EDITOR: Qun Li, liqun@cs.wm.edu

DEPARTMENT: EMERGING INTERNET TECHNOLOGIES

Distributed Quantum Machine Learning: Federated and Model-Parallel Approaches

Jindi Wu, Tianjie Hu, and Qun Li, *William & Mary, Williamsburg, VA, 23187, USA*

In this article, we explore two types of distributed quantum machine learning (DQML) methodologies: quantum federated learning and quantum model-parallel learning. We discuss the challenges encountered in DQML, propose potential solutions, and highlight future research directions in this rapidly evolving field. Additionally, we implement two solutions tailored to the two types of DQML, aiming to enhance the reliability of the computing process. Our results show the potential of DQML in the current Noisy Intermediate-Scale Quantum era.

Distributed quantum machine learning (DQML) is an emerging field that combines quantum machine learning (QML) with distributed computing. QML utilizes distinctive quantum mechanics properties, like superposition and entanglement, to potentially enhance traditional machine learning algorithms. However, the current stage of quantum computing technology, often referred to as the *Noisy Intermediate-Scale Quantum* (*NISQ*) *era*, imposes limitations on the size and complexity of QML models implemented with variational quantum circuits (VQCs). These constraints can restrict the performance and applicability of QML methods. To address these challenges, integrating QML with distributed computing has emerged as a strategic and forward-looking approach. This hybrid approach aims to overcome the individual limitations of quantum devices by harnessing distributed quantum computing's power to manage and process complex tasks across multiple quantum computing nodes, thus amplifying the capabilities of QML models.

DQML holds promise for a diverse array of applications, especially those demanding complex computations that can utilize the distinct advantages of quantum computing. For instance, DQML can significantly enhance molecular simulation and drug discovery by enabling more efficient modeling of molecular interactions. Furthermore, DQML can be highly beneficial in fields such as financial modeling and medical image processing, which require the efficient handling of large datasets while preserving local privacy. The current cost of training even a modest QML model is quite high. For instance, for a QML circuit of a quantum convolutional neural network (CNN) that utilizes eight qubits and includes approximately 150 trainable parameters on a training set of 500 training instances, it may take approximately $20,000 to train a model on current quantum computers. QML and DQML are generally not practical at present, but in the future, we believe that they have the potential to perform better than classical computers.

The DQML approaches, while promising, confront unique and significant challenges distinct from those in classical distributed machine learning. In this article, we focus on two specific areas within the realm of DQML: quantum federated learning (QFL) and quantum model-parallel learning.

QFL is a case of distributed learning with data parallelism.[1] In this approach, multiple quantum computing nodes, each with its own local dataset, collaborate to train a shared QML model. Each node processes its own data independently, ensuring privacy and security by not transferring raw data between nodes. Instead, only model updates are communicated across the network. The updates generated by each computing node are collectively aggregated, effectively synthesizing the insights learned from distinct local datasets. These consolidated updates are then redistributed to each node. This approach effectively combines computational power and data from diverse sources, enhancing the learning process while maintaining data confidentiality, a key aspect in scenarios where data privacy is crucial.

Digital Object Identifier 10.1109/MIC.2024.3361288
Date of current version xx Month 2024.

Quantum model-parallel learning is a method of distributed learning characterized by the paradigm of model parallelism. The model parallelism in quantum model-parallel learning is particularly advantageous for handling large-scale QML models that exceed the computational and memory capacities of individual quantum nodes.[2] In this framework, the QML model is partitioned into submodels, distributed across multiple quantum computing nodes, with each node handling a distinct submodel of the entire model. The computing nodes process their assigned partition of data and compute intermediate results in parallel, which are then communicated to other nodes or a central coordinator for generating the final outcome. The unique aspect of quantum model-parallel learning is its ability to leverage the individual computational strength of each node while jointly contributing to the construction of a comprehensive model. Through the distribution of computational workload and the facilitation of parallel processing, quantum model-parallel learning opens up new possibilities for addressing more intricate QML tasks.

QFL and quantum model-parallel learning each present unique benefits for specific QML tasks, yet they also face several challenges. In our study, we conduct a thorough examination of these challenges and propose potential solutions to mitigate these issues. These challenges include quantum errors, scalability, communication, and hardware diversity. Moreover, we implement two specific solutions to enhance the reliability of the two types of DQML approaches. In particular, a key challenge in QFL arises from the global model being susceptible to a wide array of local errors. Because the local models are trained under device-specific errors, the aggregation of these error-impacted local models can significantly compromise the efficacy of the overall QML model. To address this issue, we minimize the impact of errors on local models as much as possible. For quantum model-parallel learning, devising a strategy to partition a large-scale QML model without compromising its functionality and reliability presents a considerable challenge. We address these issues by carefully designing the submodels of the QML model, taking into account both the architecture and the reliability of the available quantum computing nodes. Our results demonstrate the significant potential of DQML in the current NISQ era.

BACKGROUND

Quantum Computing

Quantum computing is a revolutionary paradigm of computation that leverages the principles of quantum mechanics to perform complex calculations.

Qubit

A qubit, which carries quantum information, is the fundamental building block of quantum computing. Due to its unique property of superposition, it can exist in multiple states simultaneously. The state of a single qubit can be represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ with $\alpha,\ \beta \in \mathbb{C}$. Here, $|\alpha|^2$ and $|\beta|^2$ represent the probabilities of measuring the qubit as $|0\rangle$ and $|1\rangle$, respectively, i.e., $|\alpha|^2 + |\beta|^2 = 1$. Superposition empowers quantum computers with increased computational power, allowing them to explore and process numerous potential solutions to a problem in parallel. Furthermore, entanglement allows two or more qubits to establish correlations in which changes in the state of one entangled qubit instantaneously impact the state of the other, irrespective of the spatial separation between them.

Quantum Gates

Quantum computation involves manipulating qubit states with quantum gates, represented by unitary matrices denoted as U (satisfying the conditions $U^{\dagger}U = UU^{\dagger} = I$). The fundamental quantum gates include the Pauli-X, Pauli-Y, Pauli-Z, Hadamard (H), and controlled-X (CNOT) gates. These gates serve as foundational building blocks for assembling more intricate quantum algorithms. Among these gates, the CNOT gate is a two-qubit gate that establishes correlations between qubits. In addition, quantum rotation gates, such as RX(θ), RY(θ), and RZ(θ), provide precise manipulation of quantum states through an angle θ.

Quantum Errors

Quantum computing is error-prone due to the inherent instability of the quantum system and the immature manufacturing of quantum computers. A quantum circuit (program) comprises a set of gates that manipulate quantum data, and the processed data are acquired through quantum measurement operations. Each operation in this process can introduce errors into the quantum system, potentially resulting in inaccuracies in the quantum circuit's outcomes. In particular, errors in quantum computing arise from various sources, including quantum state preparation, quantum gates, measurement, and crosstalk. Moreover, the state of qubits can be influenced by errors due to decoherence and dephasing.

QML

VQCs are a widely adopted approach for constructing QML models. These models are tailored for executing particular tasks, including optimization and classification, by harnessing the capabilities of quantum circuits and integrating them with classical optimization methods.

Ansatz

A VQC, implementing a QML model, is composed of three parts: 1) the *encoding unit* is responsible for converting classical data into quantum data, which can be processed by the quantum circuit. Popular encoding techniques include angle encoding and amplitude encoding. 2) The *variational block* constitutes the core of the VQC, featuring a sequence of parameterized quantum rotation gates. The rotation gates equipped with trainable parameters function analogously to neurons in classical neural networks. The entire variational block is organized into layers, and a block consisting of L layers can be represented as

$$U(\theta) = U_L(\theta_L)U_{L-1}(\theta_{L-1})\ldots U_1(\theta_1) \tag{1}$$

(3) The *measurement unit* involves the measurement of one or more qubits to obtain the outcome corresponding to the input data with observable O.

Optimization

QML employs a hybrid quantum–classical approach to train the model. Once a model on a quantum computer processes specific input data, the classical computer takes over to optimize the model's parameters, guided by a predefined cost function

$$C = \langle 0|U^{\dagger}(\theta)OU(\theta)|0\rangle. \tag{2}$$

This optimization can be executed through various methods, including gradient-based approaches like the stochastic gradient descent algorithm or non-gradient-based techniques, such as the parameter-shift algorithm. The entire training procedure in QML involves repeating the optimization step until the parameters of the model converge.

QFL

QFL is a sophisticated process that blends the principles of quantum computing with federated learning's distributed model training approach. The QFL system consists of a central server and multiple quantum computing nodes, each allocated to different clients. In this configuration, clients maintain their data locally on their assigned quantum computing nodes. The objective is to collaboratively train a QML model that benefits from the aggregated data across all nodes, while ensuring that each client's private information remains unshared and secure.[3] Formally, the QFL procedure for a VQC-based QML model can be represented as follows:

1) *Initialization*: Each client initiates a local model $U(\theta)$ with the same ansatz and parameters on their respective local devices.
2) *Local training*: On the i-th quantum computing node, the client trains its individual model $U(\theta_i)$ using the private dataset with several update steps.
3) *Local model submission*: Each participating client submits their updated local parameters θ to the central server.
4) *Model aggregation*: The central server aggregates the received local parameters from clients and integrates them into a global model represented as $\theta' = \sum_{i=1}^{N} n_i\theta_i$, where N is the number of clients involved in this procedure, and n_i denotes the corresponding weight of the i-th client.
5) *Global model distribution*: The updated global parameters θ' are distributed to clients for the subsequent round of training.
6) *Iteration*: Repeat steps 2–5 for multiple rounds until convergence or the desired model performance is achieved.

Quantum Model-Parallel Learning

Quantum model parallelism is a distributed approach in QML, especially for large-scale QML models. In this method, a complex quantum model is partitioned into submodels, which are then distributed across multiple quantum computing nodes.

In such a large-scale QML model, the overall unitary operation $U(\theta)$ is partitioned into K submodels $\{U_1(\theta_1), U_2(\theta_2), \ldots, U_k(\theta_k)\}$. Each submodel, denoted as $U_i(\theta_i)$, is allocated to a distinct quantum computing node, with i ranging from 1 to K. Similarly, the input data D are divided into subsets $\{D_1,\ D_2,\ \ldots,\ D_n\}$, corresponding to the submodels $\{U_1(\theta_1), U_2(\theta_2), \ldots, U_n(\theta_n)\}$ within the input layer of the QML model, where $n \leq K$. Starting with the submodels in the input layer, each submodel within the same layer operates independently. The output generated by a submodel is then transmitted to the submodel in the subsequent layer, using either classical or quantum communication channels. The submodel in the final layer of the QML model generates the ultimate output that corresponds to the specific input data. This process guarantees the seamless flow of information and computation across the distributed quantum system.

RELATED WORK

The design of the QFL model can be varied to align with the specifications of the available quantum devices and data. For instance, a hybrid quantum–classical transfer learning model is developed and deployed on each computing node for federated learning.[4] This model utilizes a pretrained classical model to compress classical input data into a small size, followed by leveraging a VQC-based QML model for decision making. In addition, a QFL approach based on quantum data is proposed.[5] Furthermore, beyond a VQC, the QFL model can be constructed using various methods. For example, the model may consist of several layers that incorporate a differing number of qubits.[6,7]

A scalable QML is an instance of quantum model-parallel learning.[2] The approach divides a large-scale QML model into two distinct layers. The first layer comprises individual subcircuits, each designed to learn from segments of a training instance. The second layer then aggregates these intermediate results, enabling further exploration of the correlations between data segments. In addition, the approach can be implemented with a single quantum computing node due to the independence between the submodels of the large QML model. Similarly, a quanvolutional neural network achieves scalability by constructing quantum convolutional kernels that emulate the functionality of the classical convolutional kernel used in classical CNNs.[8] This quantum kernel slides over the input data to extract abstract features, mirroring the process in classical CNNs.

CHALLENGES

Although QFL and quantum model-parallel learning offer benefits by integrating distributed computing with QML, such as enhanced privacy protection and improved model scalability, there are still several challenges that need to be addressed.

Quantum Errors

Quantum errors present a significant challenge in distributed quantum computing systems as the errors accumulate through noisy operations and vary over time in an unpredictable manner.

During the training process of a QML model, parameter updates serve a dual purpose: they learn from the training dataset and simultaneously capture the error pattern to mitigate the impact of quantum errors. Nevertheless, QML models cannot completely eliminate the influence of these errors. The error pattern captured by a QML model is influenced by both the ansatz of the model and the specific quantum device. When the level of error is sufficiently high, the reliability of the model can be significantly compromised. Furthermore, error patterns vary across different quantum devices. Consequently, in QFL approaches, the aggregated model suffers from varying errors that originate from multiple computing nodes, which can lead to suboptimal performance. Additionally, quantum model-parallel learning produces final outcomes based on noisy intermediate results. As the complexity of the model increases, errors accumulate, consequently diminishing the fidelity of the outcomes.

Error-mitigation strategies such as circuit optimization and result postprocessing can be utilized to effectively improve the reliability of each computing node. For instance, the solution presented in the subsequent section aims to alleviate the influence of various errors in the overall model. It achieves this by specifically reducing the error impact on each computing node through the application of circuit-optimization techniques. However, a comprehensive solution for error mitigation has not yet been fully realized. In addition, quantum errors continuously change in unpredictable ways. Therefore, a model trained to mitigate errors during the training phase may not be effective against errors encountered during testing. A potential solution to address fluctuating quantum errors involves continuously updating the model to accommodate current quantum error conditions, but this method may lead to considerable overhead. Alternatively, the model can be trained while accounting for shifted errors.[9]

Scalability

Although distributed approaches can enhance the scalability of QML tasks to a certain extent, scalability remains a significant challenge in the field. For QFL, the scale of the QML model deployed on an individual quantum computer is constrained by the limited quantum computing resources. The circuit width, for instance, is confined by the number of qubits available on the quantum hardware. These qubits, serving as the register for data processing, limit the data's dimensionality that the model can process. To mitigate this, a common strategy is to compress data to fit the available qubit capacity. Additionally, various encoding methods have been proposed to represent data within a limited number of qubits. However, these methods can potentially diminish the utility of data for specific tasks or introduce significant overhead and errors, posing a tradeoff between data representation efficiency and the fidelity of the information processed. Moreover, the depth of the circuit is constrained due to the accumulation of errors and the instability of the quantum system. Therefore, for QFL tasks, the scale and complexity of local models remain limited. This limitation can subsequently constrain the performance and effectiveness of these local models.

Within quantum model-parallel learning, the scale of the QML model is dictated by the specifications of available quantum devices and the requirements of circuit partitioning. The capacity of each individual quantum computing node plays a critical role in determining how the QML model should be partitioned. Excessive partitioning of the QML model into numerous small segments can compromise its integrity. Hence, achieving an optimal balance in the partitioning process is crucial for the effective functioning of a large-scale QML model.

Furthermore, as the scale of the QML model increases, the challenge of the barren plateau emerges.[10] This phenomenon, encountered in the training of large-scale quantum neural networks, is characterized by the gradient of the cost function becoming extremely small. As a result, it becomes inefficient to train the QML model. To tackle the barren plateau problem, recommended strategies include careful parameter initialization and the adoption of problem-specific ansatzes. Nevertheless, there remains a need for more advanced solutions to effectively address this issue.

Communication

The communication channels, both classical and quantum, between quantum computing nodes in a distributed QML system present challenges. In quantum model-parallel learning, the outputs of subcircuits, obtained through measurements, are transmitted to the central node via classical communication channels. However, these measurement results may not entirely represent all the information of the intermediate states as measurements are typically made on a single basis, which can result in the loss of significant information. One potential solution to this issue is the classical shadow technique, which involves performing a series of measurements on a quantum state and using the results to create a "shadow" or a classical approximation of the state. However, the overhead will rise exponentially as the number of qubits increases. An effective strategy for reducing measurement overhead is to reconstruct the complete state of a single qubit and fully utilize the information of this single qubit to scale up the size of the problem that the model can solve.[11] Alternatively, the processed quantum states at the nodes could be transmitted through quantum communication channels. Yet, the development of stable and reliable quantum communication channels, essential for a functional quantum network, is still in its nascent stages. Overcoming these communication hurdles is crucial for the efficient operation of distributed QML systems.

Hardware Diversity

Hardware diversity, particularly the various techniques used to implement qubits, poses a significant challenge in distributed QML. Different quantum systems use different physical implementations for qubits, such as trapped ions, superconducting circuits, or topological qubits, each with unique characteristics and limitations. This diversity poses a challenge when attempting to create a standardized QML model capable of running on various quantum hardware platforms. Furthermore, the interoperability between diverse quantum systems becomes complex due to differences in their control and measurement protocols. This necessitates the development of adaptable QML algorithms that can be efficiently transpiled and optimized for various hardware architectures. In essence, hardware diversity in distributed QML demands a robust and flexible approach to algorithm design and system optimization to ensure effective and reliable performance across heterogeneous quantum computing platforms.

APPROACHES

In this section, we introduce two strategies designed to enhance the reliability of QFL and quantum model-parallel learning by applying an error-mitigation technique. These techniques primarily involve minimizing gate errors by optimizing quantum circuit design. In particular, we design the QML models to minimize the necessity for SWAP gates according to the qubit topology on the machines. These gates are inserted into the circuit to enable the application of multiqubit gates to nonadjacent qubits. Moreover, we prioritize choosing qubits and gates with lower error rates to further reduce the error rate of the QML model.

Based on the distinctive characteristics of QFL and quantum model-parallel learning, various strategies can be employed to design the QML model. In the context of QFL, all participating nodes train the QML model using the same logical ansatz. Nevertheless, the transpiled circuits of the model vary and capture distinct error patterns, adversely affecting the performance of the aggregated model. In contrast, in the quantum model-parallel learning approach, a large-scale QML model is partitioned into several submodels, each potentially with a different ansatz. The overall performance of the entire model depends on the reliability of each submodel. Therefore, our approach to QFL centers around designing the QML model while taking into account the qubit topology and quality of all the participating computing nodes. On the contrary, for quantum model-parallel learning, we design submodels of the entire QML model by separately considering the qubit topology and quality of each node involved.

QFL

In QFL, our strategy begins with a careful selection process, where we carefully select a subset of qubits from each participating node. These chosen subsets are characterized by similar qubit connectivity and relatively lower error rates. Subsequently, we construct the QML model to adapt the qubit connectivity of these selected subsets. This method guarantees that the transpiled QML models require the minimum number of SWAP gate insertions across all nodes. By doing so, we aim to minimize the impact of errors on each local model and, consequently, reduce the disturbance in the aggregated model caused by differing local error patterns. This method provides a more unified and error-resilient approach to QFL, optimizing both individual and aggregated model performance.

For concreteness, given a QML task focused on a binary classification task that categorizes images composed of four pixels [as shown in Figure 1(a)], we construct a QML model that incorporates four qubits. There are several open source tools available for implementing QML. For instance, Qiskit by IBM, PennyLane by Xanadu, and TensorFlow Quantum by Google. These tools provide support for QML, but currently, there are no tools specifically designed for DQML. In our research, we created a simulated federated learning environment utilizing Qiskit on the IBM Quantum platform. For this setup, three quantum computers, *ibm_lagos*, *ibm_perth*, and *ibm_nairobi*, were selected as individual quantum computing nodes to train local QML models. These three computers have identical qubit topologies, as depicted in Figure 1(b). From each computer, we select a subset of qubits within which three qubits are adjacent to a central qubit, and these are chosen for their lower error rates, as illustrated in the green box in Figure 1(b). Then we construct the QML model as depicted in Figure 1(c). This specific ansatz ensures that no noisy SWAP gates are required during the circuit transpilation, thereby preserving the reliability of the circuit. This QML model is then deployed on nodes, selecting the highest-quality qubit set that meets this topology requirement. This strategy not only enhances the fidelity of the outcomes but also minimizes the impact of errors on the trained model.

We assess the effectiveness of our proposed approach by conducting the training phase on a simulator that incorporates the noise models of specific quantum devices. Subsequently, we test the trained model on actual quantum devices. The trained model attained accuracies of 93%, 96%, and 96% on the three respective nodes involved, as illustrated in Figure 1(d). For a baseline comparison, the accuracies of a model trained without accounting for the specific architecture of the quantum devices across nodes are 90%, 86%, and 88%. The observations indicate that the model with circuit optimization using our approach outperforms the baseline model without circuit optimization in terms of accuracy across all computing nodes, highlighting the efficacy of our method. Moreover, the lower accuracy achieved by the baseline indicates that although the trained QML model can offset errors to a certain degree, its effectiveness is still notably diminished by these errors. This is mainly because the transpiled circuit of the model incurs a large volume of errors due to the extensive insertion of SWAP gates, which are intricately eliminated in our proposed approach.

Quantum Model-Parallel Learning

Typically, the width of a QML model is determined by the size of the data to be processed. For instance, to encode a data instance consisting of eight components, a QML model would require eight qubits using the angle-encoding method. In general, a large-scale

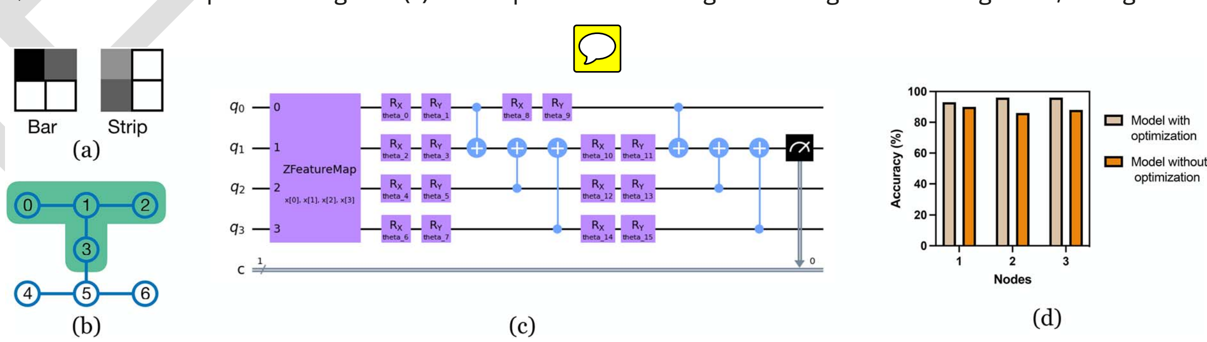

FIGURE 1. QFL with error mitigation. (a) The dataset comprises two classes, with each image containing four pixels. (b) The qubit topology of quantum computing nodes involved in the QFL environment. (c) The specifically designed QML ansatz, tailored with the qubit topology to eliminate the need for inserting SWAP gates. (d) The test accuracy of the model on actual quantum devices.

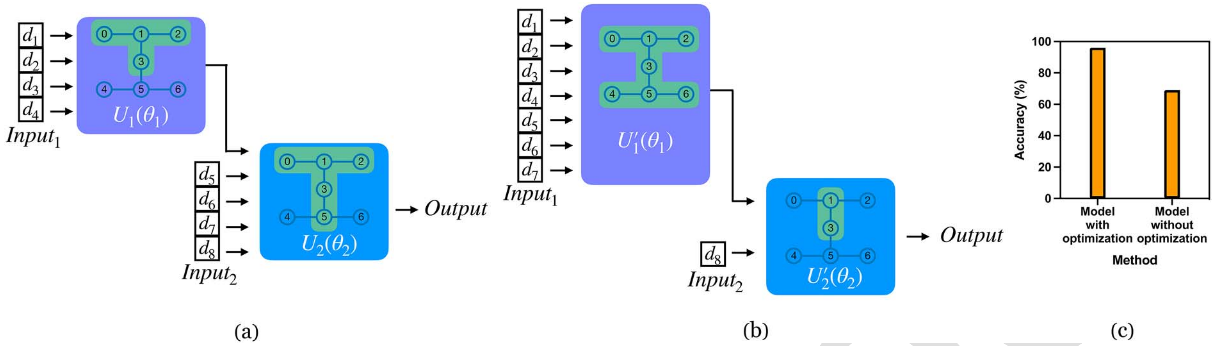

FIGURE 2. Quantum model-parallel learning with error mitigation. (a) A QML model is designed with two subcircuits, each constructed with consideration of the architecture and reliability of the involved nodes. (b) A QML model is constructed and then partitioned based on the number of qubits available on the nodes. (c) The test accuracy of the models implemented with different approaches.

QML model is first built and then divided into submodels according to the capacities of the available quantum computing nodes.

This approach frequently necessitates inserting additional gates and can compromise the correlation within data instances, leading to a significant reduction in the model's reliability. Conversely, our approach entails designing the submodels while considering both the qubit topology and the quality of resources available on the quantum computing nodes. These submodels are then integrated into a complete QML model. This strategy aims to preserve the fidelity of the output of each submodel, thereby enhancing the overall performance of the model.

For example, we consider a QML-based classification task focused on handwritten digits 0 and 1, where images from the Modified National Institute of Standards and Technology dataset are downscaled to eight components using the principal component analysis method. The quantum system in this scenario comprises two quantum computing nodes: *ibm_lagos* and *ibm_perth*, with their qubit topology illustrated in Figure 2. Based on the characteristics of these two quantum computing nodes, we design a QML model consisting of two submodels: the first subcircuit, consisting of four qubits, processes the first half of the image, while the second subcircuit, encompassing five qubits, handles the second half of the image and integrates the processed results from the first half. The complete QML model is depicted in Figure 2(a). As a baseline for comparison, another method implemented for this task involves partitioning the QML model based solely on the resource capacity of the computing nodes. For processing data instances with eight values, the first subcircuit is constructed using all the qubits (seven qubits) of one node to process seven components of the data instance. Meanwhile, the second subcircuit utilizes two qubits to process the remaining component and to integrate the intermediate results, as shown in Figure 2(b).

In Figure 2(c), we depict the accuracy of two QML models implemented using distinct methods: one with circuit optimization achieving 96% accuracy and another without optimization achieving 69% accuracy. It's clear that the optimized model exhibits significantly superior accuracy compared to the baseline model without circuit optimization. There are three main reasons for the observed performance difference. First, our proposed method for designing subcircuits using circuit-optimization techniques eliminates the need for noisy SWAP gates, thereby reducing the overall error rate. Second, data instances are partitioned evenly in our approach. This method preserves coherence within each data segment and ensures that the final result is evenly influenced by both parts. In contrast, the baseline method partitions the data unevenly, leading to a biased final result. Third, although the baseline method can also be optimized to minimize additional SWAP gates, the first submodel in this approach still includes many gates necessary for the model's functioning. Given that the fidelity of a quantum circuit exponentially decreases with the increasing number of gates, the larger size of the subcircuit in the baseline method likely results in significantly lower fidelity. Therefore, carefully partitioning the model across available computing nodes is advantageous for maintaining higher fidelity.

CONCLUSION

In this article, we explore two methodologies within DQML: QFL and quantum model-parallel learning.

A significant challenge in distributed QML is managing quantum errors. To address this, we introduce an error-resilient approach to QFL and quantum model-parallel learning, employing an ansatz construction based on qubit topology. Empirical evaluations highlight the effectiveness of our solutions, underscoring the potential of distributed QML methodologies.

REFERENCES

1. C. Ren et al., "Towards quantum federated learning," 2023, *arXiv:2306.09912*.
2. J. Wu, Z. Tao, and Q. Li, "wpScalable quantum neural networks for classification," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Piscataway, NJ, USA: IEEE Press, 2022, pp. 38–48, doi: 10.1109/QCE53715.2022.00022.
3. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, PMLR, 2017, pp. 1273–1282.
4. S. Y.-C. Chen and S. Yoo, "Federated quantum machine learning," *Entropy*, vol. 23, no. 4, pp. 460–473, 2021, doi: 10.3390/e23040460.
5. M. Chehimi and W. Saad, "Quantum federated learning with quantum data," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Piscataway, NJ, USA: IEEE Press, 2022, pp. 8617–8621, doi: 10.1109/ICASSP43922.2022.9746622.
6. Q. Xia and Q. Li, "QuantumFed: A federated learning framework for collaborative quantum training," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Piscataway, NJ, USA: IEEE Press, 2021, pp. 1–6, doi: 10.1109/GLOBECOM46510.2021.9685012.
7. Q. Xia, Z. Tao, and Q. Li, "Defending against byzantine attacks in quantum federated learning," in *Proc. 17th Int. Conf. Mobility, Sens. Netw. (MSN)*, Piscataway, NJ, USA: IEEE Press, 2021, pp. 145–152, doi: 10.1109/MSN53354.2021.00035.
8. M. Henderson, S. Shakya, S. Pradhan, and T. Cook, "Quanvolutional neural networks: Powering image recognition with quantum circuits," *Quantum Mach. Intell.*, vol. 2, no. 1, p. 2, 2020, doi: 10.1007/s42484-020-00012-y.
9. Z. He, B. Peng, Y. Alexeev, and Z. Zhang, "Distributionally robust variational quantum algorithms with shifted noise," 2023, *arXiv:2308.14935*.
10. J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Commun.*, vol. 9, no. 1, 2018, Art. no. 4812, doi: 10.1038/s41467-018-07090-4.
11. J. Wu, T. Hu, and Q. Li, "MORE: Measurement and correlation based variational quantum circuit for multi-classification," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Piscataway, NJ, USA: IEEE Press, 2023, pp. 208–218, doi: 10.1109/QCE57702.2023.00031.

JINDI WU is a Ph.D. student focusing on quantum error mitigation and quantum machine learning in the Department of Computer Science, William & Mary, Williamsburg, VA, 23187, USA. Contact her at jwu21@wm.edu.

TIANJIE HU is a Ph.D. student focusing on quantum error correction in the Department of Computer Science, William & Mary, Williamsburg, VA, 23187, USA. Contact him at thu04@wm.edu.

QUN LI is a professor with the Department of Computer Science, William & Mary, Williamsburg, VA, 23187, USA. Contact him at liqun@cs.wm.edu.