To the Graduate Council:
I am submitting herewith a thesis written by Shikha Bangar entitled "Novel Quantum Algorithms for Noisy Intermediate-Scale Quantum (NISQ) Devices." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Physics.

George Siopsis, Major Professor

We have read this thesis
and recommend its acceptance:

Dr. George Siopsis

Dr. Adrian Del Maestro

Dr. Joon Sue Lee

Dr. Himanshu Thapliyal

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

To the Graduate Council:
I am submitting herewith a thesis written by Shikha Bangar entitled "Novel Quantum Algorithms for Noisy Intermediate-Scale Quantum (NISQ) Devices." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Physics.

George Siopsis, Major Professor

We have read this thesis
and recommend its acceptance:

Dr. George Siopsis

Dr. Adrian Del Maestro

Dr. Joon Sue Lee

Dr. Himanshu Thapliyal

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Novel Quantum Algorithms for Noisy Intermediate-Scale Quantum (NISQ) Devices

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Shikha Bangar

August 2024

© by Shikha Bangar, 2024
All Rights Reserved.

To my family

Acknowledgements

I want to extend my deepest gratitude to the many individuals who have supported and enriched my journey through graduate school. First, I must thank Dr. George Siopsis for being a wonderful supervisor. He supported me throughout my Ph.D. to explore different ideas and work hard to get desirable results. My sincere thanks also go to my Ph.D. committee—Dr. Adrian Del Maestro, Dr. Joon Sue Lee, and Dr. Himanshu Thapliyal, whose insightful suggestions significantly shaped my dissertation. Additionally, I am grateful for the collaboration and support of Kubra Yeter-Ayedeniz and Leanto Sunny. Kubra, in particular, was instrumental in guiding my early research efforts, always ready to assist and offer advice on Slack.

I am fortunate to have forged lasting friendships during this period, beginning with my cohort friends Igor and Jinu. Our weekly meals, where we shared our academic struggles and triumphs, provided a comforting sense of community as I navigated life in a new country. I also cherish the bonds formed with Elias, Anjali, Vanessa, and Harini, who have supported me during challenging times and celebrations. Additionally, I thank my therapy friends, Amanda and Rachel, for sharing their unique experiences through graduate school, which helped us grow closer despite our diverse backgrounds. A special mention goes to my best friend, Ritwika, who, despite the distance, has been a constant source of stability for nearly a decade. Weekly, at least an hour-long call with her kept me going.

Next, I would like to express my heartfelt gratitude to my family, which includes my incredibly supportive husband, Rahul, and our two stress-relieving cats, Casper

and Apsara. Rahul has been my rock, always there to help with homework and research, keep our home clean, and cook fantastic meals. His unwavering support has undoubtedly made my Ph.D. journey a hundred times easier. And our cats, who sometimes sleep on my study table, remind me of the importance of taking breaks and self-care.

Finally, I am grateful to my parents (Surinder and Sunita) back in India, who have always supported and encouraged me to achieve my goals. Their daily conversations and gentle reminders of my values have been a constant source of strength. My sister Shivani, in her unique way, has been a great support, sending me memes and providing me with any help I need. Their unwavering support has paved the way for me to pursue a Ph.D. in another country.

Apart from friends and family, I am deeply grateful to the Physics Department staff for their unwavering support and guidance. This includes my excellent professors, Dr. Batista, Dr. Brienig, and Dr. Johnston. I also connected with other women professors in the department as they were always available when I needed them - Dr. Jones, Dr. Fomin, and Dr. Holmes. I also felt supported by the student health center and counseling center during the times when I struggled with my physical and mental health. Finally, I am grateful to the Aroma restaurant in Knoxville for providing good Indian food away from home. Lastly, I am also thankful to Taylor Swift for releasing new and old albums, which have brought some happiness and fun during the challenging time of my Ph.D.

This acknowledgment would be incomplete without mentioning each of you who touched my life during these crucial years. Thank you all for being part of my journey.

The universe doesn't allow perfection.

— Stephan Hawking

Abstract

Quantum computers hold the immense potential to revolutionize the current computing technology. However, the available quantum hardware, known as noisy intermediate-scale quantum (NISQ) devices, poses a challenge as these devices have noise and imperfect qubits. Hence, there is a critical need to develop novel algorithms specifically designed to harness the capabilities of these devices. This work aims to develop and optimize such quantum algorithms for implementation on current quantum technologies by integrating principles from physics and machine learning. First, we developed the Quantum Imaginary Time Evolution (QITE) and Quantum Lanczos (QLanczos) algorithms on discrete-variable quantum computing settings to study the eigensystem of a many-body quantum system. We applied them to a system of collective oscillating neutrinos. The final results agreed with the exact results demonstrating that a neutrino system can be studied on quantum hardware. Subsequently, we constructed continuous-variable (CV) quantum neural networks, which can be experimentally realized as they are made up exclusively of Gaussian gates, with nonlinearity introduced via measurements. The network was tested across various scenarios in a supervised learning setting —curve fitting, fraud detection, image classification, and state preparation. The outcomes were consistently positive, achieving over 93% accuracy across all studied cases. Furthermore, we constructed a quantum Boltzmann machine in the CV framework. This machine is designed to generate probability distributions for classical (synthetic aperture radar images) and quantum data (Gaussian and non-Gaussian (cat) states). In each instance, the

Kullback-Leibler (KL) divergence between the target and generated distributions was calculated and found to be very close to zero, indicating an excellent match between the generated and actual distributions.

In this work, we have successfully advanced the field of quantum computing by developing and optimizing algorithms tailored for NISQ devices. Future research should expand the scope of systems studied using these algorithms, continually refining these algorithms through advanced error mitigation and machine learning techniques. The next goal in the CV setting is constructing quantum neural networks and Boltzmann machines in a lab using advanced photonic quantum computers. These algorithms hold promise for diverse applications, from medical imaging and gravitational wave detection to preparing complex quantum states.

Table of Contents

1 Introduction **1**

1.1 Quantum Computing 4

1.1.1 Discrete-variable quantum computing 4

1.1.2 Continuous-variable quantum computing 6

1.2 NISQ devices era 9

1.3 Quantum Algorithms for NISQ Devices 10

1.3.1 Variational Quantum Algorithms 10

1.3.2 Quantum Machine Learning 14

2 Discrete-variable Quantum Eigensolver **18**

2.1 Quantum Algorithms 21

2.1.1 QITE Algorithm 21

2.1.2 Quantum Lanczos 24

2.2 Collective Neutrino Oscillation 26

2.2.1 Background 26

2.2.2 Model 28

2.3 Implementation of Quantum Algorithms 29

2.4 Results and Discussions 38

2.5 Conclusion 46

3 Continuous-variable Quantum Neural Network **48**

3.1 The Method 51

3.2 Case Studies . . . 57
3.2.1 State Preparation . . . 57
3.2.2 Curve Fitting . . . 67
3.2.3 Binary classification . . . 72
3.2.4 Multi-label classification . . . 79
3.3 Conclusion . . . 81

4 Continuous-variable Quantum Boltzmann Machine 85
4.1 The Method . . . 90
4.2 Classical Data Probability Distribution Generation . . . 97
4.3 Quantum Data Probability Distribution Generation . . . 105
4.4 Experimental Feasibility . . . 108
4.5 Conclusion . . . 114

5 Conclusions and Future Work 116

Bibliography 121

Vita 142

List of Tables

1.1 Comparison between the DV and CV quantum computing 7

2.1 Calibration data for three different 7-qubit IBM Quantum systems. . 42

2.2 Comparison of the mean values for three different 7-qubit IBM Quantum systems. 43

4.1 Description of parameters used for different distributions and their comparison with real-world applications. 104

4.2 Range of trained parameters of gates in the CVQBM circuit 109

4.3 Achieved fidelity and success rates in use cases 111

List of Figures

1.1 Variational Quantum Algorithm setup 11
1.2 Architecture of CV quantum neural network 15

2.1 Quantum circuit obtained using IBM Qiskit's *isometry* function . . . 34
2.2 Readout-error mitigated (ROEM) energy eigenvalues of the 4-neutrino system 36
2.3 Energy vs imaginary time plot for (a)2-qubit and (b)3-qubit case . . . 37
2.4 Transition probability amplitude comparison 37
2.5 IBM Quantum 7-qubit devices qubit layout 41
2.6 Comparison between the transitional probability calculated using the three different IBM Quantum devices 41

3.1 Two-qumode Gaussian quantum circuit element introducing non-linearity 50
3.2 Detailed CV neural network architecture 53
3.3 Counts vs. Number of loops plot 55
3.4 Training and testing loss of MNIST classification models 55
3.5 CV quantum NN architecture for state preparation 58
3.6 Study of the cost function for a different number of layers for preparation of single-photon state 58
3.7 Study of the effect of changing the number of layers of the QNN for the preparation of a single-photon state 61

3.8 Preparation of an even cat state 62
3.9 Preparation of a realistic GKP state 62
3.10 Impact of transmissivity (T) on fidelity for single photon, cat, and GKP states 65
3.11 Wigner functions of the single photon state at different transmissivity 66
3.12 CV quantum NN architecture for curve fitting. 68
3.13 The effect of changing the number of layers on curve fitting of a noisy sine function 69
3.14 Cost function values vs. the number of layers plot 69
3.15 The effect of increasing the error in the data on curve fitting of a noisy sine function 71
3.16 Plot of cost function value vs. noise in data 71
3.17 Data preparation for binary classification problem 73
3.18 CV hybrid NN for fraudulent transaction detection 73
3.19 Results of fraud detection in credit card transactions 75
3.20 Effect of classical: quantum layer ratio on the accuracy 75
3.21 CV hybrid NN architecture used for MNIST image classification. . . 78
3.22 MNIST hybrid NN 78
3.23 Loss plots for selected ratios of classical-quantum layers 80

4.1 The CVQBM framework 89
4.2 Designing the CVQBM circuit 89
4.3 Studying SAR image for forest monitoring in Brazil 96
4.4 Studying the SAR image for sea ice monitoring 99
4.5 Generating probability distributions for commonly used distributions 101
4.6 Gadget for quantum data 104
4.7 Generating continuous probability distributions from quantum data . 107
4.8 Studying various distributions under the loss channel 111
4.9 KL divergence vs. loss parameter 113

Chapter 1

Introduction

> *Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly, it's a wonderful problem because it doesn't look so easy.*
>
> — Richard Feynman

In 1981, Richard Feynman proposed the idea of quantum computers by suggesting that classical computers are inherently inadequate for simulating quantum systems accurately and efficiently. Instead, he envisioned machines built on quantum mechanical principles that could one day perform these simulations with outstanding efficiency [1]. *Quantum Computers* are machines that use the concepts of quantum physics to solve problems. Specifically, the goal of a quantum computer is to solve problems more efficiently than a classical computer by utilizing quantum mechanical properties, such as entanglement and superposition, to offer exciting computational capabilities necessary for the simulation of complex quantum systems. The outperforming of the classical computer by a quantum computer is known as the quantum advantage or supremacy, achievable through improvements in speed, accuracy, or resource efficiency. In recent times, there have been various claims of quantum advantage, for example, Google's quantum supremacy claim [2], using shallow circuits [3, 4] and machine learning techniques [5, 6, 7]. However, the industry

and academia primarily view these claims as theoretical or have challenged them, as no one has yet conclusively demonstrated a clear quantum advantage on existing hardware.

Quantum computers are intricate to build and program, frequently encountering errors such as noise, faults, and loss of quantum coherence. These challenges have spurred the development of a significant sub-field known as Noisy Intermediate-Scale Quantum (NISQ) technology. Despite their limitations, NISQ devices are invaluable for exploring complex many-body quantum physics problems [8]. Indeed, quantum computers will not instantly offer a groundbreaking way to look at the world, but NISQ devices will bring us closer to building powerful quantum technologies. In contrast, the fault-tolerant quantum computer represents the other end of the spectrum. This type of quantum computer incorporates a framework that shields qubits from quantum errors and can identify and control error propagation [9]. The concept of quantum fault tolerance specifically addresses the uncontrollable cascade of errors caused by qubit interactions, which are critical for preserving quantum information integrity [10].

In recent years, quantum computation has been a topic of interest for academia and industry. On one hand, researchers are working on using quantum computers to study various domains in sciences [11, 12, 13, 14, 15]. On the other hand, in industry, big companies like IBM [16], Microsoft [17], Amazon [18], and Google [19] are racing with startup companies like Xanadu [20], DWave [21], and IonQ [22] to build the most successful quantum computer that can offer a quantum advantage. These entities employ diverse strategies to achieve this goal: IBM utilizes superconducting qubits, Xanadu focuses on photonic quantum computers, and IonQ employs ion-trapping technologies. Parallelly, there has been a massive development in the field of quantum algorithms.

Quantum algorithms are rules (usually in the form of a quantum circuit) that one can apply on a quantum computer to get desired results. These algorithms leverage the quantum properties, allowing quantum computers to solve problems more

efficiently or faster than classical computers [23]. In 1994, Peter Shor developed Shor's algorithm, which runs exponentially faster than the best-known classical algorithm for factoring into prime numbers [24]. Another algorithm demonstrating quantum computer speed-up is Grover's algorithm for searching an unstructured database. Although these algorithms are powerful, they are challenging to implement on current quantum computers, primarily due to the absence of fault tolerance. This limitation makes developing efficient quantum algorithms for existing hardware particularly challenging.

This thesis focuses on devising novel quantum algorithms specifically for NISQ devices, aiming to overcome the inherent noise challenges and demonstrate practical quantum advantages for specific applications. The proposed algorithms, developed using principles of quantum physics combined with error correction and/or machine learning techniques, include a study on discrete-variable quantum eigensolver with collective neutrino oscillations [25] as an application (discussed in Chapter 2), an experimentally realizable continuous-variable (CV) quantum neural network [26] (discussed in Chapter 3), and a CV Quantum Boltzmann Machine [27] (discussed in Chapter 4). This chapter lays the foundational knowledge required to understand the research projects detailed in this thesis. It elucidates the current research landscape, setting the stage for the discussions in subsequent sections.

First, in Sec. 1.1, we will delve into quantum computing and its different types - discrete and continuous variables. Then, in the next Sec. 1.2, we will focus on the NISQ era of quantum computing and discuss its limitations and challenges. Lastly, in Sec. 1.3, we will explore the development of quantum algorithms tailored for NISQ devices, addressing their potential and constraints. During this journey, we will build the connection between the fundamental topics and their consequent research project.

1.1 Quantum Computing

As defined earlier, quantum computers are advanced machines that harness quantum physics principles to solve problems, potentially surpassing classical computers by exploiting unique properties such as superposition and entanglement. The foundational steps towards physical implementation were outlined in a 2000 article [28], which established essential criteria for constructing a functional quantum computer. These criteria, known as DiVincenzo's criteria, include:

- A well-defined state space for qubits
- The ability to initialize the state of qubits
- Sufficiently long decoherence times to perform operations
- A universal set of quantum gates
- The capability for qubit-specific measurements

These criteria have guided researchers in developing and refining quantum computers, setting benchmarks that ensure the practical feasibility of quantum computations.

Researchers broadly categorize the study of quantum computing into two approaches: discrete-variable (DV) and continuous-variable (CV) quantum computing. Each approach offers distinct advantages and disadvantages. In this section, we will delve deeper into the properties and operational mechanisms of both DV and CV quantum computing setups, exploring their unique characteristics, potential applications, and the challenges they face in current research.

1.1.1 Discrete-variable quantum computing

The discrete-variable (DV) quantum computers are based on the principles of superconducting and ion-trapped quantum computers. The fundamental information unit of DV quantum computers is known as *qubits*, which comes from combining the

two words - quantum and bits. A qubit is a two-state quantum mechanical system $\{|0\rangle, |1\rangle\}$. A single qubit can be described as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.1}$$

where α and β are probability amplitudes. If we measure this qubit, then there is $|\alpha|^2$ probability of finding it in $|0\rangle$ and $|\beta|^2$ for $|1\rangle$.

There are various ways to build superconducting qubits [29] using electrical circuit components like capacitors, Josephson junctions, and inductors. They use superconductivity and Josephson junction to construct a 2-level energy system, transforming a circuit into an artificial atom. The key property of superconducting circuits based on the Josephson effect that makes them particularly effective for use as qubits is the non-linear inductance of the Josephson junction. This non-linearity allows for the creation of anharmonic (non-equally spaced) energy levels, which are crucial for selectively addressing and controlling quantum states without affecting other states, enabling precise qubit manipulation. There has been much research on this kind of qubits; hence, they perform faster and have low errors. Also, most quantum algorithms published work is being tested on these qubits. The best superconducting devices can achieve a coherence time of millisecond and gate fidelity around 99.5-99.9% [30]. However, the current NISQ superconducting devices still do not outperform the best classical computers in all the scenarios. There are still some disadvantages like they experience decoherence quickly and can only operate in a very cold environment. Also, these qubits are connected to the qubits next to them, limiting the circuits' size and depth.

In this work, we used the DV quantum computing setup while studying quantum eigensolver. We developed quantum algorithms - quantum imaginary time evolution (QITE) and quantum Lanczos (QLanczos) that can determine the eigenvalues and eigenvectors of a many-body quantum system. We applied this set of algorithms to study the collective neutrino oscillations on the quantum hardware. Using the

error correction techniques and the symmetry of the Hamiltonian, we analyzed the eigensystem and demonstrated the collective neutrino oscillating on a quantum computer. This research project is detailed in Chapter 2 and published in [25].

1.1.2 Continuous-variable quantum computing

Continuous-variable (CV) quantum computing is based on physically continuous systems, such as photonic ones, as their names suggest. These systems offer an alternative to qubit-based quantum computation. The CV model naturally fits with continuous quantum operators such as position and momentum and bosonic systems such as harmonic oscillators, electromagnetic fields, and phonons. A quantum mechanic model, such as the CV model, can be studied in wavefunction representation and phase space formulation. They offer equivalent perspectives, making the CV model successful in wave-like and particle-like problems. For this work, we will consider only wavefunction representation.

The fundamental element of CV quantum computing is qumodes. A qumode is a bosonic mode that carries essential information. Consider the most elementary CV system, the bosonic harmonic oscillator, then the operators $\hat{x}$ and $\hat{p}$ are defined as:

$$\hat{x} = \sqrt{\frac{\hbar}{2}}(\hat{a} + \hat{a}^\dagger) \tag{1.2}$$

$$\hat{p} = -i\sqrt{\frac{\hbar}{2}}(\hat{a} - \hat{a}^\dagger) \tag{1.3}$$

where $\hat{a}$ and $\hat{a}^\dagger$ are creation and annihilation operators, respectively. The qumode can be considered as a fixed harmonic mode. The continuous nature of qumodes is obvious when we compare them with qubits. The qubits use a discrete set of coefficients, while qumodes use the continuum.

$$\begin{aligned} \text{Qubit} \quad |\phi\rangle &= \alpha\,|0\rangle + \beta\,|1\rangle \\ \text{Qumode} \quad |\psi\rangle &= \int dx\,\psi(x)\,|x\rangle \end{aligned}$$

Table 1.1: Comparison between the DV and CV quantum computing

Properties	DV	CV
Fundamental unit	qubit	qumode
Basis vectors	$\|0\rangle, \|1\rangle$	$\|0\rangle, \|1\rangle, \|2\rangle \ldots \|n\rangle \ldots$
Quantum state	$\|\psi\rangle = \alpha \|0\rangle + \beta \|1\rangle$	$\|\psi\rangle = \int_{-\infty}^{\infty} dx \psi(x) \|x\rangle$
Physical examples	electron spins, photon polarization	photon quadratures
Types	superconducting, neutral atoms	photonic

Also, the relevant operators for qubits are Pauli matrices ($\hat{\sigma}_x, \hat{\sigma}_y, \hat{\sigma}_z$), and for qumodes are quadrature operators ($\hat{x}$ and $\hat{p}$) and mode operators ($\hat{a}$ and $\hat{a}^\dagger$). A summary of the characteristic differences between the two types of computation is presented in Table . 1.1

An efficient way to build a CV quantum computer is through the use of photons [31]. The photons do not interact with each other or the environment, giving them very high coherence time. Unfortunately, this characteristic also makes implementing quantum gates difficult. Nevertheless, the photonic quantum computer has advantages over the other available quantum computers - as it allows room-temperature quantum computing (as photons rarely interact with their surroundings) compared to the superconducting qubits, which require extremely low temperatures to operate. Also, photons also allow distributed quantum computing, which involves connecting multiple quantum processors or nodes to create a more robust and scalable quantum network. Photons are ideally suited for this role because they can act as "flying qubits", quantum bits that transmit quantum information over long distances without significant loss. Since photons are the quanta of light, they are inherently capable of traveling at the speed of light through optical fibers or free space, making them natural carriers for quantum information in a distributed computing setup. Lastly, photonic quantum computers are compatible with the Complementary Metal-Oxide-Semiconductor (CMOS) technologies. This compatibility means that photon-based quantum devices could potentially be integrated with traditional electronic circuits. This integration could lead to more compact and efficient quantum computing systems that combine optical quantum computing elements with standard electronic components, driving the advancement of quantum computing towards more practical and accessible applications.

In this work, we have used the CV quantum computing setup for constructing the quantum neural network (details in Chapter 3 and published in the [26]) and quantum Boltzmann machine (details in Chapter 4 and published in [27]). The CV

setup offers a natural fit for many machine learning concepts discussed in Section 1.3.2, especially for NISQ devices.

1.2 NISQ devices era

Since the inception of quantum computing, the feasibility of this field has been rigorously questioned, primarily due to the phenomenon of decoherence. Decoherence is a process where a quantum system interacts with its environment, gradually losing its quantum properties. This interaction presents significant challenges in the management and control of quantum computers, thus highlighting the complexity of this field and the need for further research.

The notable algorithms like Shor's and Grover's have been demonstrated on early-stage quantum computers, but scaling them to leverage real quantum advantages remains challenging due to currently available noisy quantum computers (as we lose precision in the results). To achieve quantum advantage, we need a fault-tolerant quantum computer. There are various ways to achieve fault tolerance - we either improve the physical qubits to be easily managed and controlled or rely on error-correcting codes. In 1995, Shor demonstrated that encoding the information of one logical qubit into a highly entangled state of multiple qubits can protect the information against errors [10]. This discovery has spurred extensive research into quantum error correction [32]. In theory, quantum computers can be scaled up by maintaining errors and noise below a critical threshold with these error-correcting codes. Unfortunately, based on current codes, constructing useful logical qubits from physical ones would require millions of qubits.

NISQ computing, which stands for Noisy Intermediate-Scale Quantum computing, was coined to precisely describe the current era of quantum computing [8]. This term emphasizes the hardware aspect of quantum computers, which directly results from the current limitations. In NISQ, "noisy" refers to the significant errors and decoherence rates restricting their computational powers. The "Intermediate scale"

indicates that these devices are large enough to demonstrate some quantum properties but insufficient to be error-free or exhibit the full quantum advantage. Therefore, the NISQ era of quantum computers represents a crucial step toward developing fault-tolerant quantum computers, but it is not expected to fully showcase the potential of quantum computing.

1.3 Quantum Algorithms for NISQ Devices

As discussed in the previous section, NISQ devices are characterized by a limited number of qubits with restricted connectivity and both coherent and incoherent errors that constrain the depth of quantum circuits. Consequently, the goal is to optimally utilize current NISQ devices to achieve some level of quantum advantage. In this section, we will explore strategic algorithms researchers have employed to meet this objective. Initially, we will examine the widely utilized variational quantum algorithms applicable to a range of scenarios. Subsequently, we will delve into quantum machine learning, which adopts a learning-based approach to quantum circuits.

1.3.1 Variational Quantum Algorithms

Variational Quantum Algorithms (VQAs) are not just theoretical concepts but practical tools that leverage the strengths of both quantum and classical computing. They are particularly well-suited to the capabilities and limitations of NISQ devices, making them a valuable asset in today's quantum computing landscape [33]. In the hybrid quantum-classical model, the quantum part involves state preparation and measurement and the classical part handles the optimization process, adjusting the parameters of the quantum circuits based on the measurements obtained.

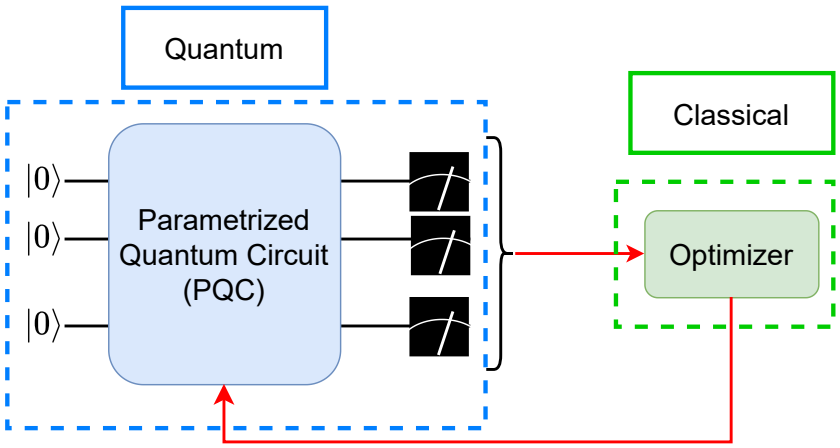


Figure 1.1: The setup of VQAs containing both quantum (state preparation, circuit, measurement) and classical (optimizer) parts.

NISQ devices, with their high error rates and limited qubit lifetimes (coherence times), pose significant challenges. However, VQAs, with their hybrid quantum-classical approach, are naturally resilient to certain types of errors. They typically use shallower (less deep) quantum circuits, effectively reducing exposure to errors over time. The continuous interaction between the quantum and classical components allows frequent adjustments and optimizations based on the output, progressively mitigating errors throughout the computation [34]. To understand this better, here is a brief description of the various components of VQAs:

1. Objective function: The problem is encoded in the objective function, also known as the cost or loss function. In a quantum context, this function typically involves calculations based on the outputs of a quantum circuit, such as the expectation value of a Hamiltonian in the case of the Variational Quantum Eigensolver (VQE). The performance of the algorithm is evaluated based on how effectively it optimizes this function.

2. Parametrized quantum circuit (PQC): The PQC contains the set of parameters that need to be optimized. Usually, these parameters are the gate parameters of the circuit. The PQC is created by the parametrized unitary operation, also known as ansatz. During the execution of a VQA, the parameters of these gates are iteratively adjusted to steer the system toward an optimal state that minimizes the objective function.

3. Optimizers: Optimizers in VQAs are classical in nature, that adjust the parameters in the PQC based on the output of the quantum measurements. Their main goal is to find parameters that optimize the objective function. This is done by analyzing the results from the quantum circuit and determining how to adjust the parameters to improve these results in the next iteration.

4. Performing measurement: Measurement is extracting classical data from the quantum system. In the context of VQAs, measurements are performed on

the quantum state prepared by the PQC to derive an empirical estimate of the objective function. These measurements provide the data that feeds back into the objective function, informing the classical optimizer of the current performance of the quantum circuit with respect to the problem at hand.

The components of a VQA are deeply interconnected, with a cyclical workflow (as shown in Fig. 1.1): the optimizer adjusts the parameters based on previous results, the PQC prepares a new quantum state using these updated parameters, and the results of new measurements are then used to update the objective function. This process repeats until the system converges to an optimal solution or reaches a predetermined number of iterations.

VQAs can be applied to solve practical and meaningful problems even with the limited capabilities of NISQ devices. For example, a variational quantum eigensolver (VQE) can find the ground states of a Hamiltonian, which is useful in physics and chemistry [35]. A quantum approximate optimization algorithm (QAOA) is a family of quantum circuits designed to find "good" solutions to optimization problems [36].

Even though full quantum advantage may not yet be achievable with NISQ devices, VQAs offer the potential for demonstrating quantum superiority in specific instances [33]. They provide a pathway to explore and potentially exploit quantum computational benefits as the technology progresses toward more advanced quantum computers.

Based on the concept of VQE, which has the same goals, are the quantum imaginary time evolution (QITE) and the quantum Lanczos (QLanczos) algorithm. Together, these algorithms are capable of finding the eigenvalues and eigenvectors of a many-body quantum system. We have used it in the research project described in chapter 2 to study collectively oscillating neutrinos on the currently available quantum computers [25].

1.3.2 Quantum Machine Learning

Machine Learning (ML) has significantly shaped the data revolution, transforming industries and technological applications worldwide. Notable applications include image and speech recognition, self-driving cars, efficient web searching, fraud detection, drug development, medical imaging, and more [37]. The success of ML stems from its ability to learn from data and iteratively improve algorithms, enabling it to identify patterns and make decisions with minimal human involvement [38].

Since ML relies heavily on data, training models with large datasets to achieve efficiency poses challenges. Advanced computing units like GPUs address some of these challenges but often fall short. In response, researchers have begun integrating ML with quantum computation, giving rise to Quantum Machine Learning (QML). QML encompasses various goals, including applying quantum properties to learning tasks, developing algorithms to expedite training, and identifying relevant features in data. QML shows promise for demonstrating quantum advantage on NISQ devices. Already, several breakthroughs have emerged at the intersection of quantum physics and ML, such as phase prediction of matter [39], solving quantum many-body problems [40], error correction [41], and applications in data-driven experimental high-energy physics [42, 43, 44].

Continuous-variable (CV) quantum computing has emerged as a powerful architecture for building neural network models on quantum computers [45]. This architecture, which readily implements the essential components of neural networks, has been successfully applied to a range of practical problems. These include function fitting, credit card fraud detection, image classification of handwritten digits, and data encryption and decryption in secure cryptography algorithms, demonstrating the real-world impact of QML.

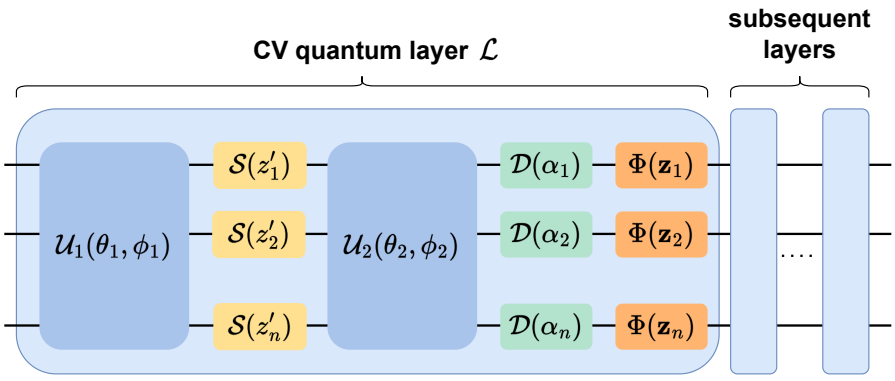


Figure 1.2: A general circuit architecture of CV quantum neural network according to [45]. The U_n represents the interferometers (combinations of beamsplitter and rotation gates), $\mathcal{S}(z_n')$ is the squeezing gate, $\mathcal{D}(\alpha_n)$ is the displacement gate, and $\Phi(\vec{z})$ is non-linear gate

A CV quantum neural network is built up as a sequence of fully connected feed-forward layers, similar to the classical fully connected neural network. Each layer, $\mathcal{L}$ contains a successive gate sequence:

$$\mathcal{L} := \Phi(\lambda) \cdot \mathcal{D}(\alpha) \cdot \mathcal{U}_2(\theta_2, \phi_2) \cdot \mathcal{S}(z) \cdot \mathcal{U}_1(\theta_1, \phi_1) \tag{1.4}$$

which is shown in figure 1.2. Here, $\mathcal{U}_i$ represents N-port linear optical interferometers consisting of beamsplitter and rotation gates. The $\mathcal{D}(\alpha)$ and $\mathcal{S}(z)$ are displacement and squeezing gates respectively. While $\Phi(\lambda)$ is the non-Gaussian gate used to introduce nonlinearity in the neural network.

By arranging each layer as a building block, the neural network constructs a multi-layer structure with the collective gate variables $(\theta, \phi, z', \alpha, \vec{z})$, denoted as $\vec{\theta}$. The goal is to optimize $\vec{\theta}$ such that the cost function $(C(\vec{\theta}))$ is minimized, using various optimization techniques (gradient descent, stochastic gradient descent, and the Adam optimizer). The parameters are updated as the following rule:

$$\vec{\theta} \longrightarrow \vec{\theta} + \eta \nabla_{\vec{\theta}} C(\vec{\theta}) \tag{1.5}$$

where η is step size, also known as the learning rate. The procedure will continue until the model cannot modify the cost function further. Then, the parameters associated with the lowest observed cost function offer the best solution to the given task. Note that the free parameters correspond to a circuit; hence, the circuit is the solution.

In principle, only one layer $(\mathcal{L})$ is sufficient to parameterize every possible unitary affine transformation on N modes [45]. The transformation of the neural network becomes universal once we add nonlinearity to it by using the non-Gaussian Φ gate. Nevertheless, we want to keep the quantum system linear, and CV encoding allows this unique nonlinear functional transformation. For that, non-Gaussian gates are used, which are not experimentally realizable.

A solution to this problem, making neural networks experimentally realizable [26], is detailed in chapter 3. In this research project, we use an ancillary mode and introduce nonlinearity by performing a measurement on this mode. Similar measurements on ancillary qumodes to create desired quantum states have been considered before. Single- and two-mode quantum gates acting on photonic qubits were generated with single photon sources, linear optical elements, and measurements of ancillary modes [46]. To generate the cubic phase gate, nonlinear quadrature measurements were implemented using ancillary states, homodyne measurements, and nonlinear feed-forwards based on the measurement results [47]. Photon-number-resolving (PNR) measurements were used for the probabilistic production of multi-mode Gaussian states, including cat (superpositions of coherent states), ON, Gottesman-Kitaev-Preskill (GKP), NOON, and bosonic code states [48]. Cat and GKP states were also created experimentally within a CV cluster state by performing PNR measurements [49]. To our knowledge, ours is the first attempt at applying nonlinearities induced by measurements to CV quantum neural networks.

Quantum neural networks have a broad range of applications, including classification, image recognition, and curve fitting. However, one significant area they have not extensively explored is generation. To address this gap, we examined the capabilities of quantum Boltzmann machines in Chapter 4, which are adept at generating probabilistic distributions. In this study, we utilized real-life data from synthetic aperture radar (SAR) images to generate a diverse array of distributions corresponding to Gaussian, Weibull, Rayleigh, and many more. We also use the quantum data to generate the distribution of a Gaussian and non-Gaussian quantum state.

Chapter 2

Discrete-variable Quantum Eigensolver

In recent years, quantum computing has experienced remarkable growth, encompassing both advancements in hardware construction and significant algorithmic developments. This progress spans a variety of platforms, including superconducting, photonic, and ion-trapped quantum computers. It extends to pivotal algorithms such as the Variational Quantum Eigensolver (VQE) [50], Quantum Approximate Optimization Algorithm (QAOA) [51], Quantum Imaginary Time Evolution (QITE) [52], Quantum Variational Classifiers [53], Quantum Walk Algorithms [54], and broader Quantum Machine Learning Algorithms [55]. Research has increasingly demonstrated a quantum advantage, showing that quantum computers can outperform classical systems in solving specific problems [30]. The focus here should be on the phrase "solving specific problems" because the current quantum computers have a limited number of qubits, which are noisy and have high decoherence. These quantum computers are known as Noisy-Intermediate Scale Quantum (NISQ) devices.

Discrete-variable (DV) quantum computing, a promising avenue for various applications, utilizes qubits as the primary units of quantum information. DV systems, which mirror classical bits but are enhanced by the quantum properties

of superposition and entanglement, offer an intuitive binary encoding. This aligns closely with existing digital computing frameworks and simplifies the implementation of a wide array of algorithms traditionally used in classical computing, instilling confidence in their potential use. Technologies for implementing DV systems, such as superconducting qubits, trapped ions, and semiconductor quantum dots, have reached a high level of maturity and thorough research [34]. This has led to significant advancements in coherence times, gate fidelities, and scalability, providing a solid foundation for further development. Furthermore, DV systems are especially conducive to established quantum error correction (QEC) techniques, such as the surface code, explicitly designed for qubits [35]. These techniques effectively protect quantum information against errors by encoding it across multiple qubits, further enhancing the reliability of DV systems. Moreover, DV systems are inherently suited for various quantum algorithms tailored explicitly for qubit-based systems. These include Shor's algorithm for factoring, Grover's algorithm for database searching, and various quantum simulation algorithms. Each of these developments contributes to the robust framework that supports the ongoing evolution and application of DV quantum computing.

Researchers from diverse fields, including physics, chemistry, and material science, are particularly interested in applying DV quantum algorithms to study quantum many-body systems. These systems, comprising electrons, atoms, molecules, and other small particles, exhibit pronounced quantum mechanical properties. Consequently, as Feynman suggested [1], studying them using quantum computers, which inherently operate based on these quantum properties, is fitting. The primary objective in such studies usually involves determining the eigenvectors and eigenvalues of the system, although more complex properties can also be explored. However, the analysis of the eigensystem is often the initial step as it provides a comprehensive framework for understanding and predicting the behavior of quantum systems. This includes studying the energy spectrum, quantum state properties, phase transitions, entanglement, thermodynamics, and material properties. The

current NISQ devices typically require hybrid quantum-classical algorithms to tackle these problems efficiently. One prominent example is the Variational Quantum Eigensolver (VQE) [50]. VQE primarily seeks to compute an upper bound for the ground-state energy of a Hamiltonian, which is a crucial first step in understanding the energetic properties of molecules and materials. Given its broad applicability, the scope of VQE extends to various domains such as chemistry, condensed matter physics, drug discovery, material science, and chemical engineering, making it a versatile tool in advancing quantum computational applications [35]. However, there are many disadvantages to using VQE, it is limited in accuracy by a fixed approach and involves high-dimensional noisy classical optimizations [52].

Consequently, researchers often turn to classical simulations, employing various numerical methods to determine the exact ground states of systems. An intriguing method among these is imaginary time evolution, which predicts the ground state as the long-time limit of the imaginary time Schrodinger equation. Unlike the VQE, which uses a fixed ansatz, imaginary time evolution inherently converges to the ground state, ensuring more reliable results. Another notable approach is the iterative Lanczos method, which constructs the Hamiltonian matrix (H) within a Krylov subspace; diagonalizing H yields a variational estimate of the ground state. The quantum adaptations of these algorithms—Quantum Imaginary Time Evolution (QITE) and Quantum Lanczos (QLanczos) were first introduced by [52]. These quantum versions offer significant advantages over their classical counterparts, requiring exponentially less memory and computational time per iteration. Moreover, they can be implemented on quantum hardware without deep circuits, ancillary qubits, or complex high-dimensional optimization strategies. These features make QITE and QLanczos particularly suited for current NISQ devices, promising substantial efficiency gains in quantum computations.

There have been various applications for these algorithms, mainly to study the quantum many-body simulation - one-dimensional spin system [56], Ising model [57], various physical and nuclear model [58, 59]. In this work, we use these algorithms

to study the collective neutrino oscillations on a quantum computer. First, we use QITE and QLanczos to calculate the eigenvalues and eigenvectors. Then, we use the first-order Trotterization method to demonstrate the oscillation of the neutrinos.

The layout of this chapter is as follows. First, we will provide a description of the quantum algorithms used in this work - QITE and QLanczos in Sec. 2.1. Then, we will focus more on the collective neutrino oscillation (in Sec.2.2). Mainly discussing the background and the Hamiltonian used in this study. Next, in Sec. 2.3, we discuss how we implement the quantum algorithms to study the neutrino system on a quantum computer, including the methods for the calculation of the energy levels of the system and the transition probabilities between neutrinos of different flavors. In Sec. 2.4, we present our results for the energy levels of a four-neutrino system as functions of the coupling parameter that depends on the radial distance from the neutrino source using the QLanczos algorithm. We also discuss the calculation of transition probability amplitudes for 3- and 4-qubit collective neutrino systems as a function of time using a simplified first-order Trotterization method. Finally, we summarize our conclusions and outlook in Sec. 2.5.

2.1 Quantum Algorithms

2.1.1 QITE Algorithm

The classical version of the quantum imaginary-time evolution (QITE) algorithm was introduced to simulate the many-body systems. It addresses the problem of exponentially increasing resource requirements for computation as a function of the number of interacting particles. The real-time in time-dependent Schrodinger's equation is replaced by the imaginary time ($t \to -i\beta$). The solution of this equation involves an imaginary time evolution operator, $\mathcal{U} = e^{-\beta H}$. This operator leads to the decay of all states except for the ground state, provided that the initial state has a non-zero overlap with the ground state. The quantum version of the QITE algorithm was

proposed in Ref. [52] and, unlike other approaches [14], does not require ancilla qubits or classical optimization. For example, Variational Quantum Eigensolver (another algorithm to calculate energy eigenvalues) requires optimization. However, it is implemented with multiple single- and two-qubit gates at each step of the imaginary-time evolution until the system converges to the ground state energy. To make the implementation more compatible with NISQ devices, we proposed adaptations that reduced the depth of the quantum circuits. [57, 58].

In the QITE algorithm, the imaginary time evolution of an initial state is given by:

$$|\Psi(\beta)\rangle = c_n(e^{-\beta H})^n|\Psi_0\rangle \tag{2.1}$$

where c_n is the normalization constant, given as $c_n = \frac{1}{\sqrt{\langle\Psi_0|\mathfrak{U}^2|\Psi_0\rangle}}$ and $e^{-\beta H}$ is the non-unitary imaginary time evolution operator. Then, one can express the non-unitary operator in terms of a unitary operator by dividing the imaginary-time evolution into n small steps with $\Delta\tau = \frac{\beta}{n}$:

$$|\Psi(\beta)\rangle = c_n(e^{-\Delta\tau H})^n|\Psi_0\rangle. \tag{2.2}$$

This step is vital because it will lead us to express the non-unitary operator as the unitary operator. The sth step of this imaginary-time evolution is approximated by real-time evolution in terms of a unitary operator as

$$|\Psi_s\rangle = \frac{c_s}{c_{s-1}}e^{-\Delta\tau H}|\Psi_{s-1}\rangle \approx e^{-i\Delta\tau A[s]}|\Psi_{s-1}\rangle \tag{2.3}$$

where c_s is a normalization constant that can be calculated recursively starting with $c_0 = 1$. The unitary updated operators can be written as a general operator of Pauli strings:

$$A[s] = \sum_{i_0,\ldots,i_{n_q-1}} a_I[s]\sigma_I \tag{2.4}$$

where $I = \{i_0, \ldots, i_{n_q-1}\}$, n_q is the number of qubits for the Hamiltonian, and $\sigma \in \{X, Y, Z\}$ are Pauli operators. The coefficients $a_I[s]$ are calculated by solving the linear system of equations $(\boldsymbol{\mathcal{S}} + \boldsymbol{\mathcal{S}}^T) \cdot \boldsymbol{a} = \boldsymbol{b}$ up to order $\mathcal{O}(\Delta\tau^2)$, where

$$\mathcal{S}_I = \langle \sigma_I \sigma_{I'} \rangle \ , \quad b_I = -i\sqrt{\frac{c_{s-1}}{c_s}} \langle \sigma_I H \rangle \tag{2.5}$$

These expectation values are calculated for the state in the previous QITE step, $|\Psi_{s-1}\rangle$. The exponentially growing size of the matrices $\boldsymbol{b}$ and $\boldsymbol{\mathcal{S}}$ challenges the scalability of the algorithm. Nevertheless, one can use a function available in the IBM Qiskit library, *isometry*, which uses the algorithm developed in [60] where a given isometry is decomposed into single-qubit and Controlled-NOT (CNOT) gates to have the least number of CNOT gates. Using the *isometry* function to simplify the circuits of the state at the sth step,

$$|\Psi_s\rangle = e^{-iA\Delta\tau[s]} e^{-iA\Delta\tau[s-1]} \cdots e^{-iA\Delta\tau[1]} |\Psi_0\rangle \ . \tag{2.6}$$

The resulting quantum circuit at each QITE step has the same gate components but different rotation angles, making the implementation of the QITE algorithm possible on NISQ devices.

There has been a lot of work within our group in applying the QITE algorithm, for example, solving the MaxCut problem [61] and polynomial unconstrained binary optimization (PUBO) problems [62]. We have used a linear Ansatz for unitary updates in both problems and achieved remarkable results. Another application of QITE was explored using continuous variables to study scalar quantum field theory on a lattice [63].

In this work, we use the QITE algorithm as the basis of QLanczos as it fills the Krylov space (discussed further in the next Section 2.1.2). We used this combination of QITE and QLanczos algorithm to study the neutrino oscillations on a quantum computer [25].

2.1.2 Quantum Lanczos

The QLanczos algorithm uses the measurement outcomes of the QITE algorithm to obtain the eigenstates of the system. At each step of the QITE algorithm, we calculate energy expectation values, which are then used as input by the QLanczos algorithm. We apply the QITE and QLanczos algorithms on the Hamiltonian, choosing initial states informed by the symmetry of the Hamiltonian to reduce the number of imaginary-time steps required for convergence.

Once the ground state is obtained, the QITE algorithm can obtain excited states by selecting an initial state $|\Psi_0\rangle$ orthogonal to the ground state. However, this is an impossible task on NISQ devices due to errors. Instead, we can apply the QLanczos algorithm, which uses the QITE states and energy expectation values at selected steps as inputs. These states span the $\mathcal{K}$ of the QLanczos algorithm such that $\{|\Phi_0\rangle, |\Phi_2\rangle, \dots\}$ where $|\Phi_l\rangle = c_l e^{-l\Delta\tau H}|\Psi_s\rangle$ and $|\Psi_s\rangle$ is the state at the s-th QITE step. After filling out the Krylov space, we build the overlap ($\mathcal{T}$) and Hamiltonian ($\mathcal{H}$) matrices from the energy expectation value measurements on quantum hardware. The elements of these matrices are defined as

$$\mathcal{T}_{l,l'} = \langle\Phi_l|\Phi_{l'}\rangle \ , \quad \mathcal{H}_{l,l'} = \langle\Phi_l|H|\Phi_{l'}\rangle \ , \tag{2.7}$$

respectively. We must express these quantities in terms of energy expectation values obtained from measurements on a quantum computer. We obtain

$$\mathcal{T}_{l,l'} = \langle\Phi_l|\Phi_{l'}\rangle = \frac{c_l c_{l'}}{c_r^2} \ , \tag{2.8}$$

and

$$\mathcal{H}_{l,l'} = \langle\Phi_l|H_{\text{block}}|\Phi_{l'}\rangle = \mathcal{T}_{l,l'}\langle\Phi_r|H|\Phi_r\rangle \ , \tag{2.9}$$

The normalization constants are found recursively through

$$\frac{1}{c_{r+1}^2} = \frac{\langle \Phi_r | e^{-2\Delta\tau H} | \Phi_r \rangle}{c_r^2} \tag{2.10}$$

where $r = \frac{l+l'}{2}$ and l, l' are even integers. For the experimental realization of these coefficients, we expand the numerator to second order in $\Delta\tau$ as $\langle \Phi_r | e^{-2\Delta\tau H} | \Phi_r \rangle = 1 - 2\Delta\tau \langle \Phi_r | H | \Phi_r \rangle + 2(\Delta\tau)^2 \langle \Phi_r | H^2 | \Phi_r \rangle + \mathcal{O}((\Delta\tau)^3)$. This second-order expansion requires measuring energy expectation values and second moments, $\langle H^2 \rangle$, at each QITE step on quantum hardware. Although this doubles the number of measurements required on quantum hardware with the method described earlier, after obtaining the QITE states from a noisy simulator, we can do circuit bundling and combine up to 900 (depending on the provider and backend) circuits in a single job and run these circuits on quantum hardware. For this second-order expansion, to improve the result with higher $\mu(r)$ values, one needs to choose a smaller value of $\Delta\tau$ since the order of magnitude of the neutrino system Hamiltonian is directly proportional to the $\mu(r)$ value. The next step is to solve the generalized eigenvalue equation

$$\boldsymbol{\mathcal{H}} \boldsymbol{x} = E \boldsymbol{\mathcal{T}} \boldsymbol{x} \ , \tag{2.11}$$

which provides an approximation,

$$|\Psi[E]\rangle = c_E \left(x_0^{(E)} |\Phi_0\rangle + x_1^{(E)} |\Phi_2\rangle + \dots \right) \ , \tag{2.12}$$

to the eigenvectors of the Hamiltonian with $c_E^{-1} = \| \sum_{l=0,1,\dots} x_l^{(E)} |\Phi_l\rangle \|$. The eigenvalues E provide an approximation to the energy levels of the Hamiltonian. Unfortunately, these eigenvalues are numerically unstable due to the noise from quantum hardware. As the number of qubits increases, the measurement outcomes become noisier, resulting in incorrect energy levels. To decide if an energy E obtained from the eq. 2.11 using experimental results is close to the exact energy eigenvalue of the Hamiltonian, we use a minimum uncertainty criterion: we calculate the

uncertainty in energy from $\Delta E = ||H|\Psi[E]\rangle - E|\Psi[E]\rangle||$. We scan the eigenvalues obtained from Krylov space at each QITE step and discard the eigenvalues with an uncertainty greater than a specific value δ, i.e., we require $\Delta E \leq \delta$.

Our group used this combination of QITE and QLanczos for the first time to study the Ising model [57]. We used the QLanczos algorithm to calculate transition probabilities and scattering amplitudes in the one-dimensional transverse Ising model with periodic boundary conditions. We also used the QITE algorithm to provide a basis for the Hilbert (Krylov) space employed in the QLanczos algorithm. Next, we wanted to apply this combination of algorithms to a more complicated system of collective oscillating neutrinos, which has been discussed in detail in the next section.

2.2 Collective Neutrino Oscillation

2.2.1 Background

According to the Standard Model of particle physics, neutrinos are neutral elementary particles in three leptonic flavors: electron, muon, and tau neutrino. Experiments conducted in over two decades [64, 65, 66, 67] have demonstrated that neutrino flavors oscillate, which implies that at least two neutrinos are massive. Mass eigenstates do not coincide with flavor eigenstates. Neutrinos have been observed to oscillate between different flavors as they propagate, which is a quantum mechanical phenomenon.

Early research in neutrino oscillations boosted observations of the flux of solar neutrinos [68], experimental studies of neutrino beams, as well as theoretical research like neutrinos that appear to be emitted and absorbed in flavor eigenstates but travel as mass eigenstates. After studying neutrino flavor transitions, it has been realized that neutrinos experience self-maintained coherent oscillations [69]. Moreover, a homogeneous and isotropic environment does not disrupt these collective neutrino

oscillations. This phenomenon was studied in detail in connection with early universe scenarios [70] and supernovae [71].

In the astrophysical events mentioned above, an abundant number of neutrinos are created. For example, in a core-collapse supernova event, the energy released is of order $\sim 10^{59}$ MeV, and 99% of this energy is carried away by neutrinos and anti-neutrinos. This large amount of energy corresponds to $\sim 10^{58}$ neutrinos [72]. Studying interactions between these many neutrinos using classical computers is a daunting task. One needs to account for strong correlations due to collective neutrino oscillations. Calculating the energy spectrum of the collective neutrino system is central to understanding the physics behind neutrino interactions, which give rise to collective behavior. In Ref. [73], a mean-field approximation was used to calculate the eigenvalues and eigenstates of a collective neutrino system. They introduced a method to systematically obtain the eigenvalues and eigenstates of a many-body Hamiltonian describing collective neutrino oscillations. Remarkably, using this method, one can perform calculations in a 15-neutrino system with a personal computer.

On the other hand, fault-tolerant quantum computers' exponential speedup provides hope to simulate and understand high-energy physics inaccessible to classical computers. Recently, there has been exciting progress, for example, towards understanding parton showers using quantum computers [74]. A similar speedup with quantum computers is also expected in studying collective neutrino oscillations. The first attempt to study neutrino oscillations on a quantum processor was discussed in [75], where a scheme to study the dynamics of neutrino oscillations on a trapped-ion quantum computer was introduced. The first experimental results on the dynamics of neutrino oscillations employing quantum hardware were presented in Ref. [76]. Two- and three-flavor neutrino oscillations were represented on a superconducting quantum processor, and the survival probability of different neutrino flavors was studied as a function of neutrino energy. Here, we use the quantum Lanczos [52] and hybrid quantum-classical algorithms to obtain energy spectra for the Hamiltonian in the mass basis of neutrino systems undergoing oscillations, in addition to transition

probabilities. Our calculations are based on the approximate many-body neutrino interaction Hamiltonian introduced in Ref. [73]. This Hamiltonian includes only two flavor/mass states for the neutrinos and ignores interactions with anti-neutrinos and the background matter.

Concurrently with the present work, Ref. [77] also studied the simulation of collective neutrino oscillations on a quantum computer. The authors of that study explore the real-time many-body evolution of a collective neutrino system using first-order Trotterization combined with error mitigation, a complementary technique to the Quantum Lanczos (QLanczos) method used in the present work.

Our experimental results for the energy eigenvalues of the $N = 4$ collective neutrino system are in excellent agreement with exact diagonalization, enabling us to observe flavor oscillations in the system with $N = 3$ and $N = 4$ neutrinos. All the experimental runs are done on IBM Q hardware [16].

2.2.2 Model

Neutrino interactions can be described by a many-body Hamiltonian, which consists of neutrino oscillations in a vacuum, interactions of neutrinos with background matter and other neutrinos, and anti-neutrinos. In [73], eigenvalues and eigenstates of the many-body collective neutrino oscillation Hamiltonian were studied after applying certain simplifications such as limiting to two flavor/mass states of neutrinos, ignoring interactions with antineutrinos as well as the background matter. With these simplifications, the Hamiltonian representing the vacuum and neutrino self-interaction terms can be written as

$$H = \sum_{\boldsymbol{p}} \omega_{\boldsymbol{p}} \vec{B} \cdot \vec{J}_{\boldsymbol{p}} + \frac{\sqrt{2} G_F}{V} \sum_{\boldsymbol{p},\boldsymbol{q}} (1 - \cos \vartheta_{\boldsymbol{pq}}) \vec{J}_{\boldsymbol{p}} \cdot \vec{J}_{\boldsymbol{q}} \ , \tag{2.13}$$

where $\vec{B} = (0, 0, -1)_{\text{mass}} = (\sin 2\theta, 0, -\cos 2\theta)_{\text{flavor}}$, θ is the vacuum mixing angle, $\omega_{\boldsymbol{p}} = \delta m^2/(2|\boldsymbol{p}|)$ are the vacuum oscillation frequencies, $\vartheta_{\boldsymbol{pq}}$ is the intersection angle

between the trajectories of neutrinos with 3-momenta $\boldsymbol{p}$ and $\boldsymbol{q}$, V is the quantization volume, and G_F is the Fermi coupling constant. The neutrino mass-basis isospin operators $\vec{J}_{\boldsymbol{p}}$ can be written in terms of fermionic creation and annihilation operators as

$$J^+_{\boldsymbol{p}} = a^\dagger_1(\boldsymbol{p})a_2(\boldsymbol{p}) \ , \tag{2.14}$$

$$J^-_{\boldsymbol{p}} = a^\dagger_2(\boldsymbol{p})a_1(\boldsymbol{p}) \ , \tag{2.15}$$

$$J^z_{\boldsymbol{p}} = \frac{1}{2}\left(a^\dagger_1(\boldsymbol{p})a_1(\boldsymbol{p}) - a^\dagger_2(\boldsymbol{p})a_2(\boldsymbol{p})\right) \ . \tag{2.16}$$

The eq. 2.13 can be further simplified by assuming that the neutrinos are emitted isotropically from a single spherical emission surface; the neutrino self-interaction term can be simplified to

$$H \approx \sum_{\boldsymbol{p}} \omega_{\boldsymbol{p}} \vec{B} \cdot \vec{J}_{\boldsymbol{p}} + \mu(r) \left(\sum_{\boldsymbol{p}} \vec{J}_{\boldsymbol{p}}\right)^2 \tag{2.17}$$

where $\mu(r) = \frac{G_F}{\sqrt{2}V}\left(1 - \sqrt{1 - \frac{R_\nu^2}{r^2}}\right)^2$, with r the distance from the center of a neutrino sphere of radius R_ν, and $\boldsymbol{p}$ the index labeling the oscillating frequencies present in the system.

2.3 Implementation of Quantum Algorithms

We use the hybrid quantum-classical QLanczos algorithm discussed in [52, 57] to find the eigenvalues of the collective neutrino many-body interaction Hamiltonian (eq. 2.17). To study a Hamiltonian on a quantum computer, one needs to express it in terms of Pauli spin matrices. Since neutrinos are fermions, observables can be represented using a two-dimensional $SU(2)$ algebra. The isospin operators can be expressed in terms of Pauli matrices as $J_i = \frac{\sigma_i}{2}$, where $\sigma_i \in \{X, Y, Z\}$ are Pauli

spin matrices. Therefore, the neutrino interaction Hamiltonian in mass basis can be written as

$$H = -\frac{1}{2}\sum_{p=1}^{M}\omega_p Z_p + \frac{\mu(r)}{4}(\boldsymbol{X}^2 + \boldsymbol{Y}^2 + \boldsymbol{Z}^2)\ , \tag{2.18}$$

where p labels the oscillation frequencies present in the system, $\boldsymbol{X} = (X_1, \ldots, X_M)$, and similarly for $\boldsymbol{Y}$ and $\boldsymbol{Z}$.

We choose a system of N neutrinos distributed evenly across M oscillation frequencies, and for simplicity, we set $M = N$. In this system, one must calculate 2^N eigenvalues, which becomes an intractable problem on classical computers as the number of neutrinos grows. Fault-tolerant quantum computers are expected to provide an exponential speedup in calculating eigenvalues and eigenvectors as functions of μ/ω_0, where the oscillation frequencies are given by $\omega_p = p\omega_0$ [78]. However, hybrid quantum-classical algorithms are utilized due to current and near-term quantum hardware limitations. We use the numerical parameters for dense neutrino gasses provided in ref. [78]. Experimentally, $\omega_0 = 1.055 \times 10^{-16}$ MeV. Here, we work in units in which $\omega_0 = 1$, thus effectively, we calculate dimensionless energy levels, E/ω_0, as functions of the dimensionless parameter $\mu(r)/\omega_0$ ($\mu(r)$ is treated as a free parameter to be varied).

In the mass basis, the system Hamiltonian, eq. 2.18, is separable into independent blocks because H commutes with the total number operator ($\mathcal{N}$):

$$\mathcal{N} = \sum_{p=1}^{N}\frac{I - Z_p}{2}. \tag{2.19}$$

Since $\mathcal{N}$ has $N+1$ distinct eigenvalues, $0, 1, \ldots, N$, the Hamiltonian splits into $N+1$ blocks that can be studied independently. The kth block has dimension $d_k = \binom{N}{k}$, and $d_0 + d_1 + \cdots + d_N = 2^N$. The largest block grows asymptotically as $\frac{2^N}{\sqrt{N}}$, requiring

$N - \frac{1}{2}\log_2 N$ qubits. This improvement is not significant for large N, but for values of N relevant to NISQ devices, it significantly reduces the required resources.

To compute the spectra, we express each reduced block Hamiltonian as a sum of unitary matrices [58]:

$$H_{\text{block}} = \sum_I c_I \sigma_I \tag{2.20}$$

where $I = \{i_0, \ldots, i_{n_q-1}\}$ and $\sigma_I = \sigma_{i_0}\sigma_{i_1}\ldots\sigma_{i_{n_q-1}}$, with $\sigma \in \{I, X, Y, Z\}$ and n_q the number of qubits used for this block Hamiltonian. The coefficients c_I are found using

$$c_I = \frac{1}{2^{n_q}}\text{Tr}[H_{\text{block}}\sigma_I] \ . \tag{2.21}$$

Here, we study the four-neutrino system Hamiltonian. The matrix elements corresponding to no particle, i.e., the state $|0000\rangle$, and four particles, i.e., the state $|1111\rangle$, form one-dimensional blocks, so the eigenvalues are readily available analytically. The subspaces of single-particle states, given as $\{|1000\rangle, |0100\rangle, |0010\rangle, |0001\rangle\}$ and three-particle states, given as $\{|1110\rangle, |0111\rangle, |1011\rangle, |1101\rangle\}$ form independent four-dimensional blocks, each of which can be represented by a two-qubit system. The reduced block Hamiltonian is given by

$$\begin{aligned} H^{\pm}_{\text{block}} =& \frac{6\mu \pm 5\omega_0}{2} I + \mu(X_0 + X_1 + X_0X_1) \\ &+ \omega_0(Z_0 + \frac{1}{2}Z_1) \ , \end{aligned} \tag{2.22}$$

where for H^{-}_{block} corresponds to 1-particle and H^{+}_{block} corresponds to 3-particle states. The largest block corresponds to the six-dimensional subspace of two-particle states, $\{|1100\rangle, |0101\rangle, |1010\rangle, |1001\rangle, |0011\rangle, |0110\rangle\}$ can be studied using a three-qubit system. Their block Hamiltonian can be written as

$$H_{\text{block}} \ = \ \frac{(\omega_0 + 6\mu)}{4} I + \frac{\mu}{2} H_1 + H_2 \tag{2.23}$$

with

$$\begin{aligned} H_1 =& X_0X_1X_2 + Z_0X_1X_2 + Y_0Y_1X_2 + X_0X_1 + X_0Z_1 \\ &+ Z_0X_1 + Y_0Y_1 + X_1X_2 + Z_1X_2 - Z_0Z_1 \\ &+ X_0 + X_1 + X_2 \ , \end{aligned} \tag{2.24}$$

$$\begin{aligned} H_2 =& \frac{(2\omega_0 - \mu)}{2} Z_0Z_1 + \frac{\omega_0}{4} Z_1Z_2 + \frac{\mu + \omega_0}{2} Z_0 \\ &+ \frac{2\mu - \omega_0}{4} Z_1 + \frac{\omega_0}{4} Z_2 \ , \end{aligned} \tag{2.25}$$

where we added arbitrary values (additional energy levels E_1', E_2') as the last two elements of the diagonal to create a matrix for a 3-qubit system. It is necessary to ensure that these additional energy levels are not at the end of the energy spectrum; otherwise, the QITE algorithm may converge to these spurious states. We set $E_1' = E_2' = \omega_0$.

We also use the information from symmetry considerations of the reduced block Hamiltonian to choose different initial states, $|\Psi_0\rangle$, for each block to obtain all energy levels of the Hamiltonian. Due to the noise in the hardware, no new information is obtained when we add more than two QLanczos vectors to the Krylov space. Therefore, we worked with only two-dimensional Krylov spaces. Using the reduced block Hamiltonian in two- and three-qubit systems, we first ran the QITE algorithm to obtain the two lowest energy eigenvalues of the symmetry sector corresponding to the initial state choice. We ran the QITE algorithm using $-H_{\text{block}}$ with a different initial state to obtain the two highest energy eigenvalues. Hence, we got all the energy eigenvalues of a two-qubit reduced Hamiltonian block. For the two remaining energy levels of a three-qubit Hamiltonian block (recall that two eigenvalues were added arbitrarily to supplement the six-dimensional block), we obtained the eigenstates exactly by exploiting the symmetry of the reduced Hamiltonian block. Then constructed quantum circuits generating the eigenstates, which we implemented on quantum hardware to obtain the experimental energy eigenvalues. As it is easily

seen from the three-qubit reduced block Hamiltonian in eq. 2.23, the eigenvalue 2μ is degenerate and the corresponding eigenvectors are $\frac{1}{\sqrt{2}}(|010\rangle - |011\rangle)$ and $\mathcal{N}[(1+\frac{2}{\mu})|010\rangle - |000\rangle - 2|001\rangle + |101\rangle + 2|100\rangle]$, where $\mathcal{N}^{-2} = 11 + \frac{4}{\mu} + \frac{4}{\mu^2}$.

In the case of larger systems described by a more significant number of qubits, in order to reach the middle energy levels, one can modify the Hamiltonian to $H'_{\text{block}} = (H_{\text{block}} - \alpha\mathbb{I})^2$, where the parameter α is adjusted so that the energy levels in the middle turn into the lowest energy levels. This method can calculate all energy levels of a system with more than four neutrinos.

Next, we calculate the transition probability amplitude of collective neutrino flavor oscillations. We need to calculate the real-time evolution of an initial state governed by the unitary time evolution operator $\mathcal{U} = e^{-iH_{\text{flavor}}t}$ with

$$\begin{aligned} H_{\text{flavor}} = &-\frac{1}{2}\sum_{p=1}^{M}\omega_p(\cos 2\theta Z_p - \sin 2\theta X_p) \\ &+ \frac{\mu(r)}{4}(\boldsymbol{X}^2 + \boldsymbol{Y}^2 + \boldsymbol{Z}^2)\ , \end{aligned} \tag{2.26}$$

where we substituted $\vec{B} = (\sin 2\theta, 0, -\cos 2\theta)_{\text{flavor}}$ in eq. 2.17 for the flavor-basis magnetic field and used $\sin^2(2\theta) = 0.1$ for the two-flavor mixing angle, which has been obtained from observations of neutrino oscillations [78]. The transition probability amplitude between a given initial state $|\Psi_{\text{initial}}\rangle$ and a final state $|\Psi_{\text{final}}\rangle$ is

$$P_{fi}(t) = |\mathcal{A}_{fi}|^2\ ,\quad \mathcal{A}_{fi}(t) = \langle\Psi_{\text{final}}|\mathcal{U}(t)|\Psi_{\text{initial}}\rangle\ . \tag{2.27}$$

First-order Trotterization is one of the commonly used methods to decompose the unitary time evolution so that the unitary time evolution operator can be approximated by operators, which can be expressed in terms of one- and two-qubit operators of the quantum computers.

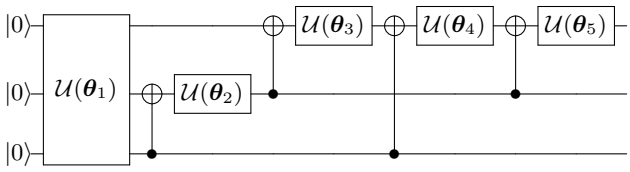


Figure 2.1: Quantum circuit obtained using IBM Qiskit's *isometry* function for the Trotterization of a three-qubit block Hamiltonian of the neutrino system.

To this end, we split the Hamiltonian into three parts, each consisting of commuting matrices, $H = H_X + H_Y + H_Z$, where

$$\begin{aligned} H_X &= \frac{\sin 2\theta}{2} \sum_{p=1}^{M} \omega_p X_p + \frac{\mu(r)}{4} \boldsymbol{X}^2 \\ H_Y &= \frac{\mu(r)}{4} \boldsymbol{Y}^2 \\ H_Z &= -\frac{\cos 2\theta}{2} \sum_{p=1}^{M} \omega_p Z_p + \frac{\mu(r)}{4} \boldsymbol{Z}^2 \end{aligned} \tag{2.28}$$

and use

$$e^{-iH\Delta t} = e^{-iH_X \Delta t} e^{-iH_Y \Delta t} e^{-iH_Z \Delta t} + \mathcal{O}((\Delta t)^2) \tag{2.29}$$

Then

$$\mathcal{U}(t) \approx \left[e^{-iH_X \Delta t} e^{-iH_Y \Delta t} e^{-iH_Z \Delta t} \right]^n \tag{2.30}$$

for $t = n\Delta t$. We need several gates to implement each step on quantum hardware. Due to the limited decoherence time in NISQ devices, it is often not possible to implement a large number of Trotter steps. To circumvent this problem, we used the *isometry* function in the IBM Qiskit library to simplify the circuits. The resulting quantum circuits can be seen in Fig. 2.1, where $\mathcal{U}(\boldsymbol{\theta}_i)$ with $i \in \{1, 2, 3, 4\}$ are a series of single-qubit U3(θ, λ, ϕ) gates. The circuits are of this form independently of the number of Trotterization steps. As we increase the number of steps, only the angles change in the circuit of Fig. 2.1, which allows us to implement Trotterization on the IBM Q devices.

Although this simplification is not scalable, it is advantageous for NISQ devices, as the angles change at each Trotter step, but the depth of the quantum circuit remains the same. Using this tool is sometimes not straightforward because the quantum circuits obtained using the *isometry* function involve connections between all qubits in the circuit, which is not the case for many IBM Q devices.

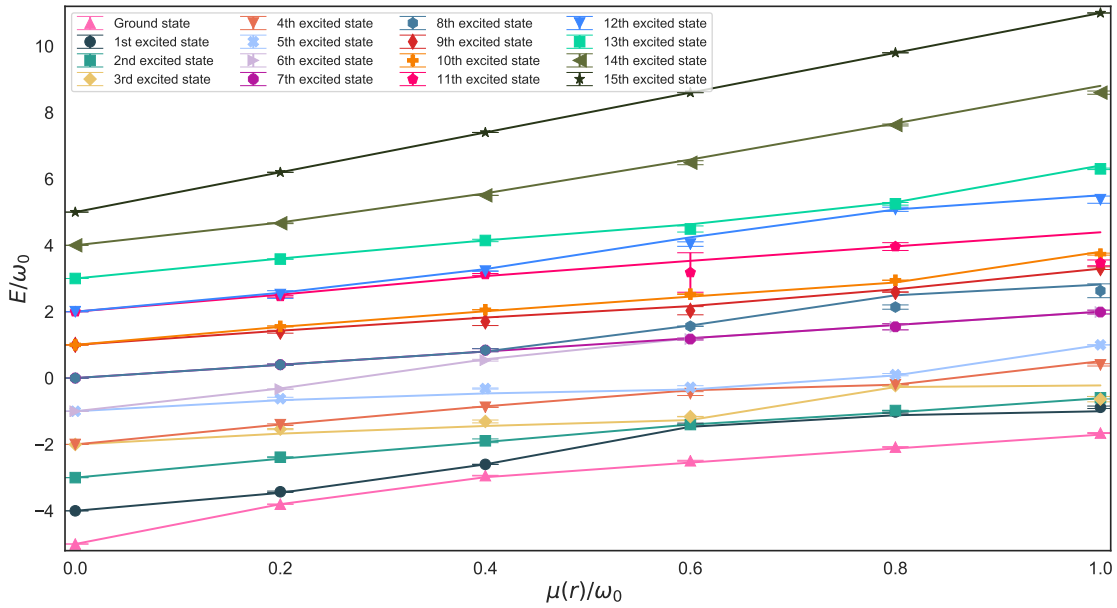


Figure 2.2: Readout-error mitigated (ROEM) energy eigenvalues of the 4-neutrino system as a function of the radial dependence of the coupling system $\mu(r)/\omega_0$ obtained using the QLanczos algorithm. The experiments were run on various IBM Q devices (Athens, Casablanca, Guadalupe, Ourense, Rome, Santiago, and Vigo) for $N_{\text{runs}} = 2$ times, and error bars represent $\pm\sigma$. The straight lines represent the exact eigenvalues obtained from exact diagonalization.

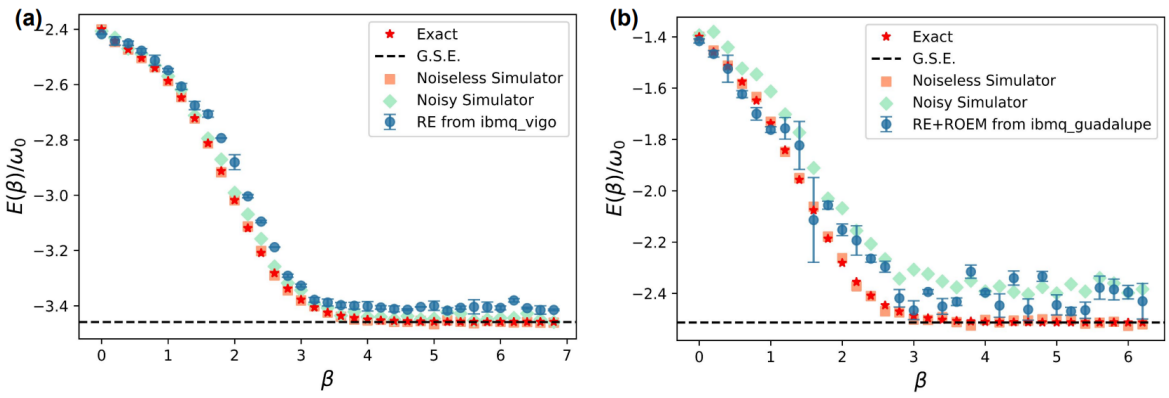


Figure 2.3: **(a)**: Energy vs. imaginary time calculated exactly using 2-qubit H_{block} and compared to IBM Q Aer QASM noiseless and noisy simulator with $\mu(r)/\omega_0 = 0.2$ and getting the measured energy expectation values from IBM Q Vigo hardware (data collected on 01/09/2021). **(b)**: Energy vs. imaginary time calculated exactly using 3-qubit $-H_{\text{block}}$ and compared to IBM Q Aer QASM noiseless and noisy simulator with $\mu(r)/\omega_0 = 0.2$ and using $A[s]$ operators from the noisy simulator to get the measured energy expectation values from IBM Q Guadalupe hardware (data collected on 05/16/2021). The experiments were run $N_{\text{run}} = 2$ times and the error bars represent $\pm\sigma$. The energies converge to the ground state energy level **(a)** -3.415 $\pm$ 0.003 with percentage error of 1.3 % and **(b)** -2.431 $\pm$ 0.083 with percentage error of 3.3 %.

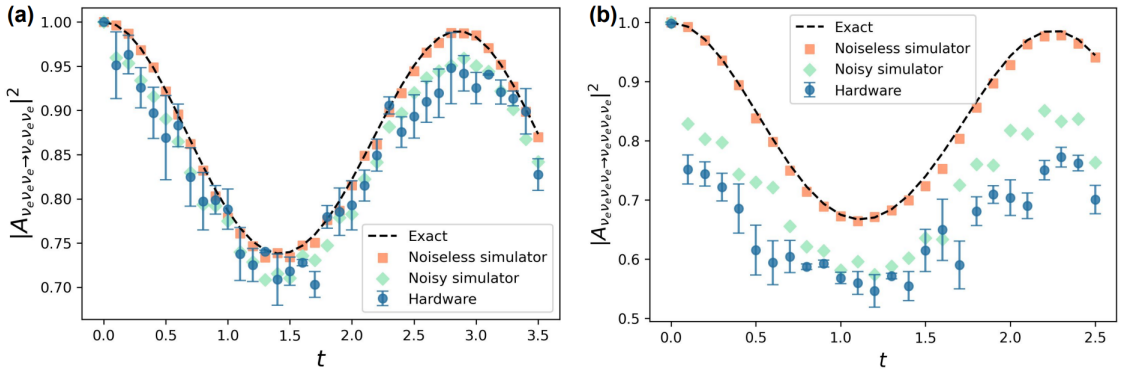


Figure 2.4: The transition probability amplitude comparison for the transition from **(a)** $|\text{initial}\rangle = |\nu_e\nu_e\nu_e\rangle$ to $|\text{final}\rangle = |\nu_e\nu_e\nu_e\rangle$, **(b)** $|\text{initial}\rangle = |\nu_e\nu_e\nu_e\nu_e\rangle$ to $|\text{final}\rangle = |\nu_e\nu_e\nu_e\nu_e\rangle$ as a function of time between exact time evolution calculations, noiseless simulated Trotterization and values obtained using Trotterization implemented on the seven-qubit IBM Q Casablanca hardware. The error mitigation method applied is readout error mitigated (ROEM). The parameters are chosen such that $\sin^2(2\theta) = 0.10$, $\omega_p = p\omega_0$ and $\mu(r)/\omega_0 = 1.0$. The experiments were run $N_{\text{run}} = 3$ times on 9/23/2021, and the error bars represent $\pm\sigma$.

If the hardware does not have this kind of connectivity, adding extra SWAP gates is required, resulting in additional noise in the system. A possible solution for this problem could be using an ion trap quantum computer with all-to-all connectivity, thus requiring no additional SWAP gates.

2.4 Results and Discussions

In Fig. 2.2, we performed calculations with the 4-neutrino system Hamiltonian expressed in the neutrino mass basis to find the eigenvalues as functions of the radial-distance-dependent coupling parameter $\mu(r)/\omega_0$ using the QLanczos algorithm and our experimental data agrees very well with the energy eigenvalues obtained from exact diagonalization. For this, we divided the Hamiltonian of the system into two- and three-qubit subsystems, which helped us obtain results with less hardware noise compared to a four-qubit system needed for the full Hamiltonian. We ran the QITE algorithm described above for two-qubit circuits on IBM Q Athens, Casablanca, Ourense, Rome, Santiago, and Vigo devices. We ran three-qubit quantum circuits on IBM Q Guadalupe and Santiago, switching between devices depending on availability. We present two sample data sets obtained with the QITE algorithm in Figs. 2.3**(a)** and 2.3**(b)** depicting QITE algorithm results for a two-qubit reduced Hamiltonian block (H_{block}) and three-qubit block with reversed sign ($-H_{\text{block}}$), for $\mu(r)/\omega_0 = 0.2$, implemented on IBM Q Vigo and Guadalupe, respectively. The convergence of the energy expectation value to the ground state energy is evident for the two-qubit H_{block} in Fig. 2.3**(a)** starting with initial state $|\Psi_0\rangle = |10\rangle$. This convergence is obscured by hardware noise in the 3-qubit case with $-H_{\text{block}}$ starting with initial state $|\Psi_0\rangle = |001\rangle$, as seen in Fig. 2.3**(b)**. We compare the energy expectation value as a function of imaginary time between those obtained from exact diagonalization, a noiseless simulator, a noisy simulator with quantum hardware noise, and experimental runs performed on the IBM Q Vigo and Guadalupe devices. We repeated this procedure with different initial states and values of the coupling $\mu(r)$ with similar

results. We then used the output of the QITE algorithm to find all eigenvalues using the QLanczos algorithm.

To obtain Fig. 2.2, we performed experiments with $N_{\text{runs}} = 2$ runs and $N_{\text{shots}} = 8192$ shots per run. The error bars represent $\pm\sigma$. To obtain the experimental energy eigenvalues in Fig. 2.2, we used readout error mitigation (ROEM) and Richardson extrapolation error mitigation techniques. In implementing the QLanczos algorithm, we discarded eigenvalues if the total uncertainty of two states in Krylov space exceeded a certain value, $\Delta E > \delta$ was minimum. Thus, for three-qubit data, we discarded data with uncertainty exceeding $\delta = 1.3$, whereas for two-qubit data, we set $\delta = 0.6$. The larger threshold value δ in the three-qubit case reflects the presence of more noise in the larger quantum circuit, resulting in no data with an uncertainty below $\delta = 1.3$. It would be desirable to reduce the noise so that a lower threshold can be attained. Nevertheless, the experimental eigenvalues obtained with the QLanczos algorithm agreed well with eigenvalues obtained from exact diagonalization.

We performed calculations for various values of the coupling $\mu(r)$. Smaller values yielded better agreement of experimental and exact energy levels. Applying the QLanczos algorithm on H_{block} and $-H_{\text{block}}$ with a two-dimensional Krylov space yields a couple of lowest and highest eigenvalues in the spectrum, respectively. While this spans the spectrum in the two-qubit case, for a three-qubit system, there are a couple of intermediate-size eigenvalues that this procedure does not calculate. Fortunately, the remaining eigenvalues correspond to eigenvectors that can be computed exactly using symmetry considerations of the block. We obtain the states $\frac{1}{\sqrt{2}}(|010\rangle - |011\rangle)$ and $\mathcal{N}[(1+\frac{2}{\mu})|010\rangle - |000\rangle - 2|001\rangle + |101\rangle + 2|100\rangle]$, where $\mathcal{N}^{-2} = 11 + \frac{4}{\mu} + \frac{4}{\mu^2}$. We obtained these eigenvalues experimentally by running the quantum circuits for the corresponding eigenstates on IBM Q hardware; see Fig. 2.2.

The error mitigation strategies helped us to obtain energy eigenvalues of a 4-neutrino system in excellent agreement with the exact eigenvalues. For the 3-qubit data presented in Fig. 2.2 we used both readout error mitigation (ROEM) and zero noise extrapolation (ZNE) to mitigate the error caused by the noise from quantum

hardware. To mitigate the errors from the measurements, we employed tools included in IBM's *qiskit-ignis*, using a constrained matrix inversion approach. The response matrix is obtained from the measurements on $2^{N_{\text{qubits}}}$ calibration quantum circuits prepared so that initialization of the qubits was followed by all combinations of the single-qubit X gate. These quantum circuits were bundled with the circuits of interest in the same job. For the error mitigation of the two-qubit gate errors, we used the ZNE proposed in [79] where we added double CNOT gates to the circuit for each CNOT gate in the original quantum circuit in order to increase the noise due to the CNOT gate. Then, we extrapolated to "noiseless" expectation values for the energy measurements and count values for the transition probability measurements. We ran the quantum circuits with $r = 1, 3, 5, 7, 9$, where r is the number of CNOT gates in the quantum circuit for each CNOT gate in the original circuit. Then we fitted the data points by a polynomial of degree n for n-qubit quantum circuits ($n = 3, 4$).

We also calculated the transition probability amplitudes for collective neutrino oscillations on a quantum computer. The transition probabilities for 3- and 4-neutrino systems are shown in Figs. 2.4 **(a)** and **(b)**, respectively. We compare exact transition probabilities, transition probabilities from noiseless simulated first-order Trotterization, and experimental results obtained from IBM Q hardware. To obtain the time evolution of the initial state, we used the Trotterization method. The experiments were performed on the IBM Q 7-qubit Casablanca hardware device. We ran the experiments $N_{\text{runs}} = 3$ times such that each run had 8192 shots. Error bars represent $\pm\sigma$. In these experiments, we only implemented ROEM since Richardson extrapolation did not improve results. The main source of error is not algorithmic but quantum hardware noise. We again compared the exact, noiseless simulator, a noisy simulator, with the quantum hardware noise. This comparison shows that the quantum device's noisy simulator with noise parameters is not good enough to reflect all noise in the quantum hardware. The cross-talk errors present in quantum hardware are not reflected in the error parameters of the so-called fake backend noise models.

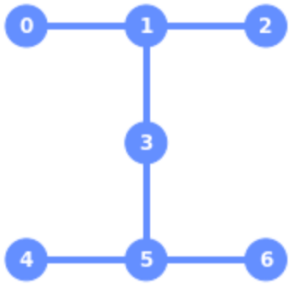


Figure 2.5: The qubit layout for IBM Quantum 7-qubit devices Casablanca, Jakarta, and Lagos.

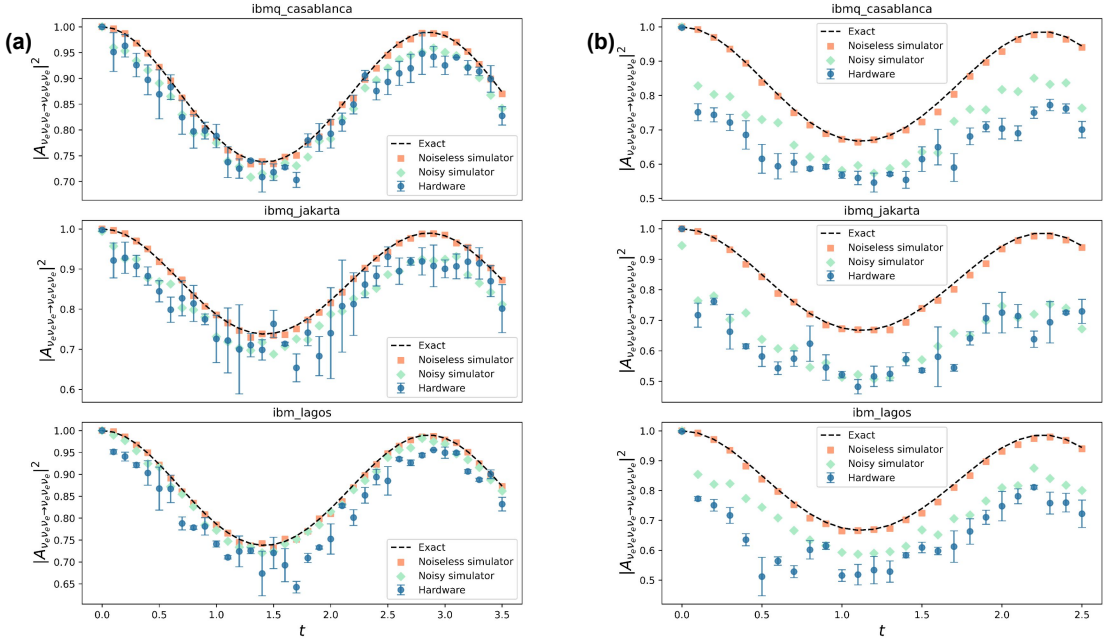


Figure 2.6: The transition probability amplitude comparison for the transition from **(a)** $|\text{initial}\rangle = |\nu_e\nu_e\nu_e\rangle$ to $|\text{final}\rangle = |\nu_e\nu_e\nu_e\rangle$, **(b)** $|\text{initial}\rangle = |\nu_e\nu_e\nu_e\nu_e\rangle$ to $|\text{final}\rangle = |\nu_e\nu_e\nu_e\nu_e\rangle$ as a function of time between exact time evolution calculations, noiseless simulated Trotterization and values obtained using Trotterization implemented on the three different 7-qubit IBM Q hardware (Casablanca, Jakarta, and Lagos, from top to bottom). The error mitigation method applied is readout error mitigation (ROEM). The parameters are chosen such that $\sin^2(2\theta) = 0.10$, $\omega_p = p\omega_0$ and $\mu(r)/\omega_0 = 1.0$. The experiments were run $N_{\text{run}} = 3$ times on 9/23/2021 for systems Casablanca and Jakarta and on 9/16/2021 for Lagos. The error bars represent $\pm\sigma$.

Table 2.1: Calibration data for three different 7-qubit IBM Quantum systems. IBM reported calibration data for experiments conducted on systems Casablanca and Jakarta on 9/23/2021 and for Lagos on 9/16/2021. For Casablanca and Jakarta, we used qubit layouts [2,1,3] and [2,1,3,5], and for Lagos, we used [3,5,6] and [1,3,5,6] for 3- and 4-qubit runs, respectively. The values reported are T_1, T_2, readout assignment error (e_{RO}), single-qubit $\sqrt{X}$ gate error (e_{SX}), and 2-qubit CNOT gate error for qubit pairs used (e_{2Q}).

Devices (Qubits layout)	T_1 (μ s)	T_2 (μ s)	e_{RO} ($\times 10^{-2}$)	e_{SX} ($\times 10^{-4}$)	e_{2Q} ($\times 10^{-3}$)	
Casablanca						
[2,1,3] [2,1,3,5]						
Q1	155.68	94.28	2.34	2.3	[1-2]	9.75
					[1-3]	7.64
Q2	88.45	218.76	4.08	3.63	[2-1]	9.75
Q3	62.16	146.71	1.82	4.36	[3-1]	7.64
					[3-5]	10.06
Q5	71.7	124.41	2.32	3.61	[5-3]	10.06
Jakarta						
[2,1,3] [2,1,3,5]						
Q1	146.55	20.94	2.71	2.05	[1-2]	11.06
					[1-3]	9.49
Q2	123.16	23.15	1.88	2.99	[2-1]	11.06
Q3	69.45	20.56	2.48	2.66	[3-1]	9.49
					[3-5]	13.97
Q5	148.45	53.89	5.99	2.19	[5-3]	13.97
Lagos						
[3,5,6] [1,3,5,6]						
Q1	71.79	100.98	2.98	6.76	[1-3]	15.18
Q3	177.05	125.42	4.08	1.9	[3-1]	15.18
					[3-5]	4.49
Q5	101.78	83.9	1.09	2.11	[5-3]	4.49
					[5-6]	4.16
Q6	145.8	128.29	2.3	3.29	[6-5]	4.16

Table 2.2: Comparison of the mean values for three different 7-qubit IBM Quantum systems. The mean values are listed for T_1, T_2, e_{RO}, e_{SX} and e_{2Q} for 3- and 4-qubit values, respectively. The last two columns represent the error values calculated from the plots of quantum hardware collected data as seen in Fig. 2.6 from the experiments run on 7-qubit devices Casablanca, Jakarta, and Lagos.

Device	T_1 (μ s)	T_2 (μ s)	e_{RO} ($\times 10^{-2}$)	e_{SX} ($\times 10^{-4}$)	3-qubit e_{2Q} ($\times 10^{-3}$)	4-qubit e_{2Q} ($\times 10^{-3}$)	$e_{3-\text{qubit}}$ (from plot)	$e_{4-\text{qubit}}$ (from plot)
Casablanca	94.5	146	2.32	3.61	8.69	9.15	0.171	0.935
Jakarta	121.9	29.6	3.26	2.47	10.27	13.97	0.262	1.104
Lagos	124.1	109.7	2.3	3.52	4.32	7.94	0.236	0.969

In Fig. 2.4**(a)** we were able to obtain the transition amplitudes within the error bars up to $t \approx 1.5$ for the 3-qubit case, and we can observe the flavor oscillation for this particular case. On the other hand, although most of the exact data points are not within the error bars of the experimental data points and there is significant drift in the experimental data, we were able to observe the flavor oscillation even for the 4-qubit case (Fig. 2.4**(b)**).

To explore how these errors depend on the device chosen, we conducted the same experiments on other IBM Q 7-qubit devices (qubit orientation shown in Fig. 2.5), i.e., Jakarta and Lagos, and compared their performance with the performance of the Casablanca device. We present Fig. 2.6, which compares the transition amplitudes obtained for 3- and 4-neutrino systems as a result of experiments run on these 3 devices. In Table 2.1, we present IBM-reported calibration data (T_1, T_2, readout assignment error, SX-error, and two-qubit CNOT error values) for these 3 devices for the qubits that are used during the experiments. We chose qubits [2,1,3] and [2,1,3,5] for devices Casablanca and Jakarta and qubits [3,5,6] and [1,3,5,6] for devices Lagos. These qubits were chosen such that they had the best CNOT error rates reported by IBM on that particular day. The experiments conducted on IBM Quantum Casablanca and Jakarta were run on 9/23/2021, and the experiments run on Lagos were run on 9/16/2021. In Table 2.1 we present the IBM-reported T_1, T_2, readout error (e_{RO}), SX gate error (e_{SX}), and two-qubit CNOT gate error (e_{2Q}) values for each qubit used on each device. These qubits were chosen such that the reported CNOT error rates for the chosen qubit pairs are minimal throughout the device. In Table 2.2 we report the mean values for the IBM-reported calibration values demonstrated in Table 2.1 and 3- and 4-qubit errors, $e_{\text{3-qubit}}$ and $e_{\text{4-qubit}}$ obtained from the plots. The errors from the plots were calculated by finding the standard deviation of the hardware data from the exact Trotterization data and then adding it all up for each data point. As a result, as seen in Fig. 2.6 there is not a significant difference between hardware data obtained from each device. 3-qubit results are better than the 4-qubit results since the 3-qubit quantum circuit is much shorter than the 4-qubit one. Even

though 4-qubit data demonstrates a significant hardware noise, the collective neutrino oscillations are visible in both cases. For 3-qubit experiments run on different 7-qubit devices some of the data points lie in error bar limits of the exact Trotterization data. Another interesting observation from these data is that the noisy simulator does not capture all quantum hardware errors since it does not include errors such as cross-talk between qubits. According to these data, even though, on average, 2-qubit errors are the least in the Lagos device, the errors obtained from plots for the Casablanca device are the least. The average error for the device Jakarta is consistent with the reported average CNOT error rate. There is ongoing research to explore how well IBM-reported error rates obtained using randomized benchmarking reflect the errors obtained from experiments.

There are two obvious sources of errors contributing to the error in the transition amplitudes in addition to other quantum hardware noise. The first one is that the *isometry* function in the Qiskit library provides quantum circuits that require a hardware layout with a connection between all physical qubits. Thus, implementing it on quantum hardware for the transition probabilities results in more noise on a four-qubit quantum circuit compared to a three-qubit quantum circuit. In most IBM quantum devices, an all-to-all connection between qubits is not available. SWAP gates are added to the quantum circuit to trigger interaction between qubits that are not connected. Each additional SWAP gate costs 3 CNOT gates on an IBM quantum system, increasing the noise in the system significantly. To address this source of error, we are going to conduct this experiment on an ion-trapped quantum computer, which has all-to-all qubit connectivity, and work in this direction is in progress. In addition to the connectivity issue, the quantum circuit is deeper in the 4-qubit case than in the 3-qubit case, and this is another source of error considering the decoherence time in quantum hardware. This source of error can be addressed using different circuit optimization tools such as QSearch [80] or QGo [81], which might provide a shorter circuit depth and reduce the number of CNOT gates in the quantum circuit. Work in this direction is also in progress. However, our error mitigation strategies allowed

us to obtain energy eigenvalues of a 4-neutrino system in very good agreement with the exact eigenvalues.

For the 3-qubit data presented in Figs. 2.2 We used both readout error mitigation (ROEM) and zero noise extrapolation (ZNE) to mitigate the error caused by the noise from quantum hardware. To mitigate the error from the measurements, we employed tools included in IBM's *qiskit-ignis*, making use of a constrained matrix inversion approach. The response matrix is obtained from the measurements on $2^{N_{\text{qubits}}}$ calibration quantum circuits prepared so that initialization of the qubits was followed by all combinations of the single-qubit X gate. These quantum circuits were bundled with the circuits of interest in the same job. For the error mitigation of the two-qubit gate errors, we used the zero-noise extrapolation [79] where we added double CNOT gates to the circuit for each CNOT gate in the original quantum circuit in order to increase the noise due to the CNOT gate. Then, we extrapolated to "noiseless" expectation values for the energy measurements and count values for the transition probability measurements. We ran the quantum circuits with $r = 1, 3, 5, 7, 9$, where r is the number of CNOT gates in the quantum circuit for each CNOT gate in the original circuit. Then we fitted the data points by a polynomial of degree n for n-qubit quantum circuits ($n = 3, 4$). For two-qubit circuits and Trotterization, we only used ROEM.

2.5 Conclusion

In this work, we performed calculations involving many-body collective neutrino-flavor oscillations on a quantum computer using the QLanczos algorithm and a first-order Trotterization algorithm. We obtained the energy levels of the Hamiltonian of the system and transition probabilities demonstrating neutrino oscillations on a quantum computer. Working with the collective neutrino oscillations Hamiltonian in the neutrino mass basis, we reduced the Hamiltonian into smaller Hamiltonian blocks to reduce the noise on quantum hardware. With the inclusion of the readout and

Richardson extrapolation error mitigation strategies, we calculated the energy levels of the system and found them to be in very good agreement with the exact values for a four-neutrino system. We could also observe collective neutrino-flavor oscillations in three- and four-neutrino systems on a quantum computer.

Besides the quantum-classical QLanczos algorithm used here, recently a number of novel methods have been proposed for the calculation of the ground state energy as well as higher energy levels of quantum systems. For example, the quantum inverse iteration algorithm [82] introduces an approximate ground state, which is prepared by a successive application of the inverse Hamiltonian. Another method that was recently proposed is the quantum version of the power method [83] in which $\mathcal{H}^n|\psi\rangle$, where $\mathcal{H}$ is the Hamiltonian, is calculated from a time-discretized version of the higher-order derivative of the time-evolution operator $\mathcal{U}(t) = e^{-i\mathcal{H}t}$ using $\mathcal{H}^n = i^n d^n \mathcal{U}(t)/dt^n|_{t=0}$. Also, the quantum power method was applied to Krylov-subspace diagonalization. On the other hand, Krylov space methods such as the classical Lanczos algorithm have also been proposed as an error mitigation method in [84]. This method is hardware agnostic and compatible with other error mitigation techniques and is worth exploring in the context of neutrino oscillations.

The quantum circuits employed in this work generally require connections between all qubits in the circuit. It would be interesting to implement our algorithm on an ion-trapped quantum computer with all-to-all connectivity and see how our results would improve with less noise. This would also allow us to increase the number of neutrinos in the system. Although all-to-all connectivity might decrease the noise due to the addition of extra SWAP gates, this might result in a significant decrease in gate and algorithmic speeds [85]. It is important to optimize the trade-off between connectivity and the speed of the algorithm.

Chapter 3

Continuous-variable Quantum Neural Network

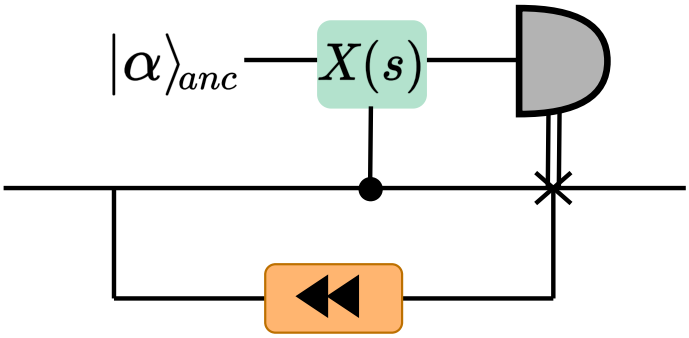
Continuous-variable (CV) quantum computing (QC) takes advantage of the wave-like properties of particles. For example, it can be realized by photonic quantum hardware manipulating electromagnetic fields. Thus, CV QC is achievable in quantum optics by utilizing continuous quadratures of the quantized electromagnetic field [86] enabling the essential steps in quantum algorithms (preparation, unitary manipulation, and measurement of (entangled) quantum states). CV quantum algorithms have been developed for various applications, ranging from quantum field theory [87, 88, 89] to machine learning [90]. Another interesting application of CV architecture is calculating quantum averages using squeezed states. This feature was studied in [91], where the advantage of quantum averaging compared to a standard arithmetic mean strategy was demonstrated.

It has recently been shown that CV is a suitable architecture for building quantum neural network (QNN) models on quantum computers [45]. The developed CV QNN architecture has been applied to various practical, real-world problems, e.g., function fitting, fraud detection of credit card transactions, image classification of hand-written digits, data encryption and decryption in a secure cryptography algorithm

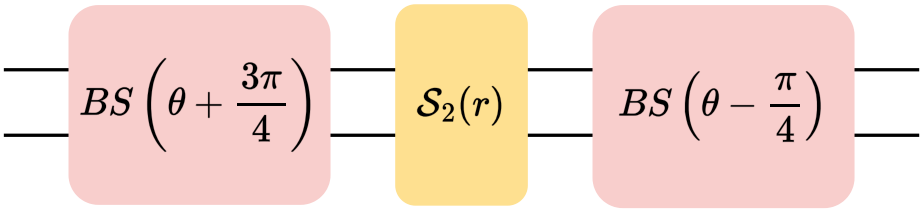
[92], and entangled state detection [93]. In addition to these applications, CV QNN has been utilized as a generator and discriminator in a CV quantum adversarial network (CV QGAN) [94] to reproduce the data outputs of the calorimeters for data collected in high energy physics experiments at CERN. The authors of [95] utilized a Quantum Convolutional Neural Network (QCNN) for detecting quantum phase transitions in axial next-nearest-neighbor Ising (ANNNI) models. While the QCNN approach emphasized bolstering out-of-distribution generalization through a partially supervised strategy and showcased resilience against barren plateaus [96], our work diverges by focusing on CV quantum systems. This adaptation is particularly advantageous for scenarios such as optical systems, enabling seamless integration into specific experimental setups, thus streamlining hardware realization.

In the CV QNN model discussed in [45], a non-Gaussian Kerr gate provides the non-linearity required for neural networks. However, experimentally realizing these non-Gaussian operations is challenging due to their weakly interacting nature. Therefore, we developed an alternative CV QNN model in which non-linearities are introduced through measurements on ancillary qumodes following the proposal in [97] based on repeat-until-success measurements. In this paper, we propose a variational hybrid quantum-classical circuit implementing a neural network that solves well-known problems such as function fitting, state preparation, binary classification, and image recognition. The quantum circuit uses only Gaussian gates that can be implemented with optical elements, such as beam splitters and squeezers. Thus, our CV hybrid neural network can be realized experimentally using current photonic quantum hardware.

Similar measurements on ancillary qumodes to create desired quantum states have been considered before. Single- and two-mode quantum gates acting on photonic qubits were generated with single photon sources, linear optical elements, and measurements of ancillary modes [98]. To generate the cubic phase gate, nonlinear quadrature measurements were implemented using ancillary states, homodyne measurements, and nonlinear feedforwards based on the measurement results [99].



(a)



(b)

Figure 3.1: Two-qumode Gaussian quantum circuit element introducing non-linearity in a CV QNN. **(a)** The ancillary qumode, initially in the coherent state $|\alpha\rangle_{\text{anc}} = \mathcal{D}(\alpha)\,|0_{\text{anc}}\rangle$, is entangled with the primary qumode with a controlled displacement ($CX(s)$) gate. The primary qumode goes through a feedback loop until the detector on the ancillary qumode clicks [97]. **(b)** The $CX(s)$ gate with parameters given in Eq. (3.1).

Photon-number-resolving (PNR) measurements were used for the production of multi-mode Gaussian states, including cat (superpositions of coherent states), ON, Gottesman-Kitaev-Preskill (GKP) [100], NOON [101], and bosonic code states [102]. Cat and GKP states were also created experimentally within a CV cluster state by performing PNR measurements [103]. To the best of our knowledge, ours is the first attempt at applying non-linearities induced by measurements to CV QNNs.

The structure of this chapter is as follows. We describe the CV neural network in Section 3.1, followed by a detailed account of our proposal to introduce non-linearities that avoid non-Gaussian gates. In Section 3.2, we study several problems involving varying degrees of hybridization between quantum and classical neural networks. We conclude with a discussion of the potential of this work in Section 3.3.

3.1 The Method

A general CV QNN was discussed in Ref. [45]. It included N-port linear optical interferometers consisting of beam splitter ($\mathcal{BS}$), rotation ($\mathcal{R}$), displacement ($\mathcal{D}$), and squeezing ($\mathcal{S}$) gates. It also featured a non-Gaussian Kerr gate (Φ), introducing non-linearity to the neural network. In our setup, we have replaced this single-mode gate with a two-mode quantum circuit element consisting of Gaussian gates and a photon detector, which is experimentally feasible with current technology (shown in Fig. 3.1).

In more detail, our nonlinear circuit element is implemented by adding an ancilla qumode in a coherent state $|\alpha\rangle = \mathcal{D}(\alpha)\,|0\rangle_{\text{anc}}$ to the primary qumode, where $\alpha \in \mathbb{R}$. The two modes are then entangled by a controlled displacement ($CX(s)$) gate that uses the q-quadrature of the primary mode to shift the q_{anc} quadrature by sq, where $s \in \mathbb{R}$. It is implemented with beam splitters and a two-mode squeezer of parameters

$$\cot 2\theta = \sinh r = -s \ . \tag{3.1}$$

If the incoming state of the primary mode is $|\psi\rangle = \int dq\, \psi(q)\, |q\rangle$, then the final two-mode state is the entangled state

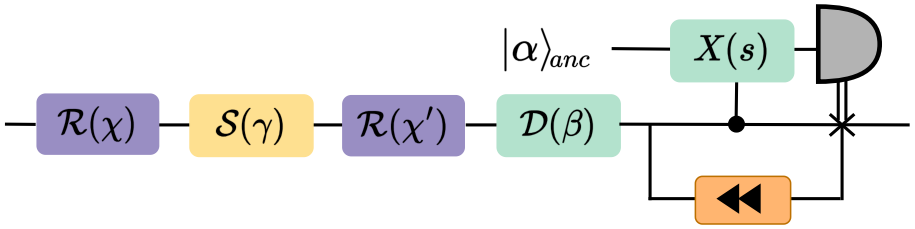
$$\int dq\, \psi(q)\, |\alpha + sq\rangle_{\text{anc}} \otimes |q\rangle \ . \tag{3.2}$$

Then, we use a photon detector to measure the photon number of the ancilla qumode. If the detector clicks, the process is successful, and the primary qumode proceeds to the next layer. For small α and s parameters, the ancilla qumode decouples. The outgoing primary qumode is in the (unnormalized) state

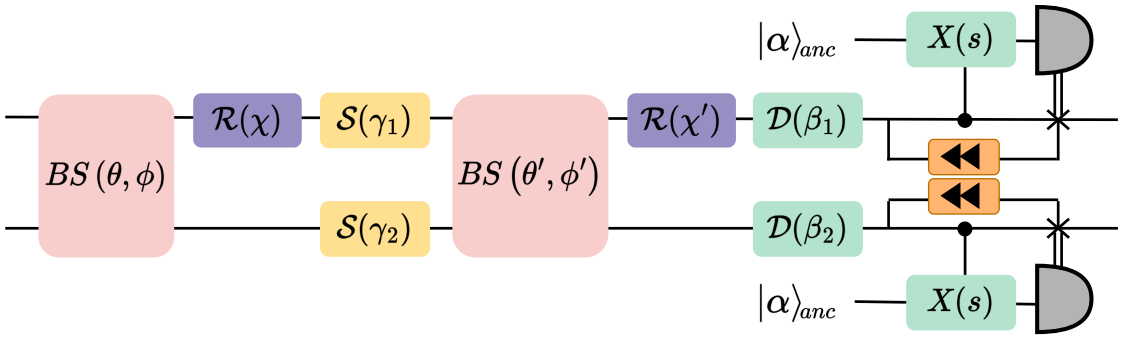
$$\int dq\, (\alpha + sq)\psi(q)\, |q\rangle \ , \tag{3.3}$$

showing that we have effectively applied the non-Gaussian gate $\alpha + sq$. For larger values of the parameters, the expression for this effective gate is more complicated, but it is still a non-Gaussian gate.

Suppose there is no click in the detector. In that case, the output state of the primary mode is approximately the same as the input state, $|\psi\rangle$ for small α and s, changing slightly for larger values of the parameters (with insertions of factors of $e^{-|\alpha+sq|^2/2}$). We feed the state back into the input port and repeat the process shown in Fig. 3.1. This loop continues until the detector clicks and the primary qumode can advance (repeat-until-success process [97]). We fix the coherence parameter α of the ancillary qumode, treating it as a hyperparameter and not a trainable parameter. In our simulations, we set $\alpha = 1$.



(a) Single-mode layer



(b) Two-mode layer

Figure 3.2: Detailed CV neural network architecture

A detailed neural network architecture for single-mode and two-mode layers is shown in Fig. 3.2. We create a multiple-layer structure by arranging each layer as a building block of the neural network with the gate variables $(\theta, \phi, \chi, x, \alpha, \beta)$ being free parameters, collectively denoted by $\vec{\zeta}$. We want to find $\vec{\zeta}$ such that the value of the cost function $(C(\vec{\zeta}))$ is minimum. This can be done using various optimization techniques available in deep learning (for example, gradient descent, stochastic gradient descent, and, most commonly, the Adam optimizer). The parameters are updated by the rule:

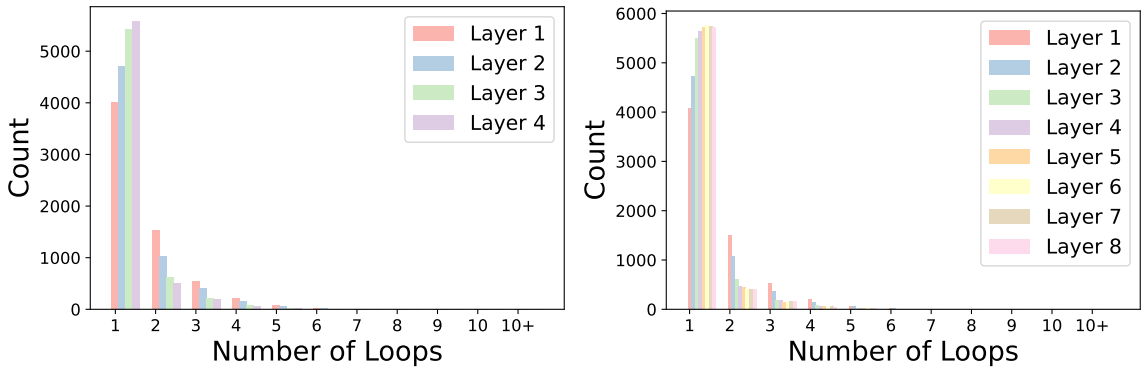
$$\vec{\zeta} \longrightarrow \vec{\zeta} + \eta \nabla_{\vec{\zeta}} C(\vec{\zeta}) \ , \tag{3.4}$$

where η is the step size, also known as the learning rate. This procedure continues until the model cannot modify the cost function further. Then, the parameters associated with the lowest observed cost function offer the best solution to the given task. Note that the free parameters correspond to a circuit; hence, the circuit is the solution.

In principle, only one layer $(\mathcal{L})$ is sufficient to parameterize every possible unitary affine transformation on N modes. However, deeper architectures provide increased expressive power, better learning capabilities, and a more efficient representation of complex transformations.

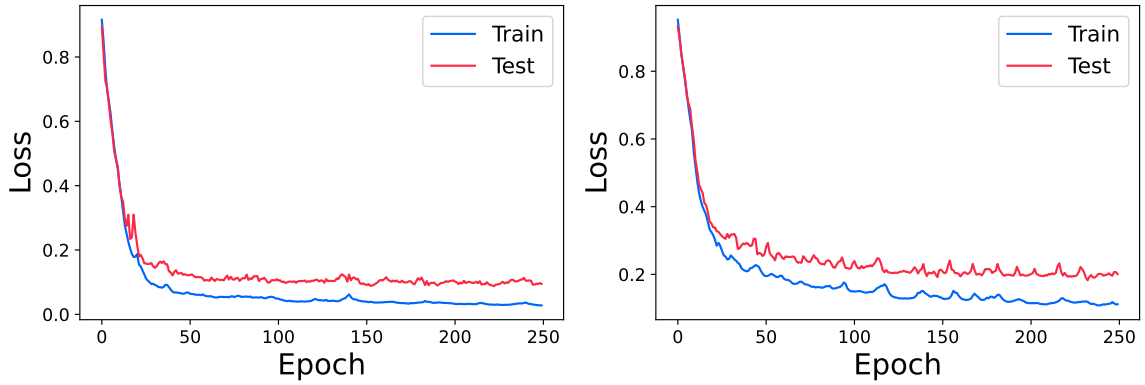
To test our neural network, we studied various machine learning and quantum computation problems, namely, state preparation, curve fitting, binary classification, and image recognition with varying degrees of hybridization in quantum and classical neural networks, as discussed in Section 3.2.

We conducted our simulations using Strawberry Fields quantum computing software. The realization of the quantum circuit element implementing non-linearity depicted in Figure 3.1 in a QNN setting turned out to be challenging. To obtain a good approximation to our setup that could be implemented with software available to us, we adjusted the coherent parameter of the ancillary qumode so that a single photon would be detected in that mode with a high success probability.



(a) **(b)**

Figure 3.3: Number of measurements necessary on the ancillary qumode to achieve successful measurement for each quantum layer of sample hybrid networks for binary classification, consisting of two classical and **(a)** four or **(b)** eight quantum layers.



(a) **(b)**

Figure 3.4: Training and testing loss of MNIST classification models with successful measurement of the ancilla within **(a)** one feedback loop (Model 1) and **(b)** two feedback loops (Model 2).

To estimate the number of ancillary qumode measurements (or feedback loops) necessary for the successful application of the quantum circuit element, we utilized Bosonic Qiskit software [104] for simulating an elementary circuit and tracking the number of measurements required for success in various applications. Sample results are illustrated in Figure 3.3. Using the architecture of the binary classification circuit discussed in Section 3.2.3 as a concrete example, we counted the number of repeated ancillary qumode measurements required for a successful pass of the primary qumodes during forward propagation through the network. We plotted the success rates per layer for setups utilizing various numbers of layers. Evidently, most successful measurements occur at the first photon detection of the ancillary qumode.

Due to the constraints in the Strawberry Fields library, we were compelled to perform post-selection of a single Fock state during the ancilla measurement step to collapse the wavefunction, thus effectively implementing measurements by photon-number-resolving detectors. It would also be interesting to simulate photon detectors that cannot resolve photon numbers and are widely available. This would alter the effective nonlinear operation slightly (depending on the choice of the laser intensity for the ancillary qumodes) but would simplify the experimental setup. To demonstrate that a high success rate can be achieved even without a photon-number-resolving detector, we trained two distinct models that performed multi-label classification on the MNIST handwritten digit data set discussed in Section 3.2.4. The training and testing loss values for these classical-quantum hybrid models are presented in Fig. 3.4. Model 1 was designed to successfully select the required state in the ancilla measurement on the initial attempt, whereas Model 2 was set up to fail the first measurement attempt and succeed on the second measurement. Even though the loss function and classification accuracy experienced a slight decline at the end of training from 97.25% to 96.49%, we were still able to train Model 2 successfully and achieve a reasonably high classification accuracy for the 4-class MNIST handwritten digit dataset classification.

3.2 Case Studies

In this section, we develop CV QNN models for various applications, including quantum state preparation (Section 3.2.1), curve fitting (Section 3.2.2), binary classification of fraud and genuine credit card transactions (Section 3.2.3), and multi-label classification of MNIST handwritten digits (Section 3.2.4). We analyze hybrid quantum-classical and fully quantum layers to observe the effect of the classical and quantum neural network layers.

We simulated our CV QNN models using the Strawberry Fields software platform [105]. The quantum machine learning toolbox application is built on top of it with Tensorflow features [106]. We used the quantum circuit simulator, optimized the algorithm, and trained the neural network to obtain the desired results.

3.2.1 State Preparation

The CV QNN model for quantum state preparation trains a quantum circuit to generate a target quantum state. To this end, we provide a canonical input state $|\Psi_0\rangle$ and target output state $|\Psi_t\rangle$ and aim to find out the circuit U (a unitary transformation) such that

$$|\Psi_t\rangle = U\,|\Psi_0\rangle \ . \tag{3.5}$$

For simplicity, we fixed the input state to be the vacuum, $|\Psi_0\rangle = |0\rangle$.

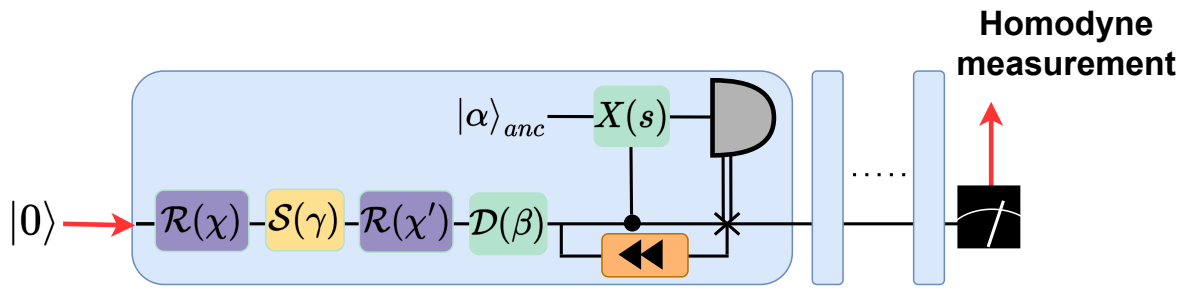


Figure 3.5: CV quantum NN architecture for quantum state preparation. The input to the network is the vacuum state, and at the end, a homodyne measurement is performed.

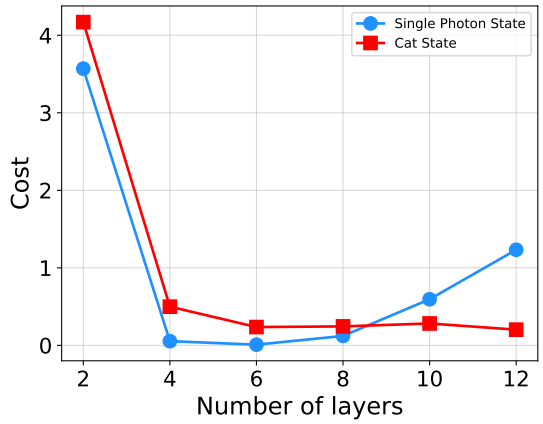


Figure 3.6: Value of cost function for a different number of layers for preparation of single-photon (blue circles) and cat (red squares) states. Values averaged over 5 independent runs. The cutoff dimension was set to 6 (10) for the single-photon (cat) state.

We considered a basic single-mode architecture of a quantum neural network with a fixed number of layers, as shown in Fig. 3.5. As described earlier, our goal was to find parameters $\vec{\zeta}$ determining a unitary $U(\vec{\zeta})$ such that $U(\vec{\zeta})\,|0\rangle = |\Psi_t\rangle$, or equivalently, $|\,\langle\Psi_t|\,U(\vec{\zeta})\,|0\rangle\,| = 1$. The fidelity of the output states is defined in terms of the overlap between the target and optimized states as

$$F = |\,\langle\Psi_t|\,U(\vec{\zeta})\,|0\rangle\,|^2\ . \tag{3.6}$$

Then, the cost function to be minimized can be defined in terms of fidelity as

$$C = |\sqrt{F} - 1| = \Big|\Big|\langle\Psi_t|\,U(\vec{\zeta})\,|0\rangle\Big| - 1\Big|\ . \tag{3.7}$$

It should be noted that a slightly different definition of the cost function has been used before, $C = \Big|\langle\Psi_t|\,U(\vec{\zeta})\,|0\rangle - 1\Big|$, which involves the phase of the output state. Including the phase adds flexibility to the search space determining the unitary $U(\vec{\zeta})$, which is lost if our definition (3.7) is used. However, this turned out not to be an impediment in all cases we considered. The advantage of using the definition (3.7) is that the cost function can be deduced experimentally through the Wigner function by performing homodyne measurement on the output state at each optimization step.

In detail, to obtain C, we perform homodyne tomography on the output state $U(\vec{\zeta})\,|0\rangle$. We measure the quadrature $X_\phi = \frac{1}{2}(e^{i\phi}a^\dagger + e^{-i\phi}a)$, and obtain a series of output pairs (ϕ_k, x_k) $(k = 1, \ldots, N)$. They allow us to obtain an estimate of the Wigner function of the final state, $W_{U(\zeta)|0\rangle}(q,p)$. By comparing with the Wigner function of the desired state, $W_{|\Psi_t\rangle}(q,p)$, we deduce the cost function C from the fidelity (overlap)

$$F = 2\pi \int dqdp\, W_{U(\zeta)|0\rangle}(q,p) W_{|\Psi_t\rangle}(q,p)\ . \tag{3.8}$$

To test the performance of the QNN, we prepared two different states, the single-photon state $|\Psi_t\rangle = |1\rangle = a^\dagger |0\rangle$, and the cat state

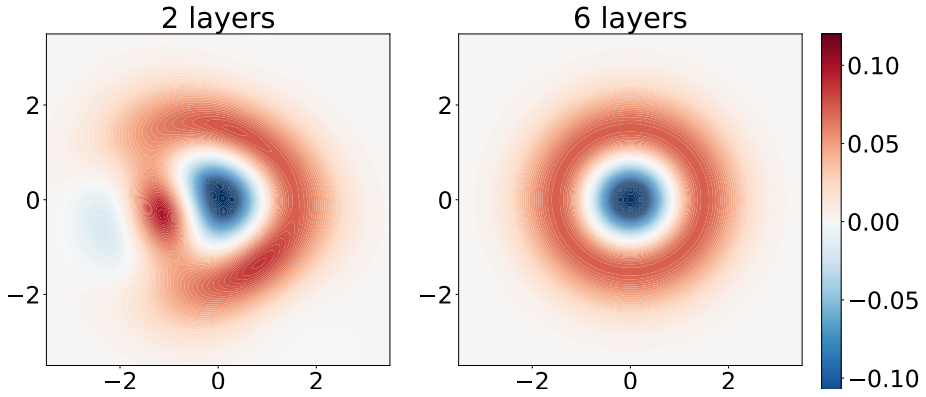
$$|\text{cat}, \theta\rangle = \frac{1}{\sqrt{2}} \left(\mathcal{D}(\alpha_0) + e^{i\theta}\mathcal{D}(-\alpha_0)\right) |0\rangle \tag{3.9}$$

where $\theta, \alpha_0 \in \mathbb{R}$.

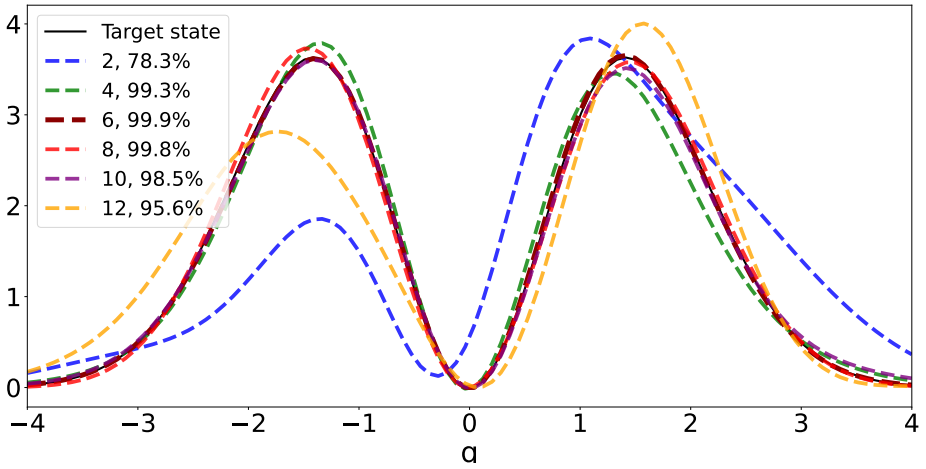
Since we were using Strawberry Fields software, we could quickly get the Wigner function after performing the homodyne measurement. However, these Wigner functions are incompatible with the commonly used efficient TensorFlow framework to perform the optimization. Hence, we used the Python library - Scipy [107] to optimize cost as Scipy provided the Nelder-Mead optimization technique [108]. This technique is non-gradient based and designed for high-dimensional minimization, which worked best for our purposes as our goal was to minimize the cost built from 2D Wigner functions obtained via homodyne detection.

We noticed that just in a few steps, the model started to learn the state. For best results, we ran the model with a different number of layers, as shown in Fig. 3.6. In these simulations, we did not include any possible quantum hardware errors. A small number of layers yields a higher cost, but fewer layers require fewer gates and lead to fewer errors due to quantum hardware imperfections. Ignoring such errors, as the number of layers increases, the cost is lowered. Notice that the cost starts to increase again beyond a certain number of layers due to overfitting, e.g., for more than 6 layers in the case of the single-photon state. There is an optimum number of layers, which would be important to determine by including a realistic model of quantum hardware.

For the single photon state, 6 quantum layers gave the best results with a fidelity of 99.9%. The cost achieved after optimization was 0.008 after 5000 steps. We used a cutoff dimension of 6; the maximum number of steps was fixed at 5000.



(a)



(b)

Figure 3.7: Study of the effect of changing the number of layers of the QNN for the preparation of a single-photon state. (a) 2D Wigner function comparison for 2 and 6 quantum layers, with fidelities 78.3% and 99.9%, respectively. (b) 1D Wigner function for various numbers of quantum layers. 6 quantum layers produce the best fidelity. The maximum number of steps was kept fixed at 5000 and the cutoff dimension at 6.

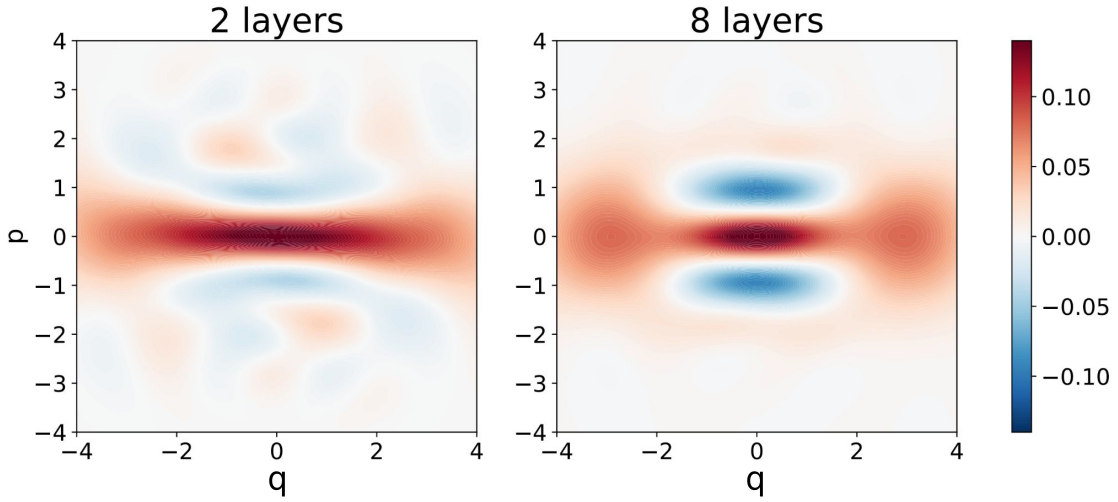


Figure 3.8: Preparation of an even cat state (Eq. (3.9) with $\alpha_0 = 1.5$, $\theta = 0$). 2D Wigner function comparison for 2 and 8 quantum layers with fidelity 79% and 99.8%, respectively. A cutoff dimension of 10 was used in both cases. 2 (8) layers were optimized in 2000 (9800) steps, resulting in a cost of 2.64 (0.03).

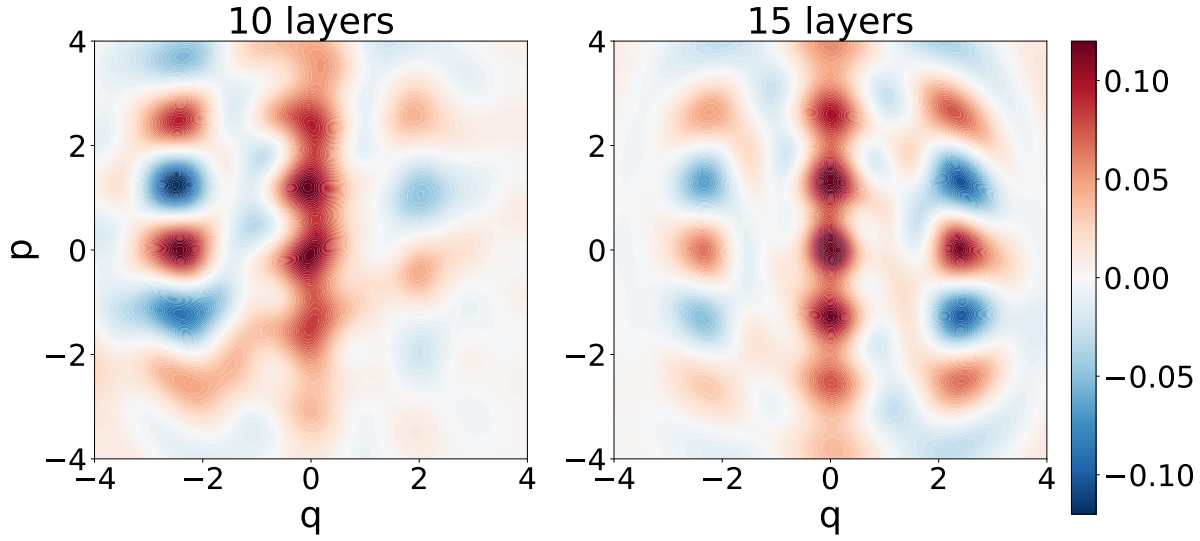


Figure 3.9: Preparation of a realistic GKP state (Eq. (3.11) with $\epsilon = 0.1$). 2D Wigner function comparison for 10 and 15 quantum layers with fidelity 70.6% and 93.9%, respectively. A cutoff dimension of 15 was used in both cases. 10 (15) layers were optimized in 18000 (15000) steps, resulting in a cost of 4.87 (1.1).

It should be noted that in the Nelder-Mead optimization, the number of steps is determined by the difference between two consecutive cost values. This difference is treated as another hyperparameter during the training process. The result of comparing other numbers of layers is shown in Fig. 3.7. We plotted the 2D Wigner function for 2 and 6 layers to demonstrate the importance of the optimal number of layers. We also showed the 1D Wigner function for different quantum layers. We integrated the 2D Wigner function over the momentum using the Scipy library to obtain the 1D Wigner function. The plot shows that the results improve up to 6 quantum layers, and beyond that, they worsen due to overfitting. Turning to state preparation of the cat state Eq. (3.9), which is a superposition of two coherent states, we concentrated on the even cat state with $\alpha_0 = 1.5$ and $\theta = 0$. As the cat state is more complicated than the single photon state, we had to increase the cutoff dimension to 10. We achieved high fidelity with 8 quantum layers. The comparison of the 2D Wigner functions of 2 and 8 layers is shown in Fig. 3.8. We obtained excellent fidelity of 99.8% with 8 layers, and after training, the cost was minimized to the value of 0.03 after 9800 steps. For comparison, with 2 layers, we obtained fidelity of 79% with cost at 2.64 after 2000 steps.

We also prepared a realistic GKP state [100]. Ideal GKP states are linear combinations of an infinite number of eigenstates of the q-quadrature. We concentrated on the state

$$|\mathrm{GKP}\rangle_{\mathrm{ideal}} = \sum_{n=-\infty}^{\infty} \left| q = 2n\sqrt{\pi\hbar} \right\rangle \tag{3.10}$$

However, such states are not normalizable and impossible to create experimentally because they have infinite energy and each component would require an infinite amount of squeezing. For a realistic case, we applied an energy cutoff and defined the realistic GKP state [109]

$$|\mathrm{GKP}\rangle_{\mathrm{real}} = e^{-\epsilon a^\dagger a} |\mathrm{GKP}\rangle_{\mathrm{ideal}} \tag{3.11}$$

We chose $\epsilon = 0.1$. Since this state is more complex than a cat state, we had to increase the cutoff dimension even further to 15 and employ 15 layers. After 15000 steps, we achieved a fidelity of 93.9% at a cost of 1.1. The comparison between 10 and 15 layers is shown in Fig. 3.9.

Comparing with the fidelity of single photon and cat states, it should be noted that the cost for the number of layers depicted in Fig. 3.6 is larger than 4. To lower the cost to levels comparable with single photon and cat states, more than 12 layers are needed. Due to computational resource limitations, we could only go up to 15 layers. Considering more layers will improve the results for GKP state preparation as the number of learning parameters would increase. Another critical factor affecting the complexity of the GKP state preparation is the decay parameter (ϵ). The smaller this parameter is, the closer we are to the ideal GKP state. Our computational resources allowed us to go as low as $\epsilon = 0.1$.

To understand the effects of loss on state preparation, we applied a loss channel operation to every qumode at the completion of each quantum layer. This operation connects a given qumode, created by $a^\dagger$, with an environmental qumode, created by $b^\dagger$, which is initially in the vacuum state. At the end of the operation, the latter is traced out. The corresponding transformation is modeled by a beam-splitting operation,

$$a \mapsto \sqrt{T}a + \sqrt{1-T}b \tag{3.12}$$

where T represents the energy transmissivity.

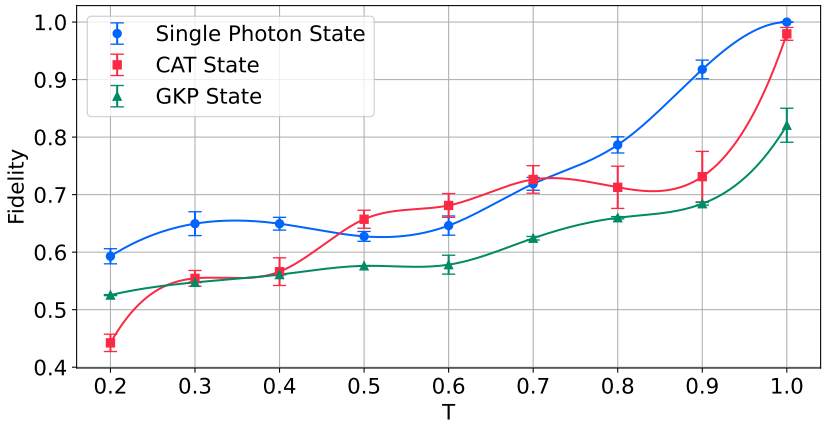


Figure 3.10: Impact of transmissivity (T) on fidelity for single photon, cat, and GKP states

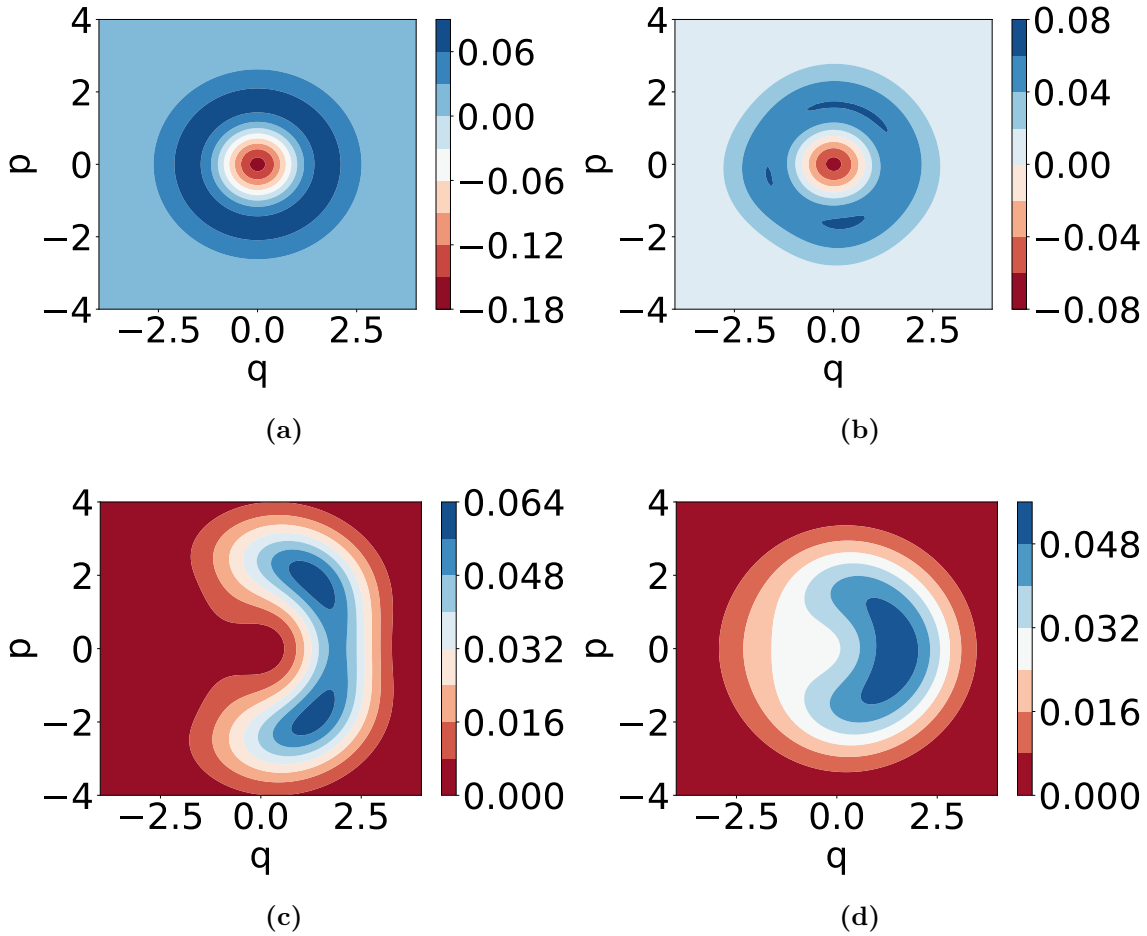


Figure 3.11: Wigner functions of the single photon state at transmissivity (T) values of (a) $T = 1$, (b) $T = 0.9$, (c) $T = 0.5$, and (d) $T = 0.2$, respectively. $T = 1$ corresponds to the noiseless case.

The effect of this operation, leading to a decrease in overall fidelity due to the introduction of loss in our simulations, is depicted in Fig. 3.10. The average fidelity remains notably high for some prepared states under a realistic loss scenario with $T = 0.9$. Specifically, the single photon state exhibits exceptional performance, achieving an average fidelity of 92%, down from peak fidelity of 96% at $T = 1$ (no loss). The Wigner function of the single photon for different cases of T is shown in Fig. 3.11. Cat states drop to an average fidelity of 74%, down from a peak of 88% in the ideal case. The GKP state exhibits the lowest performance, with average fidelity at $T = 0.9$ dropping to 69%, showing that a lot more layers are needed for a decent performance.

3.2.2 Curve Fitting

Next, we build a CV QNN in a supervised learning setting to learn the relationship between input (x) and output ($f(x)$), also known as curve fitting. It is an essential part of data analysis and a classic machine-learning problem.

We encoded the classical input, x, sampled from a noisy function, $f(x)$, as the coherence parameter of the input qumode, $|x\rangle = \mathcal{D}(x)\,|0\rangle$. The objective was to train the CV QNN to generate output states $|\psi_x\rangle$ that have an expectation value of the quadrature, q, close to $f(x)$ (i.e., $\langle\psi_x|\,q\,|\psi_x\rangle \approx f(x)$) for a given input x. We studied the noisy sine function. The data were prepared as $f(x) = f_0(x) + \Delta f$ where Δf is a normal distribution with zero mean and standard deviation ϵ. The parameter ϵ determines the amount of error present in the training data. We chose the noisy sine function with $f_0(x) = \sin x$ in the range of $x \in (-2, 2)$.

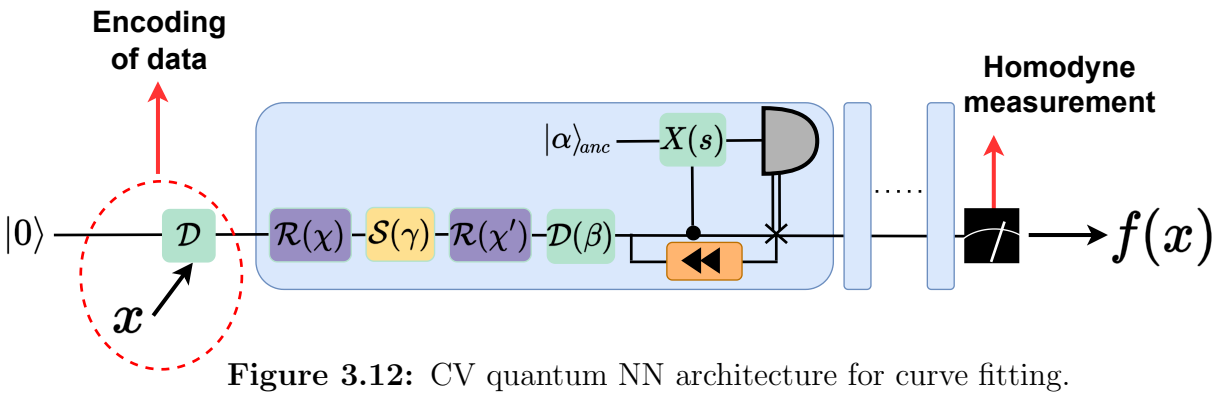


Figure 3.12: CV quantum NN architecture for curve fitting.

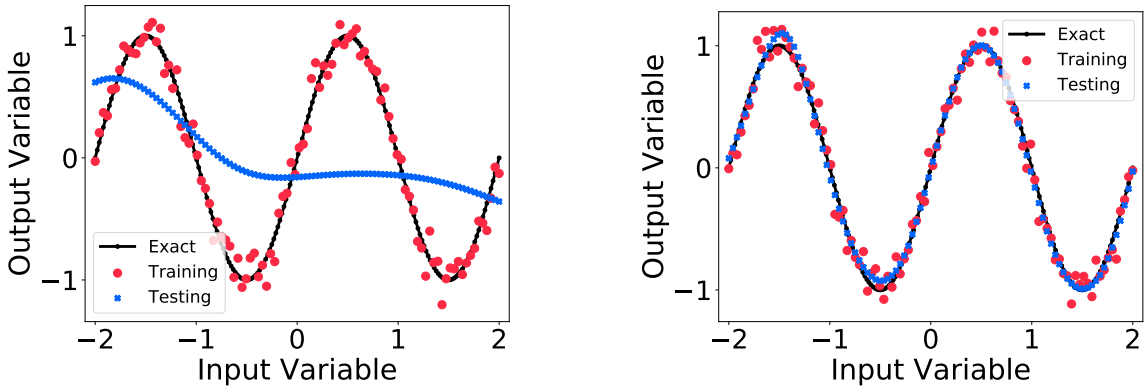


Figure 3.13: The effect of changing the number of layers on curve fitting of a noisy sine function: (a) 2 layers, cost = 0.318, and (b) 6 layers, cost = 0.008. 100 data points were used for training and testing in 1000 steps with error parameter $\epsilon = 0.1$ and cutoff dimension 6.

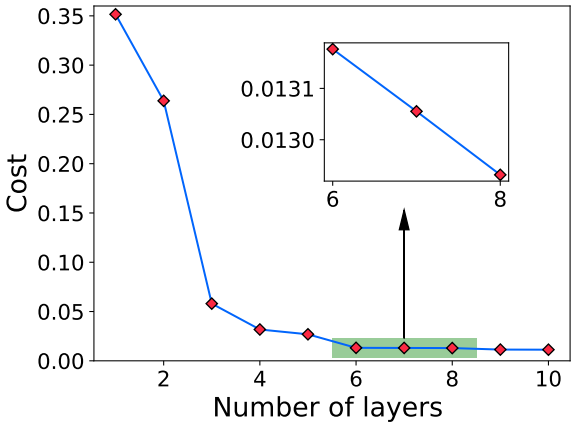
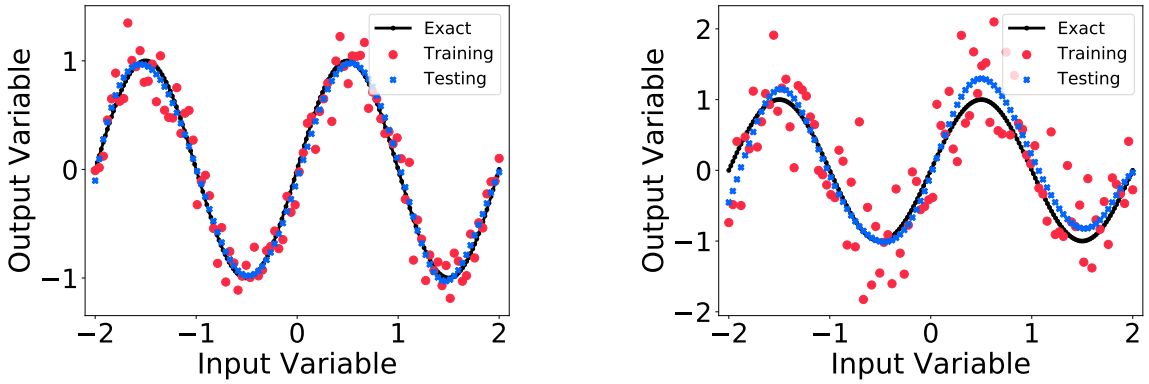


Figure 3.14: Cost function values vs. the number of layers plot obtained after performing learning on our CV quantum neural network model for fitting the noisy sine function curve. Each data point in the scatter plot corresponds to the 5 independent runs average. The training was done for 1000 steps with 100 data points (with $\epsilon = 0.1$) and a cutoff dimension of 6.

The architecture for our CV QNN is shown in Fig. 3.12. We used 6 quantum layers in this process. The training was done on 1000 steps with a Hilbert-space cutoff dimension of 6. The training and test data were prepared as tuples $(x_i, f(x_i))$, and x_i was chosen uniformly at random in the chosen interval. For training, we chose the cost function to be the mean square error (MSE) value between the circuit outputs and the desired function values,

$$C = \frac{1}{N}\sum_{i=1}^{N}\left[f(x_i) - \langle\psi_{x_i}|\, q\, |\psi_{x_i}\rangle\right]^2 \ . \tag{3.13}$$

To learn about the performance of CV QNN, we studied how the cost would change if we increased the number of layers. This helped us determine the optimum number of layers required for the desired results. We started with a single quantum layer and increased the number of quantum layers up to 10. We kept the number of steps fixed at 1000 with 100 data points and a cutoff dimension of 6. We used the Adam optimizer to minimize the cost function value. Some interesting results of how testing data behaved with a changing number of layers are shown in Fig. 3.13. The final result of the study is summarized in the cost vs. the number of layers in Fig. 3.14. We found that the cost function value decreases as we increase the number of layers, making the curve fit better. However, this improvement saturates around 6 quantum layers. One also has to keep in mind that more layers correspond to more number of training parameters. Hence, finding a number that yields good results and keeps the training parameters manageable is important. In the case considered here, optimal results were obtained with 6 layers.



(a) **(b)**

Figure 3.15: The effect of increasing the error in the data on curve fitting of a noisy sine function: (a) $\epsilon = 0.2$ and cost $= 0.037$, and (b) $\epsilon = 0.5$ and cost $= 0.232$. 100 data points were used for training and testing in 1000 steps with cutoff dimension 6 using 6 quantum layers.

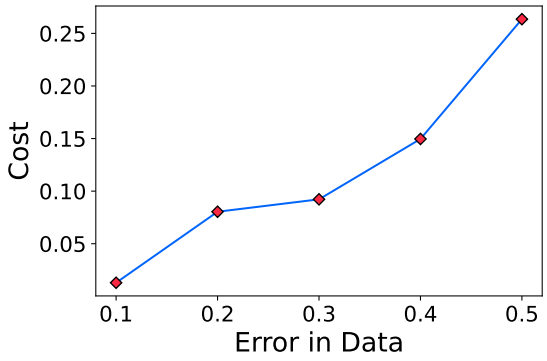


Figure 3.16: Plot of cost function value vs. noise in data determined by the parameter ϵ, obtained after learning on a QNN to fit the noisy sine function curve. Each data point in the scatter corresponds to the average of 5 independent runs. The training was done for 1000 steps with 100 data points, a cutoff dimension of 6, and using 6 quantum layers.

We also studied how the noise present in data affected our results by varying the error parameter ϵ discussed above. We kept the number of steps fixed at 1000 with 100 data points, a cutoff dimension 6, and 6 quantum layers. The results of $\epsilon = 0.2, 0.5$ are shown in Fig. 3.15. The value of cost increases from 0.037 to 0.232 as we increase the error from $\epsilon = 0.2$ to $\epsilon = 0.5$, and the fitting worsens as we increase the noise; this is expected as the model is training on noisy data. Although the fitting is getting worse with increased noise, the CV QNN still performs well in learning the shape of the sine function. The complete study of the dependence of the cost function on data noise (ϵ) is shown in Fig. 3.16.

3.2.3 Binary classification

For the third problem, we constructed a CV quantum-classical hybrid neural network to demonstrate its effectiveness in detecting fraudulent transactions on credit card purchase data. This is a binary classification problem, a canonical problem in machine learning. The main reason for including the classical layers in this problem is that we want to encode the data with the help of classical layers.

The credit card transaction data is taken from Kaggle [110], a publicly available database. Each transaction was flagged as either genuine or fraudulent with 28 features. Only 0.172% of transactions were fraudulent out of a total number of 284,807. The first step was to perform principle component analysis (PCA) to select the top 10 features containing the vital information. Since the data is highly unbalanced, the first step would be to perform data preparation.

First, we split the data into training and testing parts. In the training data set, we undersampled the genuine transactions by selecting them randomly and ensuring that the genuine-to-fraudulent transaction ratio was 3:1. All the remaining genuine transactions were added to the test data set.

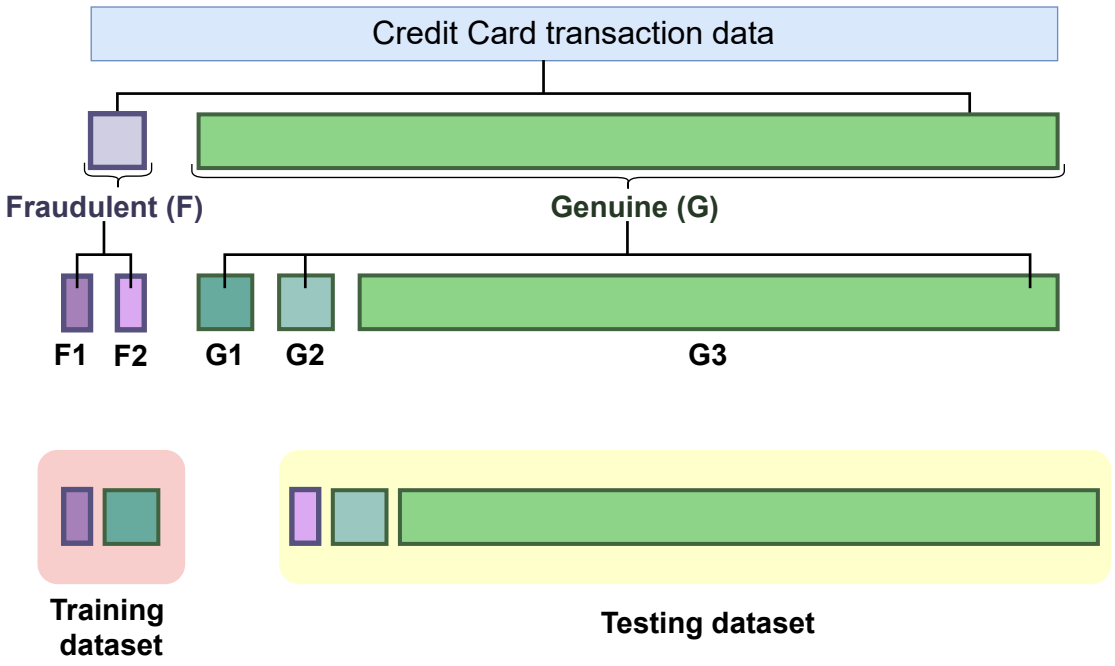


Figure 3.17: The description of data preparation done for binary classification problem. Note that the smaller parts of fraudulent and genuine transactions are denoted by the letters F1 and F2 and G1 and G2, respectively. The size of G1 and G2 are equal but three times more than F1 and F2 because of 3:1 undersampling. In the end, the training dataset consists of F1 and G1, and the testing dataset consists of F2, G2, and G3.

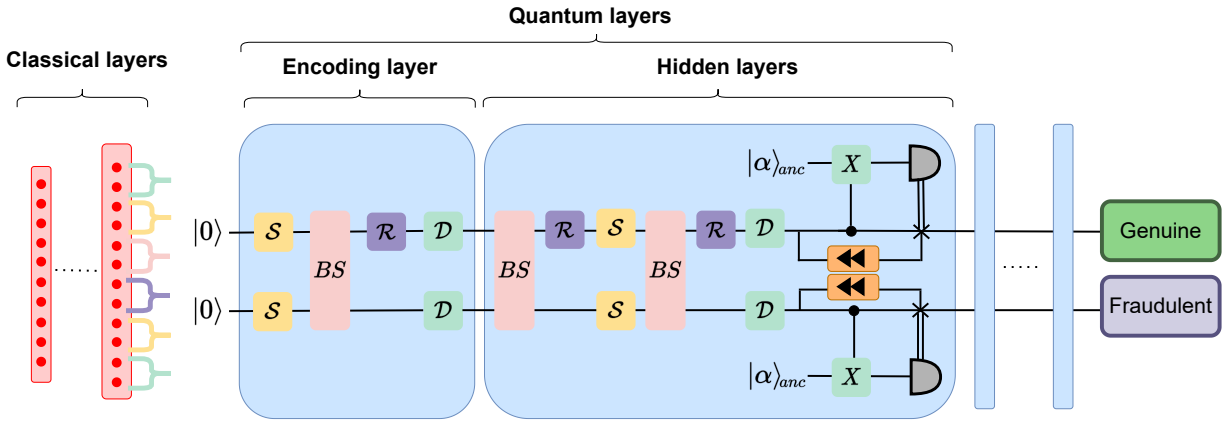
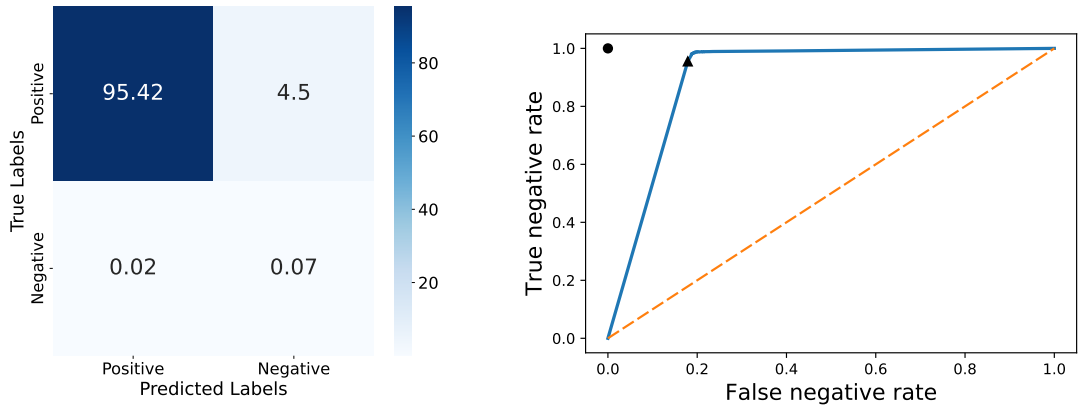


Figure 3.18: CV hybrid NN for fraudulent transaction detection in credit card data. The parameters of each gate in the encoding layer are obtained from the values of the last classical layer.

In this process, we ensured that the neural network learned the difference between genuine and fraudulent data by having just three times the genuine data in the training dataset. Also, the testing data is almost as unbalanced as the initial dataset. This data preparation is explained in detail in Fig. 3.17.

The network architecture is shown in Fig. 3.18. Four fully connected feed-forward classical layers are followed by five quantum layers with four modes (two are ancillary qumodes). The credit card data is fed into the first classical layer of size 10, followed by two hidden layers of the same size. The last classical layer of size 12 controls the gate parameters in the first quantum input layer. This layer marks the beginning of the quantum part of the neural network. And because we are letting the last classical layer control the gate parameters, the quantum layer is the encoding layer. We start with four vacuum qumodes. These layers contain two single-mode squeezing gates, S, one interferometer gate, U, two displacement gates, D, and two CX gates, which provide non-linearity through measurement on the ancilla qumode. At the end of the encoding quantum layer, a photon number measurement is performed on the two ancillary qumodes. The two primary qumodes are allowed to advance when the detectors on the ancillary qumodes click. If they do not click, the main qumodes are fed back into the quantum circuit, as shown in Fig. 3.18. The feedback loops are repeated until the detectors on the ancillary qumodes click, thus implementing the desired non-linearity in the CV quantum neural network [97]. We repeat this process for four more hidden layers. Finally, we measure the photon number on the two output primary qumodes that emerge after the last quantum output layer. If we find the photon in the first qumode, we call it a genuine transaction; if we find it in the second qumode, it is a fraudulent transaction.



(a) **(b)**

Figure 3.19: Results of fraud detection in credit card transactions: (a) Confusion matrix showing the accuracy of 95.48%, and (b) ROC curve with area under it 0.9. The CV hybrid NN had 2 classical and 4 quantum hidden layers.

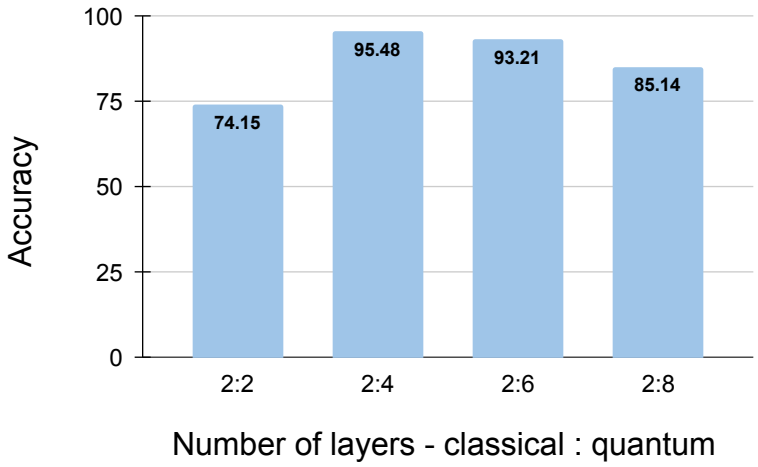


Figure 3.20: Plot showing the effect of changing the number of classical and quantum layers in a hybrid CV NN on the accuracy of the results for binary classification.

The training was performed using the Adam Optimizer with a batch size of 24. We minimized the cost function defined by:

$$C = \Sigma_{i \in \text{data}} \left(1 - p_i\right)^2 \ , \tag{3.14}$$

where p_i is the probability of detecting a photon for input i in the correct mode. We used a cutoff dimension of 8 in each mode for 10,000 batches. Once the model was trained, we tested it by choosing a threshold probability closest to the optimal ROC required for a transaction to be classified as genuine.

There are various ways to measure the accuracy of binary classification problems; for this case, we used a confusion matrix and Receiver Operating Characteristic (ROC) curve. The confusion matrix is a table containing four actual and predicted value combinations. The four quadrants of the confusion matrix contain results for true positive (TP), false negative (FN), false positive (FP), and true negative (TN) events. Various quantities of interest can be deduced from the confusion matrix, such as accuracy, precision, recall, and specificity. We calculated accuracy using the matrix:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.15}$$

The confusion matrix is shown in Fig. 3.19. The accuracy of the model calculated from the confusion matrix came out to be 95.48%. The TP quadrant shows that the model predicts the genuine data correctly more than 95% of the time. The number in the second quadrant representing FN events appears to be high. However, credit card companies can alert their users about such transactions, and by verifying these transactions, the FN rate can be brought down. The essential quadrant to consider is the third one that represents FP events, i.e., the fraudulent transactions that are incorrectly predicted as genuine. Fortunately, this number is very low for the trained model. Also, the testing data set (which was used to plot the confusion matrix) contains a tiny number of fraudulent transactions, all identified almost correctly, as

the percentage of fraudulent transactions in the testing data set matches the fourth quadrant.

The ROC curve is an essential tool in analyzing a classification model. It shows the trade-off between the TP and FP rates at different threshold values. It can help determine the optimal threshold for a given model and evaluate its overall performance. The solid black circle in Fig. 3.19(b) represents the ideal binary classifier at a TN rate of 1 and an FN rate of 0, whereas the solid black triangle is the closest point to the optimal within the chosen threshold. The area under the curve (AUC) is 0.90, which is close to the ideal value of 1. The AUC value is a good measure of the separability of the data being classified.

It should be pointed out that the number of features of the credit card transaction data we are interested in is equal to the number of parameters in the quantum circuit used for data encoding. Therefore, we investigated the role of the classical layer in this hybrid quantum-classical neural network architecture. We kept the classical layers constant at 2 and changed the quantum layer at an increment of 2. The results are plotted as accuracy calculated from the confusion matrices vs. different layers in Fig. 3.20. There is an optimal number of hybrid layer scenarios which give out the best results. When we use 2 classical and 2 quantum layers, the accuracy is around 74%, meaning there are not enough layers for learning. However, when we increased the quantum layers to 8 then, also the accuracy went slightly down, indicating the overtraining for a simple case of binary classification. We also understand that hyperparameters play a major role in training, but we found this for the set of hyperparameters we chose.

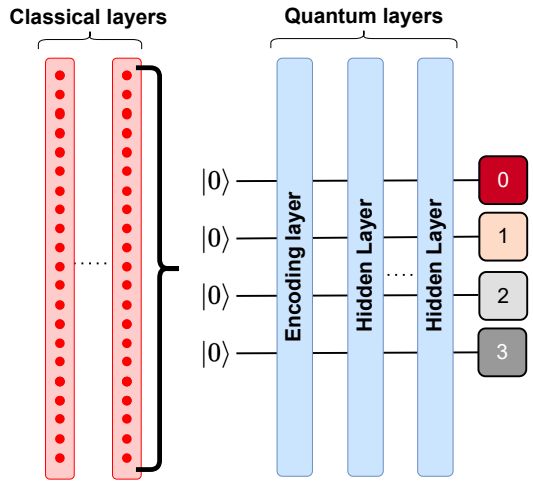
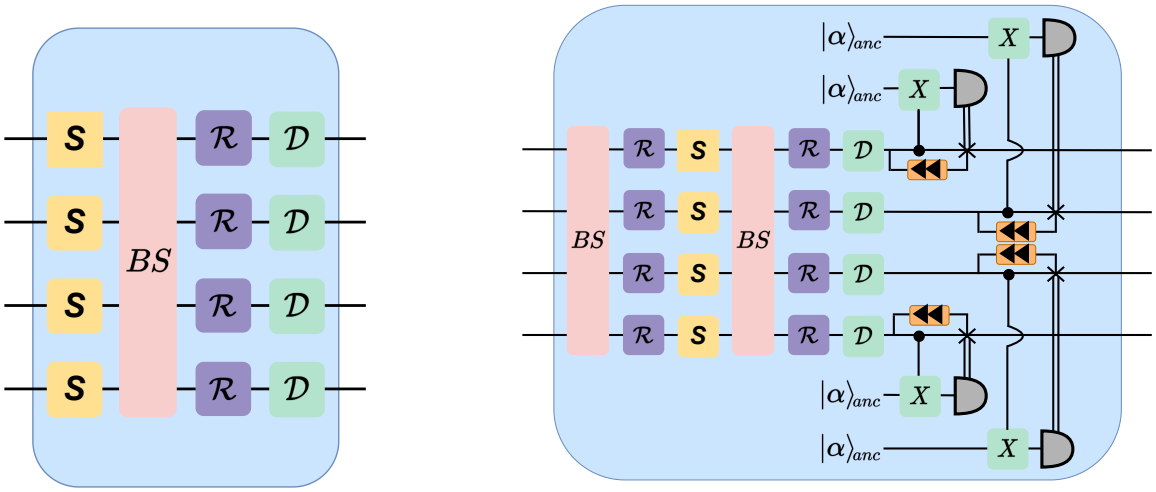


Figure 3.21: CV hybrid NN architecture used for MNIST image classification.



(a) **(b)**

Figure 3.22: MNIST hybrid NN. (a) Encoding layer of MNIST hybrid neural network that embeds the output of the classical layers into the quantum circuit. (b) Detailed architecture of quantum layer.

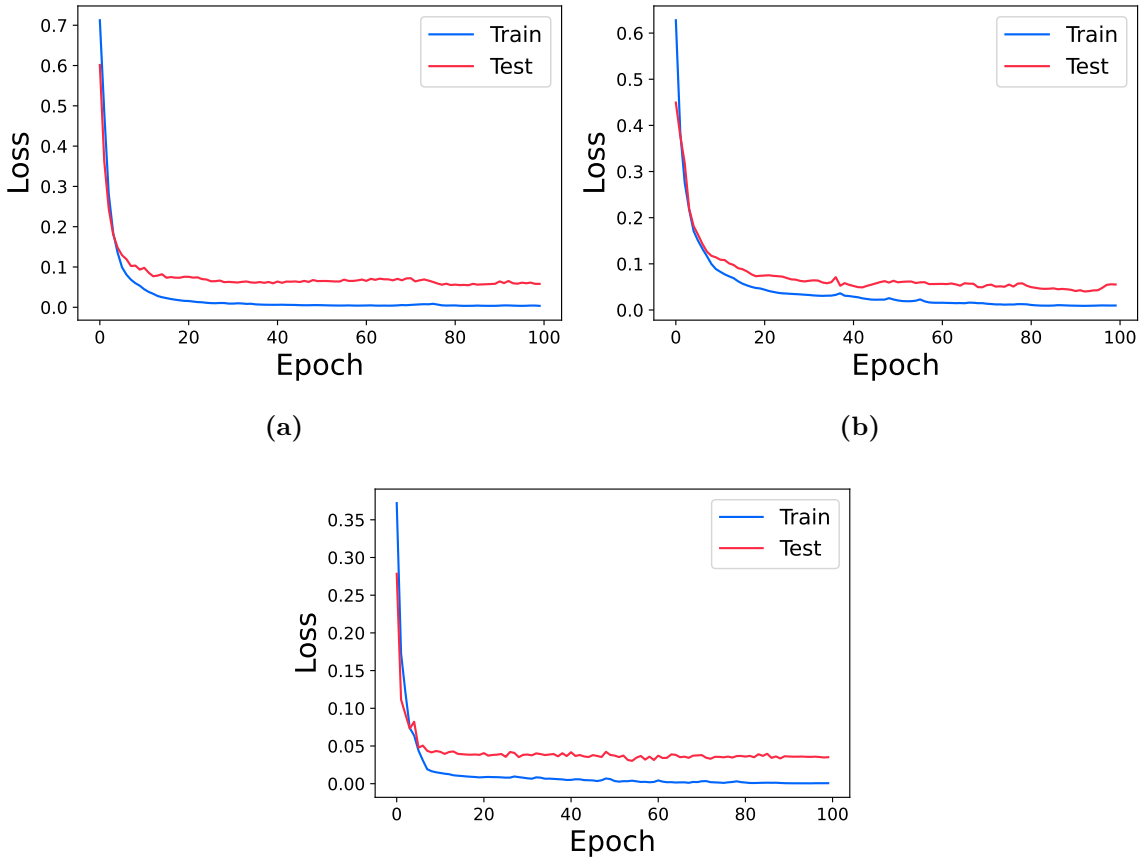
3.2.4 Multi-label classification

Extending the results for binary classification, we developed a CV quantum-classical hybrid NN to classify MNIST handwritten digits [111] into their respective classes. The MNIST dataset comprises 60,000 training images and 10,000 testing images; each normalized to 28×28 pixels in size and grayscale in color. Each data point is labeled with the corresponding digit (0-9). Our current hardware limitations allowed our model to train and classify images up to 4 classes (0-3).

The network architecture we used is illustrated in Fig. 3.21 with details of the encoding layer in Fig. 3.22(a) and the quantum layers in Fig. 3.22(b). The network consists of fully connected feed-forward classical neural network layers that take the input data, feeding into an encoding quantum layer, followed by regular quantum layers that can be repeated as needed. Each quantum layer comprises several primary qumodes, each representing a class of the MNIST dataset, as well as ancillary qumodes that implement non-linearity.

During training, we calculated the probability or accuracy of classification by considering the overlap of the final state of the circuit, which comprises only the primary qumodes, with the one-hot encoded ground truth value of the training data sample. We counted all probabilities corresponding to non-zero values of the Fock number of the correct class towards the accuracy, indicating a "click" or "non-click" on a detector. The loss was then calculated using Eq. (3.14). For training, we used the Adam optimizer with a batch size of 16 and a decaying learning rate beginning at 0.001 and decreasing by a factor of 0.9 every 5,000 steps. During testing runs, we interpreted the probabilities corresponding to each primary qumode as logits. The predicted class was determined by selecting the logit with the highest value.

To assess the versatility of our model, we performed multiple experiments, exploring different configurations of classical and quantum layers in the hybrid model. We considered a range of ratios, starting from 1 classical layer and 5 quantum layers (1:5) up to 5 classical layers and 1 quantum layer (5:1).



(a)

(b)

(c)

Figure 3.23: Training and testing loss plots for select ratios of classical and quantum layers, networks (a) 1:5, (b) 3:3, and (c) 5:1, with the first (second) number indicating the number of classical (quantum) layers.

In total, we examined 8 different layer ratios, including ratios such as 2:2, 2:6, and 2:8. The classical layers comprised 128 nodes each, except for the final layer (or only layer in the case of a single classical layer model), which had the same number of nodes as required by the quantum encoding layer. The latter is determined by the formula $7p - 2$, where p denotes the number of classes or primary qumodes.

The loss function values over the epochs for select ratios of classical and quantum layers are displayed in Fig. 3.23. All of our models converged within 100 epochs and demonstrated successful training and testing, achieving testing accuracies of $96.47\% \pm 0.86\%$. Interestingly, we did not observe a significant impact on changing the number of classical and quantum layers. Nevertheless, our hybrid networks demonstrated high levels of accuracy for the 4-class MNIST classification problem.

3.3 Conclusion

We proposed CV QNN models that can be realized experimentally. We introduced a quantum circuit element that involves an ancillary qumode with a controlled-X, i.e., CX gate, on the primary qumode. For a good success rate, it relies on repeat-until-success measurements of the photon number of ancillary qumodes [97]. It offers a simple and feasible solution to introducing non-linearity using current photonic quantum hardware, considering the high complexity of implementing non-Gaussian operators experimentally. Our study demonstrated that our experimentally realizable circuit element could efficiently solve many machine-learning and quantum computation problems.

We created a CV quantum circuit that can prepare a single photon state with 99.9% fidelity, a cat state with 99.8% fidelity, and a GKP state of fidelity 93.9%. In [112], they performed the state preparation using CV QNNs and used the Kerr gate for non-linearity, making the entire process experimentally hard to achieve. However, our CV QNN can prepare the states in a way that can be realized experimentally. Although we could not achieve high fidelity for the GKP state because of the

high computational requirements, we would need a higher cutoff dimension with more layers and optimization steps, leading to numerous training parameters for preparing such a complex state. However, it can be done, in principle, using high computational resources. GKP states could be a key factor in creating a scalable photonic fault-tolerant quantum computer [113]. We delved into the impact of noise, particularly loss, on quantum state preparation and discerned that different states respond variably to noise, with an overall fidelity reduction in all cases. Single-photon and cat state preparations performed well under realistic loss, whereas GKP states exhibited low performance under the influence of loss. Looking ahead, it is essential to investigate strategies to improve the fidelity of GKP states in the presence of loss. This could involve exploring novel quantum error correction techniques to mitigate the impact of loss and extending the study to other types of noise and losses, thus broadening our understanding of quantum state preparation in noisy environments. For example, it is worth exploring using the continuous-variable quantum erasure-correcting code introduced in Ref. [114], based on experimentally realizable linear optical elements.

We developed models (from CV QNNs) capable of fitting functions using noisy data sets. We thoroughly analyzed the layers and noise effects on the curve fitting. As we increased the noise 5 times, the accuracy only went down by 24%. Our model could still learn and accurately reproduce the curve's shape. This insight is valuable as our model can perform well even with noisy data sets. We encountered some challenges while attempting to approximate complex functions using our hybrid model for curve fitting. We discovered that these functions required a higher cutoff dimension and more layers to achieve a lower cost. This, in turn, necessitated additional computational resources for fitting more complex functions. Despite the issues, we successfully performed curve fitting on various functions like sine decay. It would be interesting to further study how quickly the QNN learns compared to its classical counterpart based on the complexity of the function and available data points for training. For the sine function we studied, the quantum and classical circuits reach the

same accuracy after training, although there has been some work showing that with few training data, the quantum circuits could learn faster [115]. They do not provide an advantage over classical machine learning, but it has been argued that quantum circuits can outperform their classical counterparts under certain assumptions. The importance of data in quantum machine learning has also been studied; see, e.g., Ref. [116], where it was discussed how one could achieve quantum advantage based on chosen data and other machine learning techniques.

The binary classification problem achieved high classification accuracy. The AUC score of 0.9 and accuracy of more than 95% on a highly unbalanced data set. Similar work has been done to study fraud detection in credit card transactions using Quantum Support Vector Machine [117]. They used a quantum-classical method to select the best features for the training process. They also focused on the importance of using quantum machine learning in selecting these features to improve the model's accuracy, which compliments the classical approach in finance.

Image recognition done on the MNIST classification model can classify hand-written digits with up to 97% accuracy. We did not observe a significant change in the results as we varied the classical and quantum layers mix. Hence, future research could investigate the efficacy of quantum layers in hybrid neural networks and quantum neural networks in general. Quantum computation has the potential to provide exponential speedup over classical computation for certain problems; quantum layers can exploit this speedup to perform computations more efficiently than classical layers for tasks that can benefit from quantum algorithms. Quantum layers can also leverage the properties of superposition and entanglement to process and represent information in impossible ways with classical layers. Quantum layers can also be utilized as non-linear feature mappings that are challenging for classical layers. Quantum Feature Maps can transform input data into higher dimensions for richer representation to later be utilized by classical neural networks, leading to potentially more accurate classification. Moreover, even though CV QGANs (quantum generative adversarial networks) have been previously studied (see, e.g.,

[94]), it would be interesting to explore the performance of CV QGANs utilizing our proposed experimentally feasible setup. The proposed prototype of QGANs in [94] requires using non-Gaussian gates within its quantum layers for both the quantum generator and the quantum discriminator. Since our prescribed quantum layer only requires Gaussian gates, we can simulate the effectiveness of an experimentally viable QGAN.

In conclusion, with our proposed CV quantum algorithm, we have obtained promising results in solving many machine-learning problems. The nonlinear quantum circuit element we introduced, which was based on an earlier proposal for universal CV quantum computing [97], offers an experimentally feasible solution to introducing non-linearity using current photonic quantum hardware, avoiding the high complexity in experimentally realizing non-Gaussian operators.

Chapter 4

Continuous-variable Quantum Boltzmann Machine

Boltzmann machines (BMs) are robust machine learning models which provide modeling of probability distributions. BMs consist of a probabilistic network that is represented as an undirected graph with hidden and visible nodes. In this machine learning model, the probability distribution of the data is approximated based on a finite set of samples. The information is encoded into the bias coefficients and weights of the neural network. After a successful training process, the learned distribution resembles the actual distribution of the data that can make correct predictions about unseen instances. However, the generalization of the model suffers from a growing number of parameters, and the training of a classical BM can become impractical. For this reason, it is worth exploring engineering quantum Hamiltonian Ansätze to construct quantum BMs [118]. In QBMs, the goal is to approximate the thermal state of the Hamiltonian Ansatz to the target density matrix of the distribution through training of the weights of the Hamiltonian Ansatz. One of the advantages of QBMs over their classical counterparts is that QBMs have higher expressive power, i.e., can fit a wider range of complex data patterns, since the Hamiltonian can include more non-commuting terms [119].

Preparing a thermal state and computing its properties is a challenging task. Hence, various heuristic methods that are compatible with noisy intermediate scale quantum (NISQ) hardware have been proposed [120]. These methods include quantum imaginary time evolution (QITE) [52], variational QITE (varQITE) [121], and the variational quantum eigensolver ($\beta-$VQE) [122]. The thermal states prepared utilizing these tools have also been used in realizing QBMs. For instance, in Ref. [123], a variational QBM was proposed and utilized for a discriminative learning task, specifically fraud detection, as well as a generative learning task, specifically learning a Bell state probability distribution. The method was implemented on IBM Quantum hardware. This work was followed by several studies including [120, 124]. In Ref. [124], similar to [123], they utilized variational QITE evolution to study QBMs and tested their model using the bars and stripes (BAS) dataset. They focused on the classical cases when the Hamiltonian for the QBM has only diagonal terms in the computational basis, and reduced the required number of qubits by half by preparing an initial pure state which was a superposition of all states. On the other hand, in Ref. [120], $\beta-$VQE was used as a subroutine for preparing the Gibbs state needed for QBM. In their QBM model, free energy, or equivalently quantum relative entropy, was used as the loss function. They tested their model both on quantum and classical data. It was proposed that, once the classical dataset is embedded into the quantum circuit, a lower rank $\beta-$VQE can be utilized for training of QBM. A similar feature was also observed for low temperature quantum data. However, for higher temperature quantum data, a higher rank $\beta-$VQE is required. They also tested their QBM model on a real-world dataset, namely salamander retina, and demonstrated that $\beta-$VQE QBM achieves lower Kuller-Leibler (KL) divergence value compared to a classical Boltzmann machine. In Ref. [119], the sample complexity of QBM machine learning was studied. It was shown that QBMs do not possess barren plateaus (optimization landscape becoming flat due to various reasons [125]) and that pre-training on a subset of the QBM parameters can only lower the sample complexity bounds. In addition to gate based quantum computing, QBMs have been demonstrated using quantum

annealers. In Ref. [126], a restricted QBM was utilized for balancing the imbalanced credit card transactions data through synthetic data generation that can be utilized for fraud detection and other cyber security applications.

Studies of quantum generative models have mainly focused on the generation of discrete probability distribution instances. One example is Ref. [127], where a variational quantum generative adversarial network (GAN) model was utilized for the generation of the probability distribution of a real dataset obtained using a variational quantum generator with fixed parameters. In Ref. [128], this method was shown to be robust to quantum hardware noise by implementing it on Rigetti quantum hardware. However, most real-world applications, such as image and sound generation, computational fluid dynamics, study continuous probability distributions. Use of discrete-variable (DV) quantum computing (based on qubits) is only partially suited for continuous-valued data. One can avoid the burden of encoding continuous valued classical data into DV by using CV quantum machine learning tools. For instance, in Ref. [129], they studied time series forecasting for continuous datasets, such as energy consumption and stock price data. It is more natural to extend generative models to continuous-variable quantum computing (CVQC) models for the study of continuous data. In CVQC, quantum information is encoded into infinite-dimensional qumodes, such as the electromagnetic field of photons in photonic quantum hardware. In Ref. [130], a CV quantum Born machine was developed which offered an efficient way of modeling continuous probability distributions, such as a Gaussian distribution. By choosing a Gaussian kernel, it was demonstrated that a CV quantum Born machine naturally provides a much better fit to a Gaussian distribution with a single qumode compared to a 10-qubit quantum circuit Born machine model. It was also shown that a 3-qumode Gaussian and cubic kernel CV quantum Born machine learns a quantum target distribution much faster than by using a classical Gaussian kernel method.

In this work, we discuss a proposal for a continuous-variable quantum Boltzmann machine (CVQBM). Our CV model is a natural fit for studying continuous

probability distributions. We utilize our method for generative modeling in real-world applications, such as SAR image probability distribution generation, as well as probability distribution of data obtained from prepared quantum states. To the best of our knowledge, this is the first demonstration of generation of various real-world applications' continuous probability distributions using a CV quantum generative model. Even though experiments were conducted on a simulator, we discuss experimental feasibility of our simulations and demonstrate that problems studied in this work can be implemented on experimentally feasible photonic quantum computing architectures. In our model, the CV quantum circuit we employ demonstrates a CV quantum generative model, i.e., CVQBM, and also presents a variational CV QITE algorithm utilized for finite temperature state preparation. Our variational CV QITE algorithm can be leveraged in other use cases, such as ground state preparation in quantum field theories [63].

Our discussion is organized as follows. In Section 4.1, we provide the details of the proposed CVQBM model. We design the CVQBM quantum circuit and discuss the procedure for setting it up as a continuous probability distribution generator. In Section 4.2, we present various case studies using our CVQBM, focusing on classical data from the SAR images. We generate data for forest and sea monitoring. In Section 4.3, we discuss quantum data, including both Gaussian and non-Gaussian cases. An essential aspect of our proposed CVQBM is that it is experimentally feasible on CV photonic quantum computers, which is discussed in Section 4.4. We introduce noise to analyze the robustness of our CVQBM, and discuss the success rate and gate parameters chosen in our case studies. Finally, in Section 4.5, we present our conclusions and indicate future directions.

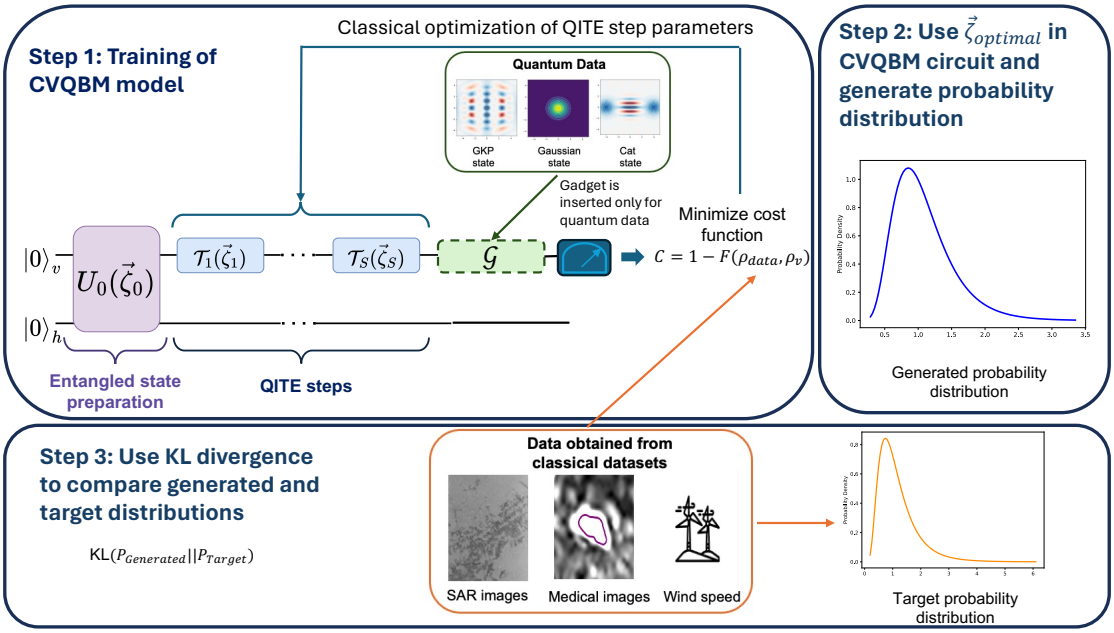
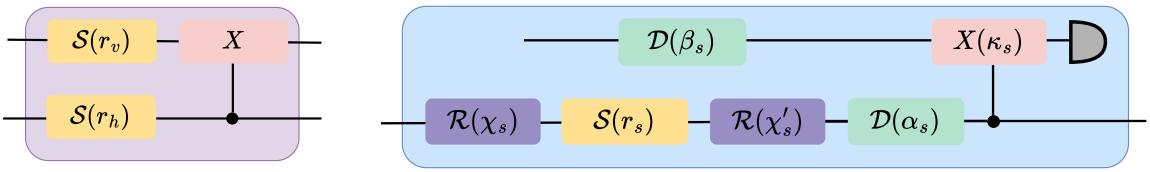


Figure 4.1: CVQBM framework: Step 1 is the training of the CVQBM model by training the parameters of the quantum circuit, which prepares an entangled state between the visible and hidden modes of the CVQBM, and then performs imaginary time evolution, minimizing the cost function (4.15), calculated using the density matrix of the data and the generated density matrix. In Step 2, the optimal quantum circuit parameters are used in the same quantum circuit for CVQBM to generate a probability distribution. Step 3 involves verifying how close the generated probability distribution is to the target distribution using metric KL divergence. For embedding quantum data, we also propose a gadget $\mathcal{G}$ (Figure 4.6) that is implemented on the visible node following the QITE steps.



(a) **(b)**

Figure 4.2: Designing the CVQBM circuit, detailed structure of **(a)** U_0 where two modes are entangled by using squeezed gates and then applying the CX gate and **(b)** $\mathcal{T}_s(\vec{\zeta}_s)$ where the gates have trainable parameters collectively represented by ζ. An ancillary mode is added, which introduces non-linearity into the network by entanglement (CX gate) and photon number measurement.

4.1 The Method

In this Section, we outline the design of a quantum circuit for the CVQBM and set up the machine as a continuous distribution generator. We show that the output can be thought of as a thermal state of an effective Hamiltonian defined by the quantum circuit. This is similar to the approximate Gibbs state that was prepared using a variational quantum imaginary time evolution (varQITE) algorithm with discrete variables [123].

The CVQBM circuit is designed to prepare a thermal state, $\frac{1}{\mathcal{Z}}e^{-\beta H(\vec{\zeta})}$, where $\mathcal{Z} = \mathrm{Tr}e^{-\beta H(\vec{\zeta})}$, for a Hamiltonian that depends on parameters collectively denoted as the vector $\vec{\zeta}$. For simplicity, we set the inverse temperature $\beta = 1$. This state represents quantum imaginary time evolution (QITE) which is not unitary. The parameters $\vec{\zeta}$ control the quantum gates of the CVQBM circuit, which are unitary, and are trained according to the problem at hand. We achieve non-unitary evolution by performing measurements on ancilla qumodes. Here, we concentrate on a Hamiltonian that depends on a single (visible) qumode of quadratures (q_v, p_v). Extension of our method to include multiple qumodes is straightforward.

The CVQBM circuit is depicted in Figure 4.1. In addition to the visible qumode, it uses an ancilla qumode of quadratures (q_h, p_h) representing a hidden mode. Both modes are initially in the vacuum state. They are entangled with the two-mode unitary U_0. No further action is taken on the hidden mode. The visible mode is taken through a series of non-unitary quantum gates $\mathcal{T}_s(\vec{\zeta}_s)$, where $s = 1, 2, \ldots$, that depend on parameters $\vec{\zeta}_s$ to be trained. Next, we proceed to describe the various quantum gates used in detail.

We start by engineering the entangled state:

$$|\psi_0\rangle_{vh} = \frac{1}{\sqrt{\mathcal{N}}} \sum_{n=0}^{\infty} e^{-\delta n} |n\rangle_v |n\rangle_h \ , \quad \mathcal{N} = \frac{1}{1 - e^{-2\delta}} \tag{4.1}$$

where $\delta > 0$ is a fixed constant, and $|n\rangle$ is the eigenstate of the number operator N with n photons. How this is implemented is shown in Figure 4.2a. We use two squeezed modes, one squeezed in the q quadrature with squeezing parameter $r_v = -\frac{1}{2}\log\delta$, and the other one squeezed in the p quadrature with $r_h = \frac{1}{2}\log\delta$. We entangle them with the CX gate $e^{iq_v p_h}$ to bring them in the state

$$\frac{1}{\sqrt{\pi}}\int dq_v dq_h e^{-\frac{\delta}{2}q_v^2} e^{-\frac{1}{2\delta}q_h^2} |q_v\rangle_v |q_v + q_h\rangle_h \ . \tag{4.2}$$

For small δ, this is approximately $|\psi_0\rangle$ (Eq. (4.1)). This is easily seen by switching to the number operator basis, in which this state can be written as

$$\frac{1}{\sqrt{\pi}}\sum_{n_v,n_h}\int dq_v dq_h e^{-\frac{\delta}{2}q_v^2} e^{-\frac{1}{2\delta}q_h^2} q_v |n_v n_h| q_v + q_h |n_v\rangle_v |n_h\rangle_h \ , \tag{4.3}$$

expressing the inner products (harmonic oscillator wavefunctions in the position representation) as

$$\Psi_n(q) = q|n = \frac{1}{\sqrt{2^n n!\sqrt{\pi}}} H_n(q) e^{-\frac{1}{2}q^2} \tag{4.4}$$

where H_n is the Hermite polynomial:

$$H_n(q) = (-)^n e^{q^2}\frac{d^n}{dq^n}e^{-q^2} \tag{4.5}$$

and performing the integrals over the q quadratures.

Next, we evolve the state $|\psi_0\rangle$ by acting with a string of non-unitary gates $\mathcal{T}_s(\vec{\zeta}_s)$ ($s = 1, 2, \ldots, S$) acting on the visible mode only. The total number of gates S depends on the problem at hand. The quantum circuit implementing a gate $\mathcal{T}_s(\vec{\zeta}_s)$ is depicted in Figure 4.2b. It consists of Gaussian gates only that act on the visible qumode and an ancilla qumode of quadratures (q_A, p_A). Non-unitarity is introduced by a measurement of the ancilla qumode, projecting it onto the state $|n_A = 1\rangle_A$. Let $|\psi_{s-1}\rangle_{vh}$ be the input at the sth step and denote by $\mathcal{U}(\vec{\zeta}_s)$ the two-mode unitary representing the collective action of the Gaussian gates on the visible and ancilla

qumodes,

$$\mathcal{U}(\vec{\zeta}_s) = \mathcal{R}(\chi_s)\mathcal{S}(r_s)\mathcal{R}(\chi'_s)\mathcal{D}(\alpha_s)X(\kappa_s) \tag{4.6}$$

where $\mathcal{R}(\chi_s)$ is phase space rotation operator, $\mathcal{S}(r_s)$ represents squeeze gate affecting position and momentum, $\mathcal{D}(\alpha_s)$, is displacement operator and $X(\kappa_s)$ is controlled-X gate, a controlled displacement in position. These gates are defined by the following expressions:

$$\mathcal{R}(\chi) = e^{i\chi\hat{a}^\dagger\hat{a}} \ , \quad \mathcal{S}(r) = e^{\frac{r}{2}\left(\hat{a}^2 - \hat{a}^{\dagger 2}\right)}, \quad \mathcal{D}(\alpha) = e^{\alpha(\hat{a}^\dagger - \hat{a})}, \quad X(\kappa) = e^{-i\kappa\hat{x}\otimes\hat{p}} \ , \tag{4.7}$$

where $\chi, r, \alpha, \kappa \in \mathbb{R}$, $\hat{x} = \frac{1}{\sqrt{2}}(\hat{a} + \hat{a}^\dagger)$, $\hat{p} = -i\frac{1}{\sqrt{2}}(\hat{a} - \hat{a}^\dagger)$, and $\hat{a}$, $\hat{a}^\dagger$ are annihilation and creation operators, respectively. Then the output of the sth step is

$$|\psi_s\rangle_{vh} \propto \mathcal{T}_s(\vec{\zeta}_s)\,|\psi_{s-1}\rangle_{vh} \ , \quad \mathcal{T}_s(\vec{\zeta}_s) = {}_A1|\mathcal{U}(\vec{\zeta}_s)|0_A \ . \tag{4.8}$$

The success probability of obtaining the desired output is $P_s = \|\mathcal{T}_s(\vec{\zeta}_s)\,|\psi_{s-1}\rangle_{vh}\,\|^2$. The circuit must be designed so as to maximize the probability of success.

It should be pointed out that the introduction of an ancilla qumode is similar to the training of CV Quantum Neural Networks (QNN) that are experimentally realizable, as we discussed in our previous work [26]. Here we aim at a similar construction that would yield experimentally realizable CVQBM.

After S steps, we obtain the output of the CVQBM,

$$|\text{out}\rangle_{vh} = |\psi_S\rangle \propto \mathcal{T}(\vec{\zeta})\,|\psi_0\rangle_{vh} \ , \quad \mathcal{T}(\vec{\zeta}) \equiv \mathcal{T}_S(\vec{\zeta}_S)\cdots\mathcal{T}_1(\vec{\zeta}_1) \tag{4.9}$$

where $\vec{\zeta} \equiv \{\vec{\zeta}_1, \ldots, \vec{\zeta}_S\}$, and after tracing over the hidden qumode, we obtain the state of the visible mode,

$$\rho_v = \text{Tr}_h\,|\text{out}\rangle\,\langle\text{out}| \propto \text{Tr}_h\mathcal{T}(\vec{\zeta})\,|\psi_0\rangle\,\langle\psi_0|\,\mathcal{T}^\dagger(\vec{\zeta}) \propto \mathcal{T}(\vec{\zeta})e^{-2\delta N}\mathcal{T}^\dagger(\vec{\zeta}) \tag{4.10}$$

where we used Eq. (4.4) and the fact that $\mathcal{T}(\vec{\zeta})$ only acts on the visible qumode. Evidently, this is a Hermitian operator. Hence we can define an effective Hamiltonian $H_{\text{eff}}(\vec{\zeta})$ (a Hermitian operator) by

$$\rho_v = e^{-H_{\text{eff}}(\vec{\zeta})} \tag{4.11}$$

which depends on the set of parameters $\vec{\zeta}$, showing that the output state of the visible qumode in the CVQBM can be viewed as a thermal state. The action of each non-unitary operator $\mathcal{T}_s(\vec{\zeta}_s)$ can be thought of as a QITE step [123, 63]. Thus, our quantum circuit *defines* the Hamiltonian and its parameters to be trained.

Having designed the CVQBM circuit, we may proceed to use the machine to solve a variety of problems, such as generative learning tasks [123, 131, 132], discriminative tasks [123, 118], reinforcement learning [133], quantum state tomography [131], and anomaly detection [134, 135]. Here, we will concentrate on using the CVQBM as a continuous distribution generator. Our goal is to make the machine learn a continuous distribution and then use the trained machine to generate samples that belong to that distribution. This type of machine can be utilized as a synthetic data generation tool. There have been various attempts to use QBM as a generation tool, such as solving a toy problem of the phase of a state at 10 spatial points [136], generating probability distributions for a state [123, 131] and Bernoulli distribution [118]. Generating a continuous distribution in a CV setting provides a natural advantage.

We simulated the CVQBM using the Strawberry Fields software platform [105]. The quantum machine learning toolbox application is built on top of it, with TensorFlow [106] features. The detailed architecture for our CVQBM is shown in Fig. 4.2. We started with two modes (visible and hidden), which were then entangled to prepare the two-mode initial state using the quantum gate U_0 depicted in Fig. 4.2a. The visible mode was then acted upon with a series of non-unitary gates $\mathcal{T}_s(\vec{\zeta}_s)$ $(s = 1, \ldots, S)$ whose detailed architecture is shown in Fig. 4.2b. The parameters of these gates are training parameters. At each step, an ancilla coherent mode was

introduced, which was entangled with the visible mode by using Gaussian gates and then measured, thus projecting it onto the $n = 1$ state. The success probability was calculated, ensuring that the setup corresponded to a reasonable rate for experimental implementation. Finally, we performed homodyne measurement on the visible mode.

In the training stage of our CVQBM, our objective was to minimize the distance between the target and output distributions. However, comparing two distributions at every training epoch is computationally intensive. We found it advantageous to work with density matrices. For a given set of data of probability distribution $P_{\text{data}}(q)$, where $\int dq P_{\text{data}}(q) = 1$, we introduced the state

$$|\Psi_{\text{data}}\rangle \approx \sum_{n=0}^{n_{\max}} a_n |n\rangle \quad , \quad a_n = \int dq \Psi_n(q) \sqrt{P_{\text{data}}(q)} \tag{4.12}$$

where $\Psi_n(q)$ is the nth state of a harmonic oscillator in the position representation given by Eq. (4.4). This can be written in a density matrix form as

$$\rho_{\text{data}} = |\Psi_{\text{data}}\rangle \langle \Psi_{\text{data}}| \quad , \quad \rho_{\text{data},mn} = a_m a_n \tag{4.13}$$

which can be compared with the output state ρ_v given in Eq. (4.10). The fidelity is defined as:

$$F(\rho_{\text{data}}, \rho_v) = \left(\text{Tr} \sqrt{\sqrt{\rho_{\text{data}}} \rho_v \sqrt{\rho_{\text{data}}}} \right)^2 \tag{4.14}$$

Our goal was to maximize this fidelity, and hence minimize the cost:

$$C = 1 - F \tag{4.15}$$

To this end, we used the Adam Optimizer.

Once the CVQBM was trained, the next step was to generate more samples by loading the trained weights into the circuits. Since, ultimately, our goal is to generate the learned distribution, we had to extract the output from the circuit that

corresponded to the output probability distribution,

$$P_v(q) = q|\rho_v|q \ . \tag{4.16}$$

Switching to eigenstates of the photon number, the output probability distribution can be written as

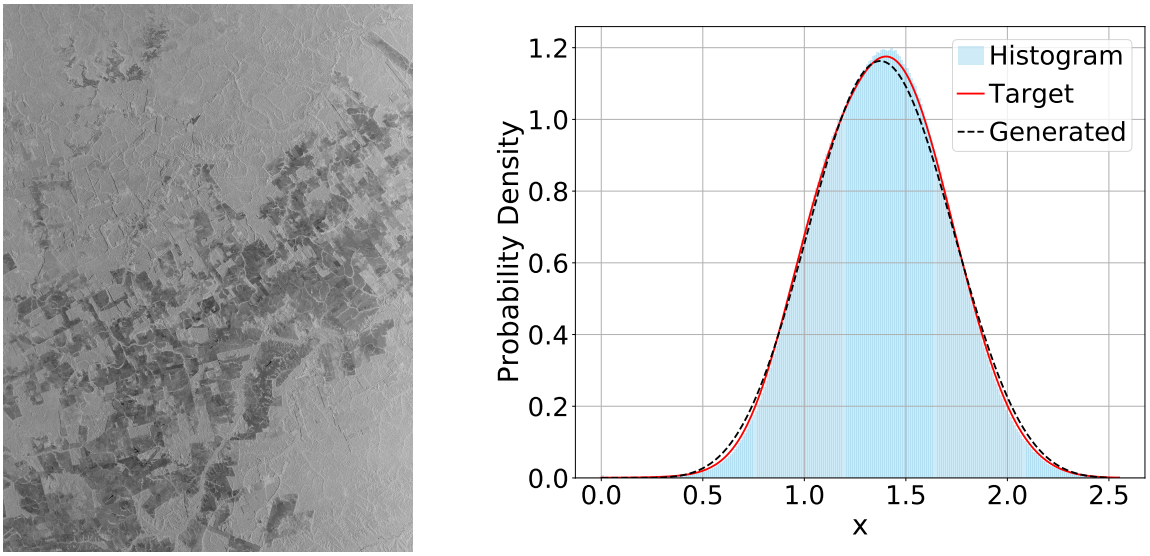
$$P(q) = \sum_{m,n} \rho_{v,mn} \Psi_m(q) \Psi_n(q) \ , \quad \rho_{v,mn} = \langle m| \, \rho_v \, |n\rangle \tag{4.17}$$

where $\Psi_n(q)$ is given by Eq. (4.4). Although this is an infinite set of basis states, in practice we need to introduce a cutoff $n_{\max}$ in the dimension of the Hilbert space. Using this result, we can generate various samples from the trained CVQBM.

A metric to compare the target with a generated distribution is provided by the Kuller-Leibler (KL) divergence,

$$\mathrm{KL}(P_{\mathrm{gen}} \| P_{\mathrm{target}}) = \sum_q P_{\mathrm{gen}}(q) \log \frac{P_{\mathrm{gen}}(q)}{P_{\mathrm{target}}(q)} \tag{4.18}$$

which indicates how much the generated probability distribution, P_{gen}, diverges from the target probability distribution, P_{target}, and takes values between 0 and ∞, with the value of 0 when $P_{\mathrm{gen}}(q) = P_{\mathrm{target}}(q)$. We compared the two distributions and obtained how much the generated distribution diverged from the target distribution by calculating the KL divergence.



(a) **(b)**

Figure 4.3: Studying SAR image for forest monitoring in Brazil, **(a)** The original SAR image of the forest in Acre Brazil taken on 22 July 21. **(b)** The final plot corresponds to the forest monitoring. The blue histogram is created from the SAR image. The target distribution (red line) is fitted Gaussian on the histogram. Finally, the generated distribution (black dashed line) comes from the trained QBM, which has the KL divergence between the target and generated distribution is 0.006.

4.2 Classical Data Probability Distribution Generation

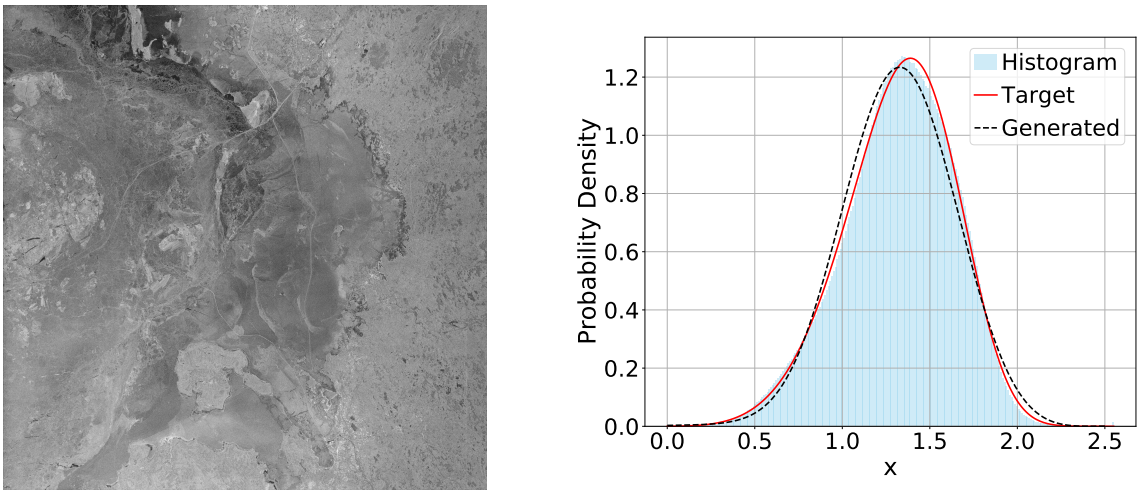
Our CVQBM can be used for generating different kinds of distributions involving classical or quantum data. In this Section, we report on probability distributions from synthetic-aperture radar (SAR) images which are classical data. We used the SAR dataset available on the ICEYE website [137]. We selected a forest monitoring strip from Acre, Brazil. The dataset contains the time-series stack of strip SAR images acquired within 12 days in Single Look Complex (SLC) in HDF5 format and Ground Range Detected (GRD) in GeoTiff format. For simplicity, we worked with the PNG image in the dataset, shown in Fig. 4.3a. The image belongs to the data collected on 22 July 2021.

We analyzed the data by plotting the image intensity probability distribution as a function of the pixel intensity. To this end, we converted the image into a Python Numpy array, and then created the histogram for the intensity values. The histogram was divided into 256 bins (for each possible intensity value in an 8-bit grayscale image) and counted how many pixels fell into each bin. The range was set from 0 to 255, including all possible intensity values. The histogram was then normalized to yield a probability distribution. The bar graph was plotted to visualize the probability distribution of the pixel intensities. Next, we applied re-scaling to both plot axes to make the problem suitable for a CV quantum computer. By doing so, the range of axes became approachable, and normalization was maintained. Because of the limitations of the simulation, it was hard to use the discrete data as this would require a higher cutoff value. Hence, the next step was to make the distribution continuous, and for that, we used the *gaussian kde* from the SciPy library [138] of Python. Once we obtained the distribution, we used Eq. (4.13) to deduce the density matrix. Using the density matrix for cost calculation provided a computational advantage over using the distribution directly.

As discussed in Section 4.1, the cost function depends on the fidelity of the circuit output density matrix compared with the target density matrix. For an optimal entangled initial state (Eq. (4.1)), we set $\delta = 1.5$. We used the Adam optimizer with the learning schedule of exponential decay to minimize the cost. The initial learning rate was 0.05; the decay steps were 100, with a decay rate of 0.96. The number of steps in this process was $S = 2$. After training for 100 epochs with a batch size of 32 and a cutoff of $n_{\max} = 12$, the best fidelity we obtained was 99.8%, corresponding to the cost value of 0.002. The low cost made us confident that the model had been trained well.

After training, the model can be used to generate more data corresponding to the target distribution. For this process, we loaded the trained weights and ran the circuit several times to get samples for the density matrix. We generated 256 samples corresponding to the original shape of the data. Then using Eq. (4.17), we generated the distribution. The generated and target distributions, as well as the original discrete probability distribution, are shown in Fig. 4.3b. Notice that the generated distribution lies on top of the target distribution, showing that we achieved high fidelity while training. We also calculated the KL divergence between the generated and target distributions, which came out to be 0.006, showing that the generated distribution is close to the target distribution.

Generating the distribution for the forest was not hard, as we could fit the probability distribution with a Gaussian. A more challenging example we studied was provided by an image that belonged to the Gulf of Bothnia between the west coast of Finland and the east coast of Sweden [137].



(a) **(b)**

Figure 4.4: Studying the SAR image for sea ice monitoring, **(a)** the original SAR image of Gulf of Bothnia, taken on 25 Feb 2022, **(b)** The final plot corresponds to the sea monitoring. The blue histogram is created from the SAR image. The target distribution (red line) is fitted Gaussian on the histogram. Finally, the generated distribution (black dashed line) comes from the trained QBM, which has the KL divergence between the target and generated distribution is 0.013

The dataset had a pair of scan mode images, which covered an area of 10,000 km^2 in the Gulf of Bothnia, shown in Fig. 4.4a. Again, we randomly picked one of the images and plotted the probability distribution vs. pixel intensity. This probability distribution was best fitted with a Weibull probability distribution:

$$W(x;\lambda,k) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \tag{4.19}$$

where $x \geq 0$, $k > 0$ is the shape parameter, and $\lambda > 0$ is the scale parameter of the distribution. For this case, the fitted Weibull distribution had $\lambda = 161.2$ and $k = 5.4$.

We adopted the same setup as in the forest case. First, we found the density matrix corresponding to the probability distribution. Then, the cost function was defined in terms of the fidelity between the target density matrix and the circuit output density matrix. Again, we used the Adam optimizer with the same learning schedule as the previous case. The number of steps used in this process was $S = 2$, with a cutoff dimension of $n_{\max} = 15$. After training for 150 epochs with a batch size of 32, the best fidelity we obtained was 98.9%. The final plot is shown in Fig. 4.4b. The generated distribution in the dashed black line matches the target Weibull distribution (shown in the red solid line). We obtained a KL divergence of 0.013.

Our model works with various probability distributions requiring an increasing amount of computational resources as their complexity increases. We demonstrated this with several distributions commonly associated with real-world data, including Rayleigh and Gamma distributions which are often used for SAR and medical images, like ultrasound and PET scans, as well as exponentiated transmuted-inverted beta (ET-IB) distributions that were recently shown [139] to outperform (in terms of goodness-of-fit measures) its competitors when it comes to fitting SAR images. We studied the performance of our model on these distributions as follows:

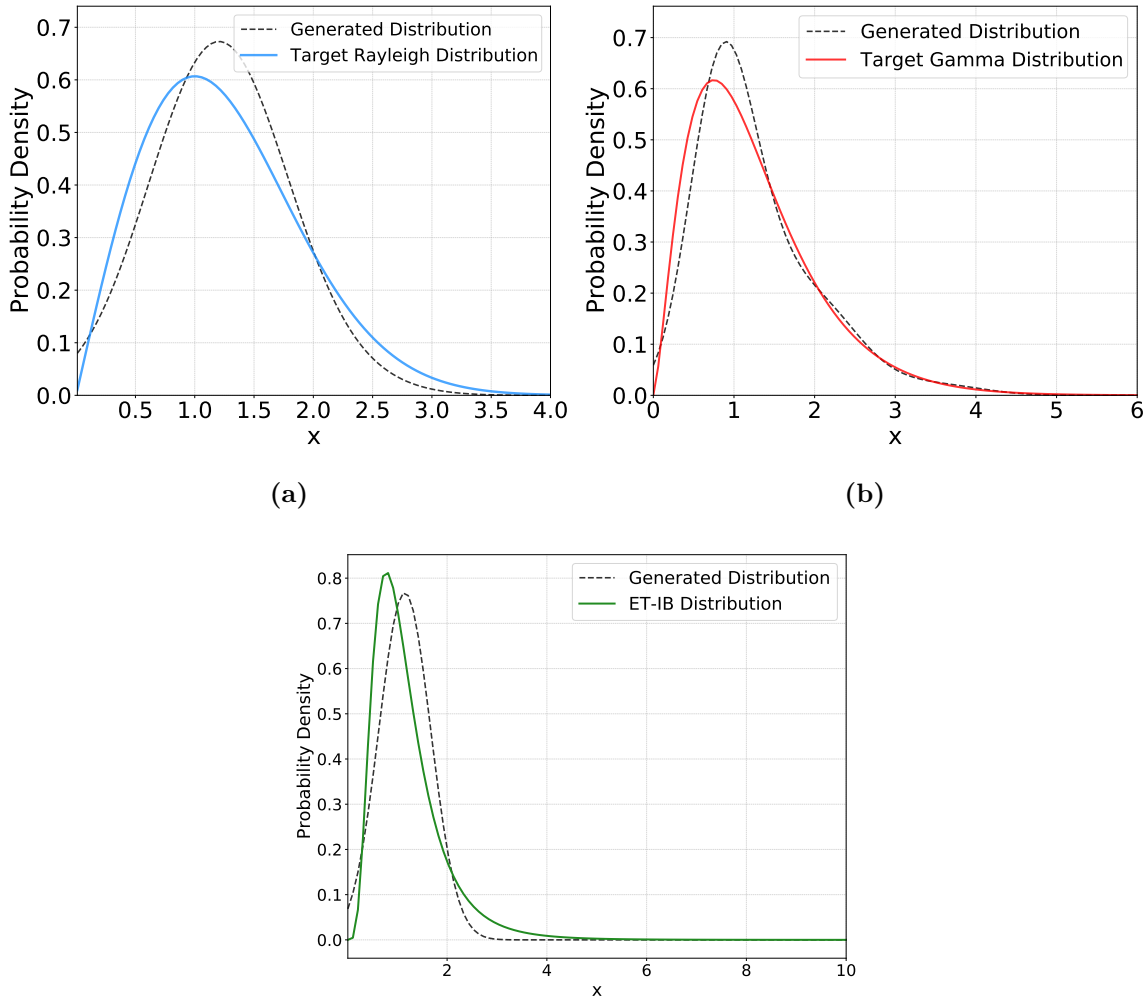


Figure 4.5: Generating continuous probability distributions implementing some commonly used distribution **(a)** Rayleigh, **(b)** Gamma and **(c)** ET-IB distribution

- Rayleigh distribution, defined as

$$R(x;\sigma) = \frac{x}{\sigma^2}\exp\left(-\frac{x^2}{2\sigma^2}\right), \quad x \geq 0 \tag{4.20}$$

where σ is the scaling parameter. In Fig. 4.5a, we chose $\sigma = 1$ and obtained a fidelity of 96.1% after training and KL divergence of 0.03. We set $\delta = 2.5$ for an optimal entangled initial state (Eq. (4.1)). A real-world application where the Rayleigh distribution is observed is the wind speed in a city, studied over a time period in Ref. [140]. It was found that $\sigma = 4$ (see Table 4.1).

- Gamma distribution, defined as

$$G(x;k,\theta) = \frac{x^{k-1}}{\Gamma(k)\theta^k}\exp\left(-\frac{x}{\theta}\right), \quad x \geq 0 \tag{4.21}$$

where k is the shape parameter, and θ is the scaling parameter. In Fig. 4.5b, we chose $k = 2.5$ and $\theta = 0.5$ and obtained a fidelity of 98.1% after training and KL divergence of 0.083. We set $\delta = 2.5$ for an optimal entangled initial state (Eq. (4.1)). There are many applications of the Gamma distribution. As an example, to study the ultrasound images displaying breast cancer, various parameter sets of distribution are used: $(k,\theta) \in \{(1,2),(2,0.5),(2,1),(2,1),(2,4),(4,2)\}$ [141] (see Table 4.1).

- ET-IB distribution, defined as

$$\begin{aligned} f(x;\alpha,\beta,\lambda,\phi) &= \frac{\phi x^{\alpha-1} I_{x/(1+x)}(\alpha,\beta)^{\phi-1}}{(1+x)^{\alpha+\beta}B(\alpha,\beta)} \\ &\quad \times \left(1+\lambda-2\lambda I_{x/(1+x)}(\alpha,\beta)\right)\left(1+\lambda-\lambda I_{x/(1+x)}(\alpha,\beta)\right)^{\phi-1} \end{aligned} \tag{4.22}$$

where

$$I_z(\alpha,\beta) = \frac{1}{B(\alpha,\beta)}\int_0^z t^{\alpha-1}(1-t)^{\beta-1}dt\ , \quad B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \tag{4.23}$$

The parameters are constrained as $\alpha, \beta, \phi > 0$ and $|\lambda| < 1$. For Fig. 4.5c, we chose the four parameters $(\alpha, \beta, \lambda, \phi) = (4.0, 5.0, 0.1, 2.0)$, and obtained a fidelity of 91.4% after training, and KL divergence of 0.229. We set $\delta = 4.0$ for an optimal entangled initial state (Eq. (4.1)). In Ref. [139], the Monte Carlo technique was used to perform parameter estimation for SAR images using different terrains. For example, the parameters $(\alpha, \beta, \lambda, \phi)$ were found to be (6.3, 36.2, -0.9, 0.4) for forest images, (3, 37.5, 0.8, 3.7) for the ocean, and (0.4, 5.3, 0.9, 66.9) for urban (see Table 4.1).

We used the same Adam optimizer with the learning schedule for all cases as before. The number of steps was $S = 2$, the cutoff dimension was $n_{\max} = 15$, and the number of epochs was 100. The final plots for all three distributions are shown in Fig. 4.5. The colored solid lines correspond to the different target distributions, and the black dashed lines correspond to the generated distribution. In all three cases, we achieved high fidelities. An important metric indicating how well the generated distribution compares to the target distribution is the KL divergence. This is evident in the plots, with the Rayleigh distribution having a lower KL divergence and better fitting the histogram compared to the ET-IB distribution, which has a higher value of KL divergence.

Table 4.1: Description of parameters used for different distributions and their comparison with real-world applications.

Names	**Parameters**	**Parameters used in this work**	**Applications**	**Parameters used in the applications**
Rayleigh	σ	1	Wind speeds [140]	4
Gamma	(k, ϕ)	(2.5, 0.5)	Breast cancer ultrasound images [141]	(1, 2), (2, 0.5), (2, 1), (2, 4), (4, 2)
ET-IB	(α, β, λ, θ)	(4, 5, 0.1, 2)	SAR images [139]: forest, ocean, urban	(6.3, 36.2, -0.9, 0.4), (3.0, 37.5, 0.8, 3.7), (0.4, 5.3, 0.9, 66.9)

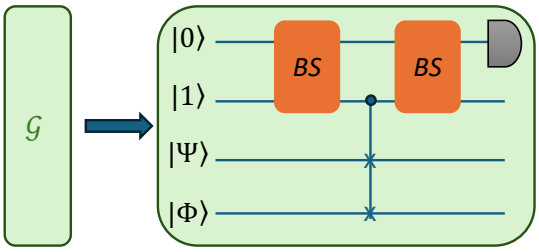


Figure 4.6: Gadget ($\mathcal{G}$) for quantum data to be attached to the CVQBM during training. $|\Psi\rangle$ is the target state and $|\Phi\rangle$ is the output of the CVQBM. The first two lines represent ancilla qumodes initialized in states $|0\rangle$ and $|1\rangle$, respectively.

As discussed above, these distributions have numerous real-world applications. In Table 4.1, we compare the distribution parameters used in this work with some of the real-world applications. The parameters used for the Gamma distribution are similar to those used in studying ultrasound images of breast cancer. However, this was not the case for Rayleigh and ET-IB distributions because they are steeper and require a higher cutoff dimension in simulations. The study of these real-world applications would require high-performance computers, or more efficient simulations, or an actual CV quantum computer, which is not currently available.

4.3 Quantum Data Probability Distribution Generation

Next, we apply our CVQBM to quantum data. We consider two cases: a Gaussian distribution and a non-Gaussian distribution (cat state) both of which are generated approximately by a quantum neural network (QNN) [26].

To handle quantum data, we need an additional gadget, $\mathcal{G}$, to attach to the CVQBM shown in Fig. 4.6. The state $|\Psi\rangle$ represents the target quantum state, generated by a quantum circuit, and $|\Phi\rangle$ represents the output state of the visible mode of our CVQBM. We introduce two control qumodes in the states $|0\rangle$ and $|1\rangle$, respectively, and put them in the state

$$|\text{ctrl}\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{4.24}$$

with the aid of a 50:50 beam splitter. Then we apply the three-mode CSWAP gate

$$U_{\text{CSWAP}} |\text{ctrl}\rangle |\Psi\rangle |\Phi\rangle = \frac{1}{\sqrt{2}}(|01\rangle |\Psi\rangle |\Phi\rangle + |10\rangle |\Phi\rangle |\Psi\rangle) \tag{4.25}$$

which can be implemented with the aid of a non-Gaussian Kerr gate [90, 97]. After channeling the beams through another 50:50 beam splitter, we obtain

$$U_{\mathrm{BS}} \cdot U_{\mathrm{CSWAP}} \left|\mathrm{ctrl}\right\rangle \left|\Psi\right\rangle \left|\Phi\right\rangle = \frac{1}{2}(\left|01\right\rangle (\left|\Psi\right\rangle \left|\Phi\right\rangle - \left|\Phi\right\rangle \left|\Psi\right\rangle) + \left|10\right\rangle (\left|\Psi\right\rangle \left|\Phi\right\rangle + \left|\Phi\right\rangle \left|\Psi\right\rangle)) \ . \tag{4.26}$$

Measurement of the first control qumode yields probabilities P_i $(i = 0, 1)$ with

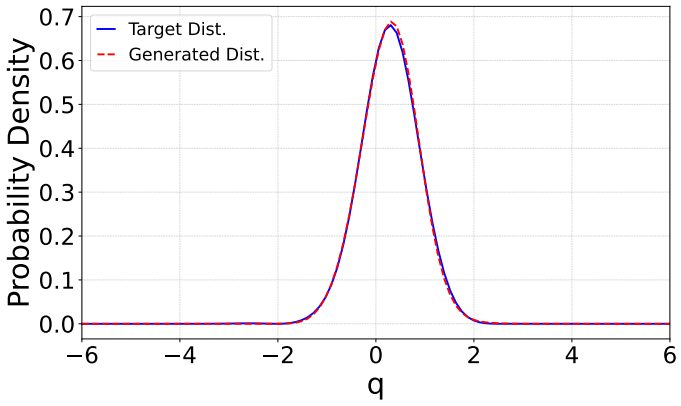
$$P_1 - P_0 = |\Phi|\Psi|^2 \ , \tag{4.27}$$

which is the fidelity to be used in the definition of the cost function (4.15).

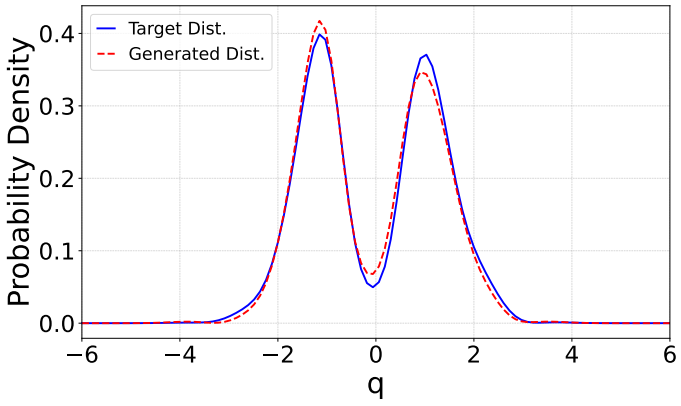
For the preparation of quantum data, we relied on our previous work on experimentally realizable CV QNNs [26]. More specifically, we used the circuits for state preparation to prepare a Gaussian and non-Gaussian state approximately.

For a Gaussian state, we constructed a target Gaussian state for our CVQBM from a single qumode in the vaccum state on which we applied a squeezing gate with a squeezing of 1.73 dB and a displacement gate with a displacement of parameter $\alpha = 0.2$. The resulting Gaussian state had a mean of 0.083 and a standard deviation of 0.145. We trained the CVQBM circuit with 1 step of our QITE algorithm and an initial entanglement of $\delta = 1.5$. Our trained CVQBM circuit attained a fidelity of 99.55% resulting in a Gaussian state with mean 0.083 and standard deviation 0.147.

For the non-Gaussian state, our target quantum state was created using an 8-layer CV QNN [26] which aimed to approximate an even squeezed cat state with a displacement magnitude of 1.2 and squeezing of 2.60 dB. It achieved this with a fidelity of 96.89%. Our CVQBM, in turn, learned this non-Gaussian state with a fidelity of 98.22%, utilizing an architecture that contained Gaussian gates only and an ancillary mode on which we performed a projective measurement, utilizing 3 steps of QITE and entangling parameter $\delta = 2.3$.



(a)



(b)

Figure 4.7: Generating continuous probability distributions from quantum data: (a) Gaussian distribution and (b) non-Gaussian distribution resulted from a cat state preparation. The target (solid blue lines) and generated (dashed red lines) distributions match well.

In the CV QNN simulation, we used a cutoff dimension of 10 and optimized the fidelity of the target state using the Nelder-Mead optimizer. For the CVQBMs, we used the Adam optimizer with a learning scheduler. The initial learning rate was 0.04 for both Gaussian and non-Gaussian data, and both the Gaussian and the non-Gaussian states were trained for 400 epochs with a cutoff of 10.

The target and generated probability distributions are shown in Fig. 4.7. The solid blue lines correspond to the target distributions, whereas the dashed red lines correspond to the generated distributions. The respective distributions match well, with KL divergence of 0.002 and 0.025 for the Gaussian and non-Gaussian (cat state) cases, respectively.

4.4 Experimental Feasibility

A main goal in developing novel quantum algorithms is achieving experimental feasibility with currently available technology. In this Section, we discuss the experimental feasibility of the gates in our proposed CVQBM, the success probability, which is an important metric, since our CVQBM relies on non-deterministic processes (projective measurements on ancilla qumodes), and the effects of noise.

For experimental feasibility, we restricted the parameters of the gates during training within an experimentally attainable range with current technology. This is especially challenging when dealing with real-world problems like studying SAR images on quantum optical hardware. The range of parameters for the gate set that were used in our case studies are shown on Table 4.2. In addition to these gates, the CX gate ($e^{i\kappa q\otimes p}$), can be decomposed into two single-mode squeezers and beam splitters. Particularly, achieving high squeezing rates is one of the most challenging tasks in photonic quantum computing. Experiments are being conducted to achieve higher squeezing values. One of the highest squeezing values (15 dB) was reported in Ref. [142]. Hence, the squeezing parameters required to realize the use cases studied in this paper are within the experimental range.

Table 4.2: Range of trained parameters of gates in the CVQBM circuit

Gates	**Parameter**	**Classical case**	**Quantum case**
		Forest	**Cat state**
Squeeze Gate	r (in dB)	2.60 - 5.21	6.05 - 15.41
Displacement Gate	α	0.8 - 1.3	0.01 - 0.51
CX	κ	0.08 - 1.7	0.01 - 0.45

Our CVQBM design is best suited for photonic quantum hardware. Unfortunately, currently available photonic quantum devices have limitations that prevent us from realizing our CVQBM with cloud-accessed photonic quantum hardware on chip. For example, the Xanadu X-series [20], which is a cloud-accessed quantum chip that includes squeezers, beam splitters, and interferometers cannot accommodate our CVQBM mainly due to the lack of online squeezers (the parameters of the available offline squeezers are fixed). The limitations of current photonic quantum hardware highlight the importance of co-designing quantum hardware, software, and algorithms. We leave the experimental realization of our CVQBM model on photonic quantum hardware to future work.

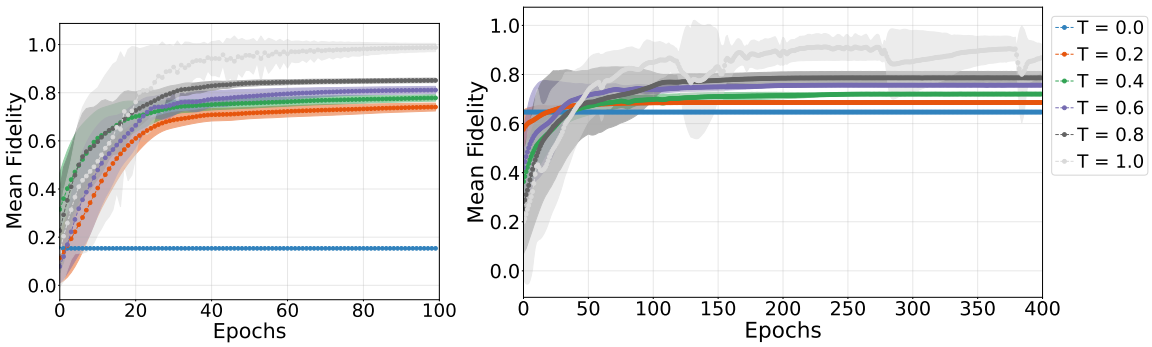
Turning to the success probability of our probabilistic algorithms, we note that the odds of a full pass-through of our CVQBM circuit is dictated by the probability of successfully measuring the required number of photons in the ancilla qumodes. Since we are post-selecting runs with a specific number of photons measured, the overall success probability is a product of success probabilities at each step of the QITE algorithm.

If p_s represents the probability of measuring 0 photons at the $s-$th QITE step, then the overall success probability is given by $P_{\text{success}} = p_1 p_2 \cdots p_S$, where S is the total number of steps in the QITE algorithm. This multiplicative nature arises because the success of each step is conditional on the success of the previous steps. Therefore, to achieve a successful pass-through of the entire circuit, each individual step must succeed, leading to the cumulative success probability described above.

We studied the success probability of generating the forest and sea SAR images using our CVQBM model. For the parameters indicated in Section 4.2, we obtained a success probability of 16.3% (27.2%) with fidelity 99.7% (98.7%) for the forest (sea) SAR images. It is possible to increase success probability at the expense of fidelity. For forest (sea) data, we obtained a success probability of 99.2% (99.7) with a fidelity of 95.1% (94.6%).

Table 4.3: Achieved fidelity and success rates in use cases

Use case	Fidelity	Success Rate
Forest SAR Images	99.7%	16.3%
	95.1%	99.2%
Sea SAR Images	98.7%	27.2%
	94.6%	98.7%
Rayleigh Distribution	96.2%	50.5%
Gamma Distribution	98.1%	1.6%
ET-IB Distribution	91.4%	49%
Quantum Gaussian State	95.0%	99.9%
Quantum Non-Gaussian (Cat) State	98.2%	21.3%



(a) **(b)**

Figure 4.8: Studying various distributions under the loss channel. Each loss parameter (T) run was done 5 times to get the mean cost values and the filled region around it represents the standard deviation of the fidelity. **(a)** Forest SAR images, and **(b)** Cat state distributions under the noise loss channel

Similar results were obtained for the other distributions we studied. For the Rayleigh, Gamma, and ET-IB distributions, we achieved fidelity of 96.2%, 98.1%, and 91.4% at a success rate of 50.5%, 1.6%, and 49%, respectively. Similarly, in the case of quantum data, we were able to train models with fidelity of 95.0% (98.2%) at a success rate of 99.9% (21.3%) for Gaussian (non-Gaussian) states. Results on achieved fidelity and corresponding success rates are shown in Table 4.3.

Finally, we added noise to our model to assess the robustness of the proposed CVQBM. We chose the simple noise model, loss channel, available in Strawberry Fields. This channel couples a given qumode in the circuit, with annihilation operator $\hat{a}$, to an unwanted qumode representing noise, with annihilation operator $\hat{b}$, prepared in the vacuum state. Thus,

$$\hat{a} \to \sqrt{T}\hat{a} + \sqrt{1-T}\hat{b} \ , \tag{4.28}$$

where $T \in [0, 1]$ is the loss parameter known as energy transmissivity. $T = 1$ represents the noiseless case (identity map) and for $T = 0$, the state is mapped to the vacuum state. We introduced loss channels for the various qumodes in our circuits.

We analyzed the cost during training as a function of the energy transmissivity T for one classical (forest SAR image as seen in Fig. 4.3a) and one quantum (non-Gaussian cat state Fig. 4.7(b)) case. Results are shown in Fig. 4.8. Each loss parameter run was done 5 times. The mean of each run is displayed with a solid line with a shaded region around it representing the standard deviation. As expected, we obtained the best fidelity for forest (cat) 99.7% (98.2%) when $T = 1$, corresponding to no loss. The fidelity decreases as we decrease the loss parameter, which makes sense because increasing the noise will make it harder for the machine to train. Eventually, at $T = 0$, no training occurs. With every 10% decrease in the loss, the fidelity changes by around 5%. At 90% loss in the channel, the fidelity change is around 30%.

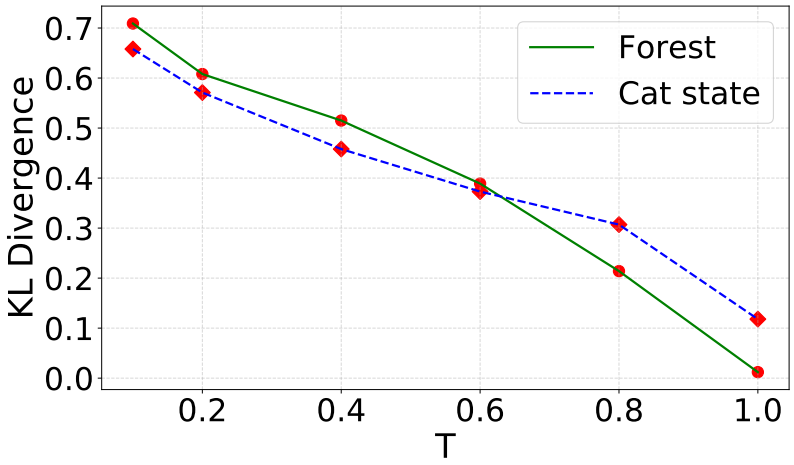


Figure 4.9: KL divergence (Eq. (4.18)) against the loss parameter (T) for the forest SAR image (classical) and cat state (quantum) data cases. Each data point corresponds to 5 runs.

In order to study how the loss channel impacts the performance of our CVQBM, we plotted the KL divergence *vs.* the loss parameter (T) in Fig. 4.9. In this plot, each data point corresponds to 5 training runs. We generated the distribution using the optimized parameters and hence calculated the KL divergence between the target and generated distributions (Eq. (4.18)). The lower value of KL divergence corresponds to the better match between the two distributions. Hence, the KL divergence decreases as $T \to 1$. As expected, best results are obtained at $T = 1$ (no loss). The difference between the classical and quantum cases is small. We note that, up to 40% loss, our CVQBM performs better the quantum case, but with more loss, the performance in the classical case is slightly better.

There is ongoing research for developing error mitigation strategies for photon loss in photonic hardware. For example, in Ref. [143], techniques were presented that can mitigate the effects of photon loss on both output probabilities and expectation values in noisy linear optical circuits. These techniques involve the creation of recycled probabilities, which are derived from output statistics affected by loss and are designed to amplify the signal of the ideal probabilities. These recycled probabilities are then processed to produce a set of loss-mitigated probabilities or expectation values. These technique were shown to perform better than post-selection and zero-noise-extrapolation (ZNE) techniques. We leave study of photon loss mitigation strategies as a future work.

4.5 Conclusion

We proposed a CVQBM which can be realized experimentally with a CV photonic quantum computer. We introduced a novel method to prepare the thermal state required by the BM setup using an entangled state followed by QITE steps. We implemented the entire CVQBM design using Gaussian gates and photon number resolving measurements, making it experimentally realizable.

We tested our CVQBM against both classical and quantum data. For classical data, we worked with the SAR images. These images are not affected by altitude, bad weather, or light, making them amenable to high-resolution remote sensing. We focused on datasets obtained from forest and sea ice monitoring. We generated distributions that were of very low KL divergence. All of the distributions we studied have various real-life applications ranging from SAR images [144] to medical images [145]. For example, SAR images are obtained from a wide-range of applications, such as agriculture monitoring, disaster management after damages done by a disaster like an earthquake, flood, and volcanic eruptions, search and rescue operations for the same, climate change studies, geology, and mining. For quantum data, we focused on a Gaussian distribution generated by a coherent state as well as one generated by a cat state, which has various applications, such as in designing qubits that are resistant to errors [146], implementing entangling gates [147], and quantum spectroscopy [148].

Chapter 5

Conclusions and Future Work

In this thesis, we have developed novel quantum algorithms for NISQ devices. These algorithms use concepts of quantum physics and/or machine learning principles. This innovative blend not only makes the algorithms theoretically robust but also practically viable on current quantum hardware, marking a significant advancement in the field.

We started with the Introduction chapter (Chapter 1), where we provided a comprehensive overview of the field's background and current state. We delved into two types of quantum computing: discrete variable (DV) and continuous variable (CV). DV was the focus of our first research project on collective neutrino oscillation, while CV took the spotlight in the second and third projects, which centered on quantum machine learning in the CV setting. We then shifted our focus to NISQ, discussing its limitations and the urgent need for novel quantum algorithms explicitly designed for NISQ devices. We introduced two such algorithm types - variational quantum algorithms and algorithms based on quantum machine learning techniques. These algorithms are crucial in the current quantum landscape, making this research highly relevant and impactful in the real world.

In Chapter 2, we studied DV quantum eigensolver - Quantum Imaginary Time Evolution (QITE) and Quantum Lanczos (QLanczos) algorithms. These algorithms

are very effective in studying many-body quantum systems. To be more specific, they can efficiently find eigenvectors and eigenvalues of a system provided we use some error mitigation techniques. We applied the algorithms to study the collective neutrino oscillations using a quantum computer. We selected a system of four neutrinos and analyzed the eigensystem. We demonstrated collective neutrino oscillations using first-order Trotterization methods. To ensure the reliability of our results, we implemented error mitigation techniques, such as readout-error mitigation and zero-noise extrapolation, during the runs on the quantum hardware. To make the entire problem solvable on the current quantum computer, we also used the symmetry properties of the Hamiltonian which enabled us to study 4 neutrinos using just 2 and 3 qubit systems.

Building on our foundational work, several researchers have expanded the study of collective neutrinos using diverse quantum platforms. These studies have been conducted on a trapped-ion quantum device [149], on a superconducting qutrit [150], and through tensor networks [151]. Further extensions of this research could explore similar systems in high-energy physics. For instance, the QITE and QLanczos algorithms, which have shown great potential in our research, can be applied to the study of hadrons within the 1D SU(2) gauge theory framework. This could extend to SU(3) gauge theory, offering deeper insights into quantum chromodynamics.

Moreover, it would be interesting to integrate QITE with machine learning techniques. In particular, a notable study has employed reinforcement learning to enhance the QITE algorithm's accuracy [152]. This approach opens the door for integrating other machine learning techniques with variational quantum algorithms, suggesting promising avenues for future research.

In Chapter 3, we developed a continuous-variable quantum neural network that can be experimentally realized on current photonic quantum devices. Instead of using the nonlinear Kerr gate, we introduced nonlinearity by performing measurements on an ancillary mode, using only Gaussian gates to construct the neural network. We evaluated the performance of our neural network across four problems: state

preparation, curve fitting, fraud detection, and image classification, all conducted in a supervised learning context. A fully quantum neural network handled the state preparation and curve fitting tasks, while more complex problems of fraud detection and image classification required a hybrid quantum-classical approach. In all cases, we achieved accuracies exceeding 95%, demonstrating the significant potential of our method.

Testing our CV quantum algorithms on photonic quantum hardware would be interesting. Unfortunately, available cloud-accessed photonic quantum devices currently have limitations that prevent the implementation of the algorithms discussed here. A prominent example is Xanadu's cloud-accessed Borealis quantum chip [20], which includes squeezers, beam splitters, and interferometers. While the parameters of the interferometers can be modified, the parameters of the squeezers are fixed. This prevents the implementation of our CV QNN, which requires squeezing parameters that can be updated. These limitations on current photonic quantum hardware highlight the importance of codesigning quantum hardware and software. Codesign is critical in overcoming hardware limitations and addressing the efficient use of limited resources. We are exploring collaborating with a national laboratory or university partner with photonic quantum computing capabilities to test our model experimentally.

A critical issue to address is benchmarking the performance of QNNs and identifying advantages over their classical counterparts. This has been addressed in the case of DV quantum computing by the authors of [153]. They compared the expressibility of classical and quantum neural networks by calculating effective dimensions for different cases. They showed that QNNs have higher effective dimensions and train faster than their classical counterparts. They used the Fisher information spectrum to demonstrate the resilience of QNNs in terms of barren plateaus and the problem of vanishing gradients. It is essential to perform a similar study with CV QNNs and benchmark the various models considered here for an appropriate measure, such as an effective dimension. A related study has

been performed on barren plateaus in bosonic variational circuits [154]. They used an energy-dependent circuit to prepare Gaussian and number states. It would be interesting to extend this to other problems in quantum machine learning by calculating effective dimensions for different models and comparing them with classical and DV quantum models.

While we have concentrated on quantum optical systems, there are other promising candidates for CV QC. One such platform is based on microwave quantum systems, also known as circuit QED (cQED), in which nonlinearities are introduced through Josephson junctions [155]. They have been shown to lead to superpositions of Fock states [156], cat states [157, 158], and GKP states [159]. Through a superconducting nonlinear asymmetric inductive element (SNAIL), a universal set of gates can be implemented for CV QC [160]. Experimental challenges remain, as high resonator quality factors ($\mathcal{O}(10^5)$) are needed [161], as well as $\mathcal{O}$(ns) pulse synthesis resolution [162]. Nevertheless, it is an interesting avenue to explore for the CV QNNs discussed here.

In Chapter 4, we advanced our exploration of CV quantum machine learning by proposing the CV Quantum Boltzmann Machine (QBM). Although the QBM has numerous applications, we specifically focused on its capabilities as a generative model. We configured our machine as a probability distribution generator, testing it against classical and quantum data. For the classical data, we used synthetic-aperture radar (SAR) images to target probability distributions: forest monitoring strips from Brazil (approximating a Gaussian distribution) and sea monitoring images from the Gulf of Bothnia (approximating a Weibull distribution). The KL divergence between the targeted and generated distributions was close to 0.01, indicating a high accuracy in the generated distributions. We tested two scenarios for quantum data: Gaussian and non-Gaussian distribution, achieving a maximum KL divergence of 0.03. This slight divergence demonstrates an excellent match between the curves, underscoring the QBM's effective generative capability. We also detailed

the experimental feasibility of our proposed QBM, including gate parameters, success rates, and the impact of the loss channel.

In this work, we focused on generating continuous probability distributions that have many real-world applications. However, our proposed CVQBM can also be further utilized for anomaly detection [163] and classification tasks, as well as variational inference as discussed in [164] by optimizing a candidate probability distribution to approximate a posterior distribution over unobserved variables. This task has many real-world applications with continuous-valued data that range from identifying diseases from symptoms to classifying economic regimes from price movements. For example, classical Boltzmann machines can efficiently analyze time series data. Extending this application to the QBM presents exciting possibilities, such as detecting gravitational waves. Given their subtle signals and high noise levels, QBMs could potentially offer a quantum advantage in identifying these elusive phenomena.

In conclusion, this thesis has pioneered the development of innovative algorithms tailored for both DV and CV NISQ devices. These algorithms have demonstrated promising potential under current technological constraints by integrating core principles from physics and/or machine learning. While quantum computers are still evolving and may require years to fully realize their quantum advantage, the work presented here leverages existing quantum hardware to forge powerful, hybrid quantum-classical algorithms. These algorithms not only challenge the existing capabilities of classical computers in specific scenarios but also lay the groundwork for the next generation of fault-tolerant quantum computing. Through this approach, we edge closer to harnessing the transformative power of quantum computing, marking a significant step forward in both theory and practical application.

Bibliography

[1] Richard P Feynman. Simulating physics with computers. In *Feynman and computation*, pages 133–153. CRC Press, 2018. 1, 19

[2] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019. 1

[3] Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018. 1

[4] Sergey Bravyi, David Gosset, Robert König, and Marco Tomamichel. Quantum advantage with noisy shallow circuits. *Nature Physics*, 16(10):1040–1045, 2020. 1

[5] Diego Ristè, Marcus P Da Silva, Colm A Ryan, Andrew W Cross, Antonio D Córcoles, John A Smolin, Jay M Gambetta, Jerry M Chow, and Blake R Johnson. Demonstration of quantum advantage in machine learning. *npj Quantum Information*, 3(1):16, 2017. 1

[6] Nikitas Stamatopoulos, Guglielmo Mazzola, Stefan Woerner, and William J Zeng. Towards quantum advantage in financial market risk using quantum gradient algorithms. *Quantum*, 6:770, 2022. 1

[7] Hsin-Yuan Huang, Richard Kueng, and John Preskill. Information-theoretic bounds on quantum advantage in machine learning. *Physical Review Letters*, 126(19):190505, 2021. 1

[8] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018. 2, 9

[9] Alexandru Paler and Simon J Devitt. An introduction to fault-tolerant quantum computing. *arXiv preprint arXiv:1508.03695*, 2015. 2

[10] Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995. 2, 9

[11] Wentao Chen, Jaren Gan, Jing-Ning Zhang, Dzmitry Matuskevich, and Kihwan Kim. Quantum computation and simulation with vibrational modes of trapped ions. *Chinese Physics B*, 30(6):060311, 2021. 2

[12] David Peral-García, Juan Cruz-Benito, and Francisco José García-Peñalvo. Systematic literature review: Quantum machine learning and its applications. *Computer Science Review*, 51:100619, 2024. 2

[13] Chien-Hung Cho, Chih-Yu Chen, Kuo-Chin Chen, Tsung-Wei Huang, Ming-Chien Hsu, Ning-Ping Cao, Bei Zeng, Seng-Ghee Tan, and Ching-Ray Chang. Quantum computation: Algorithms and applications. *Chinese Journal of Physics*, 72:248–269, 2021. 2

[14] Sam McArdle, Suguru Endo, Alán Aspuru-Guzik, Simon C Benjamin, and Xiao Yuan. Quantum computational chemistry. *Reviews of Modern Physics*, 92(1):015003, 2020. 2, 22

[15] Dmitry Solenov, Jay Brieler, and Jeffrey F Scherrer. The potential of quantum computing and machine learning to advance clinical research and change the practice of medicine. *Missouri medicine*, 115(5):463, 2018. 2

[16] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023. 2, 28

[17] Microsoft Azure. Azure quantum, 2024. 2

[18] Amazon. Amazon braket, 2024. 2

[19] Cirq Quantum Computing services. Cirq, google, 2024. 2

[20] Juan M Arrazola, Ville Bergholm, Kamil Brádler, Thomas R Bromley, Matt J Collins, Ish Dhand, Alberto Fumagalli, Thomas Gerrits, Andrey Goussev, Lukas G Helt, et al. Quantum circuits with many photons on a programmable nanophotonic chip. *Nature*, 591(7848):54–60, 2021. 2, 110, 118

[21] D wave systems. Dwave, the quantum computing company. 2

[22] IonQ contributors. Ionq. 2

[23] Ashley Montanaro. Quantum algorithms: an overview. *npj Quantum Information*, 2(1):1–8, 2016. 3

[24] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999. 3

[25] Kübra Yeter-Aydeniz, Shikha Bangar, George Siopsis, and Raphael C. Pooser. Collective neutrino oscillations on a quantum computer. *Quantum Information Processing*, 21(3):84, 2022. 3, 6, 13, 23

[26] Shikha Bangar, Leanto Sunny, Kübra Yeter-Aydeniz, and George Siopsis. Experimentally realizable continuous-variable quantum neural networks. *Physical Review A*, 108(4):042414, 2023. 3, 8, 17, 92, 105, 106

[27] Shikha Bangar, Leanto Sunny, Kübra Yeter-Aydeniz, and George Siopsis. Continuous-variable quantum boltzmann machine. *arXiv preprint arXiv:2405.06580*, 2024. 3, 8

[28] David P DiVincenzo. The physical implementation of quantum computation. *Fortschritte der Physik: Progress of Physics*, 48(9-11):771–783, 2000. 4

[29] He-Liang Huang, Dachao Wu, Daojin Fan, and Xiaobo Zhu. Superconducting quantum computing: a review. *Science China Information Sciences*, 63:1–32, 2020. 5

[30] Jonathan Wei Zhong Lau, Kian Hwee Lim, Harshank Shrotriya, and Leong Chuan Kwek. Nisq computing: where are we and where do we go? *AAPPS bulletin*, 32(1):27, 2022. 5, 18

[31] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020. 8

[32] Daniel A Lidar and Todd A Brun. *Quantum error correction*. Cambridge university press, 2013. 9

[33] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021. 10, 13

[34] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S Kottmann, Tim Menke, et al. Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, 94(1):015004, 2022. 12, 19

[35] Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, Edward Grant, Leonard Wossnig, Ivan Rungger, George H Booth, et al. The variational quantum eigensolver: a review of methods and best practices. *Physics Reports*, 986:1–128, 2022. 13, 19, 20

[36] Edward Farhi and Aram W Harrow. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv preprint arXiv:1602.07674*, 2016. 13

[37] Manas Sajjan, Junxu Li, Raja Selvarajan, Shree Hari Sureshbabu, Sumit Suresh Kale, Rishabh Gupta, Vinit Singh, and Sabre Kais. Quantum machine learning for chemistry and physics. *Chemical Society Reviews*, 51(15):6475–6573, 2022. 14

[38] Lorenzo Buffoni and Filippo Caruso. New trends in quantum machine learning (a). *Europhysics letters*, 132(6):60004, 2021. 14

[39] Juan Carrasquilla and Roger G Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431–434, 2017. 14

[40] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, 2017. 14

[41] Savvas Varsamopoulos, Koen Bertels, and Carmen G Almudever. Decoding surface code with a distributed neural network–based decoder. *Quantum Machine Intelligence*, 2:1–12, 2020. 14

[42] Wen Guan, Gabriel Perdue, Arthur Pesah, Maria Schuld, Koji Terashi, Sofia Vallecorsa, and Jean-Roch Vlimant. Quantum machine learning in high energy physics. *Machine Learning: Science and Technology*, 2(1):011003, 2021. 14

[43] Andrew Blance and Michael Spannowsky. Quantum machine learning for particle physics using a variational quantum classifier. *Journal of High Energy Physics*, 2021(2):1–20, 2021. 14

[44] Sau Lan Wu and Shinjae Yoo. Challenges and opportunities in quantum machine learning for high-energy physics. *Nature Reviews Physics*, 4(3):143–144, 2022. 14

[45] Nathan Killoran, Thomas R. Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolás Quesada, and Seth Lloyd. Continuous-variable quantum neural networks. *Phys. Rev. Res.*, 1:033063, Oct 2019. 14, 15, 16, 48, 49, 51

[46] Stefan Scheel, Kae Nemoto, William J Munro, and Peter L Knight. Measurement-induced nonlinearity in linear optics. *Physical Review A*, 68(3):032310, 2003. 17

[47] Shunya Konno, Atsushi Sakaguchi, Warit Asavanant, Hisashi Ogawa, Masaya Kobayashi, Petr Marek, Radim Filip, Jun-ichi Yoshikawa, and Akira Furusawa. Nonlinear squeezing for measurement-based non-gaussian operations in time domain. *Physical Review Applied*, 15(2):024024, 2021. 17

[48] Daiqin Su, Casey R Myers, and Krishna Kumar Sabapathy. Conversion of gaussian states to non-gaussian states using photon-number-resolving detectors. *Physical Review A*, 100(5):052301, 2019. 17

[49] Miller Eaton, Carlos González-Arciniegas, Rafael N Alexander, Nicolas C Menicucci, and Olivier Pfister. Measurement-based generation and preservation of cat and grid states within a continuous-variable cluster state. *Quantum*, 6:769, 2022. 17

[50] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1):4213, 2014. 18, 20

[51] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014. 18

[52] Mario Motta, Chong Sun, Adrian TK Tan, Matthew J O'Rourke, Erika Ye, Austin J Minnich, Fernando GSL Brandao, and Garnet Kin-Lic Chan. Determining eigenstates and thermal states on a quantum computer using

quantum imaginary time evolution. *Nature Physics*, 16(2):205–210, 2020. 18, 20, 22, 27, 29, 86

[53] Weikang Li and Dong-Ling Deng. Recent advances for quantum classifiers. *Science China Physics, Mechanics & Astronomy*, 65(2):220301, 2022. 18

[54] Miklos Santha. Quantum walk based search algorithms. In *International Conference on Theory and Applications of Models of Computation*, pages 31–46. Springer, 2008. 18

[55] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015. 18

[56] Shi-Ning Sun, Mario Motta, Ruslan N Tazhigulov, Adrian TK Tan, Garnet Kin-Lic Chan, and Austin J Minnich. Quantum computation of finite-temperature static and dynamical properties of spin systems using quantum imaginary time evolution. *PRX Quantum*, 2(1):010317, 2021. 20

[57] Kübra Yeter-Aydeniz, George Siopsis, and Raphael C Pooser. Scattering in the ising model with the quantum lanczos algorithm. *New Journal of Physics*, 23(4):043033, 2021. 20, 22, 26, 29

[58] Kübra Yeter-Aydeniz, Bryan T Gard, Jacek Jakowski, Swarnadeep Majumder, George S Barron, George Siopsis, Travis S Humble, and Raphael C Pooser. Benchmarking quantum chemistry computations with variational, imaginary time evolution, and krylov space solver algorithms. *Advanced Quantum Technologies*, 4(7):2100012, 2021. 20, 22, 31

[59] Bela Bauer, Sergey Bravyi, Mario Motta, and Garnet Kin-Lic Chan. Quantum algorithms for quantum chemistry and quantum materials science. *Chemical Reviews*, 120(22):12685–12717, 2020. 20

[60] Raban Iten, Roger Colbeck, Ivan Kukuljan, Jonathan Home, and Matthias Christandl. Quantum circuits for isometries. *Physical Review A*, 93(3):032318, 2016. 23

[61] Rizwanul Alam, George Siopsis, Rebekah Herrman, James Ostrowski, Phillip C Lotshaw, and Travis S Humble. Solving maxcut with quantum imaginary time evolution. *Quantum Information Processing*, 22(7):281, 2023. 23

[62] Nora M Bauer, Rizwanul Alam, George Siopsis, and James Ostrowski. Combinatorial optimization with quantum imaginary time evolution. *Physical Review A*, 109(5):052430, 2024. 23

[63] Kübra Yeter-Aydeniz, Eleftherios Moschandreou, and George Siopsis. Quantum imaginary-time evolution algorithm for quantum field theories with continuous variables. *Physical Review A*, 105(1):012412, 2022. 23, 88, 93

[64] Takaaki Kajita. Atmospheric neutrinos and discovery of neutrino oscillations. *Proceedings of the Japan Academy, Series B*, 86(4):303–321, 2010. 26

[65] Yoshiyuki Fukuda, T Hayakawa, E Ichihara, K Inoue, K Ishihara, Hirokazu Ishino, Y Itow, T Kajita, J Kameda, S Kasuga, et al. Evidence for oscillation of atmospheric neutrinos. *Physical review letters*, 81(8):1562, 1998. 26

[66] Sno Collaboration et al. Measurement of the rate of nu_e+ d–¿ p+ p+ eˆ- interactions produced by 8b solar neutrinos at the sudbury neutrino observatory. *arXiv preprint nucl-ex/0106015*, 2001. 26

[67] FP An, JZ Bai, AB Balantekin, HR Band, D Beavis, W Beriguete, M Bishai, S Blyth, K Boddy, RL Brown, et al. Observation of electron-antineutrino disappearance at daya bay. *Physical Review Letters*, 108(17):171803, 2012. 26

[68] Lincoln Wolfenstein. Neutrino oscillations in matter. In *Solar neutrinos*, pages 294–299. CRC Press, 2018. 26

[69] Huaiyu Duan, George M Fuller, and Yong-Zhong Qian. Collective neutrino oscillations. *Annual Review of Nuclear and Particle Science*, 60:569–594, 2010. 26

[70] Kevork N Abazajian, John F Beacom, and Nicole F Bell. Stringent constraints on cosmological neutrino-antineutrino asymmetries from synchronized flavor transformation. *Physical Review D*, 66(1):013008, 2002. 27

[71] Meng-Ru Wu, Yong-Zhong Qian, Gabriel Martínez-Pinedo, Tobias Fischer, and Lutz Huther. Effects of neutrino oscillations on nucleosynthesis and neutrino signals for an 18 m supernova model. *Physical Review D*, 91(6):065016, 2015. 27

[72] Gabriel Martínez-Pinedo, Tobias Fischer, Karlheinz Langanke, Andreas Lohs, Andre Sieverding, and Meng-Ru Wu. Neutrinos and their impact on core-collapse supernova nucleosynthesis. *Handbook of Supernovae*, page 1805, 2017. 27

[73] Amol V Patwardhan, Michael J Cervia, and A Baha Balantekin. Eigenvalues and eigenstates of the many-body collective neutrino oscillation problem. *Physical Review D*, 99(12):123013, 2019. 27, 28

[74] Benjamin Nachman, Davide Provasoli, Wibe A De Jong, and Christian W Bauer. Quantum algorithm for high energy physics simulations. *Physical review letters*, 126(6):062001, 2021. 27

[75] Changsuk Noh, BM Rodriguez-Lara, and Dimitris G Angelakis. Quantum simulation of neutrino oscillations with trapped ions. *New Journal of Physics*, 14(3):033028, 2012. 27

[76] CA Argüelles and BJP Jones. Neutrino oscillations in a quantum processor. *Physical review research*, 1(3):033176, 2019. 27

[77] Benjamin Hall, Alessandro Roggero, Alessandro Baroni, and Joseph Carlson. Simulation of collective neutrino oscillations on a quantum computer. *Physical Review D*, 104(6):063009, 2021. 28

[78] Michael J Cervia, Amol V Patwardhan, Akif B Balantekin, Susan N Coppersmith, and Calvin W Johnson. Entanglement and collective flavor oscillations in a dense neutrino gas. *Physical Review D*, 100(8):083001, 2019. 30, 33

[79] Eugene F Dumitrescu, Alex J McCaskey, Gaute Hagen, Gustav R Jansen, Titus D Morris, T Papenbrock, Raphael C Pooser, David Jarvis Dean, and Pavel Lougovski. Cloud quantum computing of an atomic nucleus. *Physical review letters*, 120(21):210501, 2018. 40, 46

[80] Marc G Davis, Ethan Smith, Ana Tudor, Koushik Sen, Irfan Siddiqi, and Costin Iancu. Towards optimal topology aware quantum circuit synthesis. pages 223–234, 2020. 45

[81] Xin-Chuan Wu, Marc Grau Davis, Frederic T Chong, and Costin Iancu. Qgo: Scalable quantum circuit optimization using automated synthesis. *arXiv preprint arXiv:2012.09835*, 2020. 45

[82] Oleksandr Kyriienko. Quantum inverse iteration algorithm for programmable quantum simulators. *npj Quantum Information*, 6(1):7, 2020. 47

[83] Kazuhiro Seki and Seiji Yunoki. Quantum power method by a superposition of time-evolved states. *PRX Quantum*, 2(1):010333, 2021. 47

[84] Philippe Suchsland, Francesco Tacchino, Mark H Fischer, Titus Neupert, Panagiotis Kl Barkoutsos, and Ivano Tavernelli. Algorithmic error mitigation scheme for current quantum processors. *Quantum*, 5:492, 2021. 47

[85] Yuri Alexeev, Dave Bacon, Kenneth R Brown, Robert Calderbank, Lincoln D Carr, Frederic T Chong, Brian DeMarco, Dirk Englund, Edward Farhi, Bill

Fefferman, et al. Quantum computer systems for scientific discovery. *PRX quantum*, 2(1):017001, 2021. 47

[86] Samuel L. Braunstein and Peter van Loock. Quantum information with continuous variables. *Rev. Mod. Phys.*, 77:513–577, Jun 2005. 48

[87] Kevin Marshall, Raphael Pooser, George Siopsis, and Christian Weedbrook. Quantum simulation of quantum field theory using continuous variables. *Phys. Rev. A*, 92:063825, Dec 2015. 48

[88] Kübra Yeter-Aydeniz and George Siopsis. Quantum computation of scattering amplitudes in scalar quantum electrodynamics. *Phys. Rev. D*, 97:036004, Feb 2018. 48

[89] Kübra Yeter-Aydeniz, Eleftherios Moschandreou, and George Siopsis. Quantum imaginary-time evolution algorithm for quantum field theories with continuous variables. *Phys. Rev. A*, 105:012412, Jan 2022. 48

[90] Hoi-Kwan Lau, Raphael Pooser, George Siopsis, and Christian Weedbrook. Quantum machine learning over infinite dimensions. *Phys. Rev. Lett.*, 118:080501, Feb 2017. 48, 106

[91] Mikael Lassen, Lars Skovgaard Madsen, Metin Sabuncu, Radim Filip, and Ulrik L Andersen. Experimental demonstration of squeezed-state quantum averaging. *Physical Review A*, 82(2):021801, 2010. 48

[92] Jinjing Shi, Shuhui Chen, Yuhu Lu, Yanyan Feng, Ronghua Shi, Yuguang Yang, and Jian Li. An approach to cryptography based on continuous-variable quantum neural network. *Scientific Reports*, 10(1):2107, Feb 2020. 49

[93] Peng-Hui Qiu, Xiao-Guang Chen, and Yi-Wei Shi. Detecting entanglement with deep quantum neural networks. *IEEE Access*, 7:94310–94320, 2019. 49

[94] Su Yeon Chang, Sofia Vallecorsa, Elías F Combarro, and Federico Carminati. Quantum generative adversarial networks in a continuous-variable architecture to simulate high energy physics detectors. *arXiv preprint arXiv:2101.11132*, 2021. 49, 84

[95] Saverio Monaco, Oriel Kiss, Antonio Mandarino, Sofia Vallecorsa, and Michele Grossi. Quantum phase detection generalization from marginal quantum neural network models. *Physical Review B*, 107(8):L081105, 2023. 49

[96] Arthur Pesah, Marco Cerezo, Samson Wang, Tyler Volkoff, Andrew T Sornborger, and Patrick J Coles. Absence of barren plateaus in quantum convolutional neural networks. *Physical Review X*, 11(4):041011, 2021. 49

[97] Kevin Marshall, Raphael Pooser, George Siopsis, and Christian Weedbrook. Repeat-until-success cubic phase gate for universal continuous-variable quantum computation. *Physical Review A*, 91(3):032321, 2015. 49, 50, 52, 74, 81, 84, 106

[98] Stefan Scheel, Kae Nemoto, William J. Munro, and Peter L. Knight. Measurement-induced nonlinearity in linear optics. *Phys. Rev. A*, 68:032310, Sep 2003. 49

[99] Shunya Konno, Atsushi Sakaguchi, Warit Asavanant, Hisashi Ogawa, Masaya Kobayashi, Petr Marek, Radim Filip, Jun-ichi Yoshikawa, and Akira Furusawa. Nonlinear squeezing for measurement-based non-gaussian operations in time domain. *Phys. Rev. Appl.*, 15:024024, Feb 2021. 49

[100] Daniel Gottesman, Alexei Kitaev, and John Preskill. Encoding a qubit in an oscillator. *Phys. Rev. A*, 64:012310, Jun 2001. 51, 63

[101] Pieter Kok, Hwang Lee, and Jonathan P. Dowling. Creation of large-photon-number path entanglement conditioned on photodetection. *Phys. Rev. A*, 65:052104, Apr 2002. 51

[102] Daiqin Su, Casey R. Myers, and Krishna Kumar Sabapathy. Conversion of gaussian states to non-gaussian states using photon-number-resolving detectors. *Phys. Rev. A*, 100:052301, Nov 2019. 51

[103] Miller Eaton, Carlos González-Arciniegas, Rafael N. Alexander, Nicolas C. Menicucci, and Olivier Pfister. Measurement-based generation and preservation of cat and grid states within a continuous-variable cluster state. *Quantum*, 6:769, July 2022. 51

[104] Timothy J Stavenger, Eleanor Crane, Kevin C Smith, Christopher T Kang, Steven M Girvin, and Nathan Wiebe. C2qa-bosonic qiskit. In *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8. IEEE, 2022. 56

[105] Nathan Killoran, Josh Izaac, Nicol'as Quesada, Ville Bergholm, Matthew Amy, and Christian Weedbrook. Strawberry Fields: A software platform for photonic quantum computing. *Quantum*, 3:129, 2019. 57, 93

[106] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 57, 93

[107] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0:

Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. 60

[108] Fuchang Gao and Lixing Han. Implementing the nelder-mead simplex algorithm with adaptive parameters. *Computational Optimization and Applications*, 51(1):259–277, 2012. 60

[109] Yunong Shi, Christopher Chamberland, and Andrew Cross. Fault-tolerant preparation of approximate gkp states. *New Journal of Physics*, 21(9):093007, 2019. 63

[110] et al. Bachmann, Janio Martinez. Dataset on kaggle / credit card fraud detection. *kaggle*, 2018. 72

[111] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 79

[112] Juan Miguel Arrazola, Thomas R Bromley, Josh Izaac, Casey R Myers, Kamil Brádler, and Nathan Killoran. Machine learning method for state preparation and gate synthesis on photonic quantum computers. *Quantum Science and Technology*, 4(2):024004, jan 2019. 81

[113] J. Eli Bourassa, Rafael N. Alexander, Michael Vasmer, Ashlesha Patil, Ilan Tzitrin, Takaya Matsuura, Daiqin Su, Ben Q. Baragiola, Saikat Guha, Guillaume Dauphinais, Krishna K. Sabapathy, Nicolas C. Menicucci, and Ish Dhand. Blueprint for a Scalable Photonic Fault-Tolerant Quantum Computer. *Quantum*, 5:392, February 2021. 82

[114] Mikael Lassen, Metin Sabuncu, Alexander Huck, Julien Niset, Gerd Leuchs, Nicolas J. Cerf, and Ulrik L. Andersen. Quantum optical coherence can survive photon losses using a continuous-variable quantum erasure-correcting code. *Nature Photonics*, 4(10):700–705, Oct 2010. 82

[115] Matthias C Caro, Hsin-Yuan Huang, Marco Cerezo, Kunal Sharma, Andrew Sornborger, Lukasz Cincio, and Patrick J Coles. Generalization in quantum machine learning from few training data. *Nature communications*, 13(1):4919, 2022. 83

[116] Hsin-Yuan Huang, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven, and Jarrod R McClean. Power of data in quantum machine learning. *Nature communications*, 12(1):2631, 2021. 83

[117] Michele Grossi, Noelle Ibrahim, Voica Radescu, Robert Loredo, Kirsten Voigt, Constantin Von Altrock, and Andreas Rudnik. Mixed quantum–classical method for fraud detection with quantum feature selection. *IEEE Transactions on Quantum Engineering*, 3:1–12, 2022. 83

[118] Mohammad H Amin, Evgeny Andriyash, Jason Rolfe, Bohdan Kulchytskyy, and Roger Melko. Quantum boltzmann machine, 2018. 85, 93

[119] Luuk Coopmans and Marcello Benedetti. On the sample complexity of quantum boltzmann machine learning, 2023. 85, 86

[120] Onno Huijgen, Luuk Coopmans, Peyman Najafi, Marcello Benedetti, and Hilbert J Kappen. Training quantum boltzmann machines with the β-variational quantum eigensolver, 2023. 86

[121] Sam McArdle, Tyson Jones, Suguru Endo, Ying Li, Simon C. Benjamin, and Xiao Yuan. Variational ansatz-based quantum simulation of imaginary time evolution. *npj Quantum Information*, 5(1), September 2019. 86

[122] Jin-Guo Liu, Liang Mao, Pan Zhang, and Lei Wang. Solving quantum statistical mechanics with variational autoregressive networks and quantum circuits. *Machine Learning: Science and Technology*, 2(2):025011, 2021. 86

[123] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. Variational quantum boltzmann machines, 2021. 86, 90, 93

[124] Yuta Shingu, Yuya Seki, Shohei Watabe, Suguru Endo, Yuichiro Matsuzaki, Shiro Kawabata, Tetsuro Nikuni, and Hideaki Hakoshima. Boltzmann machine learning with a variational quantum algorithm. *Physical Review A*, 104(3), September 2021. 86

[125] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):4812, 2018. 86

[126] Vivek Dixit, Raja Selvarajan, Tamer Aldwairi, Yaroslav Koshka, Mark A Novotny, Travis S Humble, Muhammad A Alam, and Sabre Kais. Training a quantum annealing based restricted boltzmann machine on cybersecurity data, 2021. 87

[127] Jonathan Romero and Alán Aspuru-Guzik. Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions. *Advanced Quantum Technologies*, 4(1):2000003, 2021. 87

[128] Abhinav Anand, Jonathan Romero, Matthias Degroote, and Alán Aspuru-Guzik. Noise robustness and experimental demonstration of a quantum generative adversarial network for continuous distributions. *Advanced Quantum Technologies*, 4(5):2000069, 2021. 87

[129] Prabhat Anand, M Girish Chandra, and Ankit Khandelwal. Time-series forecasting using continuous variables-based quantum neural networks. In *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 994–999. IEEE, 2024. 87

[130] Ieva Čepaitė, Brian Coyle, and Elham Kashefi. A continuous variable born machine, 2022. 87

[131] Mária Kieferová and Nathan Wiebe. Tomography and generative training with quantum boltzmann machines, 2017. 93

[132] Nathan Wiebe and Leonard Wossnig. Generative training of quantum boltzmann machines with hidden units, 2019. 93

[133] Daniel Crawford, Anna Levit, Navid Ghadermarzy, Jaspreet S Oberoi, and Pooya Ronagh. Reinforcement learning using quantum boltzmann machines, 2016. 93

[134] Jonas Stein, Daniëlle Schuman, Magdalena Benkard, Thomas Holger, Wanja Sajko, Michael Kölle, Jonas Nüßlein, Leo Sünkel, Olivier Salomon, and Claudia Linnhoff-Popien. Exploring unsupervised anomaly detection with quantum boltzmann machines in fraud detection, 2023. 93

[135] Thomas Templin, Milad Memarzadeh, Walter Vinci, P Aaron Lott, Ata Akbari Asanjan, Anthony Alexiades Armenakas, and Eleanor Rieffel. Anomaly detection in aeronautics data with quantum-compatible discrete deep generative model, 2023. 93

[136] Siddhartha Srivastava and Veera Sundararaghavan. Generative and discriminative training of boltzmann machine through quantum annealing, 2023. 93

[137] Sar datasets, 2022. 97, 98

[138] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. 97

[139] Murilo Sagrillo, Renata R Guerra, Fábio M Bayer, and Renato Machado. A new probability distribution for sar image modeling, 2022. 100, 103, 104

[140] Alexandru Serban, Lizica Simona Paraschiv, and Spiru Paraschiv. Assessment of wind energy potential based on weibull and rayleigh distribution models, 2020. 102, 104

[141] Filipa Borlinhas, Ricardo Loução, Raquel C. Conceição, and Hugo A Ferreira. Gamma distribution model in the evaluation of breast cancer through diffusion-weighted mri: a preliminary study, 2019. 102, 104

[142] Henning Vahlbruch, Moritz Mehmet, Karsten Danzmann, and Roman Schnabel. Detection of 15 db squeezed states of light and their application for the absolute calibration of photoelectric quantum efficiency. *Physical review letters*, 117(11):110801, 2016. 108

[143] James Mills and Rawad Mezher. Mitigating photon loss in linear optical quantum circuits: classical postprocessing methods outperforming postselection. *arXiv preprint arXiv:2405.02278*, 2024. 114

[144] Gui Gao. Statistical modeling of sar images: A survey, 2010. 115

[145] Greg Slabaugh, Gozde Unal, Micheal Wels, Tong Fang, and Bimba Rao. Statistical region-based segmentation of ultrasound images, 2009. 115

[146] Mazyar Mirrahimi, Zaki Leghtas, Victor V Albert, Steven Touzard, Robert J Schoelkopf, Liang Jiang, and Michel H Devoret. Dynamically protected cat-qubits: a new paradigm for universal quantum computation, 2014. 115

[147] Ye-Hong Chen, Roberto Stassi, Wei Qin, Adam Miranowicz, and Franco Nori. Fault-tolerant multiqubit geometric entangling gates using photonic cat-state qubits, 2022. 115

[148] Mackillo Kira, Stephan W Koch, Ryan P Smith, Andrew E Hunter, and Steven T Cundiff. Quantum spectroscopy with schrödinger-cat states, 2011. 115

[149] Valentina Amitrano, Alessandro Roggero, Piero Luchi, Francesco Turro, Luca Vespucci, and Francesco Pederiva. Trapped-ion quantum simulation of collective neutrino oscillations. *Physical Review D*, 107(2):023007, 2023. 117

[150] Ha C Nguyen, Bao G Bach, Tien D Nguyen, Duc M Tran, Duy V Nguyen, and Hung Q Nguyen. Simulating neutrino oscillations on a superconducting qutrit. *Physical Review D*, 108(2):023013, 2023. 117

[151] Michael J Cervia, Pooja Siwach, Amol V Patwardhan, AB Balantekin, SN Coppersmith, and Calvin W Johnson. Collective neutrino oscillations with tensor networks using a time-dependent variational principle. *Physical Review D*, 105(12):123025, 2022. 117

[152] Chenfeng Cao, Zheng An, Shi-Yao Hou, DL Zhou, and Bei Zeng. Quantum imaginary time evolution steered by reinforcement learning. *Communications Physics*, 5(1):57, 2022. 117

[153] Amira Abbas, David Sutter, Christa Zoufal, Aurélien Lucchi, Alessio Figalli, and Stefan Woerner. The power of quantum neural networks. *Nature Computational Science*, 1(6):403–409, 2021. 118

[154] Bingzhi Zhang and Quntao Zhuang. Energy-dependent barren plateau in bosonic variational quantum circuits. *arXiv preprint arXiv:2305.01799*, 2023. 119

[155] Xiu Gu, Anton Frisk Kockum, Adam Miranowicz, Yu xi Liu, and Franco Nori. Microwave photonics with superconducting quantum circuits. *Physics Reports*, 718-719:1–102, 2017. Microwave photonics with superconducting quantum circuits. 119

[156] Max Hofheinz, H. Wang, M. Ansmann, Radoslaw C. Bialczak, Erik Lucero, M. Neeley, A. D. O'Connell, D. Sank, J. Wenner, John M. Martinis, and A. N. Cleland. Synthesizing arbitrary quantum states in a superconducting resonator. *Nature*, 459(7246):546–549, May 2009. 119

[157] Brian Vlastakis, Gerhard Kirchmair, Zaki Leghtas, Simon E. Nigg, Luigi Frunzio, S. M. Girvin, Mazyar Mirrahimi, M. H. Devoret, and R. J. Schoelkopf.

Deterministically encoding quantum information using 100-photon schrödinger cat states. *Science*, 342(6158):607–610, 2013. 119

[158] Raphaël Lescanne, Marius Villiers, Théau Peronnin, Alain Sarlette, Matthieu Delbecq, Benjamin Huard, Takis Kontos, Mazyar Mirrahimi, and Zaki Leghtas. Exponential suppression of bit-flips in a qubit encoded in an oscillator. *Nature Physics*, 16(5):509–513, May 2020. 119

[159] P. Campagne-Ibarcq, A. Eickbusch, S. Touzard, E. Zalys-Geller, N. E. Frattini, V. V. Sivak, P. Reinhold, S. Puri, S. Shankar, R. J. Schoelkopf, L. Frunzio, M. Mirrahimi, and M. H. Devoret. Quantum error correction of a qubit encoded in grid states of an oscillator. *Nature*, 584(7821):368–372, Aug 2020. 119

[160] Timo Hillmann, Fernando Quijandría, Göran Johansson, Alessandro Ferraro, Simone Gasparinetti, and Giulia Ferrini. Universal gate set for continuous-variable quantum computation with microwave circuits. *Phys. Rev. Lett.*, 125:160501, Oct 2020. 119

[161] A. Grimm, N. E. Frattini, S. Puri, S. O. Mundhada, S. Touzard, M. Mirrahimi, S. M. Girvin, S. Shankar, and M. H. Devoret. Stabilization and operation of a kerr-cat qubit. *Nature*, 584(7820):205–209, Aug 2020. 119

[162] M. Sandberg, C. M. Wilson, F. Persson, T. Bauch, G. Johansson, V. Shumeiko, T. Duty, and P. Delsing. Tuning the field in a microwave resonator faster than the photon lifetime. *Applied Physics Letters*, 92(20):203501, 05 2008. 119

[163] Elie Bermot, Christa Zoufal, Michele Grossi, Julian Schuhmacher, Francesco Tacchino, Sofia Vallecorsa, and Ivano Tavernelli. Quantum generative adversarial networks for anomaly detection in high energy physics. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 1, pages 331–341. IEEE, 2023. 120

[164] Marcello Benedetti, Brian Coyle, Mattia Fiorentini, Michael Lubasch, and Matthias Rosenkranz. Variational inference with a quantum computer. *Physical Review Applied*, 16(4):044057, 2021. 120

Vita

Shikha Bangar was born in New Delhi, India. In 2011, she began her higher education at the Indian Institute of Science Education and Research (IISER), Bhopal, where she pursued a dual major in Physics and a minor in Earth and Environmental Science. For her Master's thesis, she focused on "Cosmic Inflation" under the guidance of Dr. Rajib Saha. She graduated with a BS-MS Dual Degree in Science, specializing in Physics, in May 2016.

In 2018, Shikha commenced her doctoral studies in Physics at the University of Tennessee. After passing her qualifying exams, she was a graduate teaching assistant for two years. In the summer of 2020, she joined Dr. George Siopsis's research group. Her tenure as a graduate research student was marked by her involvement in various projects that led to multiple scholarly publications. Her academic excellence was recognized with the Graduate Advancement & Training Education (GATE) Fellowship for the academic year 2022-23, which was subsequently renewed due to her outstanding performance. Additionally, she received several travel awards to present her research at national and international conferences. Shikha defended her Ph.D. in July 2024. Starting in the fall of 2024, she will continue her academic career as a Physics lecturer in the Physics department at UTK.