

ϵ -DQML: A Framework for Distributed Quantum Machine Learning over Noises and Link Failures

Danhua Zhao

The University of Texas at Arlington
Arlington, Texas, USA
danhua.zhao@uta.edu

Jiayi Meng

The University of Texas at Arlington
Arlington, Texas, USA
jiayi.meng@uta.edu

Xiaojun Shang

The University of Texas at Arlington
Arlington, Texas, USA
xiaojun.shang@uta.edu

Abstract

Quantum machine learning (QML) offers a promising approach to learning complex, high-dimensional data distributions. However, scaling QML beyond a single quantum machine requires distributed quantum computing, which introduces additional noise and link failures through quantum communication protocols such as quantum gate teleportation (QGT). In this paper, we present ϵ -DQML, a simulation-based framework for studying distributed QML under QGT-induced noise and failures. ϵ -DQML models the state-of-the-art QGT protocol and its induced noise/failure, and supports efficient noise/failure-aware distributed QML training through a hybrid differentiation method. Preliminary studies on contrastive language-image pretraining and network traffic classification show that the impact of QGT-induced noise is task- and noise-level-dependent, motivating further study on noise/failure-aware distributed QML system design.

CCS Concepts

• **Computer systems organization** → **Quantum computing**; • **Computing methodologies** → *Machine learning; Modeling and simulation*; • **Networks** → *Network reliability*.

Keywords

distributed quantum machine learning, quantum gate teleportation, quantum networks, noise modeling, link failures

ACM Reference Format:

Danhua Zhao, Jiayi Meng, and Xiaojun Shang. 2026. ϵ -DQML: A Framework for Distributed Quantum Machine Learning over Noises and Link Failures. In *ACM SIGCOMM Workshop on Quantum Networks and Distributed Quantum Computing (QuNet '26)*, August 17–21, 2026, Denver, CO, USA. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3833409.3833439>

1 Introduction

Machine Learning (ML) is widely regarded as one of the most promising and impactful domains for unlocking the practical advantages of quantum computing over classical computing, an emerging area known as *quantum machine learning (QML)*. Specifically, quantum computing provides access to exponentially large Hilbert spaces and supports expressive modeling of complex, high-dimensional distributions needed by ML at scale [4].

Despite its promise, there remain fundamental, system-level challenges in fully harnessing the power of QML. In particular, it is infeasible to train a large-scale QML model on a *single* quantum computer. This is fundamentally constrained by the limited number of available qubits. Inspired by classical distributed computing, *distributed* QML represents a new paradigm. It enables the scaling of the computational power required for training large QML models by interconnecting multiple quantum machines over a shared network.

However, such a paradigm introduces network-related noise and failures. They are mainly due to the fragility of shared quantum gates and states across qubits over different quantum machines. Existing work on noisy QML mostly focuses on a single quantum machine. On one hand, sufficiently strong noise can cause QML training to fail by exponentially suppressing the gradients of a loss function [21]. On the other hand, noise can prevent overfitting and improve QML generalization [10, 22]. There exist limited studies on the impact of network-induced noise/failures on QML training under distributed settings.

In this paper, we propose and develop a simulation-based framework, ϵ -DQML, that models quantum gate teleportation (QGT) along with its induced noise and failures to study their impact on the training of distributed QML. QGT is a protocol that efficiently implements remote entangling gates between qubits in separate quantum machines. It consumes only one Bell pair and exchanges two classical bits [1, 6]. Existing simulation frameworks either focus on non-distributed QML without considering QGT-induced noise and failures [2, 3, 8], or accurately model such noise and failures but lack support for QML training and inference [5].

Modeling QGT-induced noise and failures for distributed QML faces several challenges. First, the state-of-the-art experimental demonstration of QGT [11] validates a single remote entangling gate (CZ). However, QML circuits also involve parameterized remote controlled gates (e.g., $CR_X(\theta)$). Second, the steps within QGT (e.g., entanglement generation and local gate operations) each introduce different noise and link failures. These hardware-level noise characteristics need to be precisely modeled in simulation. Third, faithful simulation requires independently sampling noise for forward and backward processes for QML training. However, existing methods either do not support this (e.g., adjoint differentiation [9]) or incur significant computational overhead (e.g., the parameter-shift rule [12, 16]).

To address these challenges, we design ϵ -DQML with two modular components, *i.e.*, QGT modeling and QGT-induced noise/failure modeling. This modular design enables the framework to be easily extended to other QGT implementations and noise/failure models in the future. Specifically, for the QGT modeling module, we first follow the QGT protocol experimentally demonstrated in [11] to develop a realistic and efficient simulation sub-circuit for remote



This work is licensed under a Creative Commons Attribution 4.0 International License. *QuNet '26*, Denver, CO, USA

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2892-1/26/08

<https://doi.org/10.1145/3833409.3833439>

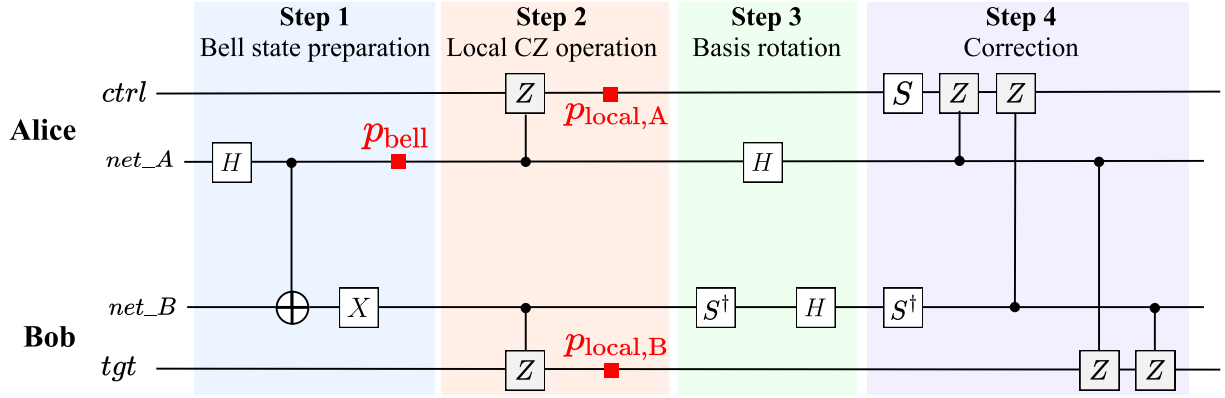


Figure 1: Our designed QGT sub-circuit for a remote CZ gate [11] between Alice’s and Bob’s circuit qubits (ctrl and tgt), using network qubits (net_A and net_B). Red markers indicate the three primary locations for modeling QGT-induced depolarizing noise with probability p_{bell} , $p_{\text{local},A}$, and $p_{\text{local},B}$, respectively.

CZ gates. Then we use the developed sub-circuit as a key building block to model other remote controlled gates, *e.g.*, CR_X , based on the controlled-unitary decomposition theorem [14].

For the noise and failure modeling, we apply the widely-adopted depolarizing noise model to the sub-circuit for the QGT protocol, following the experimentally measured gate fidelities reported in [11]. We also adopt a try-until-success strategy and develop a mask-based mechanism to model failures during the entanglement preparation of the protocol. Unlike existing frameworks [2, 8], we sample noise/failures in the forward and backward processes of QML training separately to capture the stochastic nature of QML training on real quantum computers. However, such sampling requires the computationally intensive parameter-shift rule [12, 16]. To accelerate the training, we develop a hybrid differentiation method. It applies the parameter-shift rule only to the parameters of remote gates, while using efficient adjoint differentiation to approximate the gradients for local gates.

We conduct a preliminary study on the effectiveness of ϵ -DQML with two practical workloads, contrastive language-image pretraining (CLIP) and network traffic classification (NTC), which represent emerging QML applications in multimodal learning and networking-oriented classification. Our results show that the impact of QGT-induced noise depends on both the task and the noise level. For CLIP, a small amount of noise improves zero-shot accuracy over the noiseless baseline, while stronger noise degrades both convergence and accuracy. For NTC, the model maintains near-baseline accuracy under moderate noise but suffers a substantial performance drop under excessive noise. We plan to further characterize these task- and noise-level-dependent behaviors and investigate their implications for noise-aware distributed QML system designs in our future work.

2 Background

QGT. To transmit quantum information across different quantum machines, quantum teleportation provides an effective interface that relies only on bipartite entanglement (*e.g.*, Bell states) shared between machines, together with local operations and classical communication. Building upon this interface, QGT is a protocol that

efficiently implements a remote gate between two distant qubits using pre-shared entanglement and classical communication [6, 7].

The state-of-the-art QGT protocol [11] focuses on a controlled-Z (CZ) gate between two remote circuit qubits across an optical network link. Its implementation includes five dedicated steps (①–⑤).
 ① **Entanglement generation:** A maximally entangled Bell state $|\Psi^+\rangle$ is prepared between two network qubits on two quantum machines (denoted as Alice and Bob) via the photonic link.
 ② **Local entangling operation:** Alice and Bob each perform a local CZ operation between its network qubit and its circuit qubit.
 ③ **Mid-circuit measurement:** Both network qubits are measured in the X and Y bases for Alice and Bob, respectively, collapsing the entanglement into classical information.
 ④ **Classical communication:** The classical information (*i.e.*, measurement outcomes) is then exchanged between Alice and Bob via a classical communication link.
 ⑤ **Feed-forward correction:** The circuit qubits of Alice and Bob perform conditional single-qubit corrections of byproducts based on the measurement.

QML Training. In QML, training parameterized quantum circuits relies on efficiently computing gradients of a loss function with respect to circuit parameters. There are two widely used gradient evaluation algorithms: (1) The **parameter-shift rule** [12, 16] calculates gradients with respect to a *single* circuit parameter by evaluating the circuit at shifted values, $+\pi/2$ and $-\pi/2$, of the parameter. While it is quantum hardware-friendly, its computational overhead scales proportionally with the number of trainable parameters. In contrast, (2) the **adjoint method** [9] significantly reduces the computational complexity of gradient evaluation in simulation environments. It computes quantum gradients on state-vector simulators by first performing a forward pass to compute and store intermediate quantum states throughout the circuit, and then propagating adjoint states backward through the circuit’s unitary operations for all the parameters.

3 Design of ϵ -DQML

As there exist limited studies on the impact of QGT-induced noise/failures on distributed QML training, we develop ϵ -DQML, a simulation framework for studying these effects.

ϵ -DQML Overview. Our developed framework comprises two key modules: QGT modeling (§3.1) and QGT-induced noise and link failure modeling (§3.2). For the QGT protocol, we adopt the state-of-the-art, physical implementation in [11], which is experimentally verified with complete per-step noise calibration data. Our modeling of QGT and its noise and failures is designed in a *modular* way to support other teleportation protocols and noise/failure models in the future. To accelerate noise/failure-aware QML training, we also propose a hybrid differentiable training method (§3.3).

3.1 QGT Modeling

Modeling Remote CZ Gate. Since the QGT protocol in [11] is implemented on trapped-ion hardware using mixed-species gates and photonic links, for which existing simulation frameworks provide no direct abstraction, we first develop a sub-circuit to emulate its remote CZ gate.

Figure 1 shows the four steps (❶–❹) of our sub-circuit. The first two steps are similar to ❶ and ❷ in the physical implementation: **❶ Bell state preparation:** Following the QGT protocol physically realized in [11], gate teleportation between Alice and Bob involves two circuit qubits, i.e., the original data qubits acted on by the remote CZ gate, where ctrl denotes Alice’s circuit qubit serving as the control qubit of the remote gate and tgt denotes Bob’s circuit qubit serving as the target qubit of the remote gate. It also requires two auxiliary network qubits (net_A and net_B) to mediate gate teleportation. We establish entanglement between the network qubits by preparing the Bell state $|\Psi^+\rangle = (|01\rangle + |10\rangle)/\sqrt{2}$ with the gate sequence $H(\text{net_A}) \cdot \text{CNOT}(\text{net_A}, \text{net_B}) \cdot X(\text{net_B})$. **❷ Local CZ operation:** Each quantum machine performs a local CZ gate between their respective network and circuit qubits, i.e., $CZ(\text{net_A}, \text{ctrl})$ and $CZ(\text{net_B}, \text{tgt})$.

However, consistent with the physical QGT implementation, steps ❶ and ❷ introduce Pauli byproducts as an intrinsic consequence of Bell-state-mediated remote gate operations. In the physical protocol, these byproducts are corrected through mid-circuit measurements and feed-forward operations in steps ❸–❹. However, such mid-circuit measurements are non-unitary. Directly simulating non-unitary operations generally requires a density-matrix representation, which stores 2^{2n} complex numbers for an n -qubit system and is therefore substantially more memory-intensive than state-vector simulation of unitary circuits, which requires only 2^n complex numbers. To efficiently correct the byproducts in simulation, we apply the deferred measurement principle [14] to redesign steps ❸–❹ into ❸ and ❹. This keeps the entire circuit unitary, enabling efficient simulation with a state-vector simulator.

❸ Basis rotation: After steps ❶ and ❷, the byproduct parity information is encoded in the network qubits but not in the computational basis. We align this information to the computational basis by applying $H(\text{net_A})$ and $S^\dagger H(\text{net_B})$, which rotate Alice’s and Bob’s network qubits from the X and Y bases to the computational basis, respectively. After this rotation, the network qubits remain unmeasured and coherently encode the correction information, which allows them to directly serve as control qubits in the subsequent correction gates.

❹ Correction: To remove the byproduct without mid-circuit measurement, we replace the original measurement-based feed-forward correction with a coherent correction circuit controlled by the rotated network qubits in ❸. Specifically, we apply an unconditional $S(\text{ctrl})$ gate followed by $CZ(\text{net_A}, \text{ctrl})$ and $CZ(\text{net_B}, \text{ctrl})$, and analogously apply $S^\dagger(\text{tgt})$ followed by $CZ(\text{net_A}, \text{tgt})$ and $CZ(\text{net_B}, \text{tgt})$. We numerically verify the correctness of this construction, with error below 10^{-15} , in Section 5.

Modeling Other Remote Controlled Gates. Since the QGT protocol in [11] focuses only on remote CZ gate, to enable other remote controlled gates, we adopt the controlled-unitary decomposition theorem [14]. For example, we decompose a remote CR_X gate into $R_X(\theta/2) \cdot CZ \cdot R_X(-\theta/2) \cdot CZ$, where R_X is a local gate and CZ is the remote gate implemented via QGT.

Proof. The theorem in [14] enables the decomposition of any controlled unitary operation, $C\text{-}U$, into $(I \otimes A) \cdot CZ \cdot (I \otimes B) \cdot CZ \cdot (I \otimes C)$, where $ABC = I$ and $A \cdot Z \cdot B \cdot Z \cdot C = U$.

For CR_X , i.e., $U = R_X(\theta)$, we denote $A = R_X(\theta/2)$, $B = R_X(-\theta/2)$, and $C = I$ to satisfy these two conditions. (1) $ABC = R_X(\theta/2) \cdot R_X(-\theta/2) \cdot I = I$. (2) $A \cdot Z \cdot B \cdot Z \cdot C = R_X(\theta/2) \cdot Z \cdot R_X(-\theta/2) \cdot Z \cdot I$. Since $Z \cdot R_X(\alpha) \cdot Z = R_X(-\alpha)$, we have $R_X(\theta/2) \cdot Z \cdot R_X(-\theta/2) \cdot Z = R_X(\theta/2) \cdot R_X(\theta/2) = R_X(\theta)$.

3.2 Modeling Noise & Link Failures in QGT

Modeling QGT-Induced Noise. We consider three primary locations within our developed sub-circuit. They correspond to the dominant error sources identified in [11]. Figure 1 highlights them in red. The first location is right after the step ❶, on Alice’s network qubit. We apply a single-qubit depolarizing channel to the location with probability p_{bell} to model imperfect remote entanglement. The second and third locations are after both local CZ gates in step ❷, on Alice’s and Bob’s circuit qubits. We apply single-qubit depolarizing channels with probability $p_{\text{local,A}}$ and $p_{\text{local,B}}$ to these locations to model imperfect local gate operations for Alice and Bob.

For other remote controlled gates, since they can be decomposed into one or more remote CZ gates and local gates, ϵ -DQML models their noise correspondingly and samples noise independently for each circuit execution to accurately replicate realistic, hardware-level noise. For example, one remote CR_X gate requires two QGT-enabled CZ gates, which leads to $2 \times 3 = 6$ locations for inducing noise. Note that when more detailed noise characterization becomes available for QGT, ϵ -DQML’s noise module can leverage the corresponding noise model.

Modeling QGT-Induced Link Failures. We focus on the Bell pair preparation (step ❶) in our sub-circuit to model the link failures. This is because establishing a Bell pair requires transmitting photons through an optical link, where photon loss and limited detector efficiency make each attempt likely to fail. For example, [11] reports that the experimentally measured success probability per attempt is only 1.41×10^{-4} . Although multiple attempts may eventually achieve successful entanglement creation, the number of attempts cannot be unlimited. This limitation arises from the finite time window imposed by qubit decoherence.

To capture this, we adopt a try-until-success strategy. Each attempt succeeds with probability p_{link} , and a maximum of N_{max}

attempts are made. If all attempts fail, the entire remote controlled gate is skipped. Specifically, we develop a mask-based mechanism. Before each circuit execution, a mask value is independently generated for each remote controlled gate following the widely-used Bernoulli distribution with success probability $1 - (1 - p_{\text{link}})^{N_{\text{max}}}$. A mask value of 1 indicates that the Bell pair (i.e., entanglement between Alice’s and Bob’s network qubit) is successfully established, and the corresponding remote controlled gate will execute normally. A mask value of 0 indicates a failed Bell pair preparation, and the corresponding remote gate is excluded from the circuit, thus no need to sample the noise on that gate.

3.3 Accelerating QML Training with QGT-Induced Noise & Link Failures

To capture the stochastic nature of the QML training process on real quantum computers, noise/failures in the forward and backward processes of QML training should be sampled separately. For example, the same gate may exhibit different noise realizations between forward and backward processes. Accounting for this stochasticity relies on the parameter-shift rule for precise gradient computation. However, this method can be computationally intensive, as computing the gradient of *each* trainable parameter θ requires two circuit executions at shifted values, $\theta + \pi/2$ and $\theta - \pi/2$. Every execution requires an independent realization of noise and link failures. Therefore, every single training iteration requires $1 + 2N_{\text{total}}$ circuit executions (one execution in the forward process for loss calculation and $2N_{\text{total}}$ executions in the backward process for gradient calculation), where N_{total} is the total number of trainable parameters.

To accelerate QML training under QGT-induced noise and link failures, we develop a hybrid differentiation method. The training loss is still computed with a single forward circuit execution under one sampled noise/failure realization. Meanwhile, a highly efficient adjoint differentiation method is applied to compute the gradients of all trainable parameters of local gates under the same noise/failure realization. The parameter-shift rule is then applied only to the parameters of *remote* controlled gates, requiring $2N_{\text{remote}}$ additional circuit executions with independently sampled noise/failure realizations. Here, N_{remote} denotes the number of *remote* controlled-gate parameters. Thus, the number of circuit executions per training iteration is reduced from $1 + 2N_{\text{total}}$ to $1 + 2N_{\text{remote}}$.

4 Implementation

We implement ϵ -DQML on top of PennyLane and PyTorch in around 3000 lines of Python code. Specifically, we build the proposed sub-circuit and model QGT-induced noise using standard quantum gates provided by PennyLane. We model link failures using independently sampled Bernoulli failure masks for each remote controlled gate.

To implement the hybrid differentiation method of QML training, we bypass PennyLane’s default tape mechanism, which performs automatic differentiation for the forward and backward processes together using the same noise and failure realization. To enable the independent noise and failure sampling for QML training, we decouple PennyLane’s quantum circuit execution from its differentiation calculation. Specifically, we implement a custom

Table 1: Validation of ϵ -DQML at three different levels.

Level	Comparison Description	Max error
SC	Our QGT sub-circuit vs. Ideal CZ	$< 10^{-15}$
GD	Our CR_X vs. Ideal CR_X	$< 10^{-14}$
SYS	Distributed vs. Non-Distributed	$< 10^{-13}$

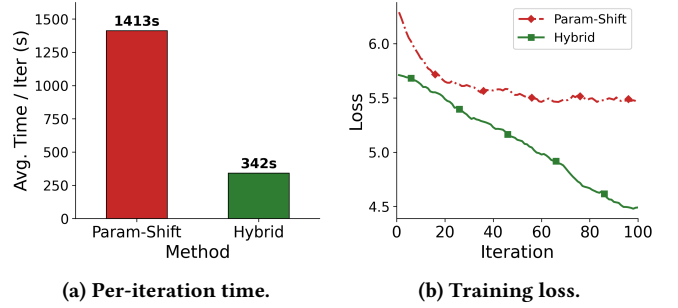


Figure 2: Comparison of hybrid and full parameter-shift differentiation methods.

autograd.Function in PyTorch. Given a QML model, for the forward pass, we first sample noise and failures and then execute the corresponding noise- and failure-induced circuit. For the backward pass, to compute gradients for local parameters, we use PennyLane’s adjoint method, reusing the sampled noise and failures in the forward process. To compute the gradient for each remote parameter, we first resample noise for the circuit corresponding to each shifted value ($+\pi/2$ or $-\pi/2$) and then execute the parameter-shift rule. To further accelerate QML training, we integrate our hybrid method into PyTorch’s optimization pipeline and use the `lightning.gpu` device of PennyLane to perform GPU-accelerated state-vector simulation.

5 Validation of ϵ -DQML

In this section, we validate ϵ -DQML’s QGT modeling and training efficiency.

5.1 QGT Modeling Validation

We validate the QGT modeling at three levels of granularity as shown in Table 1: (1) **sub-circuit (SC)**, whether our QGT sub-circuit accurately reproduces a single remote CZ operation, i.e., ideal CZ, assuming no noise or failures; (2) **gate decomposition (GD)**, whether a remote $CR_X(\theta)$ gate constructed from the sub-circuit behaves correctly, compared with ideal $CR_X(\theta)$; and (3) **system (SYS)**, whether one entire distributed QML circuit (i.e., the same circuit as in §6) without noise or failures produces outputs consistent with the original non-distributed circuit.

Specifically, for the SC level, we consider 7 input states, including 4 computational basis states and 3 superposition states. The three superposition states are prepared as $R_Y(\alpha) \otimes R_X(\beta)|00\rangle$ with $(\alpha, \beta) \in \{(0.7, 1.3), (1.5, 2.1), (\pi/3, \pi/4)\}$ to cover both real and complex amplitude components. The values of α and β are chosen to prevent implementation errors from being masked, since their $\cos(\alpha/2)$, $\sin(\alpha/2)$, $\cos(\beta/2)$, and $\sin(\beta/2)$ are neither zero nor ± 1 .

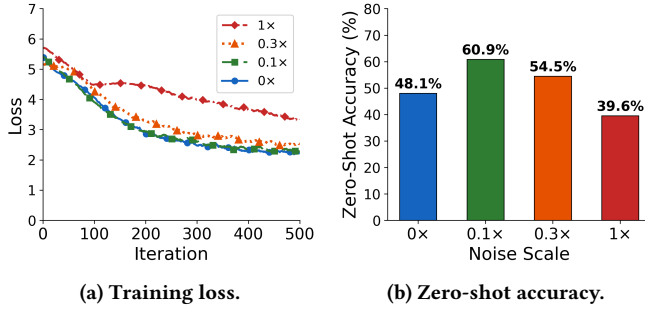


Figure 3: Task 1 – CLIP under different noise levels.

For each input state, we compare $\langle X \rangle$, $\langle Y \rangle$, and $\langle Z \rangle$ on each circuit qubit between the ideal CZ gate and the QGT sub-circuit. Similarly, for the GD level, we consider the same 7 input states along with 8 values of $\theta \in \{0, \pi/4, \pi/2, \pi, 0.3, 1.7, 2.5, -0.8\}$. We select the last four values to also avoid masking implementation errors. In addition, the negative value -0.8 is to verify our implementation handles negative θ correctly. For the SYS level, we execute 10 independent trials on the full 24-qubit distributed circuit (8 circuit and 16 network qubits) with randomly generated parameters (48 per trial), and compare the $\langle Z \rangle$ output of each circuit qubit.

Table 1 shows that the maximum differences over all measurements at the three validation levels remain below 10^{-13} . This confirms that our QGT sub-circuit, CR_X decomposition, and full distributed circuit without noise or failures achieve numerically equivalent behavior to their non-distributed counterparts.

5.2 Training Acceleration Validation

To validate our hybrid differentiation method, we use the same distributed circuit as in Section 6. We train it using our method and the full parameter-shift rule, respectively. The latter applies the parameter-shift rule to both local and remote parameters. Figure 2 shows that our hybrid method takes only 342 seconds per iteration and achieves a 4.1 $\times$ speedup over the full parameter-shift method, which takes around 1413 seconds per iteration.

6 Case Study

To demonstrate the effectiveness of ϵ -DQML, we select two tasks for case studies. Task 1 is CLIP [15], contrastive language-image pretraining, which jointly trains image and text encoders to learn cross-modal representations. Task 2 is NTC, network traffic classification, which identifies normal network traffic and various malicious network connections.

6.1 Experimental Setup

Task 1: CLIP. We follow the proposed model architecture of QCLIP [20], a quantum extension of the classical CLIP model designed for a single quantum machine. QCLIP uses a classical linear model to compress 512-dimensional image/text features into 16-dimensional input to the quantum circuit. We implement a distributed 8-qubit circuit evenly partitioned across two quantum machines (i.e., Alice and Bob): qubits $\{q_0, q_1, q_2, q_3\}$ for Alice and qubits $\{q_4, q_5, q_6, q_7\}$ for Bob. For cross-machine operations, we use remote CR_X gates,

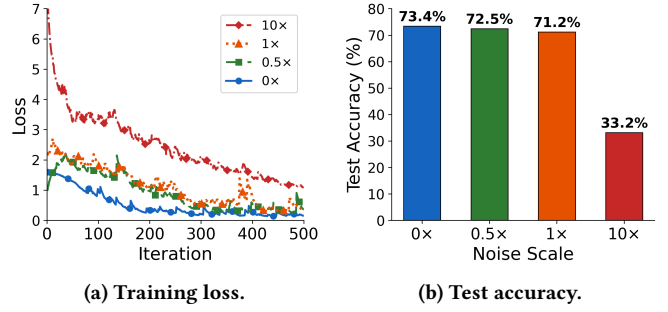


Figure 4: Task 2 – NTC under different noise levels.

i.e., $CR_X(q_3, q_4)$ and $CR_X(q_7, q_0)$, to maintain the original ring topology in QCLIP. In total, there exist 4 remote gates and 28 local gates.

We train the distributed circuit on the CC3M dataset [17] for 500 iterations with a batch size of 128, Adam optimizer, learning rate of 10^{-3} , and symmetric cross-entropy contrastive loss. We evaluate the model's zero-shot classification accuracy on another dataset, CIFAR-10, for all 10 classes without fine-tuning the model.

Task 2: NTC. To study the effects of noise and failures across different tasks, we adopt the same distributed circuit architecture as in Task 1. Different from Task 1, we compress 122-dimensional traffic features into 16-dimensional embeddings and train 16-dimensional embeddings for traffic labels, which serve as feature and label inputs to the quantum circuits, respectively.

To train the model, we use NSL-KDD [19], a standard benchmark in network intrusion detection, which considers 5 traffic classes: Normal, DoS, Probe, R2L, and U2R.

Noise and failure configuration. To study the impact of different levels of QGT-induced noise and failures on QML training, we first derive the noise probability of each location from the experimentally calibrated fidelities in [11]: $p_{\text{bell}} = 1 - 0.9689 = 0.031$, $p_{\text{local},A} = \frac{5 \times (1 - 0.976)}{4} = 0.030$, and $p_{\text{local},B} = \frac{5 \times (1 - 0.98)}{4} = 0.025$, following [13]. We then scale them by $\{0\times, 0.1\times, 0.3\times, 1\times\}$ for Task 1 and $\{0\times, 0.5\times, 1\times, 10\times\}$ for Task 2. For link failures, we set $p_{\text{link}} = 0.75$ and $N_{\text{max}} = 5$.

Hardware. We run all experiments on a single NVIDIA A100 80 GB PCIe GPU.

6.2 Preliminary Results

Task 1: CLIP. Figure 3a shows that slightly increasing the level of noise does not significantly affect QML training. The 0 $\times$ (noise-free) and 0.1 $\times$ settings achieve similar training performance over 500 iterations with final loss values of 2.150 and 1.958, respectively. In contrast, scaling the noise by 0.3 $\times$ marginally degrades the performance and slows the convergence, resulting in a final loss value of 2.503. Using 1 $\times$ (the same noise level as the physical implementation) yields the worst performance with a final loss value of 3.164.

Figure 3b shows that the 0.1 $\times$ noise level achieves the highest zero-shot accuracy, 60.87% versus 48.07% for the noiseless baseline. The 0.3 $\times$ level (54.49%) also surpasses the baseline. However, the 1 $\times$ level has the lowest accuracy (39.56%).

We find that for CLIP, a small amount of noise improves zero-shot accuracy over the noiseless baseline, while stronger noise degrades

both convergence and accuracy. A possible explanation is that low-level QGT-induced noise could act as an implicit regularizer during training. It could prevent overfitting to the training data and improve generalization to unseen data through some random perturbations to remote gate parameters at each circuit execution.

Task 2: NTC. Figure 4a shows that, unlike Task 1, the $0\times$ noise level achieves the smallest final loss value, *i.e.*, 0.224, over 500 iterations. The $0.5\times$ and $1\times$ noise levels converge to slightly higher final loss values, *i.e.*, 0.535 and 0.344. Adding $10\times$ more noise causes the highest initial loss value of 7.463 and finally converges to the largest final loss value of 0.971. For inference, Figure 4b shows that the $0\times$, $0.5\times$, and $1\times$ levels achieve similar accuracy of 73.4%, 72.5%, 71.2%, respectively. The $10\times$ level has the lowest accuracy of 33.17%.

Unlike Task 1, none of the evaluated noise levels improves accuracy over the noise-free baseline. This suggests that Task 2's data and model are already robust, so additional QGT noise does not help.

7 Limitations & Future Work

Modeling Different Gate Failure Rates. In this paper, for link failures, we focus on a specific success probability of $p_{\text{link}} = 0.75$ with $N_{\text{max}} = 5$ attempts, which produces a gate failure rate of approximately 0.1%. In practice, hardware constraints may impose stricter limits on the number of entanglement attempts per remote gate and produce higher failure rates. We plan to explore different rates (e.g., 1%, 5%, and 10%) to investigate the robustness of QML training against such failures.

Modeling Various Noise. For modeling QGT-induced noise, we apply a single depolarization channel to each of the three primary locations within a single remote CZ gate and scale their corresponding probabilities (p_{bell} , $p_{\text{local,A}}$, $p_{\text{local,B}}$) by the same factor simultaneously. We plan to explore different noise models and independently vary noise probabilities to quantify their contributions to QML training performance.

Exploring Different Differentiation Methods. In this paper, we compute gradients for distributed QML by combining the parameter-shift rule and the adjoint method into a hybrid differentiation method (§3.3). There are other differentiation methods, such as simultaneous perturbation stochastic approximation (SPSA) [18], which estimates the full gradient by perturbing all parameters along K random directions. This requires only $1 + 2K$ circuit executions per training iteration, where K is independent of the number of parameters. This reduction comes at the cost of higher gradient variance, which can slow convergence. We plan to extend ϵ -DQML to systematically compare various differentiation methods in terms of training convergence speed and final model accuracy for distributed QML training with QGT-induced noise and failures.

Exploring Different Circuit Structures for QML. Different QML models may use various remote gates (e.g., CR_Y , CR_Z , CNOT, and SWAP). Due to their different sub-circuit structures, they may respond to noise differently. In this paper, we mainly focus on remote CZ and CR_X gates. We plan to extend ϵ -DQML to include more remote gates and study their performance differences under different noise patterns and varying numbers of simulated quantum machines across diverse QML models.

References

- [1] Charles H. Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K. Wootters. 1993. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical Review Letters* 70, 13 (1993), 1895–1899. <https://doi.org/10.1103/PhysRevLett.70.1895>
- [2] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, et al. 2022. PennyLane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968* (2022).
- [3] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J. Martinez, et al. 2020. TensorFlow Quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989* (2020).
- [4] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. 2021. Variational quantum algorithms. *Nature Reviews Physics* 3, 9 (2021), 625–644. <https://doi.org/10.1038/s42254-021-00348-9>
- [5] Tim Coopmans et al. 2021. NetSquid, a network simulator for quantum information using discrete events. *Communications Physics* 4 (2021), 164. <https://doi.org/10.1038/s42005-021-00647-8>
- [6] Jens Eisert, Kurt Jacobs, Polykarpus Papadopoulos, and Martin B. Plenio. 2000. Optimal local implementation of nonlocal quantum gates. *Physical Review A* 62, 5 (2000), 052317. <https://doi.org/10.1103/PhysRevA.62.052317>
- [7] Daniel Gottesman and Isaac L. Chuang. 1999. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature* 402 (1999), 390–393. <https://doi.org/10.1038/46503>
- [8] Ali Javadi-Abhari et al. 2024. Quantum computing with Qiskit. *arXiv preprint arXiv:2405.08810* (2024).
- [9] Tyson Jones and Julien Gacon. 2020. Efficient calculation of gradients in classical simulations of variational quantum algorithms. *arXiv preprint arXiv:2009.02823* (2020).
- [10] Viacheslav Kuzmin, Wilfrid Somogyi, Ekaterina Pankovets, and Alexey Melnikov. 2024. Method for noise-induced regularization in quantum neural networks. *arXiv preprint arXiv:2410.19921* (2024).
- [11] D. Main, P. Drmota, D. P. Nadlinger, E. M. Ainley, A. Agrawal, B. C. Nichol, R. Srinivas, G. Araneda, and D. M. Lucas. 2025. Distributed quantum computing across an optical network link. *Nature* 638 (2025), 383–388. <https://doi.org/10.1038/s41586-024-08404-x>
- [12] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. 2018. Quantum circuit learning. *Physical Review A* 98, 3 (2018), 032309. <https://doi.org/10.1103/PhysRevA.98.032309>
- [13] Michael A. Nielsen. 2002. A simple formula for the average gate fidelity of a quantum dynamical operation. *Physics Letters A* 303, 4 (2002), 249–252. [https://doi.org/10.1016/S0375-9601\(02\)01272-0](https://doi.org/10.1016/S0375-9601(02)01272-0)
- [14] Michael A. Nielsen and Isaac L. Chuang. 2010. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press. <https://doi.org/10.1017/CBO9780511976667>
- [15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*. PMLR, 8748–8763.
- [16] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. 2019. Evaluating analytic gradients on quantum hardware. *Physical Review A* 99, 3 (2019), 032331. <https://doi.org/10.1103/PhysRevA.99.032331>
- [17] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset for Automatic Image Captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 2556–2565. <https://doi.org/10.18653/v1/P18-1238>
- [18] James C. Spall. 1992. Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. *IEEE Trans. Automat. Control* 37, 3 (1992), 332–341. <https://doi.org/10.1109/9.119632>
- [19] Mahbod Tavallaei, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*. IEEE, 1–6. <https://doi.org/10.1109/CISDA.2009.5356528>
- [20] Ruhan Wang, Philip Richerme, and Fan Chen. 2023. A hybrid quantum-classical neural network for learning transferable visual representation. *Quantum Science and Technology* 8, 4 (2023), 045021. <https://doi.org/10.1088/2058-9565/acf1c7>
- [21] Samson Wang, Enrico Fontana, Marco Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J. Coles. 2021. Noise-induced barren plateaus in variational quantum algorithms. *Nature Communications* 12 (2021), 6961. <https://doi.org/10.1038/s41467-021-27045-6>
- [22] Jiaqi Yang, Wei Xie, and Xiaohua Xu. 2025. Stability and Generalization of Quantum Neural Networks. *arXiv preprint arXiv:2501.12737* (2025).