and the SNR is increased from 4.256 to 6.455. We also add a loss term \( ||y - Q(y)||_2^2 \) to the training loss, as shown on the right side, to encourage outcomes to be near to the quantization centroids to improve error tolerance and reduce the chance of being quantized to a wrong centroid. Besides improving robustness, quantization also reduces the control complexity of rotation gates.

### 4 EXPERIMENTS

#### 4.1 Experiment Setups

**Datasets.** We conduct experiments on 8 classification tasks including MNIST [14] 10-class, 4-class (8, 1, 2, 3), and 2-class (3, 6); Vowel 4-class (hid, hld, had, hdd); Fashion [30] 10-class, 4-class (t-shirt/top, trouser, pullover, dress), and 2-class (dress, shirt), and CIFAR [13] 2-class (frog, ship). MNIST, Fashion, and CIFAR use 95% images in ‘train’ split as training set and 5% as the validation set. Due to the limited real QC resources, we use the first 300 images of ‘test’ split as test set. Vowel-4 dataset (990 samples) is separated to train/validation:test = 6:1:3 and test with the whole test set. MNIST and Fashion images are center-cropped to 24×24, and then down-sample to 4×4 for 2- and 4-class, and 6×6 for 10-class; CIFAR images are converted to grayscale, center-cropped to 28×28, and down-sampled to 4×4. All down-samplings are performed with average pooling. For vowel-4, we perform feature principal component analysis (PCA) and take 10 most significant dimensions. **QNN models.** QNN models for 2 and 4-class use 4 qubits; 10-class uses 10. The first quantum block’s encoder embeds images and vowel features. For 4×4 images, we use 4 qubits and 4 layers with 4 RY, 4 RX, 4 RZ, and 4 RY gates in each layer, respectively. There are in total 16 gates to encode the 16 classical values as the rotation angles. For 6×6 images, 10 qubits and 4 layers are used with 10 RY, 10 RX, 10 RZ, and 6 RY gates in each layer, respectively. 10 vowel features, uses 4 qubits and 3 layers with 4 RY, 4 RX, and 2 RZ gates on each layer for encoding. For trainable quantum layers, we use U3 and C3 layers interleaved as in Figure 2 except for Table 2. For measurement, we measure the expectation values on Pauli-2 basis and obtain a value \([-1, 1]\) from each qubit. The measurement outcome goes through post-measurement normalization and quantization and is used as rotation angles for RY gates in the next block’s encoder. After the last block, for two-classifications, we sum the qubit 0 and 1, 2 and 3 measurement outcomes, respectively, and use Softmax to get probabilities. For 4 and 10-class, Softmax is directly applied to measurement outcomes.

**Quantum hardware and compiler configurations.** We use IBMQ quantum computers via Qiskit [8] APIs. We study 6 devices, with qubits from 5 to 15 and Quantum Volume from 8 to 32. We also employ Qiskit for compilation. The optimization level is set to 2 for all experiments. All experiments run 8192 shots. The noise models we used are off-the-shelf ones updated by IBMQ team.

#### 4.2 Main Results

**QNN results.** We experiment with four different QNN architectures on 8 tasks running on 5 quantum devices to demonstrate QuantumNAT’s effectiveness. For each benchmark, we experiment with noise factor \( T = \{0.1, 0.5, 1, 1.5\} \) and quantization level among \([3, 4, 5, 6]\) and select one out of 16 combinations with the lowest loss on the validation set and test on the test set. Normalization and quantization are not applied to the last block’s measurement outcomes as they are directly used for classification. As in Table 1, QuantumNAT consistently achieves the highest accuracy on 26 benchmarks. The third bars of Athens are unavailable due to its retirement. On average, normalization, noise injection and quantization improve accuracy by 10%, 9%, and 3%, respectively. A larger model does not necessarily have higher accuracy. For example, Athens’ model is 7.5× larger than...
Table 1: QuantumNAT consistently achieves the highest accuracy, with on average 22% better. 'B' for Block, 'L' for Layer.

| Model       | Method         | MNIST-4  | Fash.-4  | Vow.-4  | MNIST-2 Fash.-2 Cifar-2 |
|-------------|----------------|----------|----------|---------|-------------------------|
| Baseline    |                | 0.30     | 0.32     | 0.28    | 0.84                    |
| 2B×12L      | + Post Norm.   | 0.41     | 0.61     | 0.29    | 0.87                    |
| Santiago    | + Gate Inser.  | 0.61     | 0.70     | 0.44    | 0.93                    |
|             | + Post Quant.  | 0.68     | 0.75     | 0.48    | 0.94                    |

Table 2: Accuracy on different design spaces.

| Design Space          | MNIST-4 | Fashion-2 |
|-----------------------|---------|-----------|
| Yorktown 'ZZ+RY'      | 0.43    | 0.57      |
| +QuantumNAT           | 0.34    | 0.60      |
| Yorktown 'RXYX'       | 0.57    | 0.61      |
| +QuantumNAT           | 0.61    | 0.70      |
| Yorktown 'ZX+XX'      | 0.29    | 0.51      |
| +QuantumNAT           | 0.38    | 0.64      |
| Yorktown 'RXYX+U1+U3' | 0.28    | 0.25      |
| +QuantumNAT           | 0.33    | 0.21      |

Table 3: Scalable noise-aware training.

| Machine | Bogota | Santiago | Lima |
|---------|--------|----------|------|
| Noise-unaware | 0.74   | 0.97     | 0.87 |
| QuantumNAT   | 0.79   | 0.99     | 0.90 |

Table 4: Compatible with existing noise mitigation.

| Method               | MNIST-4 | Fashion-4 |
|----------------------|---------|-----------|
| Normalization only   | 0.78    | 0.81      |
| Normalization + Extrapolation | 0.81    | 0.83      |

Yorktown with higher noise-free accuracy. However, due to more errors introduced by the larger model, the real accuracy is lower.

Performance on different design spaces. In Table 2, we evaluate QuantumNAT on different QNN design spaces. Specifically, the trainable quantum layers in one block of 'ZZ+RY' [17] space contains one layer of ZZ gate, with ring connections, and one RY layer. 'RXYX' [20] space has five layers: \(V_\theta, RX, RY, RZ, \) and CZ. 'ZZ+XX' [5] space has two layers: ZZ and XX. 'RXYX+U1+U3' [7] space, according to their random circuit basis gate set, has 11 layers in the order of RX, S, CNOT, RY, T, SWAP, RZ, H, \(\sqrt{SWAP}\), U1 and U3. We conduct experiments on MNIST-4 and Fashion-2 on 2 devices. In 13 settings out of 16, QuantumNAT achieves better accuracy. Thus, QuantumNAT is a general technique agnostic to QNN model size and design space.

Table 5: Post-measurement norm. improves acc. & SNR.

| Quantum Devices | QNN Models | 2 Blocks | 4 Blocks |
|-----------------|------------|----------|----------|
|                 | \(x_2\)Layers | \(x_8\)Layers | \(x_2\)Layers | \(x_4\)Layers |
| Santiago        | Acc | SNR | Acc | SNR | Acc | SNR | Acc | SNR |
| Baseline        | 0.61 | 16.59 | 0.79 | 1.48 | 0.70 | 11.36 | 0.68 | 6.55 |
| +Norm           | 0.66 | 13.92 | 0.71 | 2.18 | 0.74 | 12.26 | 0.72 | 4.54 |
| Quito           | Baseline | 0.58 | 6.64 | 0.35 | 1.43 | 0.60 | 5.98 | 0.29 | 1.73 |
|                 | +Norm | 0.66 | 13.92 | 0.71 | 2.18 | 0.74 | 12.26 | 0.72 | 4.54 |
| Athens          | Baseline | 0.59 | 8.91 | 0.60 | 2.14 | 0.63 | 9.52 | 0.55 | 3.54 |
|                 | +Norm | 0.64 | 20.27 | 0.78 | 3.47 | 0.74 | 14.07 | 0.69 | 6.09 |

Scalability. When classical simulation is infeasible, we can move the the noise-injected training to real QC using techniques such as parameter shift [3]. In this case, the training cost is linearly scaled with qubit number. Post-measurement normalization and quantization are also linearly scalable because they are performed on the measurement outcomes. Gradients obtained with real QC are naturally noise-aware because they are directly influenced by quantum noise. To demonstrate the practicality, we train a 2-class task with two numbers as input features [11] (Table 3). The QNN has 2 blocks; each with 2 RY and a CNOT gates. The noise-unaware baseline trains the model on classical part and test on real QC. In QuantumNAT, we train the model with parameter shift and test, both on real QC. We consistently outperform noise-unaware baselines.

Compatibility with existing noise mitigation. QuantumNAT is orthogonal to existing noise mitigation such as extrapolation method. It can be combined with post-measurement normalization (Table 4). The QNN model has 2 blocks, each with three U3+CU3 layers. For "Normalization only", the measurement outcomes of the 3-layer block are normalized across the batch dimension. For "Extrapolation + Normalization", we use extrapolation to estimate the standard deviation of noise-free measurement outcomes. We firstly train the QNN model to convergence and then repeat the 3 layers to 6, 9, 12 layers and obtain four standard deviations of measurement outcomes. Then we linearly extrapolate them to obtain noise-free std. We normalize the measurement outcomes of the 3-layer block to make their std the same as noise-free and then apply the proposed post-measurement norm. Results show that the extrapolation can further improve the QNN accuracy thus being orthogonal.

4.3 Ablation Studies

Ablation on post-measurement normalization. Table 5 compares the accuracy and signal-to-noise ratio (SNR) before and after post-measurement normalization on MNIST-4. We study 4 different QNN architectures and evaluate on 3 devices. The normalization can significantly and consistently increase SNR.

Ablation on different noise injection methods. Figure 7 compares different noise injection methods. Gaussian noise statistics for perturbations are obtained from error benchmarking. The left side shows accuracy without quantization. With different noise factors, the gate insertion and measurement outcome perturbation have similar accuracy, both better than rotation angle perturbation. A possible explanation is that the rotation angle perturbation does not significantly and consistently increase SNR. Gate insertion outperforms perturbation by 11% on average on 3 different devices and QNN models. The reason is: directly added perturbation on measurement outcomes can be easily canceled by quantization, and thus it is harder for noise injection to take effect.
Combining noise injection & quantization improve accuracy. Measured Accuracy.

Table 2

| Norm | Norm + Noise Injection | Norm + Quant | Norm + Noise Injection + Quant |
|------|------------------------|-------------|--------------------------------|
| MNIST-4 Santiago | 0.4133333333 | 0.41 | 0.41 |
| Vowel-4 Santiago | 0.49 | 0.49 | 0.49 |
| MNIST-2 Yorktown | 0.72 | 0.72 | 0.72 |
| Fashion-2 Yorktown | 0.74 | 0.74 | 0.74 |
| Average | 0.64 | 0.64 | 0.64 |

Figure 8: Left: Accuracy contours of quantization levels and noise factors. Right: Feature visualization.

Figure 9: Ablation of applying noise injection and quantization individually or jointly.

Noise factor and post-measurement quantization level analysis. We visualize the QNN accuracy contours on Fashion-4 on IBMQ-Athens with different noise factors and quantization levels in Figure 8 left. The best accuracy occurs for factor 0.2 and 5 levels. Horizontal-wise, the accuracy first goes up and then goes down. This is because too few quantization levels hurt the QNN model capacity; too many levels cannot bring sufficient denoising effect. Vertical-wise, the accuracy also goes up and then down. Reason: when the noise is too small, the noise-injection effect is weak, thus cannot improve the model robustness; while too large noise makes the training process unstable and hurts accuracy.

Visualization of QNN extracted features. MNIST-2 classification result is determined by which feature is larger between the two: feature one is the sum of measurement outcomes of qubit 0 and 1; feature 2 is that of qubit 2 and 3. We visualize the two features obtained from experiments on Belem in a 2-D plane as in Figure 8 right. The blue dash line is the classification boundary. The circles/stars are samples of digit ‘3’ and ‘6’. All the baseline points (yellow) huddled together, and all digit ‘3’ samples are misclassified. With normalization (green), the distribution is significantly expanded, and the majority of ‘3’ is correctly classified. Finally, after noise injection (red), the margin between the two classes is further enlarged, and the samples are farther away from the classification boundary, thus becoming more robust.

Breakdown of accuracy gain. Figure 9 shows the performance of only applying noise-injection, only applying quantization, and both. Using two techniques individually can both improve accuracy by 9%. Combining two techniques delivers better performance with a 17% accuracy gain. This indicates the benefits of synergistically applying three techniques in QuantumNAT.

5 CONCLUSION

PQC is a promising candidate to demonstrate practical quantum advantages over classical approaches. The road to such advantage relies on: (1) the discovery of novel feature embedding that encodes classical data non-linearly, and (2) overcome the impact of quantum noise. This work focuses on the latter and show that a noise-aware training pipeline with post-measurement normalization, noise injection, and post-measurement quantization can elevate the PQC robustness against arbitrary, realistic quantum noises. We anticipate such robust PQC being useful in exploring practical QC applications.

ACKNOWLEDGMENT

We acknowledge NSF CAREER Award #1943349, MIT-IBM Watson AI Lab, Qualcomm Innovation Fellowship. This work is funded in part by EPIQC, an NSF Expedition in Computing, under grants CCF-1730082/1730449; in part by STAQ under grant NSF Phy-1818914; in part by DOE grants DE-SC0020829 and DE-SC0820331; and in part by NSF GMA-2016136 and the Q-NEXT DOE NQI Center. We acknowledge the use of IBM Quantum services for this work.

A APPENDIX

A.1 Quantum Basics and Quantum Noise

A quantum circuit uses quantum bit (qubit) to carry information, which is a linear combination of two basis state: $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$. For $\alpha, \beta \in \mathbb{C}$, satisfying $|\alpha|^2 + |\beta|^2 = 1$. An n-qubit system can represent a linear combination of $2^n$ basis states. A 2-length complex statevector of all combination coefficients is used to describe the circuit state. In quantum computations, a sequence of quantum gates are applied to perform unitary transformation on the statevector, i.e., $|\psi(x, \theta)\rangle = \cdots U_2(x, \theta_2) U_1(x, \theta_1) |0\rangle$, where $x$ is the input data and $\theta$ is the trainable parameters of rotation quantum gates. As such, the input data and trainable parameters are embedded in the quantum state $|\psi(x, \theta)\rangle$. Finally, the computation results are obtained by qubit readout/measurement which measures the probability of a qubit state $|\psi\rangle$ collapsing to either $|0\rangle$ (i.e., output $y = +1$) or $|1\rangle$ (i.e., output $y = -1$) according to $|\alpha|^2$ and $|\beta|^2$. With sufficient samples, we can compute the expectation value: $\mathbb{E}[y] = (+1)|\alpha|^2 + (-1)|\beta|^2$. By cascading multiple blocks of quantum
gates and measurements, a non-linear network can be constructed to perform ML tasks.

In real quantum computer systems, errors would likely occur due to imperfect control signals, unwanted interactions between qubits, or interference from the environment [2, 12]. As a result, qubits undergo decoherence error (spontaneous loss of its stored information) over time, and quantum gates introduce operation errors (e.g., coherent errors and stochastic errors) into the system. These noisy systems need to be characterized [18] and calibrated [9] frequently to mitigate the impact of noise on computation. Noise modeling helps to paint a realistic picture of the behavior and performance of a quantum computer and enables noisy simulations [4]. While exact modeling and simulation is challenging, many approximate strategies [18, 24] have been developed based on Pauli/Clifford Twirling [21, 22].

A.2 General Framework for Quantum Noise Analysis

In this work, we examine how to characterize and mitigate the impact of quantum noises on quantum neural networks. We observe that the trainable quantum gates and post-measurement information processing play a huge role in boosting the algorithmic robustness to realistic quantum noises. In the following analysis, we restrict attention to: (1) a general (mixed) quantum state $\rho(x, \theta)$ resulting from a QNN for input data $x$ and trainable parameters $\theta$, (2) single-qubit measurement output, and (3) any fixed but unknown quantum noise. For multi-qubit quantum neural networks, similar analysis follows when considered qubit by qubit.

A.2.1 Measurement of Quantum Neural Networks.

Definition A.1. (Measurement procedure). We measure a quantum state $\rho$ in the computational basis $\{b\}$ : $b \in \{0, 1\}$ and output $z = +1$ if we obtain $|0\rangle \langle 0|$ and $z = -1$ if we obtain $|1\rangle \langle 1|$. The expectation value of such measurement contains useful information about the quantum state $\rho$:

$$E_Z \equiv \mathbb{E}[z] = \text{tr}(Z \rho),$$

where $Z$ is the Pauli-Z matrix: $Z = (+1) |0\rangle \langle 0| + (-1) |1\rangle \langle 1|$ and $\text{tr}(\cdot)$ is the trace. We can estimate the expectation value by repeating the experiment by $s$ times, obtaining $z_1, \ldots, z_s$, with each $z_j \in \{+1, -1\}$, and calculate their empirical mean: $y = \sum_{j=1}^{s} z_j / s$. Throughout this work, we use $s = 8192$ shots for the experiments to keep the variance.

Definition A.2. (Noise processes). A physical process (such as quantum noises) that can happen to a mixed quantum state $\rho$ can be described as a linear map: $\rho \rightarrow E(\rho)$, such that

$$E(\rho) = \sum_k O_k \rho O_k^\dagger.$$

The $O_k$'s are Kraus operators satisfying $\sum_k O_k^\dagger O_k = I$. The noise process for a quantum neural network can be challenging to characterize, as it depends not only on the input to the network but also on the qubits and quantum gates used in the network. We wish to analyze this noise process by decoupling its dependence on the input data. We assume that each $O_k$ has no explicit dependence on the classical input data; this is reasonable when we fix the model architecture.

A.2.2 Proof of Theorem 3.1. Now we are ready to analyze the effect of quantum noises on the measurement result from a quantum neural network. For classical data $x_i$ from the input data set $x$, we construct a quantum neural network that embeds the classical data in the quantum state $\rho_i = \rho(x_i, \theta_i)$ where $\theta_i$ is some training parameters. The output of the network is the expectation value of the measurement outcome $E_{Z,i} \equiv \text{tr}(Z \rho_i)$. However, in reality, the results are transformed by some unknown process $\rho_i \rightarrow E(\rho_i)$. The goal is to quantify the impact of the quantum noise on the expectation value.

Theorem 3.1. (formal version). There exists some real parameters $\beta_i$ and $y_i$ such that the expectation value of the measurement results $E_{Z,i}$ for input data $x_i$ with the presence of any valid quantum noise $E_i(\rho)$ can be described as a linear map from the noiseless value $E_{Z,i}^*$:

$$E_{Z,i} = y E_{Z,i}^* + \beta_i,$$

where $y$ is a scaling constant independent of the input data $x_i$.

Proof. Suppose in the noiseless scenario the quantum state obtained from a quantum neural network is denoted as $\rho$, and the expectation value of its measurement results is $E_Z = \mathbb{E}[\rho] = \text{tr}(\rho Z)$. Assume now the $\rho$ undergoes some quantum noise processes $E_i(\rho) = \sum_k O_k \rho O_k^\dagger$. Therefore, in the presence of noise, the expectation value becomes:

$$E_i = \mathbb{E}[E_i(\rho)] = \text{tr}(E_i(\rho)Z) = \sum_k \text{tr}(O_k \rho O_k^\dagger Z) = \sum_k \text{tr}(\rho O_k^\dagger Z O_k),$$

where the third and fourth equality is from the properties of trace. We can further utilize the fact that an arbitrary quantum state can be expanded as:

$$\rho = \frac{1}{2} \left( \text{tr}(\rho) I + \text{tr}(X \rho) X + \text{tr}(Y \rho) Y + \text{tr}(Z \rho) Z \right).$$

If we denote $\Omega = \sum_k O_k^\dagger Z O_k$, we obtain

$$E_i = \frac{1}{2} \text{tr}(\Omega) + \frac{1}{2} \text{tr}(X \Omega) \text{tr}(X \rho) + \frac{1}{2} \text{tr}(Y \Omega) \text{tr}(Y \rho) + \frac{1}{2} \text{tr}(Z \Omega) \text{tr}(Z \rho).$$

Notice that $\text{tr}(\Omega) = 0$ and $\text{tr}(Z \rho) = E^*_Z$. We can set $y = \frac{1}{2} \text{tr}(\Omega) \in [-1, 1]$ and $\beta_i = \frac{1}{2} \text{tr}(X \Omega) \text{tr}(X \rho) + \frac{1}{2} \text{tr}(Y \Omega) \text{tr}(Y \rho)$. We arrive at the linear map as desired. □

A.3 Additional Experiments

A.3.1 Importance of hardware-specific noise model. We train three QNN models for Fashion-2 with the same architecture but different noise models from 3 devices and then deploy each model. Results in Table 6 show a diagonal pattern: the best accuracy is achieved when the noise model and inference device are the same. This is due to various noise magnitude and distribution on different devices. For instance, the gate error of Yorktown is 5× larger than Santiago, so using Yorktown noise information for model running on Santiago is too large. Therefore, a hardware-specific noise model is necessary for proper noise injection. However, this also marks the limitation of this work, as repeated training may be required when the noise model is updated. A future direction is to explore how to finetune already trained QNN for fast adaption to a new noise setting, thus reducing the marginal cost.

Table 6: Hardware-specific noise model can achieve best accuracy than other settings.

| Method               | Santiago | Yorktown | Lima   |
|----------------------|----------|----------|--------|
| Baseline             | 0.68     | 0.83     | 0.83   |
| +Norm                | 0.87     | 0.86     | 0.91   |
| +Noise & Quant       | 0.92     | 0.92     | 0.91   |

Table 7: MNIST-2 accuracy with noise-adaptive compilation enabled (Qiskit optimization level=3).

| Method               | Santiago | Yorktown | Belem   | Athens  |
|----------------------|----------|----------|---------|---------|
| Baseline             | 0.68     | 0.83     | 0.83    | 0.54    |
| +Norm                | 0.87     | 0.86     | 0.91    | 0.51    |
| +Noise & Quant       | 0.92     | 0.92     | 0.91    | 0.93    |
Therefore, using the statistics of validation set can bring similar accuracy to using statistics of test set itself; thus the QuantumNAT can support small test batch size using validation set stats.

A.4 Hyperparameters for main results

Table 14 shows the detailed noise factor and quantization level for all the tasks in Table 1.

REFERENCES

[1] Mohammad H Amin, Evgeny Andriyash, Jason Rolle, Bohdan Kukhtynsky, and Robert M. Machko. 2018. Quantum boltzmann machine. Physical Review X 8, 2 (2018).
[2] Colin D Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M Sage. 2019. Trapped-ion quantum computing: Progress and challenges. Applied Physics Reviews 6, 2 (2019), 021314.
[3] Kevin E Crooks. 2019. Graduands of parameterized quantum gates using the parameter-shift rule and gate decomposition. arXiv preprint arXiv:1905.13311 (2019).
[4] Yongshan Ding and Frederic T Chong. 2020. Quantum computer systems: Research for noisy intermediate-scale quantum computers. Synthesis Lectures on Computer Architecture 15, 2 (2020), 1–227.
[5] Edward Farhi and Hartmut Neven. 2018. Classification with quantum neural networks on near term processors. arXiv preprint arXiv:1802.06002 (2018).
[6] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00419 (2015).
[7] Maxwell Henderson, Samriddhi Shyaka, Shashindra Pradhan, and Tristan Cook. 2020. Quantum neural networks: powering image recognition with quantum circuits. Quantum Machine Intelligence 2, 1 (2020), 1–9.
[8] IBM. [n. d.]. IBM Quantum. (n. d.). https://quantum-computing.ibm.com/
[9] Quiskit QMIB. 2021. https://qiskit.org/textbook/ch-quantum-hardware/calibrating-qubits-pulse.html
[10] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML. PMLR, 448–456.
[11] Weifen Jiang, Jinjun Xiong, and Yiyu Shi. 2011. A co-design framework of neural networks and quantum circuits towards quantum advantage. Nature Communications 12, 1 (2021), 1–13.
[12] Philip Krantz, Morten Kjaergaard, Feri Yan, Tery P Orlando, Simon Gustavsson, and William D Oliver. 2019. A quantum engineer’s guide to superconducting qubits. Applied Physics Reviews 6, 2 (2019), 021318.
[13] Alex KRizhevsky, Vinod Nair, and Geoffrey Hinton. [n. d.]. CIFAR-10 (Canadian Institute for Advanced Research). (n. d.).
[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 11 (1998), 2278–2324.
[15] Zhidong Liang, Zhepeng Wang, Junhuan Yang, Lei Yang, Yiyu Shi, and Weifen Jiang. 2021. Can Noise on Qubits Be Learned in Quantum Neural Networks? A Case Study on QuantumFlow. ICCAD. IEEE, 1–7.
[16] Ji Lin, Chuan Gan, and Song Han. 2018. Training quantized neural networks via gradient-based learning applied to document recognition. Proc. IEEE 136, 1 (2018), 2278–2324.
[17] Seth Lloyd, Maria Schuld, Aroosa Ijaz, Josh Izaac, and Nathan Killoran. 2020. Quantum embeddings for machine learning. arXiv preprint arXiv:2003.06422 (2020).
[18] Eswar Magesan, Jay M Gambetta, and Joseph Emerson. 2012. Characterizing quantum gates via randomized benchmarking. Physical Review A 85, 4 (2012), 042311.
[19] K. Matsuoka. 1992. Noise injection into inputs in back-propagation learning. IEEE Transactions on Systems, Man, and Cybernetics 22, 3 (1992), 436–440.
[20] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. 2018. barren plateaus in quantum neural network training landscapes. Nature Communications 9, 1 (2018), 1–6.
[21] Michael A Nielsen and Isaac Chuang. 2002. Quantum computation and quantum information.
[22] Marcus Silva, Eswar Magesan, David W Kribs, and Joseph Emerson. 2008. Scalable protocol for identification of correctable codes. Physical Review A 78, 1 (2008), 012347.
[23] Kristian Temme, Sergey Bravyi, and Jay M Gambetta. 2017. Error mitigation for short-depth quantum circuits. Physical review letters 119, 18 (2017), 180509.
[24] Joel J Wallman and Joseph Emerson. 2016. Noise tailoring for scalable quantum computation via randomized compiling. Physical Review A 94, 5 (2016), 052353.
Table 8: Effect of QuantumNAT on fully quantum models.

| IBMQ Machine | Model       | Method    | MNIST-4 | Fashion-4 | Vowel-4 | MNIST-2 | Fashion-2 | Cifar-2 |
|--------------|-------------|-----------|---------|-----------|---------|---------|-----------|---------|
| Santiago     | 3 Layer     | Baseline  | 0.64    | 0.78      | 0.41    | 0.94    | 0.89      | 0.59    |
|              |             | QuantumNAT| 0.78    | 0.82      | 0.53    | 0.96    | 0.90      | 0.58    |
| Santiago     | 6 Layer     | Baseline  | 0.61    | 0.37      | 0.22    | 0.51    | 0.52      | 0.52    |
|              |             | QuantumNAT| 0.62    | 0.69      | 0.22    | 0.84    | 0.89      | 0.56    |
| Yorktown     | 3 Layer     | Baseline  | 0.49    | 0.53      | 0.4     | 0.88    | 0.85      | 0.51    |
|              |             | QuantumNAT| 0.55    | 0.66      | 0.42    | 0.9     | 0.91      | 0.55    |
| Yorktown     | 6 Layer     | Baseline  | 0.22    | 0.33      | 0.26    | 0.73    | 0.8        | 0.54    |
|              |             | QuantumNAT| 0.42    | 0.35      | 0.25    | 0.78    | 0.8        | 0.52    |
| Belem        | 3 Layer     | Baseline  | 0.53    | 0.6        | 0.37    | 0.64    | 0.81      | 0.51    |
|              |             | QuantumNAT| 0.58    | 0.42      | 0.39    | 0.93    | 0.85      | 0.55    |
| Belem        | 6 Layer     | Baseline  | 0.27    | 0.18      | 0.21    | 0.54    | 0.48      | 0.43    |
|              |             | QuantumNAT| 0.43    | 0.31      | 0.22    | 0.54    | 0.54      | 0.52    |

Table 9: Effect of number of intermediate measurements.

| Task      | 1 Block × 6 Layers | 2 Blocks × 3 Layers | 3 Blocks × 2 Layers | 6 Blocks × 1 Layer |
|-----------|--------------------|--------------------|--------------------|--------------------|
| MNIST-4   | 0.62               | 0.74               | 0.71               | 0.66               |
| Fashion-4 | 0.69               | 0.82               | 0.78               | 0.68               |

Table 10: Direct comparison between QNN models with measurement in between and fully-quantum QNN models.

| Machine | Task     | Fully-Quantum (6L) | Original (2B × 3L) |
|---------|----------|--------------------|--------------------|
| Santiago| MNIST-4  | 0.62               | 0.74               |
| Santiago| Fashion-4| 0.69               | 0.82               |
| Santiago| MNIST-2  | 0.84               | 0.86               |
| Belem   | MNIST-4  | 0.43               | 0.37               |
| Belem   | Fashion-4| 0.31               | 0.34               |
| Belem   | MNIST-2  | 0.54               | 0.60               |

[25] Hanrui Wang, Yongshan Ding, Jiaqi Gu, Yujun Lin, David Z Pan, Frederic T Chong, and Song Han. 2022. QuantumNAS: Noise-adaptive search for robust quantum circuits. *HPCA* (2022).

[26] Hanrui Wang, Zirui Li, Jiaqi Gu, et al. 2022. QOC: Quantum On-Chip Training with Parameter Shift and Gradient Pruning. *DAC* (2022).

[27] Hanrui Wang, Zhekai Zhang, and Song Han. 2021. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In *HPCA*. IEEE, 97–110.

[28] Zhepeng Wang, Zhiding Liang, Shanglin Zhou, et al. 2021. Exploration of Quantum Neural Architecture by Mixing Quantum Neuron Designs. In *ICCAD*. IEEE.

[29] Robert Wille, Lukas Burgholzer, and Alwin Zulehner. 2019. Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations. In *DAC*. IEEE, 1–6.

[30] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv* 1708.07747 (2017).
### Table 11: Accuracy gap between evaluation using noise model and real QC.

| Machine | Model Method | Method | MNIST-4 | Fashion-4 | Vowel-4 | MNIST-2 | Fashion-2 | Cifar-2 |
|---------|--------------|--------|---------|-----------|---------|---------|-----------|---------|
| Santiago | 2 Blocks × 12 Layer | Noise model | 0.73 | 0.74 | 0.51 | 0.95 | 0.92 | 0.65 |
|            |               | Real QC | 0.68 | 0.75 | 0.48 | 0.94 | 0.88 | 0.59 |
| Yorktown  | 2 Blocks × 2 Layer | Noise model | 0.68 | 0.7 | 0.44 | 0.92 | 0.9 | 0.59 |
|            |               | Real QC | 0.62 | 0.65 | 0.44 | 0.93 | 0.86 | 0.6 |
| Belem     | 2 Blocks × 6 Layer | Noise model | 0.64 | 0.72 | 0.41 | 0.96 | 0.82 | 0.64 |
|            |               | Real QC | 0.58 | 0.62 | 0.41 | 0.88 | 0.8 | 0.61 |

### Table 12: Improvements are still significant as the number of classes increases.

| Task           | Average Accuracy | Baseline | QuantumNAT | Absolute Improvement | Relative Improvement |
|----------------|------------------|----------|------------|----------------------|----------------------|
| 2-classification | 0.58             | 0.76     | 0.28       | 48%                  |
| 4-classification | 0.31             | 0.57     | 0.26       | 84%                  |
| 10-classification| 0.1              | 0.33     | 0.23       | 230%                 |

### Table 13: Statistics of test and validation set; Accuracy of test set using test stats and validation stats.

| Task          | Stats MEAN | Stats STD | Accuracy |
|---------------|------------|-----------|----------|
| Fashion-4-Santiago | Test Stats [0.0469, 0.0025, -0.0581, -0.0191] | Valid Stats [0.0679, 0.0025, -0.0519, -0.0473] | 0.75 |
| Fashion-4-Yorktown | Test Stats [-0.0396, 0.0478, 0.0995, 0.1375] | Valid Stats [0.1279, 0.3368, 0.1761, 0.1538] | 0.65 |
| Fashion-4-Belem  | Test Stats [0.1118, 0.0075, 0.0901, -0.0005] | Valid Stats [0.1508, 0.0130, 0.0533, 0.0478] | 0.62 |
| Vowel-4-Santiago | Test Stats [0.1091, 0.0526, 0.0290, 0.2172] | Valid Stats [0.0551, 0.0260, 0.0554, 0.0422] | 0.48 |
| Vowel-4-Yorktown | Test Stats [0.0900, -0.3700, -0.2524, 0.1645] | Valid Stats [0.0997, 0.0580, 0.0663, 0.1198] | 0.44 |
| Vowel-4-Belem   | Test Stats [0.0841, -0.3869, -0.2948, 0.1736] | Valid Stats [0.0946, 0.0651, 0.0615, 0.1199] | 0.41 |
| MNIST-2-Santiago | Test Stats [-0.0581, -0.0657, 0.0088, 0.0170] | Valid Stats [0.0737, 0.1090, 0.1561, 0.1351] | 0.94 |
| MNIST-2-Yorktown | Test Stats [0.0892, -0.0007, 0.0548, 0.0485] | Valid Stats [0.1281, 0.3501, 0.2100, 0.2975] | 0.93 |
| MNIST-2-Belem   | Test Stats [-0.0549, 0.1949, 0.0540, 0.1313] | Valid Stats [0.0856, 0.1137, 0.1553, 0.1688] | 0.88 |
| Average        | Test Stats — | Valid Stats — | 0.67 |

### Table 14: Hyperparameters of Table 1.

| Task, (noise-factor, quantization level) | MNIST-4 | Fashion-4 | Vowel-4 | MNIST-2 | Fashion-2 | Cifar-2 |
|------------------------------------------------|---------|-----------|---------|---------|-----------|---------|
| QNN (2 Blocks × 12 Layers) on Santiago         | (1, 3)  | (0.5, 6)  | (0.5, 6) | (1, 4)  | (1, 6)    | (0.5, 6) |
| QNN (2 Blocks x 2 Layers) on Yorktown          | (0.5, 6) | (1, 5)    | (0.1, 5) | (0.5, 5) | (0.1, 6)  | (0.1, 3) |
| QNN (2 Blocks x 6 Layers) on Belem             | (0.5, 5) | (1.5, 6)  | (0.5, 5) | (0.1, 4) | (0.1, 4)  | (0.5, 6) |
| QNN (3 Blocks x 10 Layers) on Athens           | (0.1, 6) | (0.1, 5)  | (0.5, 6) | (0.1, 6) | (0.5, 6)  | (0.1, 6) |

| Task, (noise-factor, quantization level) | MNIST-10 | Fashion-10 |
|-----------------------------------------|----------|-------------|
| QNN (2 Blocks x 2 Layers) on Melbourne   | (0.1, 6) | (0.1, 5)    |