Quantum-Classical-Quantum Workflow in Quantum-HPC Middleware with GPU Acceleration

Kuan-Cheng Chen[*†‖], Xiaoren Li[‡], Xiaotian Xu[*†], Yun-Yuan Wang[§] Chen-Yu Liu[¶],
[*]Centre for Quantum Engineering, Science and Technology (QuEST), Imperial College London, SW7 2BX London, UK
[†]Department of Materials, Imperial College London, SW7 2BX, London, UK
[‡]Department of Physics and Astronomy, University of Waterloo, ON N2L 3G1 Waterloo, Canada
[§]NVidia AI Technology Center (NVAITC), NVIDIA Corp., 114 Taipei, Taiwan
[¶]Graduate Institute of Applied Physics, National Taiwan University, 10663 Taipei, Taiwan
[‖]Email: kuan-cheng.chen17@imperial.ac.uk

***Abstract*—Achieving high-performance computation on quantum systems is challenging, requiring integration between quantum and classical computing resources. This study presents a distribution-aware Quantum-Classical-Quantum (QCQ) architecture that combines advanced quantum software frameworks with high-performance classical computing to improve quantum simulations for materials and condensed matter physics, including the prediction of quantum phase transitions. The architecture employs Variational Quantum Eigensolver (VQE) algorithms on Quantum Processing Units (QPUs) for efficient quantum state preparation, and Tensor Network states and Quantum Convolutional Neural Networks (QCNNs) on classical hardware for state classification. Utilizing the cuQuantum SDK and PennyLane's Lightning plugin, the QCQ architecture achieves up to tenfold increases in computational speed for complex phase transition classification tasks compared to traditional CPU-based methods, demonstrating 99.5% accuracy in predicting phase transitions in models like the transverse field Ising and XXZ systems. This framework integrates quantum algorithms, machine learning, and Quantum-HPC capabilities, offering transformative insights into the behavior of quantum systems across different scales. As quantum hardware continues to improve, the QCQ framework will play a crucial role in realizing the full potential of quantum computing by seamlessly integrating distributed quantum resources with state-of-the-art classical computing infrastructure.**

***Index Terms*—Quantum Machine Learning, Distributed Quantum Computing, Quantum Software, Quantum Simulation, cuQuantum SDK, Tensor Network**

I. Introduction

Quantum Machine Learning (QML) integrates the disciplines of machine learning and quantum computing [1], employing parameterized quantum circuits as statistical models [2]. This technology has seen an increasing array of applications in the natural sciences [3], generative modelling [4], and classification problems [5], [6]. Due to its high expressivity [7], QML demonstrates superior performance over conventional models across numerous domains. This includes stellar classification within large datasets [8], leveraging an architecture that effectively bridges classical data with quantum algorithms. However, current applications are constrained by the limited number of Quantum Processing Unit (QPU) qubits and the fidelity of qubits in the Noisy Intermediate-Scale Quantum (NISQ) era [9]. Despite these challenges, recent research conducted by Q-CTRL and IBM Quantum has made progress in enhancing the success probability of algorithms through an automated deterministic error-suppression workflow and quantum error mitigation technique [10], [11]. Nonetheless, the availability of qubits remains a significant issue in the design of quantum machine learning algorithms [9].

On the other hand, the work of Huang et al. has demonstrated quantum advantage through the utilization of quantum data with a quantum computer [12]. For instance, Monaco et al.'s application of a Variational Quantum Eigensolver (VQE)-based quantum neural network model for quantum phase detection in the axial next-nearest-neighbour Ising (ANNNI) model illustrates its superiority [13]. However, such architectures necessitate an increased number of qubits for learning more complex models [14]. The demand for a large number of qubits can be addressed by integrating architectures through distributed quantum computing [15], also referred to as quantum-centric supercomputing by IBM Quantum [16]. In the realm of compact quantum devices, achieving high-fidelity qubit operations and their subsequent verification is relatively manageable. Nonetheless, the shift towards modular approaches mandates the transfer of quantum information between QPUs, thereby fostering effective qubit interactions across various modules. While this approach offers certain benefits, the ensuing inter-module communication is often characterized by slower speeds and reduced reliability, leading to the emergence of a challenge known as the quantum interconnect bottleneck (QIB) [17]. Furthermore, the adoption of distributed quantum algorithms introduces vulnerabilities that necessitate the formulation of robust quantum protocols to ensure system integrity and security [18].

In this study, we endeavor to develop a scheme that is conducive to high-performance computing (HPC) within the quantum domain [19], specifically designed to enable distributed quantum computing through a hybrid quantum-classical workflow. This framework incorporates multi-GPU acceleration (via the cuQuantum SDK [20]) to support quantum simulators (which serve as counterparts to noiseless QPUs, if QPUs are

The first two authors contributed equally to this work.

not available). Additionally, we integrate the latest Pennylane Lightning plugins [21]. This component leverages CUDA-aware MPI (Message Passing Interface) boosted by NVLink's 600 GB/s bidirectional bandwidth for optimized GPU-to-GPU communication, enabling rapid distributed simulation tasks [22]. This middleware scheme offers a synergy between classical GPU acceleration and quantum computing acceleration [19], [23]. The relevant conceptual architectures for Quantum-HPC middleware are discussed by Saurabh et. al [19].

Echoing the architectural paradigm introduced by Monaco et al. [13], our methodology adopts a VQE-based ansatz to explore quantum phase transitions in both the transverse field Ising and the XXZ models [24]. Different from the quantum-to-quantum or the quantum-to-classical architecture [13], [24]–[26], we propose an quantum-classical-quantum sandwich (QSandwich) architecture. This architecture capitalizes on the capabilities of classical convolutional neural networks (CNNs) to significantly reduce the qubit requirements for the quantum classifier.

For the implementation of the VQE-based ansatz, we also explore the potential of distributed-VQE approaches [25], [27], [28] or the employment of multi-GPU configurations [29] to meet the demands of large qubit requirements in complex scenarios. The QSandwich framework is designed to ensure that each quantum layer is effectively managed and utilized within a distributed quantum computing environment, leveraging small, yet high-fidelity QPU resources. This approach underscores our commitment to advancing the frontier of quantum computing by optimizing computational resources and fidelity in the execution of quantum simulations.

II. Applications and Algorithms

A. *QSandwitch: Quantum-Classical-Quantum framework for Quantum-HPC Processing*

In the domain of Quantum-HPC [19], we proposed a novel distribution-aware hybrid quantum-classical-quantum (QCQ) processing framework shown in Fig. 1, representing an optimized approach to the computational challenges of distributed quantum computing for classifying phase transition. Initiated by the VQE algorithm, this framework begins its operation within the quantum realm, where a specific quantum state is prepared to encode potential solutions. This quantum state is then converted into classical information via a process termed state feature selection, during which essential characteristics of the quantum state are identified and extracted for subsequent processing in classical CNN layers. This methodology introduces a structured integration of quantum and classical computing techniques, aiming to optimize the solution-finding process through the strategic manipulation and analysis of quantum data.

The classical segment of the QCQ framework harnesses the capabilities of CNNs to scrutinize the quantum state prepared by the VQE algorithm. This involves the application of convolutional layers to refine the data, augmented by pooling layers that distil the essence of the information. The refined data is then adeptly re-encoded into quantum form, traversing another quantum layer tasked with extracting a scalar output to classify the phase transition problem.

The scalar output generated by the process is not the end point but rather an intermediate stage requiring additional refinement via a sigmoid layer to improve the accuracy of the final result. The following section will provide a detailed explanation of this hybrid QCQ architecture, focusing on the interaction between its quantum and classical components. The integration of these computational realms aims to enhance the capabilities of the near-term Quantum-HPC ecosystem, addressing challenges with efficiency beyond what quantum or classical computing can achieve individually.

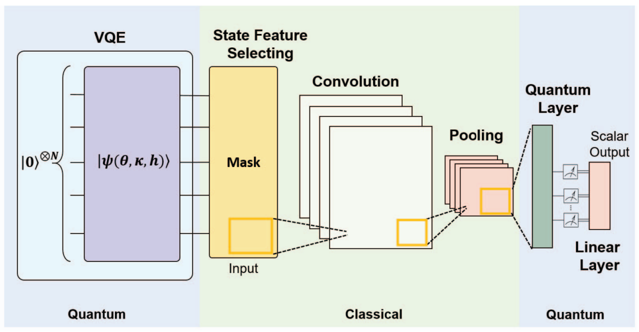

Figure 1. Pipeline of the Hybrid QSandwich Architecture.

1) VQE for State Preparation: VQE is a common approach in finding ground state in quantum computing [30]. In our study, the ground states are needed as the training data for the phase-transition problem. In this section, we focus on preparing the ground states with VQE and tensor network ansatz [24]. These methods enable efficient and accurate approximation of quantum states. Here are the key points of our approach:

- **Ground State Preparation**: The ground states information is needed for the following study. There are some open source datasets, for example Pennylane dataset [31], offers classical-shadow information of ground states which is ready for use. However, we decide to employ the VQE to approximate the ground states by ourselves. The advantage of using VQE for state preparation, instead of loading state information from the dataset, is that we can reproduce the ground state completely in the following study, since we know the structure and the parameters of the VQE circuit we used. And the dataset does not contain the circuit they used. On the other hand, the classical-shadow information from the Pennylane dataset will be helpful to reproduce the state more accurately [32], however, it could not be the exactly same state. Knowing the circuit structure also reduces the size of datasets which speeds up the data reading process in quantum machine learning later on. A n-qubits states information from pennylane datasets contains 2^n floats. On the other hand, our VQE states preparation datasets only records the parameters for the ansatz circuits. The number of parameters varies depending on the structure and depth of the ansatz. For example, we used 100 float parameters for a 10-qubit circuit in state preparation which reduced the size of datasets (otherwise 1024 floats needed) and still performance well in classification. Additionally, building the state preparation ourselves allows us to prepare more ground states (100 states from the Pennylane dataset [31] and 1000 states from our preparation) and makes it possible to apply data augmentation in II-A2.

- **Tensor Network Ansatz**: Our approach replaces the standard unitary coupled cluster ansatz with tensor network ansatz states introduced by Uvarov et. al [24], which was inspired by tensor networks, to prepare the ground states efficiently. There are several families of variational ansatz states, including rank-one circuits, tree tensor network circuits, and checkerboard-shaped circuits with varying depths. The structure of the checkerboard-shaped circuit is shown in Fig. 2 (b). These ansatz circuits are designed to capture different amounts of entanglement and are critical for accurately approximating the ground states of various Hamiltonians. We choose the checkerboard-shaped circuits in our VQE due to the best performance in reducing the error of ground state energy [24]. For each entangled blocks in a checkerboard-shaped circuit, we used Ising entangle gates shown in Fig. 2 (c).
- **Hamiltonian Decomposition**: We represent the Hamiltonian as a sum of tensor products of Pauli operators. This decomposition allows us to estimate individual terms of the Hamiltonian expectation value variational and minimize the energy using a hybrid process. More math details are shown in II-B
- **Optimization**: The parameters of the ansatz states are optimized using classical algorithms to minimize the expectation value of the Hamiltonian. This process iterates until the ground state is approximated within the desired accuracy.

2) Data Augmentation: We generate 1000 ground states for 1000 Hamiltonians using VQE, which serve as the training data for our subsequent machine learning process. To enhance the robustness and accuracy of our model, we aim to gather as much data as possible. However, obtaining ground states via VQE is time-consuming. To mitigate this, we apply data augmentation techniques that effectively double our dataset.

This technique exploits the symmetries of the Hamiltonian. The Hamiltonians we use (the TFIM model and the XXZ model as detailed in section II-B) are symmetric with respect to spin flips. Although its ground state is non-degenerate, applying these symmetries to a VQE state results in different states with identical energy, which serve as equally valid data points. Notably, these transformations can be applied to the checkerboard states without altering their structure.

In the ansatz we developed, the two-qubit blocks are followed by Z rotations. Therefore, applying this symmetry involves adjusting the angles in the Z rotations of the final checkerboard layer by ϕ (shown in Fig 2(c)).

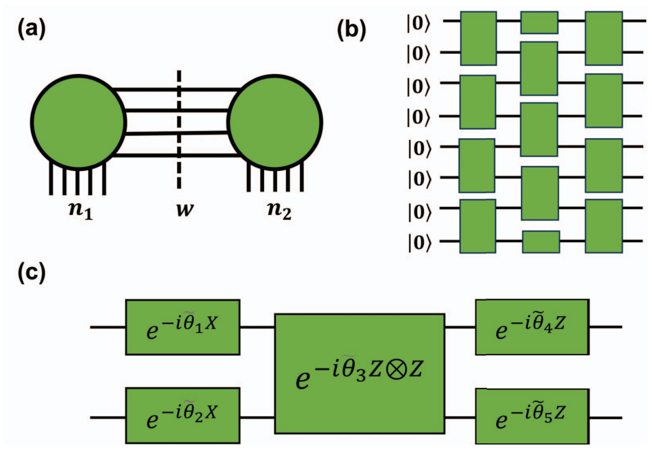
(a)
(b)
n_1
w
n_2
$|0\rangle$
(c)
$e^{-i\theta_1 X}$
$e^{-i\theta_2 X}$
$e^{-i\theta_3 Z\otimes Z}$
$e^{-i\tilde{\theta}_4 Z}$
$e^{-i\tilde{\theta}_5 Z}$

Figure 2. (a) Quantum circuit depicted as a tensor network with bonds of dimension 2. (b) Checkerboard tensor network circuit. Each green block refers to a two-qubit entangler circuit. (c) The entangled block in the checkerboard circuit [24]

Spin flips are treated as a Pauli X gate on all qubits simultaneously. In the checkerboard ansatz that we use, the two-qubit blocks are begin with X rotations and followed by Z rotations. Leveraging the anticommutativity of the X and Z Pauli matrices, we can bring the Pauli X gates to the front:

$$Xe^{i\frac{\theta}{2}Z} = \cos\left(\frac{\theta}{2}\right)X + i\sin\left(\frac{\theta}{2}\right)XZ = e^{-i\frac{\theta}{2}Z}X. \quad (1)$$

Thus, by shifting the Pauli X gates to the left and inverting the angles of the Z rotations, the circuit remains invariant.

The next gate to consider is the $Z \otimes Z$ rotation. Since $[X \otimes X, Z \otimes Z] = 0$, the X gates can pass through the RZZ gate unchanged. Finally, the Pauli X gates combine with the X rotations by incrementing the angle by $\pi/2$. Therefore, to augment the data with spin-flipped states, one should invert the angles of the Z rotations in the final layer and increase the angles of the X rotations in the final layer by $\pi/2$.

Our methodology exemplifies an innovative approach to predicting phase transitions in quantum systems by integrating quantum simulation and quantum machine learning. By leveraging tensor network ansatz states, a quantum classifier, and the computational power of multi-GPU frameworks provided by PennyLane, we achieve significant improvements in speed and scalability, thereby advancing the study of quantum phase transitions.

3) Quantum Convolutional Neural Network Classifier: Quantum Convolutional Neural Network (QCNN) is utilized to predict phase transitions, harnessing the combined strengths of convolutional layers and quantum computing layers [33]. This approach allows for efficient feature extraction from extensive databases and the natural simulation of quantum data. The QCNN algorithm, as utilized in the QCQ architecture, incorporates a single convolutional layer connected with a pooling layer, multiple fully connected layers, and a quantum circuit layer. This architecture is illustrated in Fig. 1 and is explained below:

- **Convolutional Layer**: The model initiates its computational flow with a single convolutional layer. It utilizes a one-dimensional convolution, generating 30 output channels, with a kernel size identical to that of the feature. This design is specifically tailored for the initial extraction of features directly from the input data.
- **Max Pooling Layer**: One max pooling layer, configured with a kernel size of 1, is placed strategically to reduce spatial dimensions and hence, the complexity of the data passing through the network.
- **Fully Connected Layers**: After flattening the feature map of the max pooling layer, the model incorporates two fully connected layers that serve to interpret the features extracted by the convolutional and pooling layers, culminating in the model's ability to make predictions. Notably, the second fully connected layer is specifically designed to interface with the quantum circuit layer, mapping the classical data into a quantum-compatible format.
- **Quantum Circuit Layer**: At the heart of QCNN approach is the quantum circuit layer, invoked through a predefined Pennylane circuit. The circuit is illustrated in Fig. 3. This layer signifies the model's capacity to perform quantum computations, potentially exploiting quantum parallelism and entanglement to enhance the model's predictive capabilities. This quantum layer is followed by another fully connected layer.
- **Dropout**: A dropout layer with a probability (p) of 0.5 is integrated to prevent overfitting by randomly omitting a subset of features during the training phase.
- **Prediction**: The sigmoid activation function is employed to transform the feature map of the final fully connected layer into a probability value ranging between 0 and 1. This mapping facilitates a clear decision-making process for binary classification tasks. The computation of the loss involves measuring the discrepancy between the sigmoid function's output and the designated target values. Upon completion of the training process, the model makes predictions based on a threshold criterion: outputs exceeding 0.5 are classified as 1, indicating the presence of phase transition, while those below 0.5 are classified as 0.

B. Hamiltonians for Solving Phase Transition

The Ising model is a mathematical framework used to describe phase transitions in statistical mechanics, particularly focusing on the behavior of ferromagnetic and antiferromagnetic systems [34]. In our study, we introduce a novel approach that combines quantum simulation with QML to classify phases of matter, addressing the computational challenges faced by classical simulation methods. By employing a variational quantum algorithm, we leverage the capabilities of quantum computers to prepare and classify labelled states derived from the VQE algorithm. This method effectively bypasses the data reading slowdown typically encountered in quantum-enhanced machine learning applications, presenting a significant advancement in the field.

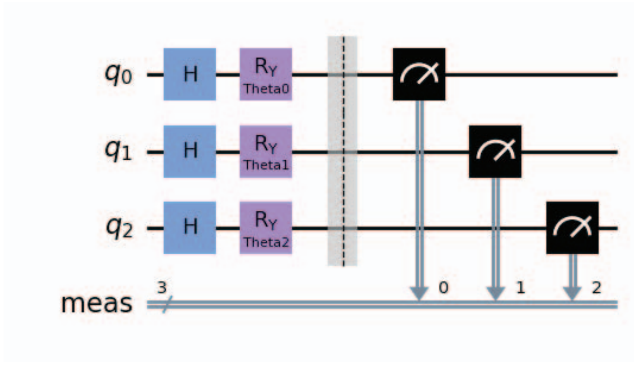

Figure 3. The quantum circuit used in QCNN. In this circuit, a Hadamard gate is first applied to each qubit, followed by a rotational-Y gate. The angles of the rotational gates are the trainable parameters (`Theta1`, `Theta2` and `Theta3`). At the end of this circuit, measurement is executed and the possibilities of finding 7 out of 8 quantum states (except "111") are the output of this circuit. Thus, when regarded as a layer in the QCNN, this circuit has 3 input channels and 7 output channels.

Our work utilizes families of variational ansatz circuits inspired by tensor networks, enabling us to exploit tensor network theory to elucidate properties of phase diagrams. This approach is instrumental in our development of a quantum neural network. These results underscore the potential of integrating quantum simulation with QML to provide deep computational insights into quantum systems.

We represent the following Hamiltonian as a sum of tensor products of Pauli operators:

$$H = \sum_{\alpha_1 \alpha_2 \dots \alpha_n} J_{\alpha_1 \alpha_2 \dots \alpha_n} \sigma_{\alpha_1} \otimes \sigma_{\alpha_2} \otimes \cdots \otimes \sigma_{\alpha_n}, \qquad (2)$$

where $\alpha_i \in \{0,1,2,3\}$ enumerate the Pauli matrices $\{1, X, Y, Z\}$. With the decomposition (1), individual terms of $\langle \psi(\theta) | H | \psi(\theta) \rangle$ can be estimated and variationally minimized elementwise using a classical-to-quantum process. In each iteration, one prepares the state $|\psi(\theta)\rangle$ and measures each qubit in the local X, Y, or Z basis, estimates the energy and updates θ. This method can become scalable only if the number of terms in the Hamiltonian is polynomially bounded in the number of spins and the coefficients $J_{\alpha_1 \alpha_2 \dots \alpha_n}$ are defined up to poly(n) digits.

In particular, we use the transverse field Ising model (TFIM):

$$H_{TFIM} = J \sum_i \sigma_i^z \sigma_{i+1}^z + h \sum_i \sigma_i^x, (J > 0, h > 0) \qquad (3)$$

where σ_i^z are Pauli Z matrix acting on the ith spin. Constant J is known as the energy scale, we set $J = 1$ in our study which means no loss of generality. Dimensionless constant h corresponds to the transverse magnetic field.

When $J > h$, the system has two ground states, with opposite signs of magnetization.When $J < h$ the system has one ground state with zero magnetization. The changing of ground states refers to the phase changing of the system, which happens at $J = h$.

The other model we used is antiferromagnetic XXZ spin chain model:

$$H_{xxz} = -\sum_i J_\perp(\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y) + J_z\sigma_i^z\sigma_{i+1}^z, (J_z < 0) \tag{4}$$

where the sign of J_z determines if the system is ferromagnetic ($J_z > 0$) or antiferromagnetic ($J_z < 0$). The sign of $J_\perp$ is not important in bipartite lattice because the states can be redefined by $\sigma^x \to -\sigma^x, \sigma^y \to -\sigma^y$. We set $J_\perp = 1$ in our study.

For antiferromagnetic XXZ model, when $|J_z| > |J_\perp|$, the ground states spins each takes their opposite values to their neighbours, and the system is in antiferromagnetic Ising state; When $|J_z| < |J_\perp|$, the system is in the planar phase. A Berezinsky–Kosterlitz–Thouless type phase transition happens at $|J_z| = |J_\perp|$ [35].

Conclusively, our research demonstrates the feasibility and effectiveness of using quantum simulation and QML to classify phases of matter. By preparing approximate ground states variationally and employing them as inputs to a quantum classifier, we avoid the limitations of traditional Monte Carlo sampling methods. The success of our nearest-neighbour quantum neural network in accurately identifying phases of matter highlights the promising future of this interdisciplinary approach. This synergy between quantum computing and machine learning opens new avenues for exploring quantum systems, significantly impacting the fields of condensed matter physics and quantum computing.

C. Towrad Distributed Quantum Computing: a Conceptual Architecture for a Distributed-Quantum-HPC Middleware

To develop a conceptual architecture for distributed quantum computing with a QCQ franework, we expand upon the concept based on the work of Saurabh et al [19]. This approach enables a deeper integration of quantum computing processes within distributed computing frameworks. Based on Fig. 4 delineates an integrated architecture designed to leverage distributed quantum resources for advanced computational tasks. The schematic begins with a classical scheduling layer, where a CPU disaggregates a complex problem into sub-problems. These are then converted into quantum states through GPUs in an encoding layer, facilitating the classical-to-quantum transition. The middle of the system can lie in the distributed-VQE layer [25], [36], where QPUs (linked by quantum internet and augmented by quantum repeaters) execute quantum algorithms. Finally, a hybrid quantum-classical neural network analyzes the quantum states, yielding a scalar output via a quantum classifier. This streamlined, multi-layered framework illustrates a scalable approach for executing computationally intensive tasks across distributed quantum systems, aiming to unlock new potentials in near-term quantum information processing.

III. Result

In Fig. 5, we have successfully implemented a quantum machine learning classifier for phase transition using the PennyLane package, enhanced by the computational acceleration

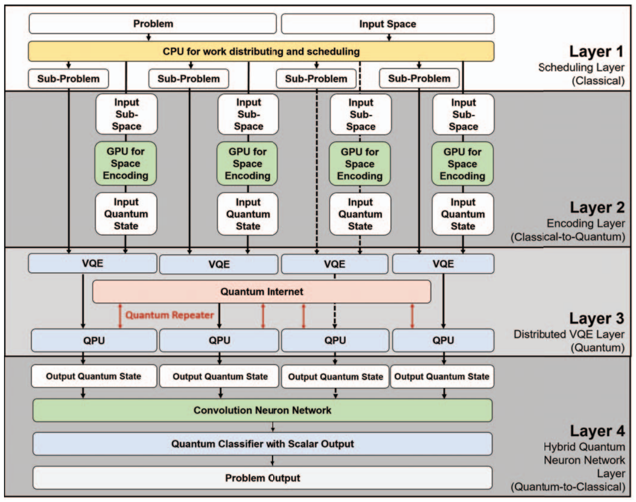

Figure 4. The schematic represents a distributed quantum computing architecture, illustrating the hybrid QCQ framework for classifying phase transitions. This architecture combines CPU-based scheduling with quantum processing across multiple layers and employs GPUs for state encoding. The dashed lines represent omitted n blocks in parallel due to size constraints of the image

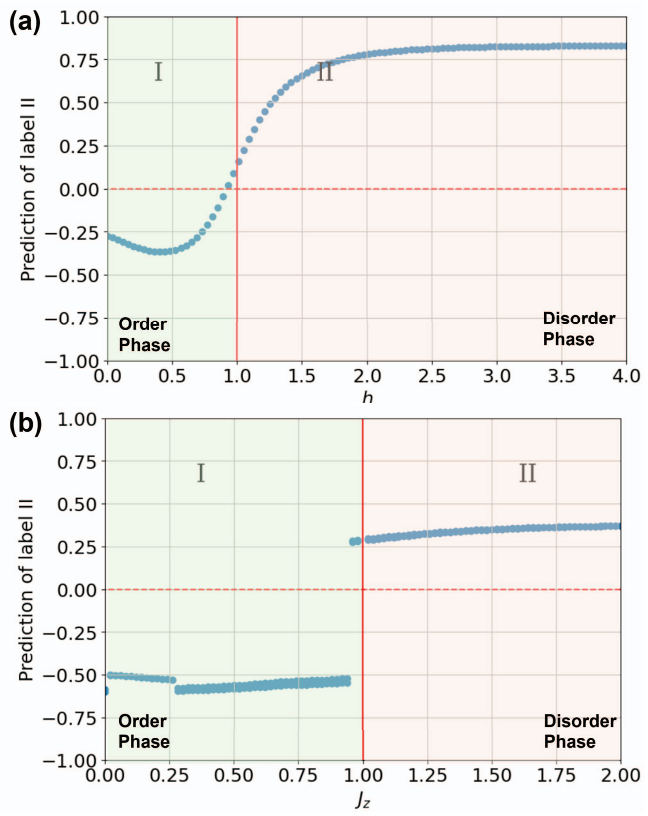

Figure 5. (a)Predicted phases as a function of h for the TFIM model. (b) The predicted probability of phases as a function of Jz for the XXZ model. Positive prediction of label II represents phase II, which is above the dashed lines. The theoretically phase II (disorder phase) is the areas on the right-hand side of the red lines (shown in red color).

of the cuQuantum SDK. In our classification task, we observed an increase in accuracy near the phase transition point when utilizing checkerboard states with increasing depth in our VQE algorithm. Notably, even lower-depth ansatz states such as rank-one, tree tensor network, and single-layer checkerboard states delivered comparable results due to the relatively simple nature of the Hamiltonian involved.

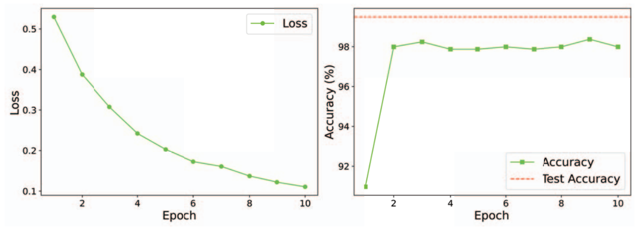

Figure 6. The evolution of loss (left) and accuracy (right) in the QCNN model. As the number of epochs increases, the model's loss decreases exponentially. The model's accuracy improves significantly in the initial epochs and then stabilizes at around 98%. And test accuracy reaches 99.5%.

We prepared a dataset of 100 data points employing a VQE with a four-layered checkerboard ansatz state. After shuffling, this dataset was divided into an 80% training set and a 20% test set. Impressively, the classifier achieved a 98.0% prediction accuracy rate. For the antiferromagnetic XXZ spin chain model, we generated 4000 data points to train the classifier, achieving a 94.6% accuracy rate on the test data after enhancing the classifier circuit with two additional layers.

The QCNN yields results that are both robust and satisfactory (Fig. 6). In the initial phases of the training epochs, the accuracy of the model escalates significantly, demonstrating a rapid learning curve which subsequently reaches a plateau at approximately 98.0%. Further evaluation on a test set reveals that the model attains a remarkable accuracy of 99.5%.

The dataset employed for the QCNN model was generated utilizing the VQE on XXZ model. The dataset comprised a total of 1000 data points. In adherence to standard machine learning practices, the dataset was partitioned in an 80-20 split, with 80% utilized for the training phase and the remaining 20% reserved for testing.

Our approach demonstrates the potential of quantum-enhanced machine learning to classify phases of matter with high accuracy, outperforming classical Monte-Carlo sampling-based methods. This technique is versatile and can be applied to any model expressible as a spin model, with the capability to be extended to multi-class classification problems. Our results underscore the advantages of integrating quantum simulations with advanced machine learning techniques, paving the way for new insights into the study of quantum phase transitions.

A. GPU Acceleration for quantum circuit simulation with cuQuantum SDK

In this research work, our main task is to develop and optimize a Hybrid QML model with a distributed-quantum-computing framework for classifying phase transitions. In our demonstration, we use the PennyLane package, with a focus on computational efficiency through GPU acceleration with cuQuantum SDK. The inherent challenge in QML is the intensive computational requirement, often necessitating QPUs with high-fidelity qubits. To address this, we employ classical hardware for initial data processing and feature extraction—a step integral to our QML model's training process. Our strategy utilizes the cuQuantum SDK, which offers a GPU-optimized collection of low-level primitives supporting Pennylane lightning backend. This approach accelerates quantum machine learning model execution on NVIDIA GPUs, substantially shortens training durations, and reduces computational costs. By running our applications on Denvr Dataworks's CUDA-compatible platform, we benefit from the enhanced processing capabilities of NVIDIA GPUs, which are designed for parallel computing and can therefore handle large datasets and complex operations with greater speed than conventional CPUs. The expected outcome of our work is a threefold acceleration in the training of our quantum machine learning models compared to CPU-only execution. This improvement is attributable to the GPUs' parallel processing power, which allows for quicker data processing and computation. The integration of cuQuantum with CUDA not only augments computational speed but also offers cost-effective solutions by eliminating the need for expensive quantum hardware and extensive infrastructure. Our objective is to make QML models more scalable and accessible, leveraging these technologies to advance research in quantum phase transitions and establish new performance benchmarks in the field of quantum computing.

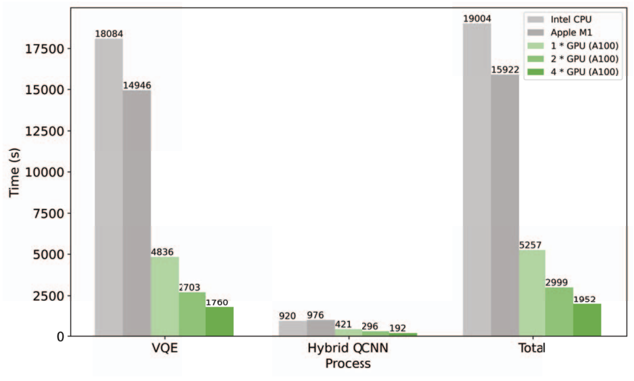

Figure 7. Benchmarking of computation times for VQE, Hybrid QCNN, and Total processes, comparing performance across Intel CPU (i7-13700KF), Apple M1 pro (10-core CPU), and 1, 2, and 4 NVIDIA A100 GPUs as quantum simulators.

In line with our objectives, the benchmarking results showcase a comparative analysis of computational performance across different hardware configurations for these quantum simulations. The bar plot distinctly illustrates the time taken to execute specific tasks like the VQE (in this case with 16 qubits), Hybrid QCNN (in this case with 3 qubits), and the aggregate computational process across Intel CPUs, Apple M1 chips, and NVIDIA A100 GPUs. The acceleration impact is significant when utilizing the A100 GPUs, with the most notable performance improvement observed with a configuration of four GPUs, suggesting a near-linear scaling in this particular benchmarking scenario. This benchmarking also agrees with the result shown in Vallero et. al's work [37].

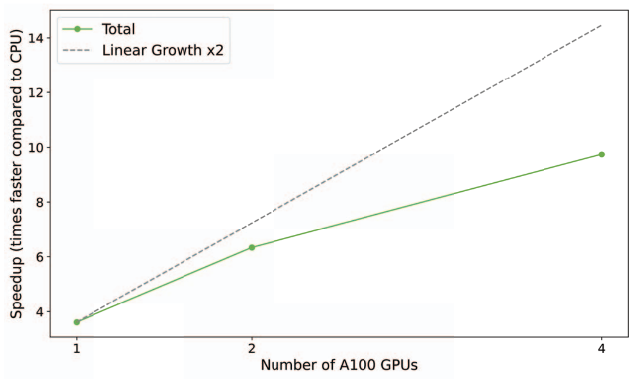

Figure 8. The line graph conducts a linearity performance check of multi-GPU settings for quantum machine learning training.

The accompanying line plot further elucidates the speedup achieved with the use of 1, 2, and 4 A100 GPUs compared to CPU benchmarks. It becomes evident that the total speedup gained from GPU acceleration does not fully align with an ideal linear progression, which is visually represented by a grey dashed line. This deviation implies that while GPU acceleration substantially benefits the computational speed, the returns diminish with the addition of more GPUs, possibly due to overhead associated with parallelization and inter-GPU communication. The less efficient may be because in our case the qubit number is not very large. In Vallero et. al's benchmarking [37], with more qubits we run, the more acceleration we will observe. These findings underscore the A100 GPUs' profound capabilities in enhancing performance for quantum simulation tasks, thereby underscoring their potential to facilitate and expedite complex quantum computations in research settings.

IV. Conclusion

In this study, we propose an innovative QCQ architecture aimed at enhancing applications in distributed quantum computing and Quantum-HPC systems. This architecture integrates VQE algorithms and QCNNs, enabling rapid and accurate quantum simulations, exemplified by phase transition classification within this research. The hybrid QCQ algorithm demonstrated an exceptional 99.5% test accuracy in predicting phase transitions for models such as the transverse field Ising and XXZ. The architecture efficiently utilizes QPUs with a limited number of high-fidelity qubits, such as superconducting circuits [38], ion traps [39], neutral atoms [40], and NV center diamonds [41], to scale up the qubit count and achieve distributed quantum computing.

At the heart of the QCQ framework is the seamless integration of variational quantum eigensolver algorithms, tensor network states, and convolutional neural networks. This synergistic amalgamation allows for the effective preparation of quantum states through quantum simulations, followed by their classification using advanced machine learning techniques. Different from existing research such as ensemble quantum computing (EQC) [42], our Quantum-Classical-Quantum (QCQ) architecture integrates multi-GPU acceleration and CPU distributed task scheduling, enhancing the practicality of Quantum-HPC and achieving better synergy between classical and quantum hardware. This approach enables researchers to use GPU-accelerated quantum circuit simulation to validate small-scale research problems and then scale up to larger problem sets using real quantum hardware within our distributed framework. The architecture also finds applications in quantum reinforcement learning [43] and can optimize the hybrid neural network structure using an $polylog(M)$ approach [44] or the fast weights approach [45].

The success of this QCQ framework highlights the immense potential of merging quantum computing with machine learning. By surmounting the limitations inherent in classical algorithms, this method paves the way for deeper insights into the behavior of quantum systems across various sizes and configurations. It finds applications in fields such as materials science, condensed matter physics, and sustainability research.

As research in quantum hardware and algorithms progresses, architectures like QCQ will play a critical role in unlocking the transformative capabilities of quantum-enhanced computing. While challenges in further enhancing accuracy and extending applicability remain, the scalability and efficiency exhibited by this framework open up exciting avenues for future exploration. Integrating the strengths of both quantum and classical computing, the QCQ architecture represents a significant advancement towards fully leveraging the potential of quantum technologies in diverse scientific and technological domains.

Acknowledgment

The authors extend their gratitude to Andreas Hehn, Tobi Odufeso, and Tom Lubowe for invaluable assistance and insights, which have been pivotal to the success of this research work. This research was supported by Xanadu, NVIDIA, Amazon, and Denvr Dataworks under QHack 2024. K.C. is grateful for the financial support from both the Turing Scheme for the Imperial Global Fellows Fund and the Imperial QuEST Seed Fund. Special acknowledgement goes to the Quantum Open Source Foundation (QOSF) and QuantumPedia AI for their contributions and their quantum computing mentorship program.

Code Availability

The source code for generating the dataset and figures presented in this research work is openly accessible at https://github.com/Louisanity/cuPhastLearn.

References

[1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.

[2] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, "Parameterized quantum circuits as machine learning models," *Quantum Science and Technology*, vol. 4, no. 4, p. 043001, 2019.

[3] V. S. Ngairangbam, M. Spannowsky, and M. Takeuchi, "Anomaly detection in high-energy physics using a quantum autoencoder," *Physical Review D*, vol. 105, no. 9, p. 095004, 2022.

[4] C. Zoufal, A. Lucchi, and S. Woerner, "Quantum generative adversarial networks for learning and loading random distributions," *npj Quantum Information*, vol. 5, no. 1, p. 103, 2019.

[5] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, "Transfer learning in hybrid classical-quantum neural networks," *Quantum*, vol. 4, p. 340, 2020.

[6] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo, "Quantum convolutional neural networks for high energy physics data analysis," *Physical Review Research*, vol. 4, no. 1, p. 013231, 2022.

[7] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, "The power of quantum neural networks," *Nature Computational Science*, vol. 1, no. 6, pp. 403–409, 2021.

[8] K.-C. Chen, X. Xu, H. Makhanov, H.-H. Chung, and C.-Y. Liu, "Quantum-enhanced support vector machine for large-scale stellar classification with gpu acceleration," *arXiv preprint arXiv:2311.12328*, 2023.

[9] S. C. Marshall, C. Gyurik, and V. Dunjko, "High dimensional quantum machine learning with small quantum computers," *Quantum*, vol. 7, p. 1078, 2023.

[10] P. S. Mundada, A. Barbosa, S. Maity, Y. Wang, T. Merkh, T. Stace, F. Nielson, A. R. Carvalho, M. Hush, M. J. Biercuk *et al.*, "Experimental benchmarking of an automated deterministic error-suppression workflow for quantum algorithms," *Physical Review Applied*, vol. 20, no. 2, p. 024034, 2023.

[11] K.-C. Chen, "Short-depth circuits and error mitigation for large-scale ghz-state preparation, and benchmarking on ibm's 127-qubit system," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 2. IEEE, 2023, pp. 207–210.

[12] H.-Y. Huang, M. Broughton, J. Cotler, S. Chen, J. Li, M. Mohseni, H. Neven, R. Babbush, R. Kueng, J. Preskill *et al.*, "Quantum advantage in learning from experiments," *Science*, vol. 376, no. 6598, pp. 1182–1186, 2022.

[13] S. Monaco, O. Kiss, A. Mandarino, S. Vallecorsa, and M. Grossi, "Quantum phase detection generalization from marginal quantum neural network models," *Physical Review B*, vol. 107, no. 8, p. L081105, 2023.

[14] N. Grzesiak, R. Blümel, K. Wright, K. M. Beck, N. C. Pisenti, M. Li, V. Chaplin, J. M. Amini, S. Debnath, J.-S. Chen *et al.*, "Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer," *Nature communications*, vol. 11, no. 1, p. 2963, 2020.

[15] D. Cuomo, M. Caleffi, and A. S. Cacciapuoti, "Towards a distributed quantum computing ecosystem," *IET Quantum Communication*, vol. 1, no. 1, pp. 3–8, 2020.

[16] Y. Alexeev, M. Amsler, P. Baity, M. A. Barroca, S. Bassini, T. Battelle, D. Camps, D. Casanova, F. T. Chong, C. Chung *et al.*, "Quantum-centric supercomputing for materials science: A perspective on challenges and future directions," *arXiv preprint arXiv:2312.09733*, 2023.

[17] D. Awschalom, K. K. Berggren, H. Bernien, S. Bhave, L. D. Carr, P. Davids, S. E. Economou, D. Englund, A. Faraon, M. Fejer *et al.*, "Development of quantum interconnects (quics) for next-generation information technologies," *PRX Quantum*, vol. 2, no. 1, p. 017002, 2021.

[18] M. Prest and K.-C. Chen, "Quantum-error-mitigated detectable byzantine agreement with dynamical decoupling for distributed quantum computing," *arXiv preprint arXiv:2311.03097*, 2023.

[19] N. Saurabh, S. Jha, and A. Luckow, "A conceptual architecture for a quantum-hpc middleware," in *2023 IEEE International Conference on Quantum Software (QSW)*. IEEE, 2023, pp. 116–127.

[20] NVIDIA, "cuQuantum SDK: Simulating quantum circuits on GPUs," https://developer.nvidia.com/cuquantum-sdk, 2021, accessed: February 23, 2023.

[21] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi *et al.*, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," *arXiv preprint arXiv:1811.04968*, 2018.

[22] H. Gao and N. Sakharnykh, "Scaling joins to a thousand gpus." in *ADMS@ VLDB*, 2021, pp. 55–64.

[23] K. Wintersperger, H. Safi, and W. Mauerer, "Qpu-system co-design for quantum hpc accelerators," in *International Conference on Architecture of Computing Systems*. Springer, 2022, pp. 100–114.

[24] A. Uvarov, A. Kardashin, and J. D. Biamonte, "Machine learning phase transitions with a quantum processor," *Physical Review A*, vol. 102, no. 1, p. 012415, 2020.

[25] I. Khait, E. Tham, D. Segal, and A. Brodutch, "Variational quantum eigensolvers in the era of distributed quantum computers," *arXiv preprint arXiv:2302.14067*, 2023.

[26] C.-Y. Liu, C.-H. A. Lin, and K.-C. Chen, "Learning quantum phase estimation by variational quantum circuits," *arXiv preprint arXiv:2311.04690*, 2023.

[27] S. DiAdamo, M. Ghibaudi, and J. Cruise, "Distributed quantum computing and network control for accelerated vqe," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–21, 2021.

[28] Y. Du, Y. Qian, X. Wu, and D. Tao, "A distributed learning scheme for variational quantum algorithms," *IEEE Transactions on Quantum Engineering*, vol. 3, pp. 1–16, 2022.

[29] M. Modani, A. Banerjee, and A. Das, "Multi-gpu-enabled quantum circuit simulations on hpc-ai system," in *International Conference on Data Management, Analytics & Innovation*. Springer, 2023, pp. 845–854.

[30] J.-G. Liu, Y.-H. Zhang, Y. Wan, and L. Wang, "Variational quantum eigensolver with fewer qubits," *Physical Review Research*, vol. 1, no. 2, p. 023025, 2019.

[31] D. G. Utkarsh Azad, "Pennylane spin datasets," https://pennylane.ai/datasets/qspin/ising-model, 2023.

[32] H.-Y. Huang, R. Kueng, and J. Preskill, "Predicting many properties of a quantum system from very few measurements," *Nature Physics*, vol. 16, no. 10, pp. 1050–1057, 2020.

[33] I. Cong, S. Choi, and M. D. Lukin, "Quantum convolutional neural networks," *Nature Physics*, vol. 15, no. 12, pp. 1273–1278, 2019.

[34] J. Yeomans, "The theory and application of axial ising models," in *Solid State Physics*. Elsevier, 1988, vol. 41, pp. 151–200.

[35] F. Franchini, *An Introduction to Integrable Techniques for One-Dimensional Quantum Systems*. Springer International Publishing, 2017. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-48487-7

[36] R. Parekh, A. Ricciardi, A. Darwish, and S. DiAdamo, "Quantum algorithms and simulation for parallel and distributed quantum computing," in *2021 IEEE/ACM Second International Workshop on Quantum Computing Software (QCS)*. IEEE, 2021, pp. 9–19.

[37] M. Vallero, F. Vella, and P. Rech, "State of practice: evaluating gpu performance of state vector and tensor network methods," *arXiv preprint arXiv:2401.06188*, 2024.

[38] J. Clarke and F. K. Wilhelm, "Superconducting quantum bits," *Nature*, vol. 453, no. 7198, pp. 1031–1042, 2008.

[39] D. Kielpinski, C. Monroe, and D. J. Wineland, "Architecture for a large-scale ion-trap quantum computer," *Nature*, vol. 417, no. 6890, pp. 709–711, 2002.

[40] D. S. Weiss and M. Saffman, "Quantum computing with neutral atoms," *Physics Today*, vol. 70, no. 7, pp. 44–50, 2017.

[41] L. Childress and R. Hanson, "Diamond nv centers for quantum computing and quantum networks," *MRS bulletin*, vol. 38, no. 2, pp. 134–138, 2013.

[42] S. Stein, N. Wiebe, Y. Ding, P. Bo, K. Kowalski, N. Baker, J. Ang, and A. Li, "Eqc: ensembled quantum computing for variational quantum algorithms," in *Proceedings of the 49th annual international symposium on computer architecture*, 2022, pp. 59–71.

[43] S. Y.-C. Chen, C.-H. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, "Variational quantum circuits for deep reinforcement learning," *IEEE Access*, vol. 8, pp. 141 007–141 024, 2020.

[44] C.-Y. Liu, E.-J. Kuo, C.-H. A. Lin, S. Chen, J. G. Young, Y.-J. Chang, and M.-H. Hsieh, "Training classical neural networks by quantum machine learning," *arXiv preprint arXiv:2402.16465*, 2024.

[45] S. Yen-Chi Chen, "Learning to program variational quantum circuits with fast weights," *arXiv e-prints*, pp. arXiv–2402, 2024.